Special Collection Article

International Journal of Distributed Sensor Networks

International Journal of Distributed Sensor Networks
2018, Vol. 14(6)

DOI: 10.1177/1550147718781750
journals.sagepub.com/home/dsn
SAGE

A low-cost solution for unmanned aerial vehicle navigation in a global positioning system–denied environment

Shahrukh Ashraf[1], Priyanka Aggarwal[2], Praveen Damacharla[3], Hong Wang[4], Ahmad Y Javaid[3] and Vijay Devabhaktuni[3]

Abstract

The ability of an autonomous unmanned aerial vehicle to navigate and fly precisely determines its utility and performance. The current navigation systems are highly dependent on the global positioning system and are prone to error because of global positioning system signal outages. However, advancements in onboard processing have enabled inertial navigation algorithms to perform well during short global positioning system outages. In this article, we propose an intelligent optical flow–based algorithm combined with Kalman filters to provide the navigation capability during global positioning system outages and global positioning system–denied environments. Traditional optical flow measurement uses block matching for motion vector calculation that makes the measurement task computationally expensive and slow. We propose the application of an artificial bee colony–based block matching technique for faster optical flow measurements. To effectively fuse optical flow data with inertial sensors output, we employ a modified form of extended Kalman filter. The modifications make the filter less noisy by utilizing the redundancy of sensors. We have achieved an accuracy of ~95% for all non-global positioning system navigation during our simulation studies. Our real-world experiments are in agreement with the simulation studies when effects of wind are taken into consideration.

Keywords

Inertial navigation, optical flow, multi-sensor data fusion, block matching

Date received: 10 July 2017; accepted: 16 May 2018

Handling Editor: Lyudmila Mihaylova

Introduction

The utility of an autonomous unmanned aerial vehicle (UAV) depends on its ability to navigate with an acceptable positioning error. The term "autonomous UAV" may have different meanings in different contexts; here, it implies a UAV capable of flying without any external guidance. The sensing and processing are performed using onboard sensors and processors. The biggest challenge is to make UAVs robust enough to autonomously navigate under global positioning system (GPS) outages or loss of signal connections and to fully realize the practical applications, such as robust and

[1]Cypress Semiconductors Corp., Livonia, MI, USA
[2]Mantra Scientific LLC, Virginia Beach, VA, USA
[3]Electrical Engineering and Computer Science Department, The University of Toledo, Toledo, OH, USA
[4]Engineering Technology Department, The University of Toledo, Toledo, OH, USA

Corresponding author:
Ahmad Y Javaid, Electrical Engineering and Computer Science Department, The University of Toledo, MS 308, 2801 W. Bancroft Street, Toledo, OH 43606, USA.
Email: Ahmad.Javaid@utoledo.edu

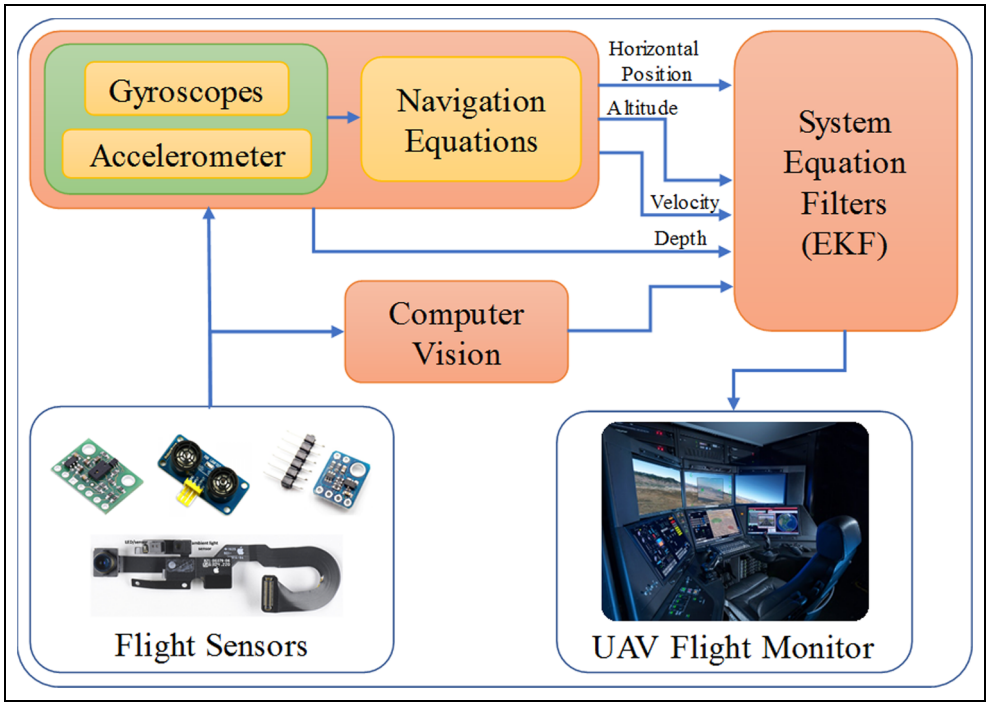

Figure 1. Outline of an autonomous UAV operation.

automated package delivery.[1] In the present scenario, UAVs use GPS or inertial navigation system (INS) for navigation and can fly safely during very short GPS outages (Figure 1). For example, an integrated system with an attitude outage of roughly 2 min is still considered safe, while all GPS outages of only 30 s can lead to roughly a 10 m error in location and approximately 1 m/s error in velocity.[2]

Many approaches have been proposed in the literature to address the problem of navigation during GPS outages. Some researchers have proposed the use of cameras to observe the surroundings and navigate, while others have used ultrasonic or laser-ranging sensors to map the surroundings, thereby localizing a UAV.[3] Although these techniques are effective for indoor navigation, they encounter limitations when it comes to outdoor navigation. In an outdoor environment, UAV range sensors might not find anything close enough to take a reference, making their mapping difficult with respect to the surroundings. Other drawbacks include power requirements, size, weight, and, most importantly, cost of the equipment involved. Optical flow and other visual motion detection algorithms have been used in the past and have shown to improve the performance of a UAV. However, published results suggest that researchers achieve an enhanced navigation precision with the fusion of data from inertial sensors, visual motion detection data, and ranging sensors.[4,5] In this article, we propose a modified extended Kalman filter (M-EKF) that allows fusing the most accurate data from sensors such as optical flow sensor (camera), accelerometer, gyroscope, and a barometric altitude sensor to estimate UAV position in GPS-denied areas. Over the past few years, the UAVs have been equipped with optical mouse sensors such as ADSN2620 for optical flow measurements.[6] Optical flow sensors can measure the velocity of the UAV by looking down at the ground and estimating position. However, these types of sensors do not work well in low-light conditions because of their poor light sensitivity,[7] thereby making measurements inaccurate. A complementary metal oxide semiconductor (CMOS) sensor-based camera can remove this drawback; PX4FLOW is the optical flow sensor developed in 2013 and is specifically designed for use in UAVs.[8]

Using the abovementioned optical sensor, this article presents the implementation of an artificial bee colony (ABC)-based block-matching algorithm that processes images and updates the flow measurement fast enough for real-time applications. We propose to use a popular optimization algorithm to efficiently accomplish block matching (BM) in optical flow sensor output images to accurately calculate the speed and direction of the motion. We use these values to predict correct GPS coordinates using the last known location before the outage. Our intelligent optical flow measurement algorithm also reduces the computational load and allows low overhead during onboard processing. The mathematical derivations for extended Kalman filter (EKF) are well defined in the study by Reif et al.,[9] and they

are used as guidelines in this article to create a modified EKF. Proposed modified EKF is used to merge the data from various sensors to navigate the UAV. We implemented the intelligent optical flow algorithm, modified EKF on a UAV, and used autonomous test flights to record experimental results. These results were logged and plotted for better visualization of the achieved performance. In addition, the results were obtained in an outdoor scenario in the presence of other environmental parameters such as wind, fluctuating lighting conditions, and changing ground patterns, making the tests more realistic. The primary contributions of this article are as follows:

1. Proposed and developed an ABC-based BM algorithm for UAV applications.
2. Designed and implemented a novel modified EKF algorithm for multi-sensor data fusion for UAV navigation.

The article is organized as follows. The next section discusses the previous work in this field of research and presents a brief comparison of related techniques. The next section, "Technical Methodology," details the proposed data fusion algorithm, optical flow algorithm, and sensors used. Moving on, experimental results and performance comparison with GPS-based navigation have been presented in section "Results and discussion." Finally, we conclude the article in section "Conclusion and future work."

Related work

Optical flow is a natural method of navigation and obstacle avoidance for many insects and birds. This is accomplished by analyzing the relative motion between the navigator and the surroundings or the ground. It can be considered as the two-dimensional (2D) representation of the three-dimensional (3D) motion of the object and has proven its application in motion estimation, video compression, and other image-processing applications.[10] The optical flow measurement techniques can be classified into four categories with respect to the image features being used to measure the flow, thereby determining the motion. These categories include the following:[11]

1. A differential technique that uses derivatives to calculate the first constraint and then solve for the orthogonal component using global methods;
2. A correlation-based technique that uses assumptions such as displaced values generally does not change between frames; between the neighborhood of the pixel in the first image and the second image;
3. A feature-based technique, for example, an algorithm for real-time detection and tracking of moving targets in terrestrial scenes using a mobile camera;
4. Hierarchal techniques, presenting a framework of combining two prevailing analysis approaches optical flow and global motion analysis.

