IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2024.0429000

Blindfolded Flight: A Novel Approach for Secure Drone Flight

KAZUMASA OIDA[1], (Member, IEEE), TOMOYA KOREZAWA[1], TAIKI YAMADA[1], AND SOJIRO ETO[1]

[1]Department of Computer Science and Engineering, Fukuoka Institute of Technology, Fukuoka, 811-0295 Japan

Corresponding author: Kazumasa Oida (e-mail: oida@fit.ac.jp).

This work was supported in part by the Japan Science and Technology Agency (JST).

ABSTRACT This study proposes a positioning system that allows a drone to determine its location independently of external data sources. The system is one of the key factors in achieving blindfolded flight, which is the ability to follow a predetermined flight route using only the aircraft's internal data, even in the presence of spoofing, interference, or unavailability of any positioning systems. The system remains operational even when the sensor signals are of low quality. The GPS spoofing detection method based on this system effectively manages slow drift attacks, and its stability is demonstrated by the small coefficients of variation (CVs) of detection delays.

The prediction of the current position is based on deep learning rather than motion equations. Existing approaches to blindfolded flight are constrained by cost, flight time, or flight distance. The proposed system does not significantly suffer from these limitations. The flight time limitations of the latest studies are approximately 100 s. The system demonstrates no tendency for prediction error to increase over time during the 360-second flight, regardless of the presence of GPS spoofing attacks.

INDEX TERMS Drone, GPS spoofing, neural network, sensor, time series

I. INTRODUCTION

Drones, which are unmanned aerial vehicles discussed in this paper, are expanding into various industrial sectors, including logistics, construction, networking, and agriculture [1], [2]. Following significant expansion in the near future, they will inevitably become primary targets of cyberattacks, similar to those experienced by smartphones. A multitude of papers have been published on the topics of GPS spoofing, jamming, and denied environments. This is not only because these challenges hinder drones from playing important roles in the aforementioned fields but also because drones have access to various alternative means to determine their positions. Examples include signals from Wi-Fi/cellular base stations, infrared/visible/ultraviolet images, reflected laser light, various beacons, and ambient signals not originally intended for positioning [3], [4].

Cyberattacks will inevitably target all these positioning methods, not just GPS [5]. There are numerous spoofing attacks (fake GPS, fake Wi-Fi/cellular stations, fake ground stations, fake beacons, etc.) and jamming attacks [5]. Relying on other external positioning sources during GPS outages [4] is a temporary measure. Additionally, because attacks and defenses are in a constant state of co-evolution, any countermeasures focused on a specific attack technique are temporary [6]. While adopting strong authentication technologies [7] (which may require infrastructure updates) can be effective against spoofing attacks, it does not offer protection against jamming or unavailability. Without long-term, broadly applicable safety mechanisms, the widespread use of drones can become both dangerous and costly.

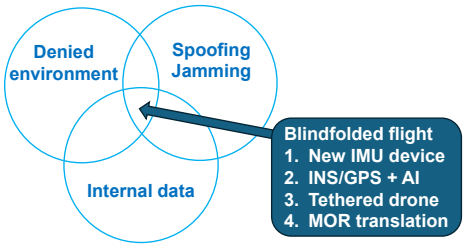


FIGURE 1. Four existing approaches aiming at the intersection of the three conditions.

Such a safety mechanism will ultimately require drones to perform "blindfolded flight," a permanently effective countermeasure against both known and unknown attack methods. In this paper, the term blindfolded flight refers to the ability of a drone to automatically follow a predetermined flight route using only internal data, even in an environment where

positioning systems are spoofed/jammed/rejected. Fig. 1 intuitively illustrates blindfolded flight as the intersection of three requirements. In addition, it presents four existing approaches (detailed later) that are expected to enable blindfolded flight under a variety of conditions; the fourth is our approach.

Approaches 1-3 focus on minimizing sensor noise or its effects by improving part of the drone's control mechanism. However, approach 4 does not modify the drone's control mechanism because its aim is to introduce a new positioning system that transmits current positions to the control mechanism. Most current drones are controlled using proportional-integral-derivative (PID) controllers, which require numerical integration and differentiation to determine control quantities. Therefore, the traditional focus has been on acquiring high-quality (low noise and frequently sampled) sensor data. Our approach is different. We use internal sensors to control drones on coarser time scales, just as humans use vision to control them on coarse time scales. One of the key contributions of this study is the demonstration that calculating drone positions on a coarse time scale (e.g., 1 to 10 s) provides the following advantages.

- Increasing sensor noise from 0 to 10,000 times the signal does not significantly degrade the performance of the proposed system. Greater noise immunity results in improved safety and reduced cost.
- Increasing the data collection interval from 1 to 10 s does not degrade the performance of the system. A longer interval requires less communication, memory, and computation.
- The system shows no tendency for prediction errors to increase over time during a 360-second flight, which is crucial for long blindfolded flights.

The remainder of this paper is organized as follows. Section II presents research relevant to this study. Section III provides an overview of the drone autopilot technology and Perceiver, the deep-learning model used in this work. Section IV describes the proposed system, including data preprocessing and the algorithm for efficiently predicting drone positions. Section V describes how the preprocessing and algorithm parameters affect system behavior. Section VI proposes a GPS spoofing detection method based on the proposed system. Section VII compares our approach with existing studies in terms of prediction errors and GPS outage times. Section VIII discusses the limitations of the system, potential applications, and ethical considerations. Finally, section IX concludes the paper and suggests directions for future research.

II. RELATED WORK

A. SENSOR SPOOFING

Sensor spoofing attacks are conducted by transmitting malformed signals, altering sensor measurements, or disrupting the normal operation of sensors. The primary objective of these attacks is to deceive systems that depend on sensors for decision-making, including navigation and security. Several sensors have recently become primary targets of these attacks

TABLE 1. Comparisons with existing approaches.

Approach	adv	disadv	app	time scale
1. new IMU [27]–[29]	high quality output	expensive	aerospace military	very fine < 1 ms
2. INS/GPS [26], [30], [31]	no prior training	limited flight time	versatile	fine INS/GPS
3. Tethered [32]–[34]	no prior knowledge	limited flight area	indoor tracking	fine PID
4. MOR [35]	data quality insensitive	prior training	versatile	coarse human

[5]: GPS [8], [9], light detection and ranging (LiDAR) [10], [11], camera [12], [13], inertial measurement unit (IMU) [14], [15], ultrasonic sensor [16], [17], and millimeter wave (MMW) radars [18], [19]. These sensors are indispensable for controlling autonomous vehicles, robots, drones, etc. Extensive research has been conducted on individual sensors and attack vectors, and new attack schemes continue to emerge [20].

B. DEAD RECKONING

Dead reckoning is a traditional navigation method used to estimate the current position of a moving object, such as a ship, vehicle, or person. The position is determined based on the initial position, speed, direction, and elapsed time. Dead reckoning is useful in environments where GPS signals are unavailable or unreliable, such as indoors, underground, or in urban areas where tall buildings block satellite signals. This technology is being explored for its potential in detecting GPS spoofing [6]. A modern implementation of dead reckoning, the inertial navigation system (INS), uses IMUs (consisting of accelerometers, gyroscopes, etc.) along with a computer to determine the position, orientation, and speed of a moving object without relying on external references.

Dead reckoning is prone to error accumulation, as changes in speed and direction can cause significant errors over time. Additionally, it does not consider external factors, such as wind and currents, that can affect the actual path of movement. Therefore, various technologies are being developed, including sensor fusion [21], [22], visual/radar odometry [23], [24], and machine learning [25], [26]. At the device level, advancements are being made with technologies, such as quantum accelerometer/gyroscope [27] and fiber-optic/ring-laser gyroscope devices [28], [29].

C. BLINDFOLDED FLIGHT

Table 1 compares four blindfolded-flight approaches illustrated in Fig. 1. The first approach focuses on developing high-quality accelerometers and gyroscopes to reduce sensor noise. The second approach aims to create an artificial intelligence (AI) system that outperforms the extended Kalman filter (EKF). These AI systems are trained using GPS signals when GPS is available, and in the absence of GPS, they work to reduce INS errors. The flight times during which the AI system is active range from 40 and 100 seconds, as noted in [31]. The third approach involves developing cable-tethered

drones that are resistant to malicious wireless signals, with the cable length determining the movable range of the drone. The fourth approach, which is the focus of this paper, involves developing a system that translates manual operation representation (MOR) signals into drone positions. MOR signals are signals from the human-operated control unit, such as roll, pitch, yaw, and throttle signals when transmitted to the quadcopter.

As mentioned in Section I, the key difference between our approach and existing approaches lies in the time scale at which they operate on the drone system. Table 1 shows that approaches 1-3 aim to improve or replace existing components (e.g., IMU and EKF) that function on fine time scales. However, our approach designs a positioning system that operates on coarse time scales. The system is attached to a drone as an additional positioning system, thereby reducing variability of prediction errors at coarser time scales. Our previous work [35] demonstrated that it is possible to convert MOR signals to autopilot Python programs. In this study, we extend that work in three ways. First, we propose an algorithm to convert MOR signals into current positions (rather than Python programs as in [35]). Second, we examine the impact of input data quality on system performance. Third, we introduce a GPS-spoofing detection method.

D. OTHER RELATED WORK

There are two approaches that, although they do not meet the blindfolded flight requirements, are highly effective against GPS attacks or GPS-denied environments. The first approach is signals of opportunity (SOPs) [36], [37], which utilizes ambient radio-frequency (RF) signals not originally intended for positioning, navigation, and timing (PNT) sources. These signals include AM/FM, digital television, cellular, and satellite communication signals. The second approach is cooperative anti-jamming [38], which combines signals from multiple homogeneous drone sensors to address intentional or unintentional GPS interference. This method is suitable for cooperation with relative positioning systems in swarm formation [39], [40].

Even when a drone operates in blindfolded flight mode, certain IMU devices may be vulnerable to exploitation, potentially disrupting its operations [15]. Experiments in [41] showed that malicious sound noise could degrade the accuracy of micro-electro-mechanical systems (MEMS) gyroscopes. Additionally, malicious acoustic injections have the potential to damage the digital integrity of MEMS accelerometers [42]. The authors in [43] demonstrated that sensor reconfiguration attacks caused abnormal sensor behavior as well as led the drone EKF to a complete halt.

III. BACKGROUND

A. DRONE AUTOPILOT

The key components for enabling drone autopilot are PID controllers and EKFs [44]. The PID controllers adjust the drone's attitude, position, and speed in real time to maintain stable flight. The EKF corrects the estimated drone states, ensuring that the PID controller receives more accurate input. The EKF fuses data from multiple sensor devices to perform optimal state estimation, considering noise and errors of each device [21], [22]. Various external signals (GPS, cameras, LiDAR, etc.) are used to correct INS errors. In the absence of external signals, these errors accumulate over time, causing a degradation in the accuracy of INS estimates.

B. TRANSFORMER

The Transformer is a deep learning model that has driven the current AI revolution in natural language processing. One of its most distinctive features is the use of self-attention mechanisms, which assess the relevance of different parts of the input sequence to one another, enabling the model to capture complex relationships and dependencies [45]. In certain aspects, Transformer outperforms traditional convolutional neural network (CNN) and recurrent neural network (RNN) models. For example, Transformers can process multiple input sequences using graphics processing units (GPUs), whereas RNNs process input sequences sequentially, leading to longer training times. Additionally, Transformers excel at capturing long-range dependencies in the input sequence, a capability that CNNs lack. This feature is essential for drones flying routes with long distances between waypoints.

Currently, Transformers are employed in the fields of text, speech, and image processing, with applications in time series analysis also underway [46], [47]. Because the CPU and memory consumption of self-attention scales quadratically the input size, more efficient Transformer-based architectures have been developed. This study uses Perceiver, one such scalable architecture [48], [49].

