



# Implementation of solution separation-based Kalman filter integrity monitoring against all-source faults for multi-sensor integrated navigation

Shizhuang Wang<sup>1</sup> · Yawei Zhai<sup>1</sup> · Xingqun Zhan<sup>1</sup>

Received: 2 June 2021 / Accepted: 18 February 2023 / Published online: 11 April 2023  
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2023

## Abstract

Accurate and safety-assured navigation is demanded by future autonomous systems such as automated vehicles and urban air mobility (UAM). These systems usually integrate multiple sensors to improve navigation accuracy and require the corresponding integrity monitoring architecture to ensure integrity. In response, we implement the Solution Separation-based Kalman filter integrity monitoring (SS-KFIM) technique to achieve fault detection and protection level evaluation for multi-sensor integrated navigation. In our implementation, the filter bank management strategies to handle sensor-in and sensor-out events are discussed. Besides, we consider the faults in state initialization and propagation phases aside from those at the measurement-update stage. Furthermore, our implementation can accommodate the cases where the all-in-view filter is not optimal in a least-squares sense. Simulations are conducted with an illustrative example where an inertial navigation system, global navigation satellite systems, and visual odometry are integrated for a UAM task. The results prove the high effectiveness and extended applicability of our implementation of the SS-KFIM algorithm.

**Keywords** Multi-sensor integrated navigation · Integrity monitoring · Solution separation · Kalman filter

## Introduction

Autonomous systems such as Highly-Automated Vehicles (HAVs), Unmanned Aerial Vehicles (UAVs), and Urban Air Mobility (UAM) have attracted worldwide interest. Accurate and safety-assured navigation is the precondition for them to achieve autonomous operation. These systems usually integrate multiple sensors such as Global Navigation Satellite Systems (GNSS) and Inertial Navigation System (INS) to improve navigation accuracy. Meanwhile, it becomes a solid demand for them to ensure the integrity of onboard navigation systems.

Integrity is a key performance metric in Safety-Critical Applications (SCAs) such as civil aviation. It measures the trust that can be placed in the correctness of a navigation output (ICAO 2009). Integrity monitoring ensures navigation integrity through real-time Fault Detection (FD) and

Protection Level (PL) evaluation. FD aims at checking whether there is any fault in the system, and PL captures the probabilistic upper bound on navigation errors.

Although Fault Exclusion (FE) improves navigation continuity, it is not always necessary in SCAs. For example in civil aviation, although FE is needed in en-route phases to meet stringent continuity requirements (e.g.,  $10^{-6}$ /hour), it is not required in most precision approach phases which have loose continuity requirements (e.g.,  $8 \times 10^{-6}$ /15 s) (Zhai et al. 2018). Similarly, continuity requirements can be satisfied without FE in specific autonomous applications, such as UAM landing and HAV automatic parking. This work focuses on these continuity-insensitive applications, and thus we implement an FD-only integrity monitoring technique.

The well-known integrity solution for GNSS is Receiver Autonomous Integrity Monitoring (RAIM) (Brown 1992), which has become a key function in airborne receivers. Recently, it has evolved into Advanced RAIM (ARAIM) with fully refined architectures and algorithms (Blanch et al. 2015; Blanch and Walter 2021; Joerger et al. 2014). Specifically, ARAIM is established based on Multiple

✉ Xingqun Zhan  
xqzhan@sjtu.edu.cn

<sup>1</sup> School of Aeronautics and Astronautics, Shanghai Jiao Tong University, Shanghai, China

Hypothesis Solution Separation (MHSS) and offers better performance than traditional RAIM approaches.