The choice of a technique from these categories depends on the application, computational complexity, and accuracy requirements. Optical flow methodologies have achieved significant pixel level accuracy on benchmark data sets used by the image-processing community. For example, Microsoft robotics researchers have started to employ these techniques for navigation and obstacle avoidance on the ground and in aerial robots.[12] In the study by Kim et al.,[13] researchers use vision navigation through image processing and target tracking. Optical flow measurement coupled with inertial measurement unit or other similar sensors have also proved capable of completing many navigational subtasks. In the study by Floris et al.,[14] the authors employed optical flow for distance estimation. In the studies by Lange et al.[15] and Herisse et al.,[16] it was used for position hold, obstacle avoidance, and vertical landing. CentEye Inc. developed optical flow chips and has achieved position estimation as well as obstacle avoidance.[17] However, optical flow algorithms require very high computational resources because of their algorithmic complexity, making it difficult for onboard computers and real-time implementation.[18]

One of the most important steps to make the optical flow data useful for a UAV is to integrate these measurements with translational and rotational motion (roll, pitch, and yaw). In addition, a rigorous analysis of flow is required before it can be used reliably for navigation. Once the calibration is complete, optical flow measurements can be used to aid in velocity, position, and orientation estimation. Another popular technique among researchers currently is vision-based simultaneous localization and mapping (SLAM).[19–21] In the study by Robert et al.,[22] researchers proposed a new architecture that constructs SLAM maps based on visual data and EKF to navigate in an indoor environment. However, the light detection and ranging (LIDAR) used in the regular SLAM algorithm is costly as well as computationally expensive. In the study by Green and Oh,[23] the authors propose a collision avoidance technique using optical flow data. In another work, optical flow was used as a backup for velocity estimation, and the velocity errors were in meter range, during the outages.[24] In the study by Mercado et al.,[25] authors fused data from GPS, INS, and optical flow for position and velocity estimation. However, after 10 s of GPS dropout, the velocity in the north direction

diverged. In the study by Rhudy et al.,[26] the wide field optical flow measurement was employed to reduce the drift error in INS, for navigation during GPS outages. Gageik et al.[27] applied the optical flow sensor for position hold and landing application. In this article, we implemented ABC-based BM to implement optical flow and then integrated the sensor data using the EKF dynamic prediction model.

Much research has been undertaken in the area of multi-sensor data fusion for UAV-related applications. Multi-sensor data fusion deals with the efficient algorithms to transform the uncertain, noisy, and sometimes conflicting data from different sensors into a single representation that can help in the decision-making process.[28] The reason for the widespread use of multi-sensor data fusion in an autonomous system is that it provides the improved reliability and scalability of data.[29,30] For example, data fusion of multiple sensors can help keep error under control by exploiting the redundant information from various noisy and imperfect sensors. This also makes the fusion of data reliable, especially in real-time situations.[28] In our case, this issue can be extremely important and sensitive during the autonomous flight of a UAV. Among the available methods for data fusion, the Kalman filter–based approaches have been widely accepted because of their simple implementation and adaptability to the system requirement.[31,32] EKF linearizes the nonlinear system around the conditional mean using Taylor series expansion and neglects the higher-order terms.[33,34] It may be unsuitable for highly nonlinear systems but works well for mildly nonlinear systems.[35] Some of GPS/INS algorithms also use EKF and unscented Kalman filter (UKF) for attitude and position estimation.[36,37]

In addition to the abovementioned works, we present a comparison of methods for ABC-based BM, EKF, and navigation using optical flow in contemporary research works in Table 1. Numerous research studies published from 2013 to 2018 underscore the superiority of ABC compared with other popular algorithms. These works conclude that ABC algorithm can converge faster than particle swarm optimization (PSO), genetic algorithm (GA), cuckoo-search, and differential evolution.[38–40] Moreover, another work

Table 1. Summary of comparisons of methods used in this research paper.

	Method/study name, year	Application studied	Comparison	Conclusion(s)
ABC-BM	Artificial Bee Colony (ABC), 2014[40]	Minimize the job scheduling process span	Genetic algorithm (GA)	ABC is competent to achieve higher accuracy and converges faster than GA
	Binary ABC (BABC), 2017[38]	Structure selection of a nonlinear auto-regressive model	Binary particles swarm optimization (BPSO)	BABC (1) significantly outperformed, (2) finds lowest possible fitness value, and (3) converged consistently faster than BPSO
	Modified ABC (MABCA), 2017[41]	Block matching (BM)	Full search methods	Among five video sequences, ABC performed better in four sequences than other Full search methods
EKF	EKF, 2018[42]	Ultrasonic positioning system	Maximum correntropy criterion EKF, weighted centroid	Maximum correntropy criterion EKF weighted centroid algorithm improves the positioning accuracy by 60.06% over the traditional EKF
	Kalman filter, 2016[43]	Accelerometer-based attitude determination	Fast complementary filter (FCF)	FCF performs faster calculation, but EKF gives a higher accuracy
Optical navigation	Visual Odometry, 2016[44]	Estimating and maintaining the UAV position	Simultaneous localization and mapping (SLAM)	Visual odometry–based approaches were faster, and memory and processing efficient than SLAM
	Survey of Optical Flow, 2014[12]	Robotic navigation	Lucas-Kanade, Horn-Schenck, and BM methods	Despite several accomplishments, most research is either limited to structured indoor environments or constrained by computational powers

EKF: extended Kalman filter; UAV: unmanned aerial vehicle.

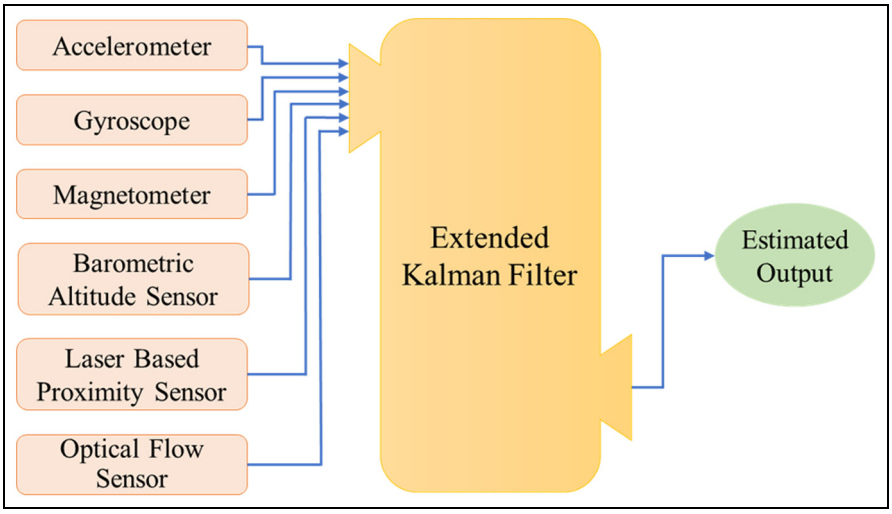

Figure 2. Centralized architecture for extended Kalman filter.

comparison to other full search (FS) methods, modified ABC algorithm (MABCA), outperformed.[40,41] The authors compared MABCA to six FS methods including FS, three-step search, new three-step search, simple and efficient search, four-step search, and diamond search. To compare these algorithms, mean square error (MSE) was used to check the BM, and peak signal-to-noise ratio (PSNR) was used to check the rebuilt image quality. The experiments were conducted with classic test video sequences that show MABCA performed 35.4 times better than any other search methods.[41]

As discussed in the previous sections, EKF is used as a state estimation method. Other state estimation methods presented in the literature include artificial neural networks (ANN), alpha-beta filter, set-membership filter, and sigma point filter. However, results from various works consistently proved that different versions of EKF performed well in state estimation or position estimation.[43,45,46] In addition, Balamurugan et al.[44] presented a survey on various methods used for UAV navigation in the GPS-denied environment, which presented a captivating comparative analysis showing the prominence of EKF-based visual navigation. This reaffirms our use of EKF and its modification that, as proved by Balamurugan et al.,[44] enhances the performance of traditional EKF. These studies are consistent with industrial applications such as Microsoft and Tesla future vision on navigation.[12,32] This analysis by various researchers compelled us to test optical flow, EKF, and ABC-BM methods for navigation and position estimation of UAVs. According to application needs, we proposed a modified EKF that can empower the ABC-based BM and can achieve navigation goals for extended GPS outages.

Technical methodology

Multi-sensor data fusion

Our goal is to efficiently combine and process the data from multiple sensors for successfully navigating a UAV, even under multiple sensors fault or missing/noisy data while GPS outages occur. In general, the EKF has two possible architectures for implementation: (1) centralized and (2) decentralized. As the name suggests, in a centralized architecture, data from different sensors fuse together at one single point, as shown in Figure 2, while in decentralized architecture, every sensor has its own EKF filter working for its estimated output, as shown in Figure 3. The main EKF in decentralized architecture considers the outputs of each parallel EKF filter as the local state estimation and then uses them to estimate the global state or solution. In this article, we selected the decentralized architecture over the centralized architecture to achieve better results based on previously discussed published literature on the effectiveness of modified EKFs.[30] Here, we first present a quick overview of the EKF followed by proposed modifications.

EKF. In most of the real-world applications, systems are described by nonlinear equations.[47] A simple model is represented by equations (1) and (2)

$$\boldsymbol{x_k} = \boldsymbol{f}(\boldsymbol{x_{k-1}}) + \boldsymbol{w_{k-1}} \quad (1)$$

$$\boldsymbol{z_k} = \boldsymbol{h}(\boldsymbol{x_k}) + \boldsymbol{v_k} \quad (2)$$

where

$$x_k\ n \times 1 - \textit{State vector}$$

$$w_k\ n \times 1 - \textit{Process noise vector}$$

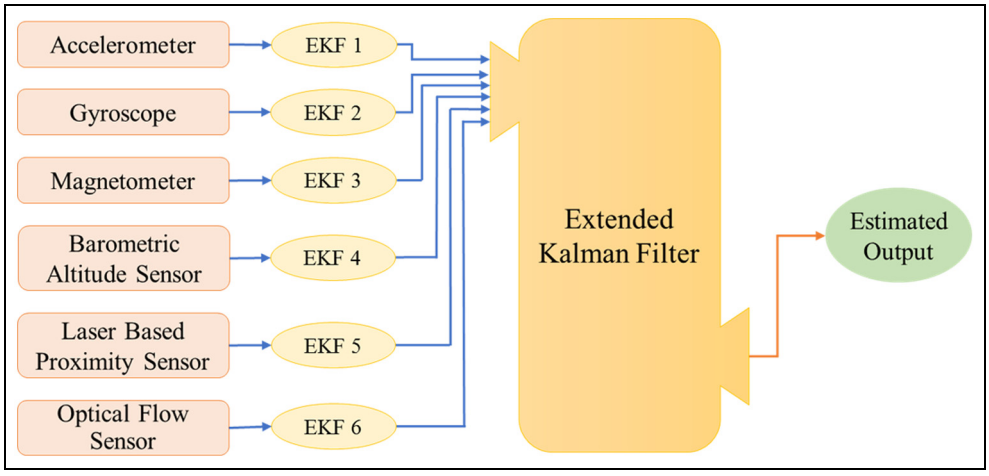

Figure 3. Decentralized architecture for extended Kalman filter.

$$z_k\ m \times 1 - \textit{Observation vector}$$

$$v_k\ m \times 1 - \textit{Measurement noise vector}$$

$$f(\cdot)\ n \times 1 - \textit{Process/system nonlinear vector function}$$

$$h(\cdot)\ m \times 1 - \textit{Observation/measurment nonlinear vector function}$$

$$Q_k\ n \times n - \textit{Process noise covariance matrix}$$

$$R_k\ m \times m - \textit{Measurement noise covariance matrix}$$

In short, here x_k is the state-vector of interest, z_k are the measured variables obtained from the N installed sensors, w_k. v_k indicates the stochastic process and measurement disturbances modeled by zero-mean white Gaussian noises with covariance matrices Q_k and R_k, respectively. The algorithm will result in an optimal/updated estimate of the state vector x_k^a at the time step k based on the previously estimated state x_{k-1}^a and sensor data z_k. The four main steps of the algorithm are initialization, prediction, measurement, and estimation.