IV. PROPOSED SYSTEM

This study proposes a system that predicts the three-dimensional positions (referred to as "the position time series") of a drone based on the time series of its state (roll, pitch, yaw, throttle) (referred to as "the state time series"). All experiments in this paper employ GPS to collect position time series, although the specific positioning system used is not critical. This study focuses on the control of quadcopters. If other types of aircraft are considered, the sensor types in the state time series may vary, and the characteristics of each aircraft type would be captured in the neural network parameters through learning.

A. CONFIGURATION

Fig. 2 shows the proposed system, which incorporates Perceiver [48], [49] to translate state time series $\{r_s\}_{1\leq s\leq t}$ into position time series $\{p_s\}_{1\leq s\leq t}$. Similar to the process used in natural language machine translation [45], the system does not perform batch translation from $\{r_s\}$ into $\{p_s\}$; instead, it infers current position p_t, using $\{r_s\}_{1\leq s\leq t}$ and all previously calculated positions $\{p_s\}_{1\leq s\leq t-1}$ (see "positions calculated before time t" in Fig. 2). The input in Fig. 2 includes p_t because it acts as the initial value of position p_t (see "initial value at time t" in Fig. 2).

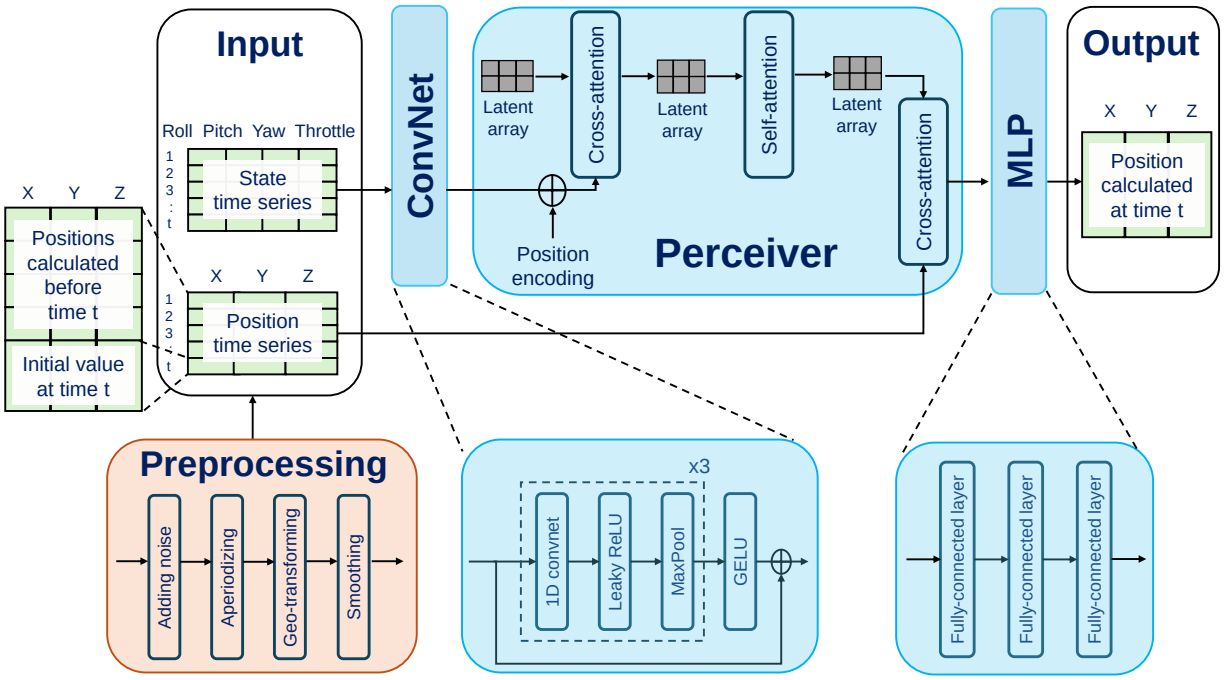


FIGURE 2. Proposed system outputs the current position of the drone (x, y, z). The state (position) time series are preprocessed and input into the first (second) cross-attention of the Perceiver model to embed (extract) features from the state time series into (from) the latent array. A ConvNet is used to efficiently extract the features, and the MLP is added to enable the system to function as a regression model.

As shown in Fig. 2, both input time series are preprocessed before being fed into the Perceiver. The state time series first passes through ConvNet, comprising a one-dimensional CNN and max pooling (MaxPool), for local feature extraction. The extracted features are transcribed into a latent array (a 4×128 matrix) through the cross-attention process. The self-attention mechanism subsequently embeds correlations between all feature pairs into the array. The second cross-attention process reconstructs the position time series from the latent array. Finally, a multi-layer perceptron (MLP) selects the optimal p_t from the reconstructed time series.

The Perceiver compresses the state time series into a small latent array, thereby reducing the computation and memory requirements for self-attention [48]. The ConvNet and MLP are implemented using the open source library Pytorch [50], while Perceiver is available on Hugging Face [51]. The hyperparameter values used in the experiments are as follows. The kernel size and stride of the CNN (MaxPool) are 3 and 1 (2 and 2), respectively. The number of heads (blocks) for cross-attention and self-attention are both set to 4 (1). The initial value of the latent array is assigned randomly.

B. PREPROCESSING

Fig. 2 shows that preprocessing consists of four operations: smoothing, aperiodization, geo-transformation, and noise addition. Fig. 3 describes how these operations transform the sensor signals (roll, pitch, yaw, throttle, longitude, latitude, and altitude) into inputs for the proposed system. Note that the position time series are used as ground truth (initial values) in

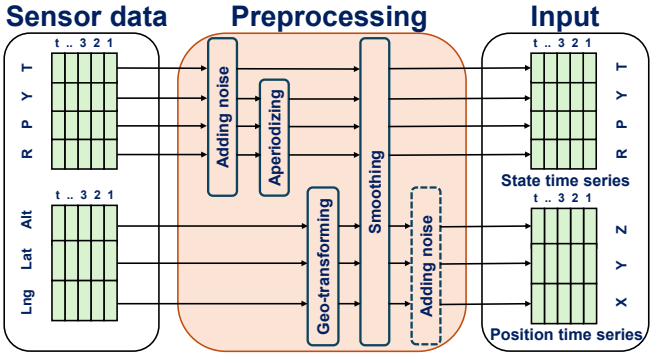


FIGURE 3. Input of the proposed system is generated through four operations: smoothing, aperiodizing, geo-transforming, and adding noise. The adding noise enclosed by the dashed rectangle is performed only in the inference phase. R, P, Y, and T denote roll, pitch, yaw, and throttle signals, respectively.

the learning (inference) phase. For performance evaluation, the noise addition operation enclosed by the dashed rectangle in Fig. 3 is performed to generate initial values. Fig. 4 illustrates smoothing and aperiodization. As shown in the figure, smoothing generates a time series consisting of the arithmetic means of the windows, with each window centered on each step. Unless otherwise specified, the window size w and step size ℓ satisfy $w = \ell/2$. All time series in this paper were collected at a sampling interval of 0.1 s. Aperiodization adds (or subtracts) 360 degrees to (from) the angle of roll, pitch, or yaw to ensure the correct amount of signal change.

Algorithm 1 describes the aperiodization operation, where

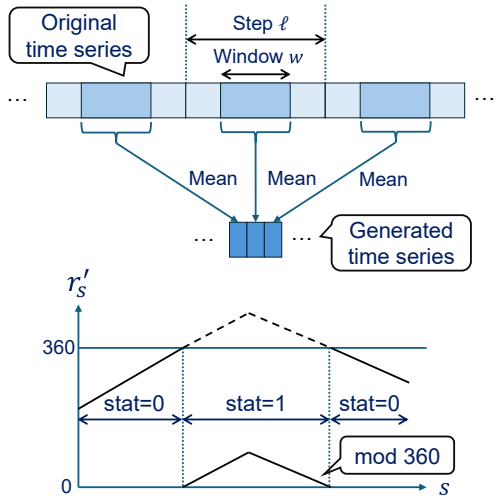


FIGURE 4. **Upper: Smoothing generates a time series of arithmetic means, each of which is derived from a window of size w, and each step of size ℓ has one window. Lower: Aperiodization transforms time series $\{r'_s\}$ into a smoother one by replacing the solid line with the dashed line.**

Algorithm 1 Aperiodization.

Precondition: Without mod 360 operation, $|r'_{s+1} - r'_s| < 180$
Input: $\{r'_s\}_{1 \le s \le t}$
Output: $\{r'_s\}_{1 \le s \le t}$

for $s = 1$ to $t - 1$ **do**
  if $r'_{s+1} - r'_s \le -180$ **then**
    $stat \leftarrow stat + 1$
  else if $r'_{s+1} - r'_s \ge 180$ **then**
    $stat \leftarrow stat - 1$
  end if
  $r'_{s+1} \leftarrow r'_{s+1} + 360 \times stat$
end for
return $\{r'_s\}_{1 \le s \le t}$

the variable *stat* indicates the number of times to add (subtract) 360 to (from) r'_s if the *stat* is positive (negative). Geo-transformation converts GPS signals (altitude, latitude, longitude) to Cartesian coordinates $x = (x_1, x_2, x_3)$, where the origin $(0, 0, 0)$ is the takeoff location of the drone. The positive x_1, x_2, and x_3 are the east, north, and up, respectively. For simplicity, x_1, x_2, and x_3 are referred to as the x, y, and z coordinates, respectively.

To verify stability against noise, preprocessing adds noise to the state time series by introducing different random numbers n_r to the roll, pitch, yaw, and throttle signals r_t as

$$r_t \leftarrow r_t + n_r, \ n_r \sim \mathcal{N}(\mu, \sigma^2), \tag{1}$$

where $\sim$ denotes n_r follows a normal distribution $\mathcal{N}(\mu, \sigma^2)$, with mean μ and standard deviation σ selected according to one of two rules: they are $g_m(\ge 0)$ times greater than $\bar{r}$, the arithmetic mean of samples $\{r_s\}$; i.e.,

$$\mu = \sigma = g_m \bar{r} \tag{2}$$

or they are $g_s(\ge 0)$ times greater than each sample r_s; i.e.,

$$\mu = \sigma = g_s r_s. \tag{3}$$

From (2)-(3), the coefficient of variation (CV) is 1. Note that (2) implies μ and σ are invariant with time step s and (3) implies they vary with s.

The preprocessing adds noise to the position time series by introducing a three-dimensional random vector n_p to the position p_t as follows:

$$p_t \leftarrow p_t + n_p, \tag{4}$$
$$n_p = (\lambda + \omega u)v, \ u \sim \mathcal{U}_{[-1,1]}, \tag{5}$$

where $\mathcal{U}_{[-1,1]}$ is the uniform distribution within the range $[-1, 1]$, and v is a 3-dimensional vector. From (4)-(5), position p_t is shifted from the correct position by $(\lambda + \omega u)\|v\|_2$ in the v direction, where $\|v\|_2 := \sqrt{|v_1|^2 + |v_2|^2 + |v_3|^2}$ if $v = (v_1, v_2, v_3)$. Therefore, real numbers λ and ω influence the size and variation of the position shift, respectively. Because the training phase uses the position time series as the correct labels, noise n_p is added only in the inference phase. This study investigates whether the system can determine the correct position p_t, starting from the noisy initial position $p_t + n_p$.

C. TRAINING AND DATASETS

The proposed system was trained using datasets consisting of pairs of state time series and position time series, with the latter serving as the true labels. Each pair corresponds to a single flight route, thus the size of a dataset (K) is determined by the number of flight routes. The loss function assesses the error between predicted and true positions using the Euclidean distance $\|\cdot\|_2$. In this study, the number of training epochs is set to 500. Unless otherwise mentioned, training and test are performed using leave-one-out cross-validation (LOOCV), which is a special case of k-fold cross-validation where $k = K$.