RAIM and ARAIM were originally developed for Least-Squares (LS) estimators. Meanwhile, there have been various studies focusing on integrity monitoring for Kalman Filters (KFs). Some approaches aim at improving navigation robustness through Fault Detection and Exclusion (FDE) without quantifying the Integrity Risk (IR). For example, various FDE methods were developed to protect GNSS/INS tight integration against GNSS faults, based on hypothesis tests (Bhatti et al. 2012; Wang et al. 2016), robust estimation (Wang et al. 2018), or machine learning (Zhong et al. 2017). These methods were further improved to accommodate both GNSS faults and Inertial Measurement Unit (IMU) failures (Wang et al. 2020). Additionally, Jurado et al. (2020) presented a residual-based multi-filter method to realize FDE and offer performance guarantees for multi-sensor integrated navigation. And there are also fault-tolerant methods for federal Kalman filters (Guo et al. 2019).

There have also been a few techniques capable of evaluating the integrity of KF-based navigation systems, which are called Kalman Filter Integrity Monitoring (KFIM) hereafter. Lee et al. (2018) employed snapshot innovations for fault detection, but this method has poor detection capability against slowly growing errors and does not consider the risk coming from previously undetected faults. Innovation or residual sequence monitors can mitigate these issues (Joerger and Pervan 2013; Tanil et al. 2018a, 2018b), but they are computationally expensive due to the search of the worst-case fault profile, especially in multi-fault cases.

Attracted by the outstanding performance of MHSS ARAIM, researchers started to apply MHSS to KFIM and proved that MHSS usually produces lower PLs than traditional methods (Gunning et al. 2018; Tanil et al. 2019). The MHSS-based KFIM, referred to as SS-KFIM, is computationally expensive because it requires running a bank of filters. Fortunately, this issue can be addressed by fault grouping and/or suboptimal subfilter techniques (Blanch et al. 2019).

To enable accurate and safety-assured navigation for autonomous systems, we implement the SS-KFIM algorithm for multi-sensor integrated navigation. First, the existing SS-KFIM algorithm is revisited with fault sources analyzed and key assumptions laid out. Then, the filter bank management strategies to handle sensor-in and sensor-out events are discussed. Next, we consider the faults in state initialization and propagation phases aside from those at the measurement-update stage. Finally, the SS-KFIM algorithm is modified to accommodate the cases where the all-in-view filter is not optimal in a least-squares sense. This work is expected to extend the applicability of the SS-KFIM technique.

## Solution separation-based Kalman filter integrity monitoring: a revisit

We revisit the existing FD-only SS-KFIM algorithm. First, the principles of the Kalman filter are given. Then, all sources of faults affecting the filter are analyzed. Finally, the basic SS-KFIM algorithm is presented with key assumptions laid out.

### Principles of the Kalman filter

The KF performs prediction and update steps in one iteration. The prediction step is to predict the state and its error covariance using the system model below:

$$\mathbf{x}_k = \mathbf{F}_{k|k-1}\mathbf{x}_{k-1} + \boldsymbol{\omega}_k \tag{1}$$

where  $\mathbf{x}$  denotes the  $n_x$ -dimensional state vector,  $\mathbf{F}_{k|k-1}$  is the state transition matrix from epoch  $(k-1)$  to epoch  $k$ , and  $\boldsymbol{\omega}$  represents the process noise vector. Accordingly, the prediction step is implemented by:

$$\bar{\mathbf{x}}_k = \mathbf{F}_{k|k-1}\hat{\mathbf{x}}_{k-1} \tag{2}$$

$$\bar{\mathbf{P}}_k = \mathbf{F}_{k|k-1}\hat{\mathbf{P}}_{k-1}\mathbf{F}_{k|k-1}^T + \mathbf{Q}_k \tag{3}$$

where  $\mathbf{Q}$  represents the covariance of  $\boldsymbol{\omega}$ ,  $\bar{\mathbf{x}}$  and  $\hat{\mathbf{x}}$  are the predicted and updated state estimates in turn, and their error covariances are  $\bar{\mathbf{P}}$  and  $\hat{\mathbf{P}}$ , respectively.