Initialization. $\boldsymbol{x_0^a} = \boldsymbol{\mu}_0$, with error covariance $\boldsymbol{P}_0$

Prediction. We get the new prediction of the state x_k^f and covariance P_k^f at time step k. Using this prediction, we compute the predicted measurements $z_k^f = h(x_k^f)$

$$\boldsymbol{x_k^f} = \mathbf{f}\left(\boldsymbol{x_{k-1}^a}\right) \tag{3}$$

$$\boldsymbol{P_k^f} = \mathbf{J}_f\left(\boldsymbol{x_{k-1}^a}\right)\boldsymbol{P}_{k-1}\mathbf{J}_f^T\left(\boldsymbol{x_{k-1}^a}\right) + \boldsymbol{Q}_{k-1} \tag{4}$$

Measurement. In this step, we acquire the actual sensor data z_k and compare it with the predicted z_k^f obtained from the predicted state vector. The difference between them is called the residual or innovation (r)

$$\boldsymbol{r_k} = \boldsymbol{z_k} - \boldsymbol{z_k^f} \tag{5}$$

Update step. Next, using the above information, we update our state vector and the covariance matrix

$$\boldsymbol{x_k^a} \approx \boldsymbol{x_k^f} + \boldsymbol{K_k}\left(\boldsymbol{z_k} - \boldsymbol{h}\left(\boldsymbol{x_k^f}\right)\right) \tag{6}$$

$$\boldsymbol{P_k} = (\boldsymbol{I} - \boldsymbol{K_k}\boldsymbol{H})\boldsymbol{P_k^f} \tag{7}$$

$$\boldsymbol{K_k} = \boldsymbol{P_k^f}\boldsymbol{H^T}\left(\boldsymbol{P_k^f}\boldsymbol{H^T} + \boldsymbol{R_k}\right)^{-1} \tag{8}$$

where the initial state x_0 is a random vector with known mean $\mu_0 = E[x_0]$ and covariance

$$\boldsymbol{P}_0 = \boldsymbol{E}\left[(\boldsymbol{x}_0 - \boldsymbol{\mu}_0)(\boldsymbol{x}_0 - \boldsymbol{\mu}_0)^{\boldsymbol{T}}\right] \tag{9}$$

where $\boldsymbol{J_f}(.)$ is the Jacobian and $\boldsymbol{K_k}$ is the Kalman gain.

While we assume that (1) the noise in individual sensors are not correlated to each other ($E[w_k v_j^T] = 0$ for all k *and* j) and to the initial state x_0, and (2) the nonlinearities in the dynamic system and observation model are smooth, we can expand $f(x_k)$ and $h(x_k)$ using Taylor's series and approximate the estimate of x_k. For additional details on the derivation of prediction and update equations, the reader is requested to please refer to the literature.[48,49] In our approach, we used the modified EKF for error correction in the optical flow sensor output and combined it with other data, such as that from an accelerometer, to accurately predict the current location. Specifically, for optical flow measurements, we have used the ABC optimization-based fast BM to calculate direction and speed of motion. These values combined with sensor inputs result in an estimate of highly accurate location value during GPS outage.

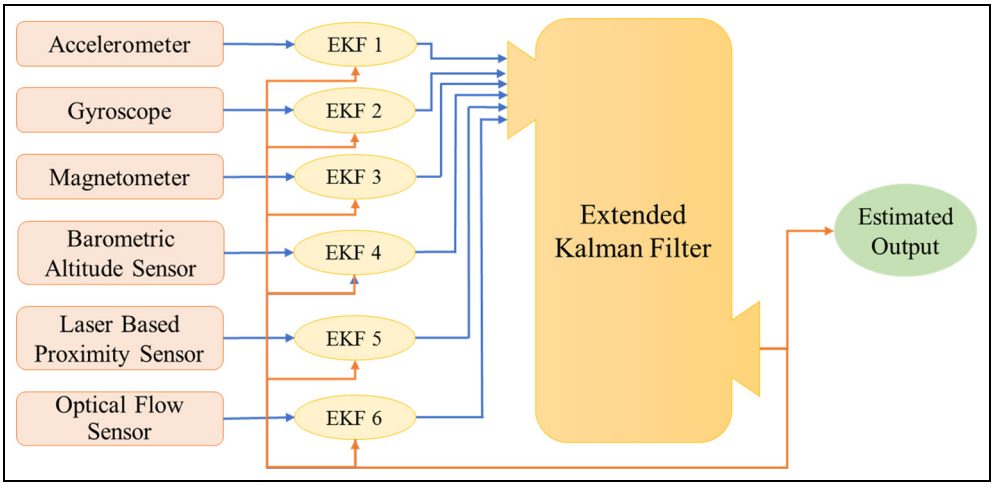

Figure 4. Modified extended Kalman filter (M-EKF) architecture with fault detection.

M-EKF. As evident from the information above, the efficiency of the EKF algorithm depends heavily on the prior knowledge of the dynamic models along with the covariance matrices. Uncertainty in the knowledge of these parameters can significantly degrade the performance of the algorithm. In this article, we dynamically reduce the impact of the noisy sensor output by implementing a fault detection feature. The architecture for the modified extended Kalman filter is shown in Figure 4.

Using the equations described in the previous section, in a real-world example, we use only accelerometer data in this experiment to predict current location in combination with optical flow data. Incorporating the last known position and velocity information obtained by the optical flow method through the BM technique, we estimate and correct the next position. We will incorporate the modifications to show the advantage of M-EKF over the traditional method using simulations. Assuming the state vector of three accelerometers as the input of the system, we have

$$\boldsymbol{x} = \left[\boldsymbol{a_x a_y a_z}\right]^{\boldsymbol{T}} \tag{10}$$

$$\boldsymbol{z} = \left[\boldsymbol{a_x^{(1)} a_y^{(1)} a_z^{(1)} a_x^{(2)} a_y^{(2)} a_z^{(2)}}\right]^{\boldsymbol{T}} \tag{11}$$

Process equations

$$\boldsymbol{a_x} = \boldsymbol{s_x^{(1)}} * \boldsymbol{a_x^{(1)}} + \boldsymbol{s_x^{(2)}} * \boldsymbol{a_x^{(2)}} \tag{12}$$

$$\boldsymbol{a_y} = \boldsymbol{s_y^{(1)}} * \boldsymbol{a_y^{(1)}} + \boldsymbol{s_y^{(2)}} * \boldsymbol{a_y^{(2)}} \tag{13}$$

$$\boldsymbol{a_z} = \boldsymbol{s_z^{(1)}} * \boldsymbol{a_z^{(1)}} + \boldsymbol{s_z^{(2)}} * \boldsymbol{a_z^{(2)}} \tag{14}$$

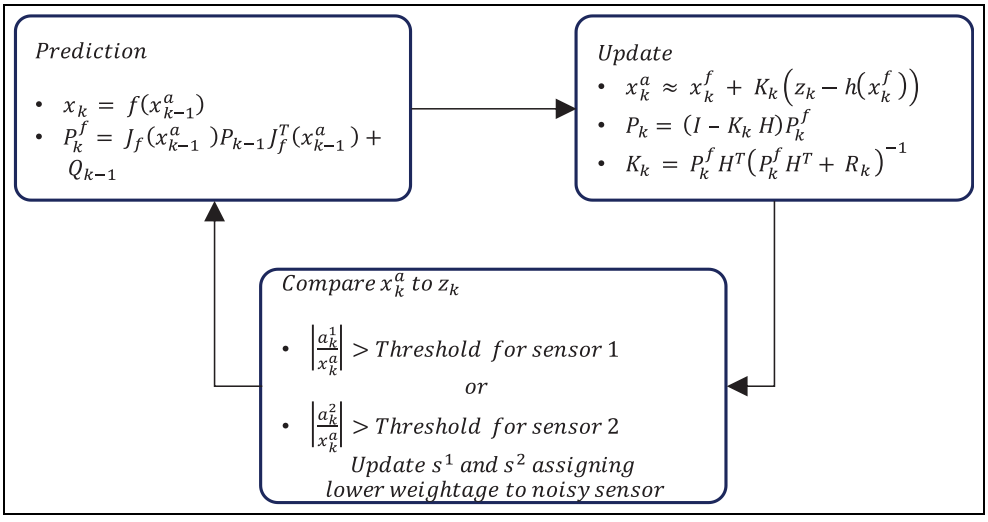

Figure 5. Modified extended Kalman filter process loop.

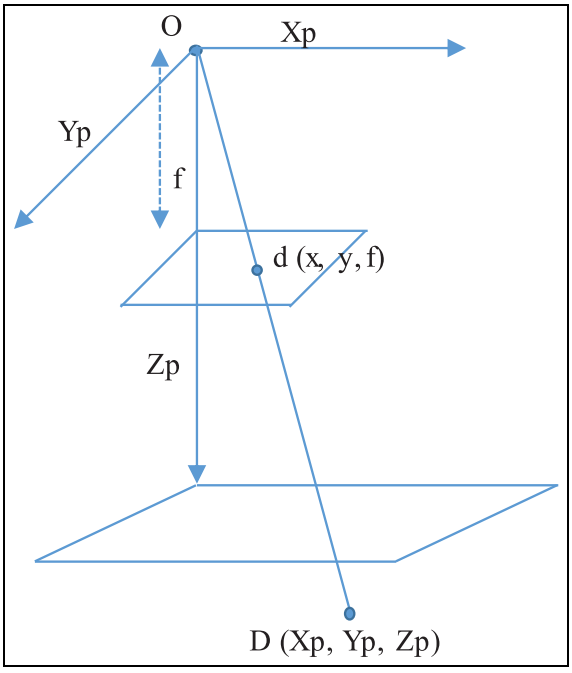

Figure 6. Pinhole model.

where $s^{(1)} = [s_x^{(1)}\, s_y^{(1)}\, s_z^{(1)}]$ and $s^{(2)} = [s_x^{(2)}\, s_y^{(2)}\, s_z^{(2)}]$ are the weights assigned to the two sensors, with $s^{(1)} + s^{(2)} = [1\,1\,1]$.