The study employed two identical quadcopters, both configured with the open source software ArduPilot [44] and the flight controller Pixhawk [52]. Two datasets were obtained from the log data of Pixhawk: one dataset ($K = 40$) was collected at the seaside during the summer and the other ($K = 31$) in an urban area during the winter. The two distinct collection sites and seasons provide datasets that account for a variety of wind strengths and temperatures. Additionally, this study uses a dataset ($K = 40$) obtained from the SITL simulator [53].

D. INFERENCE PHASE

Algorithm 2, which employs the trained proposed system, includes the following notations (see Appendix A for quick reference):

- $\bmod$: Modulo operation, which returns the remainder of a division.
- $[P]$: Iverson bracket of statement P; i.e., $[P] = 1$ if P is true; otherwise, $[P] = 0$. $[q = 0]$ implies bit inversion of

q if $q \in \{0,1\}$; i.e., $[q=0]=0$ ($[q=0]=1$) if $q=1$ ($q=0$).
- $\beta[i]$: $(i+1)$-th element of two-dimensional vector β; i.e., $\beta=(\beta[0],\beta[1])$.
- $\odot$: Hadamard product; i.e., $(\delta[0],\delta[1])\odot([q=0],[q=1])=(\delta[0][q=0],\delta[1][q=1])$.
- $R(\theta)\mathbf{1}^t$: Product of two-dimensional rotation matrix $R(\theta)$ and transposed vector of $\mathbf{1}=(1,1)$; i.e.,

$$R(\theta)\mathbf{1}^t=\begin{pmatrix}\cos(\theta) & -\sin(\theta)\\ \sin(\theta) & \cos(\theta)\end{pmatrix}\begin{pmatrix}1\\1\end{pmatrix}. \tag{6}$$

If all elements of a vector have the same value, the vector can be expressed in terms of the value of its single element. For example, $B=10$ implies $B=(10,10)$.

Algorithm 2 Algorithm for approaching p_t^*

Input: $M,\theta,B,\delta,p_t^0,\{\bar{p}_s^*\}_{1\le s\le t-1},\{r_s\}_{1\le s\le t}$
Output: $\bar{p}_t^*$

Initialization: $\gamma\leftarrow 4$
$\ell\leftarrow(\infty,\infty)$, $c\leftarrow(0,0)$, $\mu\leftarrow(0,0)$, $\beta\leftarrow(0,0)$
1: **for** $k=1$ to M **do**
2: $q\leftarrow k \mod 2$
3: $\delta'\leftarrow\delta\odot([q=0],[q=1])$
4: $B'\leftarrow B\odot([q=0],[q=1])$
5: $\beta[q]\leftarrow\mathcal{N}(\mu[q],\delta[q]^2)$
6: $p_t^k\leftarrow p_t^0+(\beta[0],\mu[1],0)[q=0]+(\mu[0],\beta[1],0)[q=1]$
7: $\bar{p}_t^k\leftarrow \mathrm{mor2pos}\left(p_t^k;\{\bar{p}_s^*\}_{s\in[1,t-1]},\{r_s\}_{s\in[1,t]}\right)$
8: **if** $\ell[q]>\|p_t^k-\bar{p}_t^k\|_2$ **then**
9: $\ell[q]\leftarrow\|p_t^k-\bar{p}_t^k\|_2$
10: $\bar{p}_t^*\leftarrow\bar{p}_t^k[\ell[q]<\ell[[q=0]]+\bar{p}_t^*[\ell[q]\ge\ell[[q=0]]]$
11: $c[q]\leftarrow c[q]+1$
12: **if** $c[q]=\gamma$ **then**
13: $\mu[q]\leftarrow\mu[q]+\delta' R(\theta-\frac{\pi}{4})\mathbf{1}^t$
14: **end if**
15: **else**
16: $c[q]\leftarrow c[q]-1$
17: **if** $c[q]=-\gamma$ **then**
18: $\mu[q]\leftarrow\mu[q]-\delta' R(\theta-\frac{\pi}{4})\mathbf{1}^t$
19: **end if**
20: **end if**
21: **if** $|c[q]|=\gamma$ **then**
22: $c[q]\leftarrow 0$
23: **if** $\mu[q]>0$ **then**
24: $\mu[q]\leftarrow\min(\mu[q],|B' R(\theta-\frac{\pi}{4})\mathbf{1}^t|)$
25: **else**
26: $\mu[q]\leftarrow\max(\mu[q],-|B' R(\theta-\frac{\pi}{4})\mathbf{1}^t|)$
27: **end if**
28: **end if**
29: **end for**
30: **return** $\bar{p}_t^*$

Algorithm 2 outputs the optimal prediction of the drone position at time t ($\bar{p}_t^*$) by iteratively executing mor2pos, the trained proposed system. Line 7 of the algorithm

$$\bar{p}_t^k\leftarrow\mathrm{mor2pos}\left(p_t^k;\{\bar{p}_s^*\}_{s\in[1,t-1]},\{r_s\}_{s\in[1,t]}\right), \tag{7}$$

implies that mor2pos derives $\bar{p}_t^k$ (the k-th prediction of the drone position at time t) from initial value p_t^k. Other inputs to mor2pos in (7), $\{\bar{p}_s^*\}_{s\in[1,t-1]}$ and $\{r_s\}_{s\in[1,t]}$, remain fixed during the calculation of $\bar{p}_t^k, k=1,2,\ldots,M$, where M is the number of iterations and $\bar{p}_t^*$ (the optimal prediction at t) is defined by lines 8-10 as:

$$\bar{p}_t^*:=\bar{p}_t^\ell \text{ if } \ell=\underset{1\le k\le M}{arg\,min}\,\|p_t^k-\bar{p}_t^k\|_2. \tag{8}$$

Note that $\{\bar{p}_s^*\}_{s\in[1,t-1]}$ in (7) represents all optimal predictions before time t calculated by Algorithm 2. Let p_t^* be the true drone position at t. Equation (8) implies Algorithm 2 approaches p_t^* by minimizing $\|p_t^k-\bar{p}_t^k\|_2$.

Fig. 5 illustrates the behavior of Algorithm 2. In the upper left panel, it indicates that a loop is formed by three operations: executing mor2pos as in (7), deriving a new input p_t^{k+1}, and replacing the previous input with the new one. In the lower left panel, starting from p_t^0, M predictions $\bar{p}_t^1,\ldots,\bar{p}_t^M$ are sequentially generated. The optimal prediction $\bar{p}_s^*$ (red triangle) defined in (8) is considered the closest to the true position p_t^* (green circle).

Line 2 indicates that $q=0$ ($q=1$) if integer k is even (odd). On line 6, the input p_t^k is selected as

$$p_t^k\leftarrow p_t^0+(\beta[0],\mu[1],0)[q=0]+(\mu[0],\beta[1],0)[q=1]. \tag{9}$$

From Iverson's notation, (9) implies that $p_t^k=p_t^0+(\beta[0],\mu[1],0)$ if $q=0$; otherwise $p_t^k=p_t^0+(\mu[0],\beta[1],0)$. From line 5, a normal distribution generates $\beta[0]$ if $q=0$; otherwise $\beta[1]$ is generated. Therefore, p_t^k moves alternately in the x and y directions as k increases; however, it does not move in the z direction. This is because the z-coordinate of $\bar{p}_t^k$ remains close to the true altitude, even when k is small.

Let us next describe how to update μ. From lines 8, 11, 15, and 16, the counter c is incremented (decremented) by one whenever the smallest $\|p_t^k-\bar{p}_t^k\|_2$ is (not) found. If the counter reaches the threshold γ ($-\gamma$), μ is increased (decreased) by $\delta' R(\theta-\frac{\pi}{4})\mathbf{1}^t$ at line 13 (line 18), ensuring that μ stays longer in regions where distance $\|p_t^k-\bar{p}_t^k\|_2$ is small. Here δ' indicates how much μ moves at a time, and θ represents the direction of p_t^* as observed from p_t^0. The threshold-based method was introduced because the loss function may not be sufficiently smooth to obtain the minimum using numerical differentiation. Lines 23-27 state that μ is bounded as:

$$-|B' R(\theta-\frac{\pi}{4})\mathbf{1}^t|\le\mu[q]\le|B' R(\theta-\frac{\pi}{4})\mathbf{1}^t|. \tag{10}$$

Fig. 5(right) shows bound (10) when $\theta=\frac{\pi}{4}$. In this case, from (10), $\mu\in\Phi$, where

$$\Phi=\{(x,y)|x\in[-B[0],B[0]],y\in[-B[1],B[1]]\}. \tag{11}$$

The rectangle rotates counterclockwise around point p_t^0 as θ increases. In the figure, the small circle in the top right-hand corner of the rectangle represents the position of μ, and the

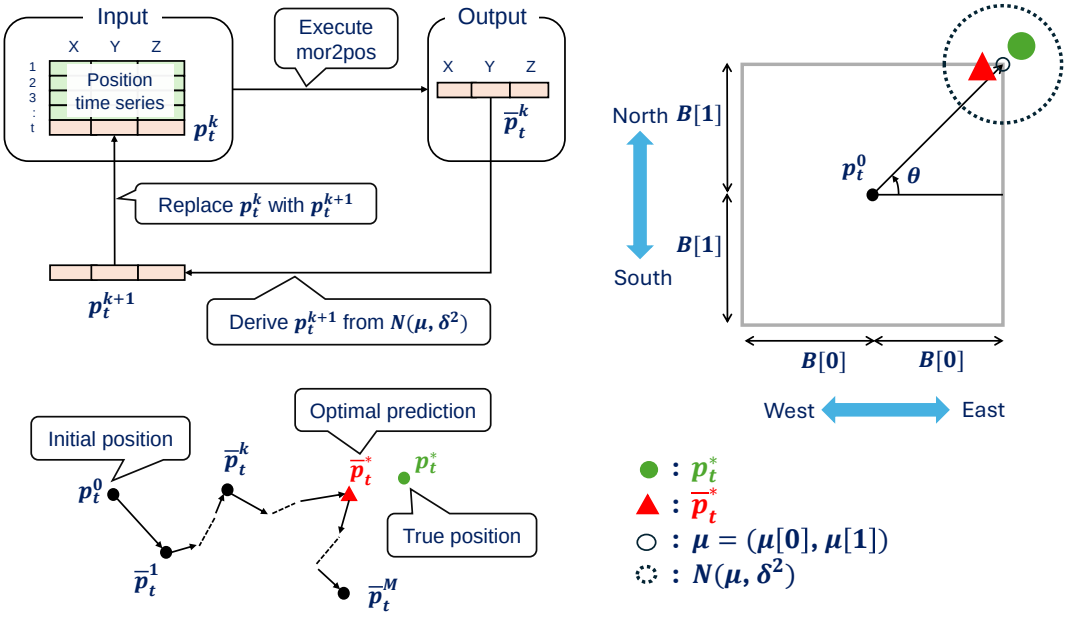


FIGURE 5. Basic behavior of Algorithm 2. Upper left: three operations are performed repeatedly to generate M predictions $\bar{p}_t^1, \ldots, \bar{p}_t^M$. Lower left: M predictions are calculated, each progressively approaching the true position p_t^*. Right: μ is restricted within the rectangle and the dashed circle intuitively represents the sampling range of new input p_t^k.

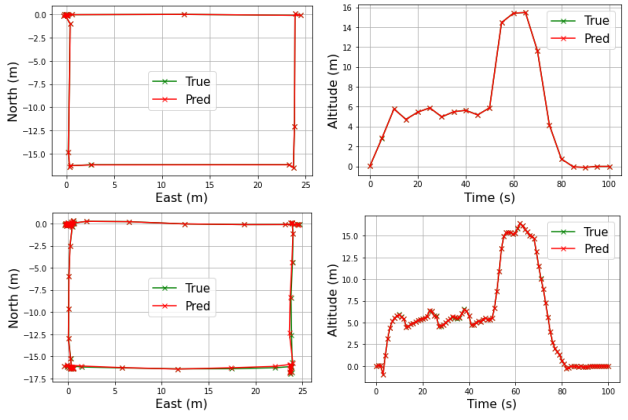


FIGURE 6. True $\{p_s^*\}$ (green) and predicted positions $\{\bar{p}_s\}$ (red) are expressed as trajectories on the x–y (east–north) plane and on the z–t (altitude–time) plane when $\{p_s^0\} = \{p_s^*\}$. The step sizes ℓ are (upper) 50 and (lower) 10. The drone departs from the origin (0, 0), follows a rectangular trajectory, ascends, and then lands. $(K, g_m, \lambda, \omega) = (66, 10, 0, 0)$.