We know the initial parameters of sensor bias and variance. For determining the threshold value for each of the sensors, the logged data are analyzed. The complete process flow can be understood from Figures 4 and 5.

Optical flow algorithm

At the bottom of the UAV, a CMOS-based camera with a resolution of 752 × 480 is placed. Optical flow algorithm uses the images acquired by the camera for flow measurement. The optical flow between two successive frames is calculated and refined to achieve subpixel accuracy. Angular rate is compensated using the onboard gyroscope. Finally, the laser-based range sonar provides the precise height from the ground; it is used to scale the optical flow reading to get the exact metric motion details. Figure 6 explains in greater detail the pinhole approach employed to scale the optical flow. Ground point is $D[Xp, Yp, Zp]$. Focal length "f" can give us the corresponding 2D point d in camera coordinate system

$$\boldsymbol{d} = \boldsymbol{f}\frac{\boldsymbol{D}}{\boldsymbol{Zp}} \tag{15}$$

Zp is the height given by laser range sensor. The most popular and easy to understand algorithms for optical flow measurement are BM, differential, and feature-based approach as described in section "BM." The problem with these other algorithms is their computational burden, limiting their use in a real-time application like ours. We need a fast and accurate algorithm for our application.

BM. BM algorithm is used widely for motion estimation in video coding.[50] We are using its modified version to reduce its computational cost and maintain the accuracy to make it fit for our application. In BM, the image is divided into blocks. For each block, the algorithm tries to search the best matching block in the previous frame. The aim is to minimize the sum of absolute difference (SAD) in the search window.

Consider a target block at position (x, y) shown in Figure 7 in the current frame and the reference block at position $(x + \breve{a}, y + \breve{b})$ in the previous frame. It can be defined as

$$\boldsymbol{SAD}(\breve{\boldsymbol{a}}, \breve{\boldsymbol{b}}) = \sum_{j=0}^{N-1}\sum_{k=0}^{N-1} \left|\boldsymbol{g}_{i-1}(\boldsymbol{x} + \breve{\boldsymbol{a}} + \boldsymbol{j}, \boldsymbol{y} + \breve{\boldsymbol{b}} + \boldsymbol{k}) - \boldsymbol{g}_{i}(\boldsymbol{x} + \boldsymbol{j}, \boldsymbol{y} + \boldsymbol{k})\right| \tag{16}$$

where g_i the gray value of a pixel in the current is frame i and g_{i-1} is the gray level of a pixel in the previous

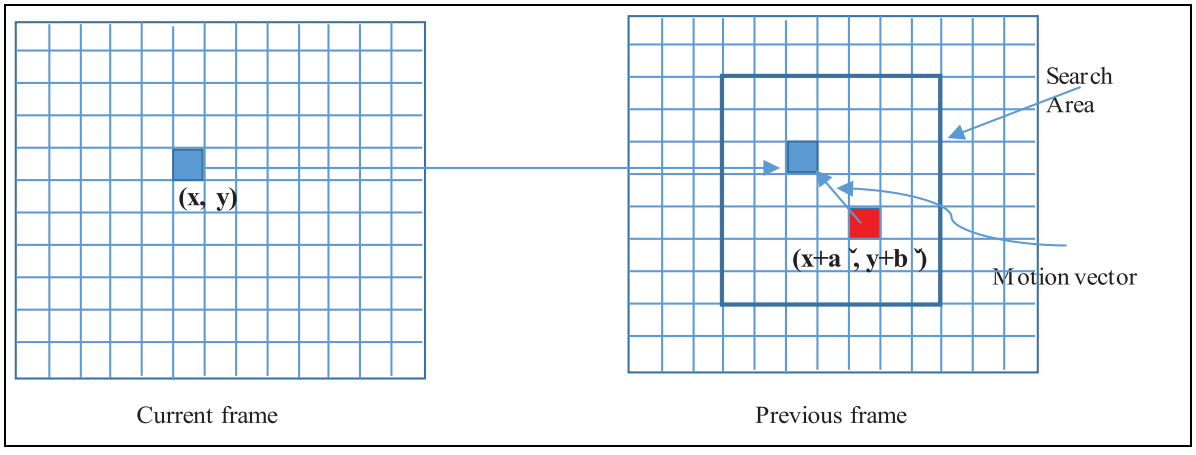

Figure 7. Block-matching algorithm to find motion vector.

frame -1. Therefore, the motion vector (MV) in (a, b) is defined as follows

$$(\boldsymbol{a}, \boldsymbol{b}) = \arg \min_{(\boldsymbol{a}, \boldsymbol{b}) \in \boldsymbol{R}} \boldsymbol{SAD}(\breve{\boldsymbol{a}}, \breve{\boldsymbol{b}}) \tag{17}$$

where

$$R = \{(\breve{a}, \breve{b}) | -W \leq \breve{a}, \breve{b} \geq W \text{ and } (x + \breve{a}, y + \breve{b}) \text{ is a valid position in } i - 1 \text{ frame}\}$$

For BM, FS is the most robust and accurate method to find MV. It tests all possible blocks in $i - 1$ frame within the search area to find the block with the minimum SAD. For the maximum displacement of W, the FS requires $(2W + 1)^2$ search points or blocks. For instance, if the maximum displacement W is ± 7, the total search locations are 225 using the formulae $(2W + 1)^2$. Each SAD calculation requires 2N2 additions, making 130,560 calculations for a 16×16 block. Such a computational requirement makes the implementation of FS difficult for real-time tasks. To make the algorithm run faster, the simplest approach is to reduce the number of SAD computations that will result in a poor accuracy of flow measurement. However, if we can reduce the number of search locations using an intelligent algorithm such as the ABC for BM, then it may perform faster without losing the accuracy.

BM based on ABC. We are employing an ABC algorithm for reducing the search locations for best BM. It uses an intelligent algorithm for finding the global minimum or the best MV. With ABC, the search locations vary from generation to generation and avoid being stuck at a local minimum.[51,52] This algorithm uses a fitness strategy for reducing the computation of SAD values while starting with randomly generated solutions. We perform the following steps until the termination condition is met:[53]

Create initial solutions. The algorithm begins by initializing N solutions, similar to the employed bees in a bee colony. Each solution is a D-dimensional vector containing the parameter values to be optimized. Parameters are randomly and uniformly distributed within the pre-defined bounds

$$\boldsymbol{x}_{j,i} = \boldsymbol{x}_j^{\boldsymbol{low}} + \boldsymbol{rand}(0, 1) * \left(\boldsymbol{x}_j^{\boldsymbol{high}} - \boldsymbol{x}_j^{\boldsymbol{low}}\right) \tag{18}$$

$j = 1, 2, \ldots, D$; $i = 1, 2, \ldots, N_p$

Here, the lower initial parameter bound is x_j^{low} and the upper bound is x_j^{high} where j and i are the parameter and solution indexed in equation, respectively. Hence, $x_{j,i}$ is the jth parameter of the ith solution.

Generate new solution in neighborhoods. At this stage, each employed bee generates a new solution (v_i) in the neighborhood of its present position $(x_{j,i})$ as follows

$$\boldsymbol{v}_{j,i} = \boldsymbol{x}_{j,i} + \varnothing_{j,i}(\boldsymbol{x}_{j,i} - \boldsymbol{x}_{j,k}) \tag{19}$$

$$k \in \{1, 2, \ldots, N_p\}; j \in \{1, 2, \ldots, D\}$$

x_i is randomly chosen and ks is one of the N_p solutions, satisfying the condition $i \neq k$. The scale factor $\varnothing_{j,i}$ has a random value between $[-1, 1]$. Once a new solution is generated, a fitness value of the solution is calculated as given in equation (20)

$$\boldsymbol{fit_i} = \begin{cases} \dfrac{1}{1 + \boldsymbol{J_i}} & \text{if } \boldsymbol{J_i} \geq 0 \\ 1 + abs(\boldsymbol{J_i}) & \text{if } \boldsymbol{J_i} < 0 \end{cases} \tag{20}$$

where J_i is the fitness function to be minimized. If the fitness of v_i is better, then the solution x_i is replaced by v_i, otherwise x_i remains and search continuous for more solutions based on a probability. Each onlooker bee selects one of the proposed solutions depending on their fitness value, which has been recently defined by employed bees. The probability that a solution is directly proportional to the above fitness value can be obtained from equation (21)

$$\boldsymbol{Prob_i} = \frac{\boldsymbol{fit_i}}{\sum_{\boldsymbol{i}=1}^{N_p} \boldsymbol{fit_i}} \tag{21}$$

After the solution is selected, onlooker bees will go to the selected solution and generate innovative solutions inside the neighborhood of the chosen solution. Scout bees perform the task similar to this in nature. In the case that the fitness of the new solution is better than before, such position is held. Otherwise, the old solution remains.

Fitness strategy. With reference from Cuevas et al.,[51] the fitness calculation model follows three important rules to evaluate or estimate fitness values. Figure 8 gives flow representation of fitness strategy.

If a new block (solution) B is located closer than the distance d from the block with best fitness value seen so far, then the fitness value of B is evaluated using the SAD calculation. If a new block B is located further than distance d from the nearest block with known fitness, then its fitness value is evaluated using SAD calculation. If a new block B is located closer than the distance d from the block whose fitness value does not correspond with the best fitness value, then its fitness value is estimated as the same fitness as of that block.

ABC-BM algorithm. Using the flow chart in Figure 9, we have presented an algorithm that is implemented on the drone-computing platform for an ABC-BM

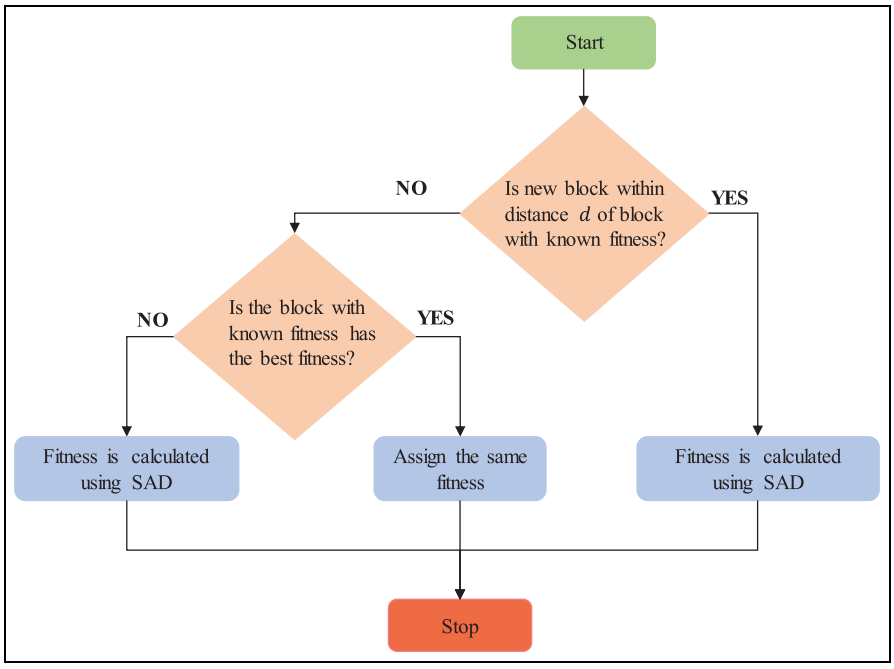

Figure 8. Fitness strategy flow chart.

Algorithm of Artificial Bee Colony—Block Matching

Input: Initialize counter; // Set the counter limits
Initialize blocks B = {B1, B2, B3, B4, B5}
Clear all counters C_i 1 $\leq i \leq$ 5
Initialize empty array A // for fitness database
1 **while** $C_i \leq 5$
2 **for** any block in distance d
3 **if** (fitness not known) \\ for each block
4 *Calculate* fitness from SAD values
5 **else if** (it is the best fitness)
6 *Assign fitness and store in A.*
7 **else**
8 *Calculate* fitness from SAD values
9 **end if**
10 **end for**
11 *Update* all blocks B_i & all counter C_i
12 *Evaluate* new fitness
13 *Update* database array A
14 *Calculate* $Prob_i$ \\ for each solution as preference index
15 *Generate* new search locations \\ according to $Prob_i$
16 *Update* counters
17 *Evaluate* fitness
18 *Update* database array A
19 *Generate* new population
20 **if** new solution B' better than old solution, B
21 Choose B'
22 **Else**
23 Choose B
24 *Update* a_{best}, b_{best}
25 **end if**
26 **end While**

algorithm, which is presented followed by a flow chart. Taking reference from Honegger et al.,[8] the average number of search points reached in different test cases using FS-BM is 422.3, whereas, for similar performance, the ABC-BM searched an average of 12.14 points with similar accuracy. This performance makes ABC-BM the best choice for implementation in real-time applications.

In our case, the camera can provide up to 60 frames per second. After applying 4 × binning, we get approximately 240 frames per second. In 4 × binning, we are getting the average of the 4 × 4 pixels as one pixel. We calculate the SAD value of 8 × 8 pixel block in the search locations given by the ABC algorithm. We now take the position having the minimum SAD block value as the flow measurement for the corresponding sample block. Up to 64 sample points are taken and passed through a histogram filter, and we select the one with highest histogram bin. Histogram filter normalizes the image after converting it into grayscale and produces a histogram of the image that represents the frequency of occurrence of each gray level in the image. Histogram filter reduces the effects of light and other radiation. In this way, every pixel has an optical flow value.

Afterward, we perform subpixel refinement, which estimates the optical flow with half-pixel step size in every direction of the best-matched pixel. Finally, we select the best match of the direction with the best-matched pixel.

Sensors

For our experiments, we have used several different sensors onboard to help navigate the UAV. These sensors

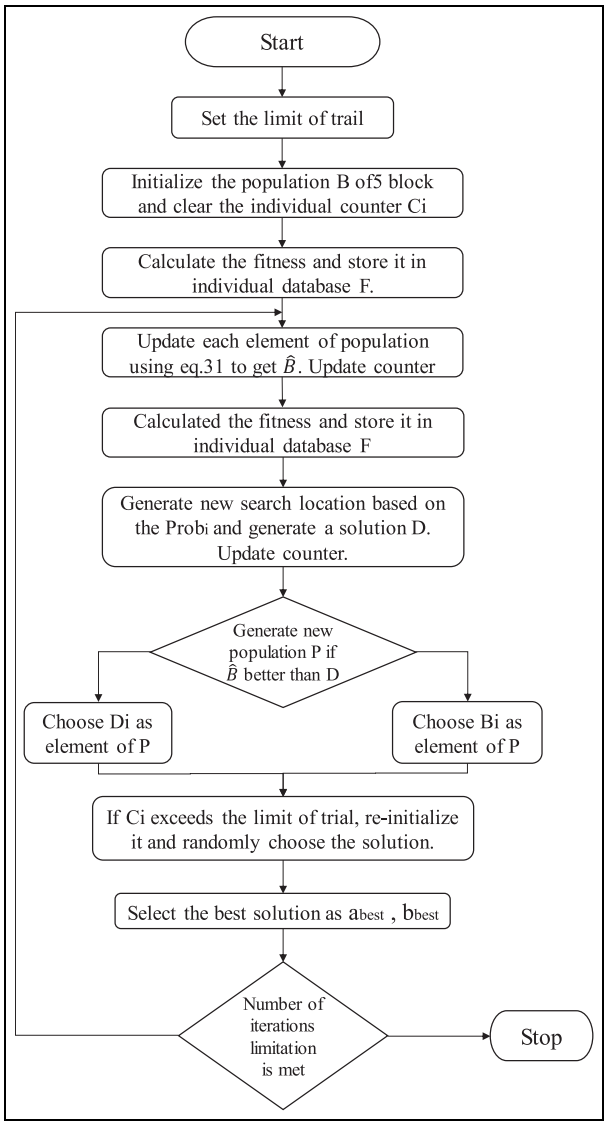

Figure 9. ABC-BM Flowchart.

include an accelerometer, gyroscope, magnetometer, barometric altitude sensor, laser-based proximity sensor, and an optical flow sensor. We are using 3DR IRIS + UAV for our experimental purpose. The reason for choosing this particular UAV is that it is based on open hardware as well as software, making it possible for us to implement our algorithms and add more sensors to it.

Pixhawk is an independent open hardware kit that specifically developed for implementing high-performance autopilot software.[9] It is available for use by academics and industry, as well as hobbyists. A 168-MHz Cortex M4F CPU (256 KB RAM, 2 MB Flash) powers the Pixhawk. It also embeds a 3D accelerometer, gyroscope, magnetometer, and barometer with the below characteristics (Table 2). There is also an integrated backup or failsafe processor, which makes it a perfect choice for real-time autonomous UAV. Figure 10 shows the optical flow sensor, laser-based range sensor, and Pixhawk board (brain of the UAV). Optical flow and range sensors were attached at the bottom of the UAV, such that they face toward the ground while it is flying.

Results and discussion

Simulation results

We first used logged accelerometer data from a test flight and used our algorithm on the recorded data on a PC. Figure 11 shows the filter output when both the sensors are assigned equal weights, while Figure 12 involves assigning lower or negligible weight to the noisy sensor using modified EKF. The filter output is closer to the less noisy sensor.

This approach enabled us to exploit the availability of multiple sensors for the same variable. During the

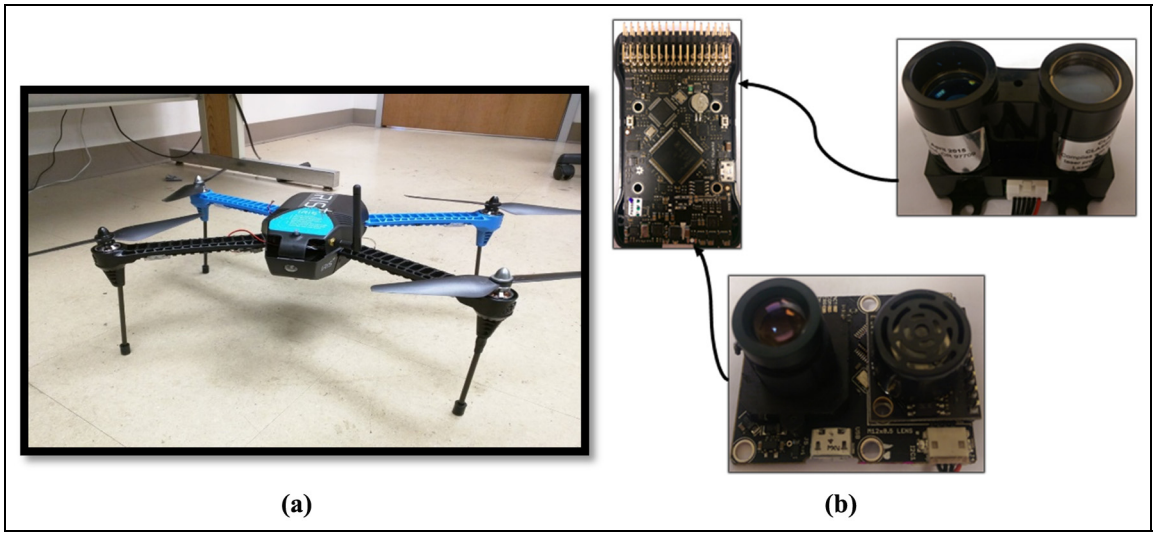

Figure 10. (a) 3DR IRIS + UAV with Pixhawk (open hardware) and (b) optical flow sensor, laser-based range sensor, and the Pixhawk board.

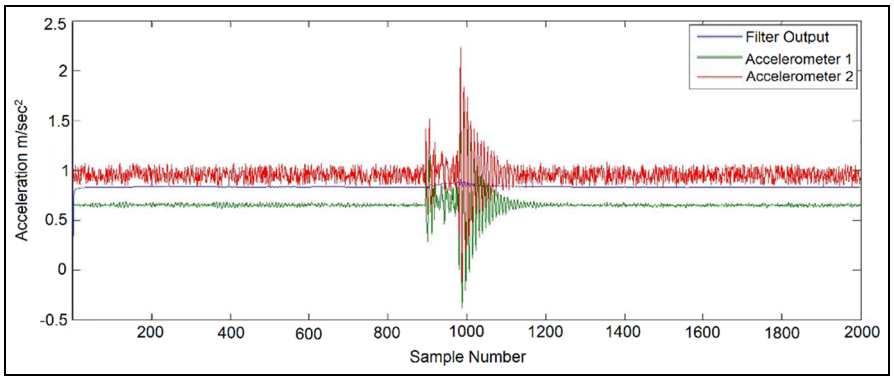

Figure 11. Plot of filter output with dynamic weight assignment.