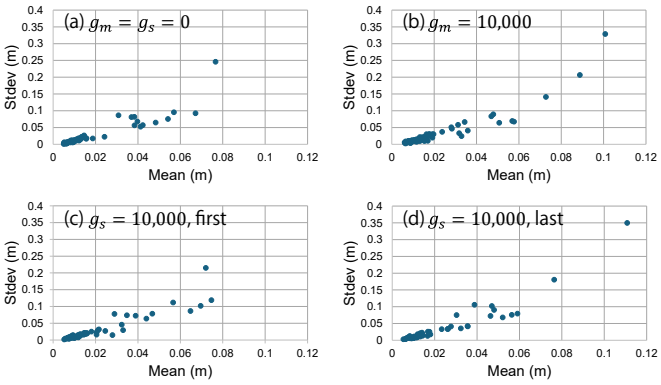


FIGURE 7. All graphs show small distances $\|\bar{p}_t - p_t^*\|_2$. Each dot denotes the mean and standard deviation of distances $\|\bar{p}_t - p_t^*\|_2$, $t \geq 1$, of one of the 66 flight routes. $\{p_s^0\} = \{p_s^*\}$ and $(K, \ell, \lambda, \omega) = (66, 100, 0, 0)$.

dashed circle around μ intuitively denotes the sampling range of p_t^k.

V. BASIC PERFORMANCE

A. INPUT QUALITY

This section first evaluates mor2pos (without using Algorithm 2) according to the LOOCV test under the condition that initial positions $\{p_s^0\}_{0 \leq s \leq t}$ are equal to the true positions $\{p_s^*\}_{0 \leq s \leq t}$. The aim is to verify whether mor2pos indeed recognizes the true position. If it does not, then $\|\bar{p}_t - p_t^*\|_2 \not\approx 0$ even if $\{p_s^0\} = \{p_s^*\}$, where $\bar{p}_t$ is the predicted position denoted by:

$$\bar{p}_t = \text{mor2pos}\left(p_t^0; \{\bar{p}_s\}_{s \in [1,t-1]}, \{r_s\}_{s \in [1,t]}\right). \quad (12)$$

Fig. 6 shows that mor2pos recognizes $\{p_s^*\}$. From the figure, $\{p_s^*\}$ (green lines) agrees with $\{\bar{p}_s\}$ (red lines) even under high noise ($g_m = 10$) and long step size ($\ell = 10, 50$) conditions. Note from (2) that $g_m = 10$ implies the amount of noise with a CV of one is ten times greater than signal r_t. Additionally, $\ell = 50$ ($\ell = 10$) corresponds to 5 s (1 s), as all time series were originally sampled every 0.1 s.

Fig. 7 statistically verifies $\|\bar{p}_t - p_t^*\|_2 \approx 0$. The figure shows the means and standard deviations of the distances $\|\bar{p}_t - p_t^*\|_2$, $t \geq 1$, for all K ($= 66$) flight routes, where each dot corresponds to one of the routes. For all graphs in the figure, $\ell = 100$; thus, the input data size is 1% of

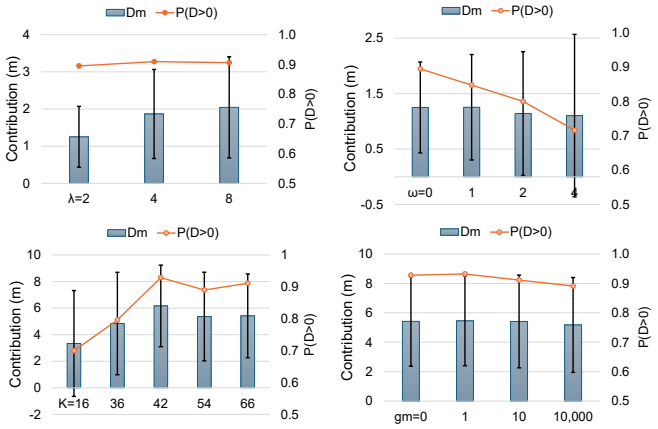


FIGURE 8. Impacts of (upper left) λ, (upper right) ω, (lower left) K, and (lower right) g_m on D_m, D_{sd}, and $P(D>0)$. Error bars denote D_{sd}. The default values are $(K, \ell, g_m, \lambda, \omega) = (66, 100, 0, 0, 0)$, $(M, \theta, B, \delta) = (120, 0, 10, 0.08)$, and $v = (1, 1, 0)$.

the original time series. Figs. 7(a)-(b) demonstrate that the means and standard deviations are less than 0.12 m and 0.4 m, respectively, regardless of whether the noise is non-existent ($g_m = 0$) or overwhelming ($g_m = 10,000$). Figs. 7(c)-(d) indicate that using another noise addition rule ($g_s = 10,000$) yields almost the same outcome, with Fig. 7(c) (Fig. 7(d)) showing the case where noise addition occurs as the first (last) operation in the preprocessing. (In Fig. 3, the addition of noise to the state time series is depicted only as the first operation). The similarity between Fig. 7(c) and Fig. 7(d) demonstrates that the effect of the law of large numbers owing to the smoothing operation is negligible.

B. ALGORITHM CONTRIBUTION

Let us evaluate Algorithm 2, which generates $\bar{p}_t^k$, $k = 1, 2, \ldots, M$, assuming that $\{p_s^0\} \neq \{p_s^*\}$. Let $\mathcal{D}_{t,i}$ represent the contribution of the algorithm in approaching the true position $p_{t,i}^*$ defined by:

$$\mathcal{D}_{t,i} := \|p_{t,i}^0 - p_{t,i}^*\|_2 - \|\bar{p}_{t,i}^* - p_{t,i}^*\|_2, \tag{13}$$

where the subscript i denotes the i-th flight route; e.g., $\bar{p}_{t,i}^k$ represents $\bar{p}_t^k$ for the i-th route. Let L be the set of all (t, i) pairs. Fig. 8 shows the impact of λ, ω, K, and g_m on the performance of the algorithm, where the blue bars and error bars represent the means (D_m) and standard deviation (D_{sd}) of $\{\mathcal{D}_{t,i}\}_{(t,i)\in L}$. Fig. 8 includes the line graphs of $P(D > 0)$, the probability of $\mathcal{D}_{t,i} > 0$, defined by

$$P(D > 0) := \frac{1}{|L|} \sum_{(t,i)\in L} [\mathcal{D}_{t,i} > 0], \tag{14}$$

where $[\cdot]$ is the Iverson bracket.

Fig. 8 reveals two challenges posed by noise n_p defined in (4)-(5).

1) From Fig. 8(upper left), as λ increases, D_m increases slowly compared to the average distance between the initial and true positions, which is equal to $\sqrt{2}\lambda$.

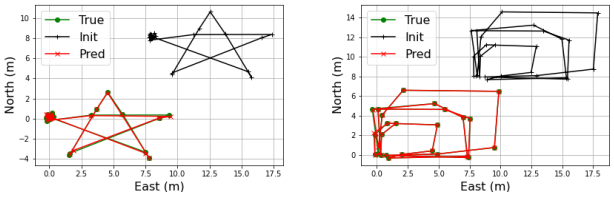


FIGURE 9. True $\{p_s^*\}$ (green), initial $\{p_s^0\}$ (black), and predicted positions $\{\bar{p}_s^*\}$ (red) on the x–y plane when $\theta = \theta^*$ and $B = \lambda$. $E_m = 0.06$ and $E_{sd} = 0.18$. $(K, \ell, g_m, \lambda, \omega) = (26, 50, 10, 8, 0)$, $(M, \theta, B, \delta) = (200, \pi, 8, 0.4)$, and $v = (1, 1, 0)$.

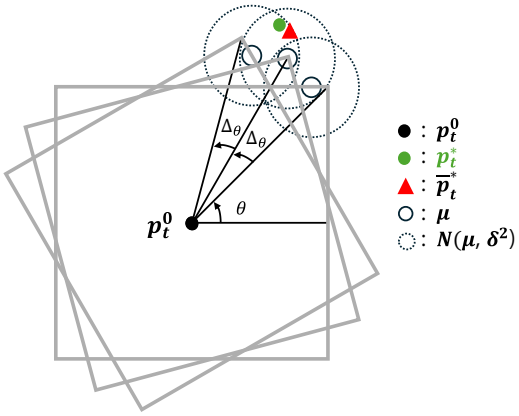


FIGURE 10. Sampling range of p_t^k (the union of the dotted circles) expands by changing θ value.

2) From Fig. 8(upper right), as ω increases, D_{sd} rises and $P(D > 0)$ drops significantly.

Meanwhile, Fig. 8(lower left) indicates that a dataset size K of around 50 is sufficient, and Fig. 8(lower right) reveals high stability against intense noise, with $g_m = 10,000$.

C. CALIBRATION

The above-mentioned two challenges originate from the slow movement of $\bar{p}_t^k$, $k = 1, 2, \ldots, M$. One approach to overcome these challenges is to adjust inputs M, θ, B, and δ of Algorithm 2. Fig. 9 demonstrates that the red and green lines overlap almost perfectly by setting the parameters as $\theta = \theta^*$ and $B = \lambda$, where $\theta^* := \arctan2(v[1], v[0])$, the correct value of θ. Thus, $B = \lambda$ implies that p_t^k stays around the boundary. In the figure, the mean E_m and standard deviation E_{sd} of prediction errors $\mathcal{E}_{t,i}$ defined by

$$\mathcal{E}_{t,i} := \|\bar{p}_{t,i}^* - p_{t,i}^*\|_2, \tag{15}$$

for all $(t, i) \in L$ are significantly small ($E_m = 0.06$ m and $E_{sd} = 0.18$ m).

Incorrect selection of parameter values ($\theta \neq \theta^*$ or $B \neq \lambda$) could cause a sharp increase in $\mathcal{E}_{t,i}$. Fig. 10 illustrates a countermeasure, where Δ_θ is added to θ twice to ensure that $\theta^* \in [\theta, \theta + 2\Delta_\theta]$. Let $\mathcal{F}(M, \theta, p_t^0)$ represent Algorithm 2, where parameters irrelevant to the explanation (B and δ) have

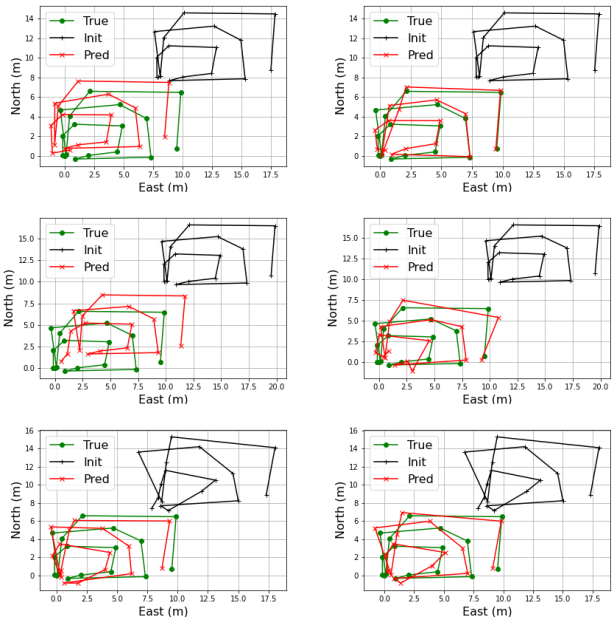


FIGURE 11. Calibration. Upper left: $E_m = 1.58$ when $\theta^* - \theta = \pi/24$. Upper right: Increasing θ by $\pi/48$ twice decreases E_m to 1.15. Middle left: $E_m = 2.96$ when $\lambda - B = 2$. Middle right: Increasing B by 2 m twice decreases E_m to 1.19. Lower left: $E_m = 0.97$ when $\delta = 0.4$ and $\omega = 1$. Lower right: Decreasing δ twice (0.5 to 0.2 and 0.2 to 0.01) decreases E_m to 0.65. $(K, \ell, g_m, \lambda, \omega) = (26, 50, 10, 8, 0)$, $(M, \theta, B, \delta) = (200, \pi, 8, 0.4)$, $(M_1, M_2, M_3) = (100, 50, 50)$, and $v = (1, 1, 0)$.

been omitted. Precisely, the countermeasure executes $\mathcal{F}$ three times as

$$\mathcal{F}(M_3, \theta + 2\Delta_\theta, \mathcal{F}(M_2, \theta + \Delta_\theta, \mathcal{F}(M_1, \theta, p_t^0))), \qquad (16)$$

where the total number of iterations remains the same ($M_1 + M_2 + M_3 = M$).