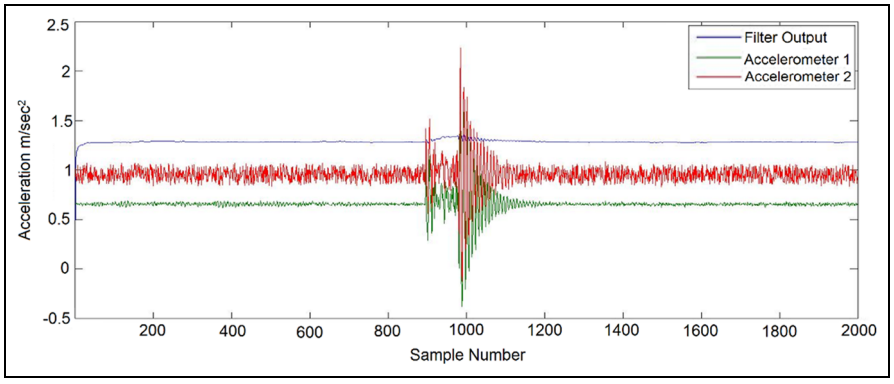

Figure 12. Plot of filter output without dynamic weight.

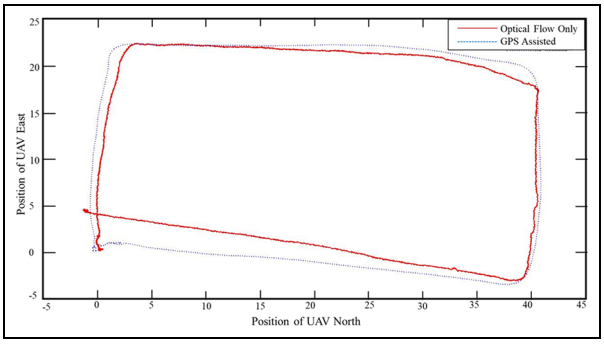

Figure 13. Position estimate: modified EKF versus GPS.

real flight, this is helpful in the scenarios when the wind increases, thereby increasing the vibrations in the UAV, making the inertial sensors noisy while the optical flow is still working fine. The reverse happens when the terrain is featureless, making the optical flow less reliable and inertial sensors more accurate. Then, we plotted output of the flight data recorded during a test flight. Figures 13 and 14 show the position and velocity estimate of the UAV, respectively, during this test flight. We fed the logged sensor data to our implemented EKF on MATLAB and the UAV. The output results, as expected, are not as accurate as GPS-assisted navigation. However, the error is within an acceptable limit. Once the algorithm was proved to be giving results within acceptable error ranges compared with actual GPS values, field tests were used to check the actual error obtained in the real-world experiments. These results are discussed in the following section.

Table 2. Sensors used.

Name	Model	Accuracy
Accelerometer/magnetometer	ST Micro LSM303D	± 16 g/ ± 12 gauss
Gyroscope	ST Micro L3GD20	2000°/s
Accelerometer/gyroscope	InvenSense MPU 6000	± 16 g/2000°/s
Barometric altitude sensor	MEAS MS5611	± 10 cm
Laser-based range sensor	Lidar Lite	± 2.5 cm
Optical flow sensor	PX4Flow	± 1.5 rad/s = ± 86°/s

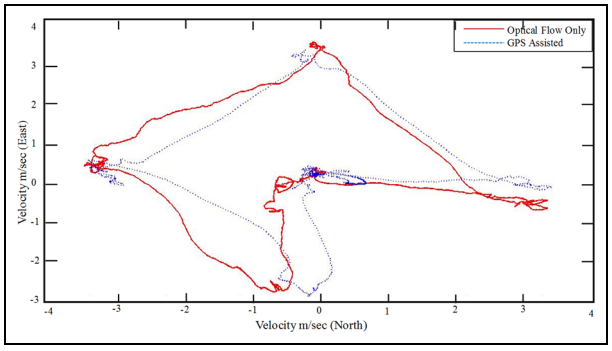

Figure 14. Velocity estimate: modified EKF versus GPS.

Experimental results

We experimented to verify the performance of our algorithm in a parking lot at the University of Toledo, shown in Figure 15. We uploaded a set of waypoints to the UAV for the autonomous flight. With each set of the waypoint, we fly it two times, the first time with GPS available to the system and the second time with no GPS. For the second flight, the UAV just had the coordinates of its initial position. Figure 16 shows the saved waypoints of the first path in the UAV.

Figure 17 shows the logged data; the red color path is the first flight, which uses the GPS data for naviga-

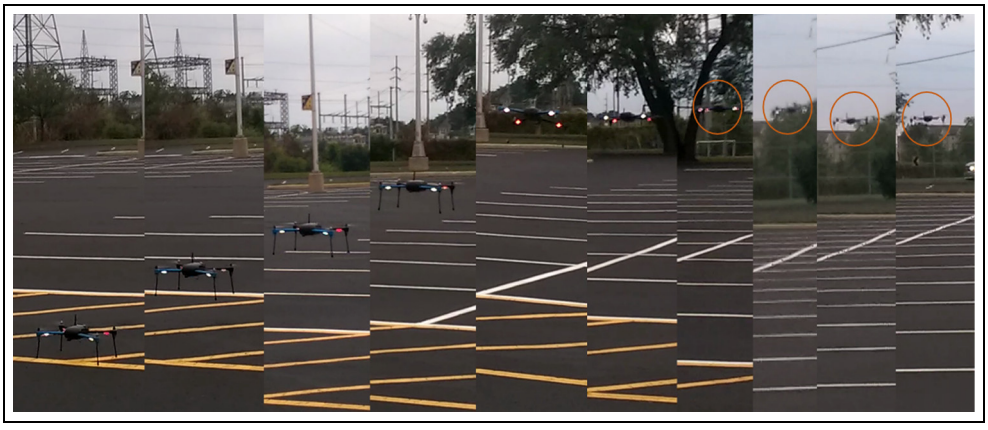

Figure 15. Outdoor autonomous navigation of Pixhawk UAV in simulated GPS-denied environment demonstration in the University of Toledo parking lot (https://youtu.be/BmhQ-FfZspQ).

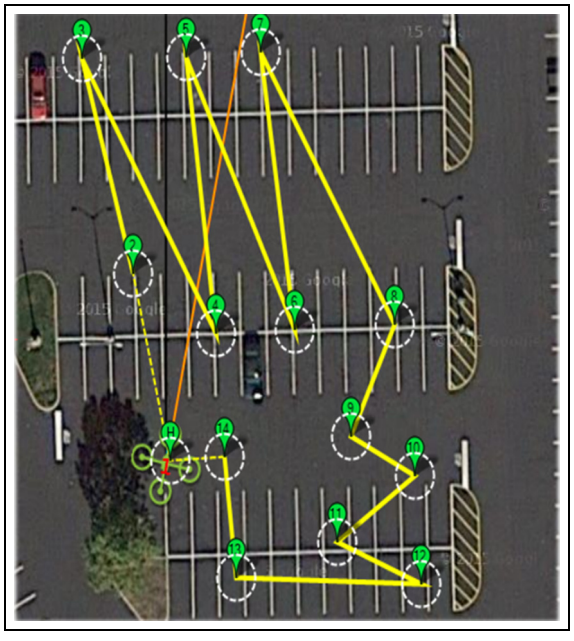

Figure 16. First set of waypoints stored on the UAV.

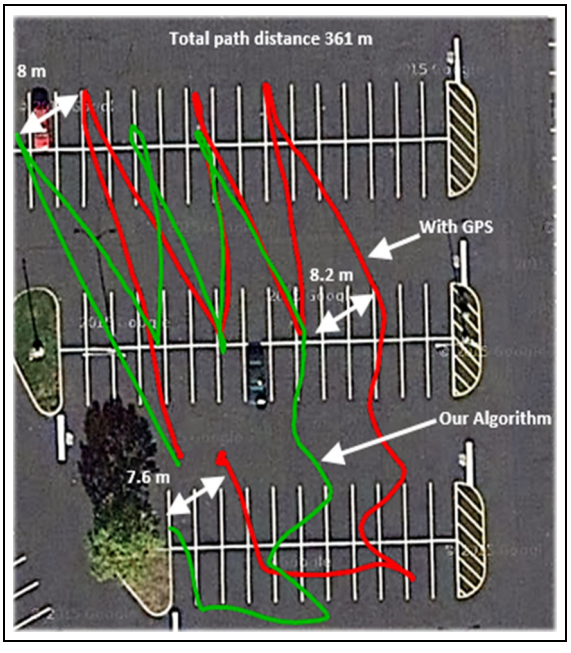

Figure 17. First path: comparison of our algorithm versus GPS-aided navigation.

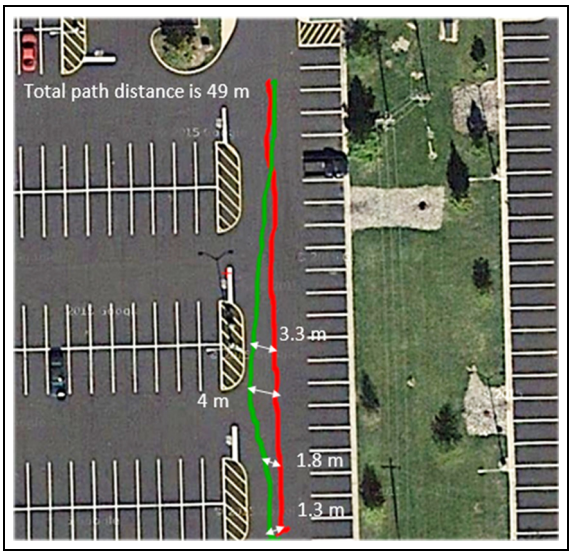

Figure 18. Third path: comparison of our algorithm versus GPS-aided navigation.

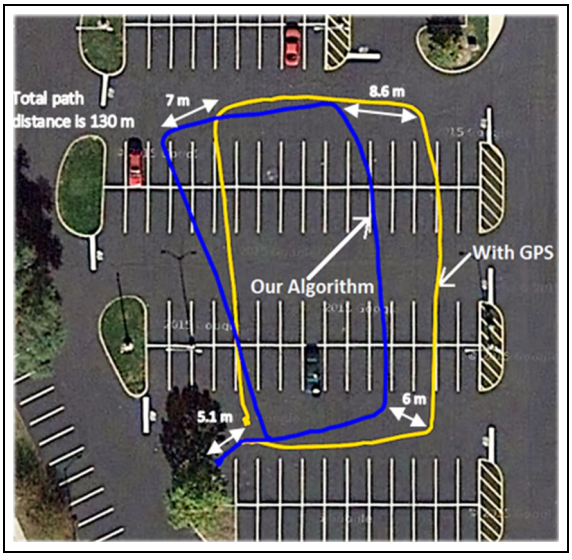

Figure 19. Second path: comparison of our algorithm versus GPS-aided navigation.

tion, and the green colored path is the second flight that does not use the GPS data for navigation. We logged the GPS data to compare both the flights with the same reference. As can be seen in Figure 18, our algorithm is slightly shifted from the actual path; this is due to the high-speed wind blowing during the experiment. Although we currently do not have any sensor to detect the wind and compensate for it, our system gave an acceptable performance without using GPS in our algorithm.

Figure 19 shows the logged data of the second test flight with new waypoints (logged within the UAV

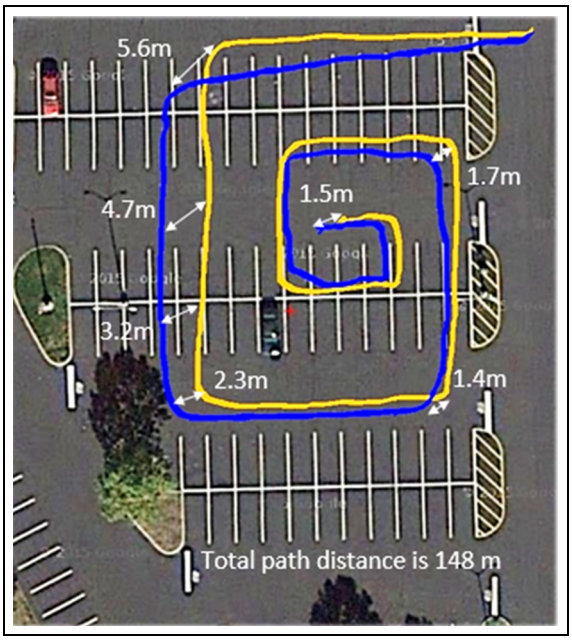

Figure 20. Fourth path: comparison of our algorithm versus GPS-aided navigation.

beforehand) to verify our algorithm further. It is a rectangular path of approximately 130 m. It still has the drifting effect from the wind. The yellow color line shows the GPS assisted flight, and the blue color line shows the path followed by the UAV without using the GPS. The third path is a simple line of approximately 49 m. Fortunately, the wind speed was minimal during this experiment; our UAV followed the GPS path quite efficiently. In Figure 19, if we consider the landing point for both the flights they differ by only 1.3 m. Figure 20 shows the spiral type of flight path completed by our UAV as the fourth experiment. We logged the wind speed data during every experiment. Table 2 shows the mathematical figures of the performance. Similarly, Figure 21 also exhibits accurate navigation.

Results presented in Figure 22(a) and (b) show flight path when (1) the UAV was guided by the GPS in a 3D outdoor environment (in red) and (2) the UAV maneuvered the same path generated in (1) when GPS outage occurs using our algorithm (in green). In these newly generated results, we used a complex 3D path compared with our previous results. In presented results, red lines indicate the path generated by Pixhawk with GPS-guided navigation, and green lines indicate the autonomous navigation of UAV that follows the original (red) path. This additional experiment demonstrates the repeatability of paths. Based on the visual results in the complex 3D navigation, the new path followed by the UAV in a GPS-denied environment is highly accurate, as seen in Figure 22.

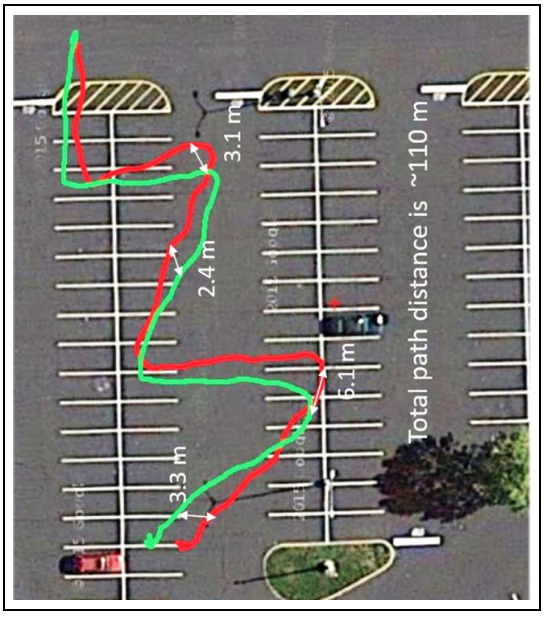

Figure 21. Fifth path: comparison of our algorithm versus GPS-aided navigation.

Conclusion and future work

Our system worked with acceptable accuracy in the GPS-denied environment in multiple instances. With the experimental data, we have proved its usability in a real-world scenario in controlled environments. Moreover, in uncontrolled environments, the performance overshoot created new paths and consumed excessive power that needs to be addressed in future work. Our algorithm will easily navigate through an environment with weak GPS signals or GPS-denied regions for a longer duration compared with a GPS/INS system. It is evident from Table 3 that when the speed of wind was lower, the accuracy of the system improved. We have achieved this by employing an intelligent and robust optical flow measurement algorithm, ABC-based BM with some modifications in traditional EKF. We achieved better noise reduction using modified EKF for data fusion. More work is needed to improve the system and increase robustness against physical parameters like the wind. We also plan

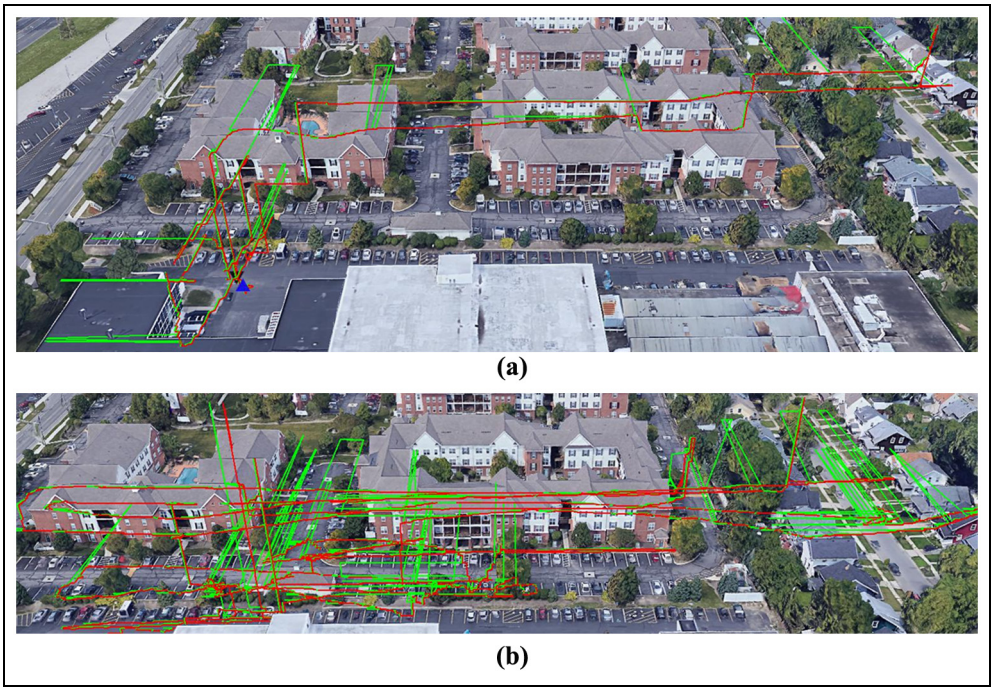

Figure 22. Path followed with GPS (red) and followed without GPS (green) in 3D at the University of Toledo campus.

Table 3. Performance of the proposed algorithm.

Name	Average drift	Wind speed[a] (mile/h)	Total path length (m)	Percentage error (average drift/path length) (%)
Path 1	(8 + 8.2 + 7.6)/3 = 7.93	14	361	2.19
Path 2	(7 + 8.6 + 6 + 5.6)/4 = 6.8	20	130	5.23
Path 3	(3.3 + 4 + 1.8 + 1.3)/4 = 2.6	2.6	49	5.30
Path 4	(5.6 + 4.7 + 3.2 + 2.3)/4 = 3.95	2.6	148	2.67

[a]Reference for wind speed: https://www.ncdc.noaa.gov

to explore algorithms that could help to further improve the overall system performance.

Acknowledgements

The authors would like to thank Department of EECS (Electrical Engineering and Computer Science) at University of Toledo for its support; authors also appreciate the support of the Paul A. Hotmer Family CSTAR (Cybersecurity and Teaming Research) Lab at the University of Toledo.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: The authors received no financial support for the research and authorship. Publication of this article was supported by the Paul A. Hotmer Family Cybersecurity and Teaming Research (CSTAR) lab at the University of Toledo.