Fig. 11(upper) demonstrates that (16) reduces E_m from 1.58 to 1.15. Similarly, Fig. 11(middle) shows a decrease in E_m from 2.96 to 1.19 by increasing B by 2 m twice. If the initial positions $\{p_s^0\}$ contain randomness ω (> 0) defined in (4)-(5), the same method can be applied as well. Fig. 11(lower) shows that E_m at $\omega = 1$ decreases from 0.97 to 0.64 by employing three different δ values ($\delta = 0.5, 0.2, 0.01$), where δ adjusts the size of the sampling range, as shown in Fig. 5. Note that Figs. 9 and 11 are obtained under high noise ($g_m = 10$).

VI. EXTERNAL ATTACK

A. DATASET TYPES

In the previous sections, p_t^0 is correlated to p_t^* because $p_t^0 = p_t^* + n_p$, where n_p is a random vector as defined in (5). This section considers the following two types of independent pair p_t^0 and p_t^*, both obtained from the same flight route:

- Type 1: $\{p_s^0\}$ is obtained from a simulator [53] and $\{p_s^*\}$ consists of real sample data.
- Type 2: Both $\{p_s^0\}$ and $\{p_s^*\}$ are real sample data collected independently.

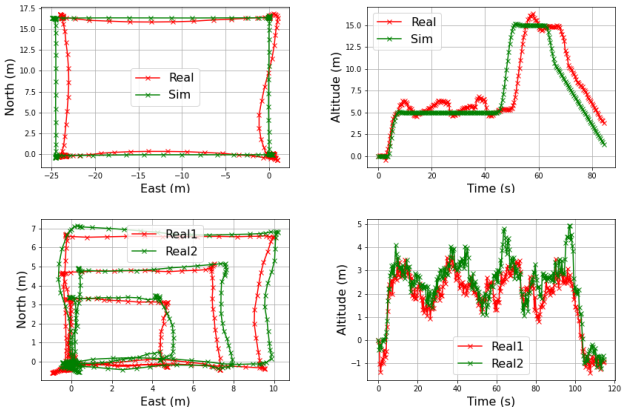


FIGURE 12. Trajectories of (upper) type 1 and (lower) type 2 pairs on the x–y and z–t planes. $\ell = 4$.

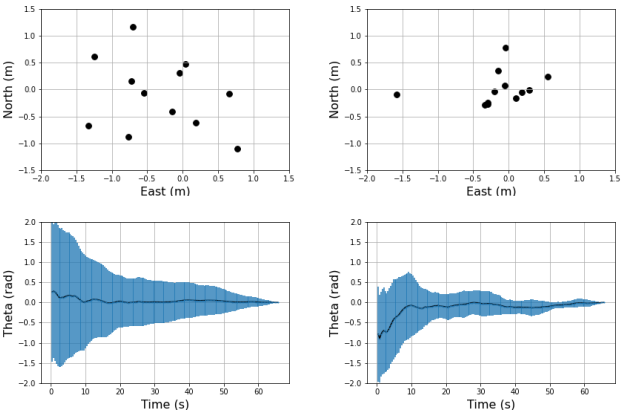


FIGURE 13. Upper: Twelve B_{t_0} points of (left) type 1 and (right) type 2 pairs. Lower: Means (black line) and standard deviations (blue error bars) of $\{\theta_{s,i} - \theta_{t_0,i}\}_{s\in[1,t_0]}$ of (left) 40 type 1 pairs and (right) 12 type 2 pairs. $\ell = 4$ and $t_0 = 165$ (66 s).

Fig. 12 shows typical trajectories for the two types. In Fig. 12(upper), the simulation data (green line) is unique for producing a trajectory that perfectly follows the flight path, showing no deviation or feedback behavior. Therefore, the real and simulation trajectories occasionally diverge significantly from each other. In Fig. 12(lower), each real sample has its unique trajectory, but the distance between the two trajectories is not large.

Algorithm 2 requires initial values of θ and B to properly select the sampling range of p_t^k. This study assumes that no attack occurs for the first t_0 seconds after takeoff, during which the initial values are calculated. Specifically, the initial values θ_{t_0} and B_{t_0} are derived from

$$\begin{cases} \theta_t = \arctan2(\bar{y}_t, \bar{x}_t), \\ B_t = (\bar{x}_t, \bar{y}_t), \end{cases} \qquad (17)$$

and $\bar{x}_t$ ($\bar{y}_t$) denotes the mean of $\{x_s\}_{s\in[1,t]}$ ($\{y_s\}_{s\in[1,t]}$), and x_s (y_s) corresponds to the x-coordinate (y-coordinate) of $p_s^* - p_s^0$.

Fig. 13(upper) shows $B_{t_0} = (\bar{x}_{t_0}, \bar{y}_{t_0})$ on the x–y plane, where $t_0 = 165$ s (66 s). The figure indicates that type 2

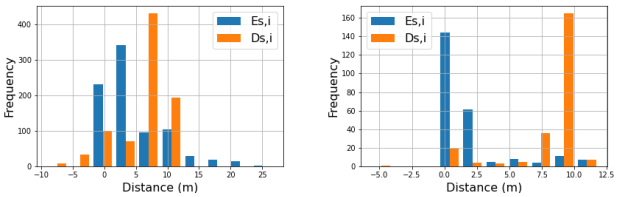


FIGURE 14. Histograms of $\{\mathcal{D}_{s,i}\}$ (orange) and $\{\mathcal{E}_{s,i}\}$ (blue) for (left) type 1 and (right) type 2. (t_0, K) is equal to (left) $(22, 40)$ and (right) $(6, 31)$. $(\ell, g_m, \lambda, \omega) = (30, 10, 7, 0)$ and $v = (1, 1, 0)$.

pairs tend to result in a shorter distance $\|B_{t_0}\|_2$. Fig. 13(lower) shows the means and standard deviations of $\{\theta_{s,i} - \theta_{t_0,i}\}_{s\in[1,t_0]}$ of all flight route i, where $\theta_{s,i}$ is θ_s for the i-th flight route and $t \le t_0 = 165$. Fig. 13(lower) visualizes how quickly $\theta_{t,i}$ approaches $\theta_{t_0,i}$. From the figure, the standard deviation of type 2 pairs decreases more quickly. Accordingly, if t_0 should be significantly small, a type 2 pair is more appropriate.

Fig. 14 shows the histograms of $\{\mathcal{D}_{s,i}\}$ and $\{\mathcal{E}_{s,i}\}$ at $t_0 = 22$ ($t_0 = 6$) for type 1 (type 2) pairs, where $\{\mathcal{D}_{s,i}\}$ and $\{\mathcal{E}_{s,i}\}$ are defined in (13) and (15), respectively. As shown in the figure, type 2 pairs mostly yield smaller $\mathcal{E}_{t,i}$ and larger $\mathcal{D}_{t,i}$ values, with the means E_m and D_m for type 1 (type 2) are 4.85 and 6.07 (2.02 and 7.96), respectively. Thus, type 2 yields better results. However, simulation data is easier to collect. A practical approach would be to initially collect real samples while using type 1 pairs, and then replace type 1 pairs with type 2 pairs immediately sufficient real samples have been gathered.

B. ATTACK DETECTION

The blindfolded flight enables a wide range of applications, one of which is reconnaissance. Assume that drones equipped with Algorithm 2 do not deviate from their originally planned flight routes even when attacked. These drones can create a map documenting any attacks experienced during the flight if they can detect the attacks. This section focuses on GPS spoofing detection.

Assume that Algorithm 2 is used for positioning as well as for attack detection. The experiments are conducted under the following conditions:

1) A drone receives current position p_t^{ex} from a malicious GPS signal, where

$$p_t^{ex} = \begin{cases} p_t^* & \text{if } t \le t_a, \\ p_{t_a}^* + v_a t & \text{if } t > t_a. \end{cases} \tag{18}$$

Thus, the position before (after) time t_a is correct (incorrect), with the incorrect information indicating that the drone deviates from the correct position at a speed of v_a m/s. Algorithm 2 uses p_t^{ex} as the initial position, and each time the algorithm outputs $\bar{p}_t^*$, the following is performed: a spoofing attack is detected if threshold Δ_a (m) satisfies:

$$\|\bar{p}_t^* - p_t^{ex}\|_2 > \Delta_a. \tag{19}$$

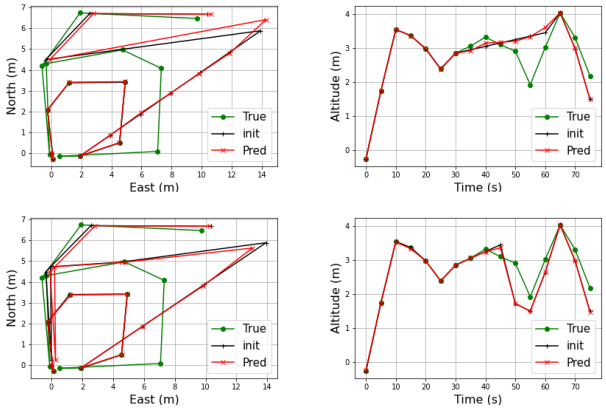


FIGURE 15. True $\{p_s^*\}$ (green), initial $\{p_s^0\}$ (black), and predicted positions $\{\bar{p}_s^*\}$ (red) when (upper) $v_a = v$ and $t_d = 12$ (60 s) and (lower) $v_a = 2v$ and $t_d = 9$ (45 s), where $v = (2, 1, 0.5)$. $t_0 = 4$ (20 s), $t_a = 6$ (30 s), $\Delta_a = 1.5$, and $(K, \ell, g_m, \lambda, \omega) = (31, 50, 10, 0, 0)$.

TABLE 2. Statistics on detection delays (s) derived from twelve samples. $(K, \ell, g_m, \lambda, \omega) = (31, 50, 10, 0, 0)$, $v = (2, 1, 0.5)$, and $(t_0, t_a, \Delta_a) = (4, 6, 0.5)$.

	mean	stdev	CV	max	min
$v_a = v$	29.2	2.9	0.10	30.0	20.0
$v_a = 2v$	15.0	4.3	0.28	25.0	10.0

2) $\{p_s^0\}$ and $\{p_s^*\}$ are type 2 pairs and $t_0 < t_a$.

Let t_d be the time at which the spoofing is detected. The detection process terminates at t_d. Fig. 15 shows the true $\{p_s^*\}$, initial $\{p_s^0\}$, and predicted positions $\{\bar{p}_s^*\}$, where the initial and predicted positions before (after) t_d are the input and output of Algorithm 2 used for detection (positioning), respectively. From Fig. 15(upper), the attack causes the black line to move linearly in the direction of vector v_a and the red line moves along the black line. The two lines gradually move apart, and finally, at time step $t_d = 12$, the condition in (19) holds, where the two lines in Fig. 15(upper left) show pointed shapes.