ORCID iD

Praveen Damacharla https://orcid.org/0000-0001-8058-7072
Ahmad Y Javaid https://orcid.org/0000-0003-4719-4941

References

1. Stolaroff J. *The need for a life cycle assessment of drone-based commercial package delivery*. LLNL-TR-652316, 25 March 2014. Livermore, CA: Lawrence Livermore National Laboratory.
2. Han S and Wang J. Integrated GPS/INS navigation system with dual-rate Kalman Filter. *GPS Solut* 2012; 16(3): 389–404.
3. Newman P, Cole D and Ho K. Outdoor SLAM using visual appearance and laser ranging. In: *Proceedings of the IEEE international conference on robotics and automation*, Orlando, FL, 15–19 May 2006. New York: IEEE.
4. Mercado DA, Flores G, Castillo P, et al. GPS/INS/optic flow data fusion for position and velocity estimation. In: *Proceedings of the international conference on unmanned aircraft system*, Atlanta, GA, 28–31 May 2013, pp.486–491. New York: IEEE.
5. Kim K, Kim W, Choi D, et al. Calibration of the drift error in GPS using optical flow and fixed reference station. In: *Proceedings of the 15th international conference on control, automation, and systems*, Busan, South Korea, 13–16 October 2015, pp.1370–1373. New York: IEEE.
6. Javaan C, Rosser K and Mizutani A. Axially displaced optical flow sensors for measuring and controlling the landing height of a UAV. In: *Proceedings of the 15th Australian international aerospace congress*, Melbourne, VIC, Australia, 25–28 February 2013, pp.221–227. Australian International Aerospace Congress.
7. Dille M, Grocholsky B and Singh S. Outdoor downward-facing optical flow odometry with commodity sensors. In: Howard A, Iagnemma K and Kelly A (eds) *Field and service robotics*. Berlin: Springer, 2010, pp.183–193.
8. Honegger D, Meier L, Tanskanen P, et al. An open source and open hardware embedded metric optical flow CMOS camera for indoor and outdoor applications. In: *Proceedings of the IEEE international conference on robotics and automation*, Karlsruhe, 6–10 May 2013, pp.1736–1741. New York: IEEE.
9. Reif K, Gunther S, Yaz E, et al. Stochastic stability of the discrete-time extended Kalman filter. *IEEE T Automat Contr* 1999; 44(4): 714–728.
10. Brox T and Malik J. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE T Pattern Anal* 2011; 33(3): 500–513.
11. Chao H, Gu Y and Napolitano M. A survey of optical flow techniques for UAV navigation applications. In: *Proceedings of the 2013 International Conference on Unmanned Aircraft Systems*, Atlanta, GA, 28–31 May 2013, pp.710–716. New York: IEEE.
12. Chao H, Gu Y and Napolitano M. A survey of optical flow techniques for robotics navigation applications. *J Intell Robot Syst* 2014; 73(1–4): 361–372.
13. Kim Y, Jung W and Bang H. Visual target tracking and relative navigation for unmanned aerial vehicles in a GPS-denied environment. *Int J Aeronaut Space* 2014; 15(3): 258–266.
14. Floris VB, Kristi M and Dickinson HM. Monocular distance estimation from optic flow during active landing maneuvers. *Bioinspir Biomim* 2014; 9(2): 025002.
15. Lange S, Sunderhauf N and Protzel P. A vision based onboard approach for landing and position control of an autonomous multirotor UAV in GPS-denied environments. In: *Proceedings of the international conference on advanced robotics*, Munich, 22–26 June 2009, pp.1–6. New York: IEEE.
16. Herisse B, Hamel T, Mahony R, et al. Landing a VTOL unmanned aerial vehicle on a moving platform using optical flow. *IEEE T Robot* 2012; 28(1): 77–89.
17. Schneider K, Conroy J and Nothwang W. *Computing optic flow with ArduEye vision sensor*. ARL-TR-6292, 2013. Fort Belvoir, VA: Army Research Lab.
18. Beauchemin SS and Barron JL. The computation of optical flow. *ACM Comput Surv* 1995; 27(3): 433–466.
19. Caballero F, Merino L, Ferruz J, et al. Vision-based odometry and SLAM for medium and high altitude flying UAVs. *J Intell Robotic Syst* 2009; 54(1–3): 137–161.
20. Cesetti A, Frontoni E, Mancini A, et al. A vision-based guidance system for UAV navigation and safe landing using natural landmarks. *J Intell Robot Syst* 2010; 57(1–4): 233.
21. Hwang S-Y and Song J-B. Monocular vision-based SLAM in indoor environment using corner, lamp, and door features from upward-looking camera. *IEEE T Ind Electron* 2011; 58(10): 4804–4812.
22. Robert CL, Timothy WM and Randal WB. Relative navigation approach for vision-based aerial GPS-denied navigation. *J Intell Robot Syst* 2014; 74(1–2): 97–111.
23. Green WE and Oh PY. Optic-flow-based collision avoidance. *IEEE Robot Autom Mag* 2008; 15(1): 96–103.

24. Ding W, Wang J, Han S, et al. Adding optical flow into the GPS/INS integration for UAV navigation. In: *Proceedings of the international global navigation satellite systems society symposium*, Australia, Qld, 1–3 December 2009, pp.1–13.
25. Mercado DA, Flores G, Castillo P, et al. GPS/INS/optic flow data fusion for position and Velocity estimation. In: *Proceedings of the 2013 international conference on unmanned aircraft systems*, Atlanta, GA, 28–31 May 2013, pp.486–491. New York: IEEE.
26. Rhudy MB, Chao H and Gu Y. Wide-field optical flow aided inertial navigation for unmanned aerial vehicles. In: *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems*, Chicago, IL, 14–18 September 2014, pp.674–679. New York: IEEE.
27. Gageik N, Strohmeier M and Montenegro S. An autonomous UAV with an optical flow sensor for positioning and navigation. *Int J Adv Robot Syst*. Epub ahead of print 1 January 2013. DOI: 10.5772/56813.
28. Khaleghi B, Khamis A, Karray FO, et al. Multisensor data fusion: a review of the state-of-the-art. *Inform Fusion* 2013; 14(1): 28–44.
29. Blösch M, Weiss S, Scaramuzza D, et al. Vision based MAV navigation in unknown and unstructured environments. In: *Proceedings of the 2010 IEEE international conference on Robotics and automation*, Anchorage, AK, 3–7 May 2010, pp.21–28. New York: IEEE.
30. Salahshoor K, Mosallaei M and Bayat AM. Centralized and decentralized process and sensor fault monitoring using data fusion based on adaptive extended Kalman filter algorithm. *Measurement* 2008; 41(10): 1059–1076.
31. Kim J, Kim Y and Kim S. An accurate localization for mobile robot using extended Kalman filter and sensor fusion. In: *Proceedings of the IEEE international joint conference on neural networks*, Hong Kong, China, 1–8 June 2008, pp.2928–2933. New York: IEEE.
32. Simon D. Kalman filtering with state constraints: a survey of linear and nonlinear algorithms. *IET Control Theory A* 2010; 4(8): 1303–1318.
33. Haseltine EL and Rawlings JB. Critical evaluation of extended Kalman filtering and moving-horizon estimation. *Ind Eng Chem Res* 2005; 44(8): 2451–2460.
34. Aggarwal P, Syed Z, Noureldin A, et al. *Integrated MEMS based navigation systems*. London: Artech House Publishers, 2010.
35. Haug AJ. The analytical linearization class of Kalman filters: the extended Kalman filter. In: Haug AJ (ed.) *Bayesian estimation and tracking: a practical guide*. Hoboken, NJ: Wiley, 12 June 2012, pp.93–114.
36. Kim KH, Lee JG and Park CG. Adaptive two-stage extended Kalman filter for a fault-tolerant INS-GPS loosely coupled system. *IEEE T Aero Elec Sys* 2009; 45(1): 125–137.
37. Almagbile A, Wang J and Ding W. Evaluating the performances of adaptive Kalman filter methods in GPS/INS integration. *J Global Position Syst* 2010; 9(1): 33–40.
38. Zabidi A, Yassin IM, Tahir NM, et al. Comparison between binary particles swarm optimization (BPSO) and binary artificial bee colony (BABC) for nonlinear autoregressive model structure selection of chaotic data. *J Fund Appl Sci* 2017; 9(3S): 730–754.
39. Civicioglu P and Besdok E. A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms. *Artif Intell Rev* 2013; 39(4): 315–346.
40. Muthiah A and Rajkumar R. A comparison of artificial bee colony algorithm and genetic algorithm to minimize the makespan for job shop scheduling. *Procedia Eng* 2014; 97: 1745–1754.
41. Ouyang W, Tombari F, Mattoccia S, et al. Performance evaluation of full search equivalent pattern matching algorithms. *IEEE T Pattern Anal* 2012; 34(1): 127–143.
42. Ma F, Liu F, Zhang X, et al. An ultrasonic positioning algorithm based on maximum correntropy criterion extended Kalman filter weighted centroid. *Signal Image Video P*. Epub ahead of print 16 March 2018. DOI: https://doi.org/10.1007/s11760-018-1272-2.
43. Wu J, Zhou Z, Chen J, et al. Fast complementary filter for attitude estimation using low-cost MARG sensors. *IEEE Sens J* 2016; 16(18): 6997–7007.
44. Balamurugan G, Valarmathi J and Naidu VPS. Survey on UAV navigation in GPS denied environments. In: *Proceedings of the 2016 international conference on signal processing, communication, power and embedded system*, Paralakhemundi, India, 3–5 October 2016. New York: IEEE.
45. Campbell ME, Lee J-W, Scholte E, et al. Simulation and flight test of autonomous aircraft estimation, planning, and control algorithms. *J Guid Control Dynam* 2007; 30(6): 1597–1609.
46. Aydogmus Z and Aydogmus O. A comparison of artificial neural network and extended Kalman filter based sensorless speed estimation. *Measurement* 2015; 63: 152–158.
47. Lee JH and Ricker NL. Extended Kalman filter based nonlinear model predictive control. *Ind Eng Chem Res 194*; 33(6): 1530–1541.
48. Terejanu GA. Extended Kalman Filter Tutorial, https://homes.cs.washington.edu/~todorov/courses/cseP590/readings/tutorialEKF.pdf (2003, accessed 16 December 2016).
49. Lewis JM, Lakshmivarahan S and Dhall S. *Dynamic data assimilation: a least squares approach*, vol. 13. Cambridge: Cambridge University Press, 2006.
50. Lu J and Liou ML. A simple and efficient search algorithm for block-matching motion estimation. *IEEE T Circ Syst Vid* 1997; 7(2): 429–433.
51. Cuevas E, Zaldívar D, Pérez-Cisneros M, et al. Block matching algorithm for motion estimation based on Artificial Bee Colony (ABC). *Appl Soft Comput* 2013; 13(6): 3047–3059.
52. Dervis K, Gorkemli B, Ozturk C, et al. A comprehensive survey: artificial bee colony (ABC) algorithm and applications. *Artif Intell Rev* 2014; 42(1): 21–57.
53. Karaboga D and Akay B. A comparative study of artificial bee colony algorithm. *Appl Math Comput* 2009; 214(1): 108–132.
54. Meier L, Tanskanen P, Fraundorfer F, et al. Pixhawk: a system for autonomous flight using onboard computer vision. In: *Proceedings of the 2011 IEEE international conference on robotics and automation*, Shanghai, China, 9–13 May 2011. New York: IEEE.