Because the attack begins at $t_a = 6$, Fig. 15(upper) indicates that the detection delay is $|t_d - t_a| = 6$ (30 s). If v_a doubles, Fig. 15(lower) demonstrates that the attack ends at $t_d = 9$, resulting in a delay of $|t_d - t_a| = 3$ (15 s). Thus, doubling the velocity v_a halves the detection delay $|t_d - t_a|$. Table 2 statistically verifies this relationship and shows small CVs regardless of v_a. These results suggest that the CV, which reflects detection stability, might be more important than the detection delay, which heavily depends on the attack strategy v_a.

Two more insights can be drawn from Fig. 15.

1) Algorithm 2 reacts slowly to the attack. As discussed in Section V, this slow response is owing to the gradual movement of $\bar{p}_t^k$, $k = 1, 2, \ldots, M$. If the initial positions $\{p_s^0\}$ are provided in advance, then the slow movement is improved by obtaining θ_{t_0} and B_{t_0}. However, in the case of an attack, the initial position $\{p_t^{ex}\}$, is not available beforehand.

TABLE 3. Statistics on distances $\|\bar{p}^*_{t,i} - \boldsymbol{p}^0_{t,i}\|_2$ (m) for no attack and type 2 pairs. FPRs are obtained with $\Delta_a = 1.0$. L denotes set $\{(s,i)\}$. $(K, g_m, \lambda, \omega) = (31, 10, 0, 0)$.

ℓ	mean	stdev	max	FPR	$\lvert L\rvert$
100	0.056	0.087	0.493	6.5×10^{-13}	84
50	0.055	0.16	1.796	4.8×10^{-4}	168
30	0.029	0.064	0.655	9.3×10^{-24}	288

2) Spoofing attacks may be detected earlier by analyzing changes in direction and velocity of the predicted drone trajectory $\bar{p}^*_t$. However, slow drift attacks [54], which gradually deviate from the correct route, can bypass such methods. The following results support the idea that as $\|p^*_t - p^{ex}_t\|_2$ increases, (19) eventually detects any type of GPS spoofing:
   - Appendix B shows that (19) holds on average if $\Delta_a \in (0, D_m)$. Furthermore, $\Delta_a \in (0, D_m)$ eventually holds as $\|p^*_t - p^{ex}_t\|_2$ increases because Fig. 8(upper left) shows that D_m increases with λ, where $\lambda = \|p^*_{t,i} - p^0_{t,i}\|_2 / \|v\|_2$ and $\|v\|_2 = \sqrt{2}$.
   - Table 2 demonstrates that detection delays exhibit small CVs.

C. FALSE POSITIVE RATE

A smaller threshold Δ_a results in a shorter detection delay, but increases false positive rate (FPR). However, the false negative rate (FNR) can approach zero if attackers enlarge $\|p^*_t - p^{ex}_t\|_2$ significantly. Let us find appropriate threshold values that achieve short detection delays and small FPRs. Table 3 shows how $\|\bar{p}^*_{t,i} - p^0_{t,i}\|_2$ fluctuates under no attack. The table indicates that the standard deviation reaches its highest value when the step size is $\ell = 50$. This implies that the FPR is highest at $\ell = 50$. Assume that $\|\bar{p}^*_{t,i} - p^0_{t,i}\|_2 \sim \mathcal{MB}$, then the FPR at $\ell = 50$ is 4.8×10^{-4} when $\Delta_a = 1.0$, where $\mathcal{MB}$ is the Maxwell-Boltzmann distribution provided in Appendix C. Therefore, $\Delta_a = 1.0$ seems to be a good threshold. However, because the maximum value at $\ell = 50$ is large (1.796), three values ($\Delta_a = 1.0, 1.5, 2.0$) are selected as appropriate thresholds.

Let $\mathcal{L}_i$ be the detection delay (s) for the i-th flight route. Table 4 presents statistics on $\{\mathcal{L}_i\}$ obtained under the condition that a malicious GPS signal guides the drone to reach destination p^a in 20 s, where $v_a = \frac{1}{4}(p^a - p^*_{t_a})$. From the table, the means of $\{\mathcal{L}_i\}$ at step size $\ell = 50$ or 100, which are between 20 s and 30 s, are smaller than those at $\ell = 30$. Assume that $\mathcal{L}_i$ follows a normal distribution, then the probability that the delay exceeds one minute is less than 10^{-6} (0.023) for all $\Delta_a \in \{1.0, 1.5, 2.0\}$ if $\ell = 50$ or 100 ($\ell = 30$). Thus, a large ℓ should be selected to avoid long detection delays.

A larger ℓ tends to result in shorter mean delays. This seems unrealistic according to conventional wisdom. Additionally, a larger threshold Δ_a tends to yield a smaller CV. Therefore, a larger threshold does not necessarily cause a rapid increase in delay.

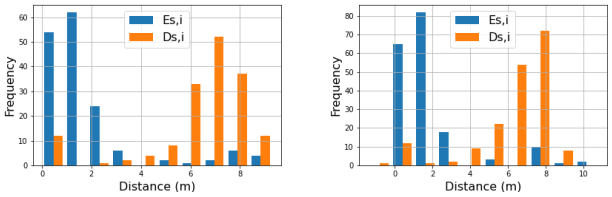


FIGURE 16. Histograms of $\{\mathcal{D}_{s,i}\}$ and $\{\mathcal{E}_{s,i}\}$. Left: $g_m = 0$. Right: $g_m = 10^4$. $(K, \ell, \lambda, \omega) = (31, 50, 6, 0)$, $(t_0, t_a, \Delta_a) = (4, 6, 1.5)$, and $p^a = 20 \times (4, 2, 1)$.

D. LOW-QUALITY INPUTS

One of the most fascinating results obtained through the experiments is the stability of the proposed system for low-quality inputs $\{r_s\}$. Fig. 16 compares two extreme cases: $g_m = 0$ and $g_m = 10^4$. Note from (2) that the noise in $\{r_s\}$ is g_m times greater than signal. As shown in the figure, the histograms for both cases are similar. This may seem unrealistic according to conventional wisdom.

Table 5 represents a statistical comparison of three cases: $g_m = 0$, $g_m = 10^4$, and a 50–50 mixture of $g_m = 0$ and $g_m = 10^4$. The table provides new insights. Over 140 samples of $\{\mathcal{D}_{s,i}\}$ and $\{\mathcal{E}_{s,i}\}$ suggest that the mixture case yields the largest algorithm contribution (the largest mean of $\{\mathcal{D}_{s,i}\}$) and the best prediction (the smallest mean of $\{\mathcal{E}_{s,i}\}$). Moreover, the mixture case demonstrates stability as it provides the smallest CVs for the three criteria $\{\mathcal{D}_{s,i}\}$, $\{\mathcal{E}_{s,i}\}$, and $\{\mathcal{L}_i\}$. Thus, datasets created under various noise intensities can enhance the stability of the proposed system.

VII. COMPARISON

Table 6 shows the root mean square (RMS) and maximum prediction errors reported in the latest INS/GPS studies (RNN [26], LSTM-PI-FGO [30], OSS-Transformer [31], and CNN-LSTM-A [55]) and those of the proposed system, where the INS/GPS studies employ AI models to reduce INS error during GPS outages. From the table, target vehicles and outage times are not homogeneous. Several studies considered multiple travel routes for experiments. All studies obtained the prediction errors for each outage period, except that the authors in [30] reported errors during each test period that has various short outages. Evaluation of our system, denoted as MOR in the table, includes errors after spoofing attack detection. In this experiment, the attacks affect input $\{p_s\}_{1 \le s \le t-1}$ ("positions calculated before time t" in Fig. 2) of the proposed system. However, as shown in the table, the attack has little impact on the performance of the system.

According to [55], existing INS/GPS methods tend to increase prediction errors over time (see Fig. 6 in [55]). Therefore, as shown in Table 6, the outages in most existing results do not exceed 115 s. Whereas, our system is evaluated with flight times of 140 and 360 s. Studies in [31] and [55] dealt with long outages (100–115 s) using land vehicles. Our system is noteworthy in that its distance errors ($\|\cdot\|_2$) are smaller than OSS-Transformer in [31] and its x and y directional

TABLE 4. Statistics on detection delays $\{\mathcal{L}_i\}_{1\le i\le 12}$. $(t_0, t_a) = (4, 6)$, $p^a = 20 \times (4, 2, 1)$, and $(K, g_m, \lambda, \omega) = (31, 10, 0, 0)$.

	$\Delta_a = 1.0$			$\Delta_a = 1.5$			$\Delta_a = 2.0$		
ℓ	mean	CV	min–max	mean	CV	min–max	mean	CV	min–max
100	22.5	0.20	20–30	24.2	0.21	20–30	30.0	0.14	20–40
50	23.8	0.26	10–30	27.5	0.23	15–35	29.2	0.19	20–35
30	38.3	0.35	3–51	52.5	0.05	45–54	54.8	0.05	51–57

TABLE 5. Statistics on $\{\mathcal{D}_{s,i}\}$ (m), $\{\mathcal{E}_{s,i}\}$ (m), and $\{\mathcal{L}_i\}$ (s) for different g_m values. $(t_0, t_a, \Delta_a) = (4, 6, 1.5)$, $p^a = 20 \times (4, 2, 1)$, and $(K, \ell, \lambda, \omega) = (31, 50, 6, 0)$.

	$\{\mathcal{D}_{s,i}\}$		$\{\mathcal{E}_{s,i}\}$		$\{\mathcal{L}_i\}$	
g_m	mean	CV	mean	CV	mean	CV
0	6.43	0.33	1.85	1.08	37.9	0.27
10^4	6.28	0.34	1.87	1.06	29.6	0.28
mix	6.47	0.3	1.73	1.02	35.4	0.09

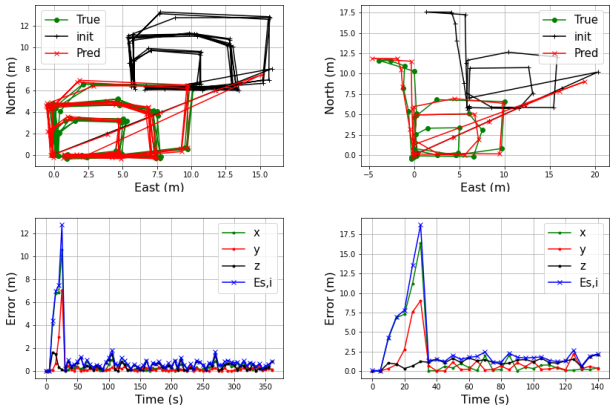


FIGURE 17. Long flight experiments, including an attack. (Upper left): 360 second flight (route 1). (Upper right): Flight with large directional changes (route 2). (Lower left): errors (m) in the x, y, and z directions and $\mathcal{E}_{s,i}$ for route 1. (Lower right): errors for route 2. $(K, \ell, w, g_m, \lambda, \omega) = (31, 50, 30, 10, 6, 0)$, $(t_a, \Delta_a) = (0.5, 0.5)$, and $p^a = 20 \times (4, 2, 1)$.

errors are smaller than CNN-LSTM-A in [55], although our system has longer GPS-independent flight times.

Fig. 17(upper) shows the prediction results on the x–y plane for 360 (route 1) and 140 (route 2) second flights. Fig. 17(lower) denotes time variability of the prediction errors for routes 1 and 2. As shown in Fig. 17(lower), because prediction $\{\bar{p}_s^*\}$ is moving away from true position p_s^* during the spoofing attack, $\mathcal{E}_{s,i}$ rises sharply and then falls as the attack is detected. From Fig. 17(lower), while the complexity of the flight route could affect the mean of the errors, our system has no tendency of error increase over time, even if an attack occurs. This is an essential nature for long-duration blindfolded flights. Note that the errors after attack detection in Table 6 are calculated from errors after the $\mathcal{E}_{s,i}$ fall.

Table 7 shows the percentages of increase or decrease in mean E_m and standard deviation E_{sd} of $\{\mathcal{E}_{s,i}\}$, where the baseline is that ConvNet is used and smoothing parameters satisfy $(\ell, w) = (50, 30)$. From the table, the disuse of ConvNet decreases E_m by 1.7% and increases E_{sd} by 3.3%. Thus, ConvNet helps reduce the variability of $\mathcal{E}_{s,i}$. If the step and window sizes (ℓ, w) decrease to $(10, 5)$, E_{sd} rises largely and E_m remains almost the same, regardless of the presence of ConvNet. Therefore, small ℓ and w should not be selected. Note that reducing ℓ increases the computational complexity, as more predictions $\bar{p}_s^*$ must be computed.

VIII. DISCUSSION

The blindfolded flight does not imply that drones are completely immune to external signal attacks. As discussed in Section II, attacks can be initiated through the exploitation of the physical nature of sensor devices or by breaching the security of companion computers (e.g., Raspberry Pi) attached to the drones. Although attack patterns are diverse, the blindfolded flight is valuable because it considerably narrows down the possible attack patterns that should be considered.

The ability to fly blindfolded can be valuable in emergencies, as a last resort, or under special circumstances such as: (1) Systems requiring a higher level of safety, such as flying taxis. (2) Environments with limited signals, such as outer space. (3) Areas where malicious signals may arrive, including anti-drone guns. (4) Areas with high GPS power consumption, such as mountainous areas.

The blindfolded flight achieved by the proposed system would have a wider range of applications. For example, if trained with a high-quality positioning system, such as real time kinematic (RTK), drones could fly with higher precision than typical GPS flights. When trained, the drone no longer requires RTK, offering an economic advantage. Additionally, the blindfolded flight could be used in the areas where sensor-signal quality is compromised, such as high electromagnetic interference (EMI) sites.

Drones can be used as weapons or for covert surveillance, and the realization of blindfolded flight derived from our system could advance applications in these areas. The blindfolded flight is resistant to anti-drone guns. In addition, high resistance to gyro noise, a side effect of our approach, makes drones more resistant to gyro disruption attacks. Consequently, drones can enter previously inaccessible areas and perform activities, such as voyeurism, espionage, and sabotage. Appropriate safeguards against blindfolded flight abuse should be considered from a variety of perspectives. These include technical as well as legal, ethical, and social perspectives.

IX. CONCLUSIONS

This study proposed a positioning system that predicts the current position using only internal sensor signals (roll, pitch,

TABLE 6. RMS and maximum prediction errors (m) during GPS outages (s) derived from the latest INS/GPS results and our experimental results (MOR). [30] reported errors during the test periods, including numerous brief GPS outages. $\|\cdot\|_2$ denotes a two-dimensional or three-dimensional distance, and x, y, and z represent the errors in the x, y, and z directions, respectively.

		No attack (RMS/max)				After attack detection (RMS/max)				Outage time	Vehicle
		x	y	z	$\|\cdot\|_2$	x	y	z	$\|\cdot\|_2$	(test time)	
MOR	1	0.41/1.01	0.20/0.55	0.57/1.51	0.73/1.84	0.42/1.01	0.20/0.56	0.57/1.50	0.73/1.84	360	
	2	0.76/2.05	0.72/2.19	1.42/2.16	1.77/2.67	0.75/2.05	0.77/2.18	1.38/2.12	1.75/2.67	140	aerial
	3	0.99/1.97	1.10/2.31	0.81/1.60	1.69/2.93	1.02/1.98	1.13/2.31	0.79/1.46	1.71/2.91	140	
	4	1.57/3.34	0.80/2.22	0.65/1.23	1.88/3.44	1.45/3.30	0.86/2.22	0.66/1.19	1.81/3.35	140	
[26], 2020	1	1.86/	2.42/	1.75/	-	-	-	-	-	15	aerial
[30], 2024	1	0.27/1.31	0.33/1.41	-	-	-	-	-	-	0 (1600)	
	2	0.27/0.90	0.37/1.19	-	-	-	-	-	-	4–9 (850)	land
	3	0.79/3.41	0.79/3.89	-	-	-	-	-	-	2–27 (850)	
[31], 2024	1	2.74/5.55	0.79/1.62	-	3.24/6.02	-	-	-	-	40	
	2	2.55/5.19	3.73/8.17	-	5.09/9.81	-	-	-	-	100	land
	3	1.40/2.37	4.23/6.07	-	4.46/6.21	-	-	-	-	40	
	4	4.10/9.20	6.85/11.49	-	8.01/11.52	-	-	-	-	115	
[55], 2025	1	4.14/7.12	3.51/10.49	-	-	-	-	-	-	100	land

TABLE 7. Impacts of ConvNet and smoothing processes on the mean and standard deviation of $\{\mathcal{E}_{s,i}\}$ (m). Their increase (+) or decrease (–) percentages are calculated compared to the baseline setting where ConvNet is used and $(\ell, w) = (50, 30)$. The four flight routes used in the calculation are the same as those used in Table 6.

Smoothing	ConvNet			
(ℓ, w)	use		disuse	
	mean	stdev	mean	stdev
(50,30)	1.42 m	0.65 m	–1.7%	+3.3%
(10,5)	–0.1%	+9.7%	–0.2%	+9.5%

yaw, throttle). The system included Perceiver, a deep learning model, which operated on coarse time scales (1 to 10 s) to reduce the variability of prediction errors on coarse time scales. The system demonstrated high tolerance to low-quality inputs. Even with noise level 10,000 times greater than the signal, the error in the predicted position remained within a few meters. The system showed smaller prediction errors than the latest methods although it experienced longer GPS-independent periods (140 to 360 s). The 360-second flight experiment showed no tendency for the prediction error to increase with time, even when a spoofing attack occurred.

An algorithm was introduced to iteratively execute the trained proposed system, aiming to refine the drone's position accuracy. The number of iterations can be reduced by using either previous flight data or simulation data as initial values. Using previous flight data resulted in better outcomes. In the case of GPS spoofing attacks, the algorithm took longer to detect the attacks, as the initial values for the iterations were unavailable beforehand. However, the coefficient of variations (CVs) in detection delays remained small.

Drones equipped with the proposed system will need to undergo various field tests in the future for stable, long-duration blindfolded flight. The system has potential applications across a variety of mobility and transportation systems. We plan to explore its application to fixed-wing, vertical takeoff and landing (VTOL), and jet airplanes.

APPENDIX A
GLOSSARY

Table 8 lists technical notations and their definitions for clarity and prevention of misunderstanding.

TABLE 8. Technical notations used in this paper and their definitions.

Notation	Definition
Iverson bracket $[P]$	$[P] = 1$ if P is true, $[P] = 0$ if P is not true
Hadamard product $\odot$	$(a_1, a_2) \odot (b_1, b_2) = (a_1 b_1, a_2 b_2)$
Mor signals	Signals from human-operated control units
mor2pos	Trained proposed system in Fig. 2
$v[i]$	$(i+1)$-th element of vector v
$\arctan2(y, x)$	Direction in which (x, y) lies relative to $(0, 0)$
$\mathcal{E}_{t,i}$	Prediction error at time t and route i
$\mathcal{D}_{t,i}$	Error reduction by Algorithm 2

APPENDIX B
(19) AVERAGELY HOLDS

Because $\|\bar{p}^*_{t,i} - p^0_{t,i}\|_2 = \|(\bar{p}^*_{t,i} - p^*_{t,i}) - (p^0_{t,i} - p^*_{t,i})\|_2$, the reverse triangle inequality argues that

$$|\|\bar{p}^*_{t,i} - p^*_{t,i}\|_2 - \|p^0_{t,i} - p^*_{t,i}\|_2| \le \|\bar{p}^*_{t,i} - p^0_{t,i}\|_2. \tag{20}$$

From (13), $\mathcal{D}_{t,i} := \|p^0_{t,i} - p^*_{t,i}\|_2 - \|\bar{p}^*_{t,i} - p^*_{t,i}\|_2$. Therefore,

$$D_m \le \frac{1}{|L|} \sum_{(t,i)\in L} |\mathcal{D}_{t,i}| \le \frac{1}{|L|} \sum_{(t,i)\in L} \|\bar{p}^*_{t,i} - p^0_{t,i}\|_2. \tag{21}$$

By replacing $p^0_{t,i}$ with $p^{ex}_{t,i}$ in (21),

$$\frac{1}{|L|} \sum_{(t,i)\in L} \|\bar{p}^*_{t,i} - p^{ex}_{t,i}\|_2 > \Delta_a \text{ if } \Delta_a \in (0, D_m), \tag{22}$$

where (22) corresponds to the arithmetic mean of (19).

APPENDIX C
FPR DERIVATION

The Maxwell-Boltzmann distribution describes the distribution of the magnitude of velocities $\boldsymbol{v} = (v_x, v_y, v_z)$ of particles

in a gas. The components of the velocity are assumed to be independent and follow a normal distribution $N(0, a^2)$, i.e.,

$$v_x, v_y, v_z \sim N(0, a^2), \tag{23}$$

where the scaling parameter a affects the spread of the distribution. Let $x := \|\boldsymbol{v}\|_2$. The probability density function of the Maxwell-Boltzmann distribution is given by

$$f(x; a) = \sqrt{\frac{2}{\pi}} \frac{x^2}{a^3} \exp\left(-\frac{x^2}{2a^2}\right). \tag{24}$$

This study assumes that $\|\bar{p}^*_{t,i} - p^0_{t,i}\|_2$ has a Maxwell-Boltzmann distribution if $\{p^0_{s,i}\}$ and $\{p^*_{s,i}\}$ are type 2 pairs and no attack occurs. In this case, from (24), the FPR is

$$\mathrm{FPR} = 1 - \int_0^{\Delta_a} f(x; a) dx, \tag{25}$$

where Δ_a is the threshold in (19). The values of the scale parameter a, which can be obtained from either the mean or the standard deviation of $\|\bar{p}^*_{t,i} - p^0_{t,i}\|_2$, were derived from the latter, which always yielded larger a values (i.e., larger FPRs).

ACKNOWLEDGMENT

We would like to express our sincere gratitude to Geng Wang and Akihiro Nonaka for providing us with most of the flight log data used in this paper through numerous field experiments.

REFERENCES

[1] H. Jahani, Y. Khosravi, B. Kargar, K.-L. Ong, and S. Arisian, "Exploring the role of drones and uavs in logistics and supply chain management: a novel text-based literature review," *International Journal of Production Research*, pp. 1–25, 2024.

[2] A. Rejeb, A. Abdollahi, K. Rejeb, and H. Treiblmaier, "Drones in agriculture: A review and bibliometric analysis," *Computers and electronics in agriculture*, vol. 198, p. 107017, 2022.

[3] Y. Chang, Y. Cheng, U. Manzoor, and J. Murray, "A review of uav autonomous navigation in gps-denied environments," *Robotics and Autonomous Systems*, p. 104533, 2023.

[4] J. L. Burbank, L. Foust, T. Greene, and N. Kaabouch, "A proposed framework for uas positioning in gps-denied and gps-spoofed environments," in *2024 Integrated Communications, Navigation and Surveillance Conference (ICNS)*. IEEE, 2024, pp. 1–9.

[5] Y. Xu, G. Deng, X. Han, G. Li, H. Qiu, and T. Zhang, "Physcout: Detecting sensor spoofing attacks via spatio-temporal consistency," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, 2024, pp. 1879–1893.

[6] X. Chen, T. Li, H. Liu, Q. Huang, and X. Gan, "A gps spoofing detection algorithm for uavs based on trust evaluation," in *2023 IEEE 13th International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*. IEEE, 2023, pp. 315–319.

[7] J. Simon, T. Rodriguez, A. Scorzolini, P. d. Silva, F. Sbardellati, I. Fernandez-Hernandez, S. Damy, and D. Ibanez, "The galileo open service navigation message authentication (osnma): The pioneer data authentication service," in *Proceedings of the 37th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2024)*, 2024, pp. 973–991.

[8] H. Sathaye, M. Strohmeier, V. Lenders, and A. Ranganathan, "An experimental study of {GPS} spoofing and takeover attacks on {UAVs}," in *31st USENIX security symposium (USENIX security 22)*, 2022, pp. 3503–3520.

[9] A. Altaweel, H. Mukkath, and I. Kamel, "Gps spoofing attacks in fanets: A systematic literature review," *IEEE Access*, vol. 11, pp. 55 233–55 280, 2023.

[10] K. M. A. Alheeti, A. Alzahrani, and D. Al Dosary, "Lidar spoofing attack detection in autonomous vehicles," in *2022 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE, 2022, pp. 1–2.

[11] T. Sato, Y. Hayakawa, R. Suzuki, Y. Shiiki, K. Yoshioka, and Q. A. Chen, "Lidar spoofing meets the new-gen: Capability improvements, broken assumptions, and new attack strategies," *arXiv preprint arXiv:2303.10555*, 2023.

[12] Z. Wu, S.-N. Lim, L. S. Davis, and T. Goldstein, "Making an invisibility cloak: Real world adversarial attacks on object detectors," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*. Springer, 2020, pp. 1–17.

[13] W. Wang, Y. Yao, X. Liu, X. Li, P. Hao, and T. Zhu, "I can see the light: Attacks on autonomous vehicles using invisible lights," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 1930–1944.

[14] Y. Tu, Z. Lin, I. Lee, and X. Hei, "Injected and delivered: Fabricating implicit control over actuation systems by spoofing inertial sensors," in *27th USENIX security symposium (USENIX Security 18)*, 2018, pp. 1545–1562.

[15] A. Khan, N. Ivaki, and H. Madeira, "A comprehensive study on drones resilience in the presence of inertial measurement unit faults," in *2024 54th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2024, pp. 316–323.

[16] W. Xu, C. Yan, W. Jia, X. Ji, and J. Liu, "Analyzing and enhancing the security of ultrasonic sensors for autonomous vehicles," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 5015–5029, 2018.

[17] T. Gluck, M. Kravchik, S. Chocron, Y. Elovici, and A. Shabtai, "Spoofing attack on ultrasonic distance sensors using a continuous signal," *Sensors*, vol. 20, no. 21, p. 6157, 2020.

[18] R. Komissarov and A. Wool, "Spoofing attacks against vehicular fmcw radar," in *Proceedings of the 5th Workshop on Attacks and Solutions in Hardware Security*, 2021, pp. 91–97.

[19] S. Nashimoto, D. Suzuki, N. Miura, T. Machida, K. Matsuda, and M. Nagata, "Low-cost distance-spoofing attack on fmcw radar and its feasibility study on countermeasure," *Journal of Cryptographic Engineering*, pp. 1–10, 2021.

[20] Y. Xu, X. Han, G. Deng, J. Li, Y. Liu, and T. Zhang, "Sok: Rethinking sensor spoofing attacks against robotic vehicles from a systematic view," in *2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2023, pp. 1082–1100.

[21] D. J. Yeong, G. Velasco-Hernandez, J. Barry, and J. Walsh, "Sensor and sensor fusion technology in autonomous vehicles: A review," *Sensors*, vol. 21, no. 6, p. 2140, 2021.

[22] A. Singh, "Transformer-based sensor fusion for autonomous driving: A survey," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 3312–3317.

[23] D. Yang, H. Liu, X. Jin, J. Chen, C. Wang, X. Ding, and K. Xu, "Enhancing vio robustness under sudden lighting variation: A learning-based imu dead-reckoning for uav localization," *IEEE Robotics and Automation Letters*, 2024.

[24] H. V. Do, Y. H. Kim, J. H. Lee, M. H. Lee, and J. W. Song, "Dero: Dead reckoning based on radar odometry with accelerometers aided for robot localization," *arXiv preprint arXiv:2403.05136*, 2024.

[25] M. Brossard, A. Barrau, and S. Bonnabel, "Ai-imu dead-reckoning," *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 4, pp. 585–595, 2020.

[26] H.-f. Dai, H.-w. Bian, R.-y. Wang, and H. Ma, "An ins/gnss integrated navigation in gnss denied environment using recurrent neural network," *Defence technology*, vol. 16, no. 2, pp. 334–340, 2020.

[27] E. Oh, M. D. Gregoire, A. T. Black, K. Jeramy Hughes, P. D. Kunz, M. Larsen, J. Lautier-Gaud, J. Lee, P. D. Schwindt, S. L. Mouradian *et al.*, "Perspective on quantum sensors from basic research to commercial applications," *AIAA Journal*, vol. 62, no. 11, pp. 4029–4053, 2024.

[28] A. A. Taranta and A. Taranta, "Novel fibres for next-generation fibre-optic gyroscopes," Ph.D. dissertation, University of Southampton, 2024.

[29] W. Zou, Y. Huang, H. Lin, and Z. Xue, "New application and research of ring laser gyroscope in the field of angle metrology," *IEEE Transactions on Instrumentation and Measurement*, 2024.

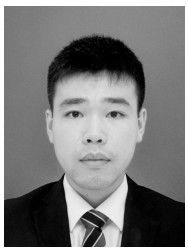
[30] F. Liu, H. Zhao, and W. Chen, "A hybrid algorithm of lstm and factor graph for improving combined gnss/ins positioning accuracy during gnss interruptions," *Sensors*, vol. 24, no. 17, p. 5605, 2024.

[31] H. Wang, F. Tang, J. Wei, B. Zhu, Y. Wang, and K. Zhang, "Online semi-supervised transformer for resilient vehicle gnss/ins navigation," *IEEE Transactions on Vehicular Technology*, 2024.

[32] A. Al-Radaideh and L. Sun, "Self-localization of a tethered quadcopter using inertial sensors in a gps-denied environment," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2017, pp. 271–277.

[33] ——, "Self-localization of tethered drones without a cable force sensor in gps-denied environments," *Drones*, vol. 5, no. 4, p. 135, 2021.
[34] A. Al-Radaideh, R. A. Selje, D. Coraspe, E. Camci, R. Dutta, L. Sun, S. Jayavelu, and X. Li, "Tethered multicopter guidance in gps-denied environments through reinforcement learning," in *AIAA SCITECH 2023 Forum*, 2023, p. 0507.
[35] G. Wang and K. Oida, "Tracking drones with manual operation representation," in *Proceedings of the 2023 11th International Conference on Information Technology: IoT and Smart City*, 2023, pp. 167–172.
[36] Z. M. Kassas, N. Khairallah, J. J. Khalife, C. Lee, J. Jurado, S. Wachtel, J. Duede, Z. Hoeffner, T. Hulsey, R. Quirarte *et al.*, "Aircraft navigation in gnss-denied environments via radio slam with terrestrial signals of opportunity," *IEEE Transactions on Intelligent Transportation Systems*, 2024.
[37] Z. Kassas, J. Khalife, A. Abdallah, N. Khairallah, S. Shahcheraghi, C. Lee, J. Jurado, S. Wachtel, J. Duede, Z. Hoeffner *et al.*, "Protecting the skies: Gnss-less aircraft navigation with cellular signals of opportunity," 2024.
[38] J. Zhou, W. Wang, X. Hong, and C. Zhang, "Multi-uav cooperative anti-jamming for gnss signals based on frequency-domain power inversion," *IEEE Sensors Journal*, 2024.
[39] F. She, Y. Zhang, D. Shi, H. Zhou, X. Ren, and T. Xu, "Enhanced relative localization based on persistent excitation for multi-uavs in gps-denied environments," *IEEE Access*, vol. 8, pp. 148 136–148 148, 2020.
[40] M. Bodi, L. Zhenbao, F. Jiang, Z. Wen, D. Qingqing, W. Xiao, J. Zhang, and W. Lina, "Reinforcement learning based uav formation control in gps-denied environment," *Chinese Journal of Aeronautics*, vol. 36, no. 11, pp. 281–296, 2023.
[41] Y. Son, H. Shin, D. Kim, Y. Park, J. Noh, K. Choi, J. Choi, and Y. Kim, "Rocking drones with intentional sound noise on gyroscopic sensors," in *24th USENIX security symposium (USENIX Security 15)*, 2015, pp. 881–896.
[42] T. Trippel, O. Weisse, W. Xu, P. Honeyman, and K. Fu, "Walnut: Waging doubt on the integrity of mems accelerometers with acoustic injection attacks," in *2017 IEEE European symposium on security and privacy (EuroS&P)*. IEEE, 2017, pp. 3–18.
[43] A. Erba, J. H. Castellanos, S. Sihag, S. Zonouz, and N. O. Tippenhauer, "Sensor deprivation attacks for stealthy uav manipulation," *arXiv preprint arXiv:2410.11131*, 2024.
[44] "Ardupilot versatile, trusted, open," https://ardupilot.org/, online; accessed 7 August 2023.
[45] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
[46] B. Lim and S. Zohren, "Time-series forecasting with deep learning: a survey," *Philosophical Transactions of the Royal Society A*, vol. 379, no. 2194, p. 20200209, 2021.
[47] Q. Wen, T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, and L. Sun, "Transformers in time series: A survey," *arXiv preprint arXiv:2202.07125*, 2022.
[48] A. Jaegle, F. Gimeno, A. Brock, O. Vinyals, A. Zisserman, and J. Carreira, "Perceiver: General perception with iterative attention," in *International conference on machine learning*. PMLR, 2021, pp. 4651–4664.
[49] A. Jaegle, S. Borgeaud, J.-B. Alayrac, C. Doersch, C. Ionescu, D. Ding, S. Koppula, D. Zoran, A. Brock, E. Shelhamer *et al.*, "Perceiver io: A general architecture for structured inputs & outputs," *arXiv preprint arXiv:2107.14795*, 2021.
[50] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
[51] "The ai community building the future." https://huggingface.co/, online; accessed 14 October 2024.
[52] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "Pixhawk: A system for autonomous flight using onboard computer vision," in *2011 ieee international conference on robotics and automation*. IEEE, 2011, pp. 2992–2997.
[53] A. D. Team, "Sitl simulator (software in the loop)," 2016.
[54] S. Dasgupta, A. Ahmed, M. Rahman, and T. N. Bandi, "Unveiling the stealthy threat: Analyzing slow drift gps spoofing attacks for autonomous vehicles in urban environments and enabling the resilience," *arXiv preprint arXiv:2401.01394*, 2024.
[55] L. Zhong and X. Li, "Ins/gnss integrated navigation algorithm based on cnn-lstm-attention," in *Journal of Physics: Conference Series*, vol. 2990, no. 1. IOP Publishing, 2025, p. 012018.

KAZUMASA OIDA received Bachelor of Information Science and Doctor of Informatics degrees from the University of Tsukuba in 1983 and Kyoto University in 2002, respectively. He engaged in the development of private network systems at NTT. He is currently a professor at Fukuoka Institute of Technology. His research interests include cybersecurity, blockchain applications, secure UAV systems, and complex networks. He is currently working with the Fukuoka Prefectural Police to analyze smishing malware and to trace malicious traffic in the Tor network. He plans to launch a drone security company with the support of the Japan Science and Technology Agency (JST).



TOMOYA KOREZAWA is currently an undergraduate student at the Fukuoka Institute of Technology. After graduating in March 2025, he plans to work in software system development at an IT company. His current research theme is the development of drone systems, with a particular focus on GPS and EKF. He plans to be involved in IT system development, making full use of his current experience.

TAIKI YAMADA is currently an undergraduate student at the Fukuoka Institute of Technology. After graduating in March 2025, he will continue his research in security at the university's graduate school. His research themes are cybersecurity, malware analysis, and phishing/smishing detection. In the future, he aims to become a white hat hacker in these fields.

SOJIRO ETO is an undergraduate student in the Department of Computer Science and Engineering, Fukuoka Institute of Technology. His research theme is the development of safe UAV systems. He is currently working on tracking drones from the viewpoint of the manual operation representation. He is interested in the fields of autonomous systems and AI robotics, and plans to work as an engineer in these fields in the future.