In the update phase,  $\bar{\mathbf{x}}_k$  and  $\bar{\mathbf{P}}_k$  are updated to form  $\hat{\mathbf{x}}_k$  and  $\hat{\mathbf{P}}_k$  by incorporating the measurements at  $k$ . The measurement model is expressed by:

$$\mathbf{z}_k = \mathbf{H}_k\mathbf{x}_k + \boldsymbol{\nu}_k \tag{4}$$

where  $\mathbf{z}$  represents the  $n_z$ -dimensional measurement vector,  $\mathbf{H}$  is the measurement matrix, and  $\boldsymbol{\nu}$  is the measurement noise. The state and its error covariance are updated by:

$$\mathbf{K}_k = \bar{\mathbf{P}}_k\mathbf{H}_k^T(\mathbf{H}_k\bar{\mathbf{P}}_k\mathbf{H}_k^T + \mathbf{R}_k)^{-1} \tag{5}$$

$$\hat{\mathbf{x}}_k = \bar{\mathbf{x}}_k + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k\bar{\mathbf{x}}_k) \tag{6}$$

$$\hat{\mathbf{P}}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\bar{\mathbf{P}}_k(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T \tag{7}$$

where  $\mathbf{K}$  denotes the filter gain,  $\mathbf{R}$  is the measurement noise covariance, and  $\mathbf{I}$  is an identity matrix.

### Fault sources and their effects on state estimation

If a filter encounters faults, it will produce a biased state estimate. The faults may occur in state initialization, propagation, or update phases, as shown below:

$$\hat{\mathbf{x}}_0 = \mathbf{x}_0 + \tilde{\boldsymbol{\omega}} + \mathbf{f}_{S,0} \tag{8}$$

$$\mathbf{x}_k = \mathbf{F}_{k|k-1}\mathbf{x}_{k-1} + \boldsymbol{\omega}_k + \mathbf{f}_{S,k} \tag{9}$$

$$\mathbf{z}_k = \mathbf{H}_k\mathbf{x}_k + \boldsymbol{\nu}_k + \mathbf{f}_{M,k} \tag{10}$$

where  $\mathbf{f}_{S,0}$  is the initialization fault vector,  $\mathbf{f}_{S,k}$  denotes the system propagation fault vector, and  $\mathbf{f}_M$  is the *Measurement Fault* (MF) vector. Figure 1 summarizes the fault sources for a KF based on its equivalence with a batch Weighted LS (WLS) (Joerger and Pervan 2013).

Considering the faults above, the state estimate bias,  $\boldsymbol{\mu}$ , can be computed recursively by (Wang et al. 2020; Arana et al. 2019)

$$\boldsymbol{\mu}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{F}_{k|k-1}\boldsymbol{\mu}_{k-1} + (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{f}_{S,k} + \mathbf{K}_k\mathbf{f}_{M,k} \tag{11}$$

with  $\boldsymbol{\mu}_0 = \mathbf{f}_{S,0}$ . Since initialization faults and propagation faults affect the current estimate in similar ways, they are grouped to be the *System Fault* (SF),  $\mathbf{f}_S$ , in this work.

Prior work hardly considered initialization faults and propagation faults, but these faults may occur, such as in the following situations. (a) An initialization fault occurs when the initial state is incorrectly set. (b) For a vehicle with a low-cost IMU, IMU faults may occur (see Appendix A), which lead to propagation faults in GNSS/INS integration. (c) For a filter using aircraft dynamic models, it may encounter a propagation fault under strong wind disturbances.

### Revisiting the existing solution separation-based KFIM algorithm

This subsection presents the principles of the SS-KFIM algorithm based on the existing approaches (Gunning et al. 2018; Tanil et al. 2019) and the following assumptions.

**Assumption 1.** The state initialization and propagation processes are always fault-free.

**Assumption 2.** The basic assumptions of KFs are all satisfied under nominal conditions, and the noise covariances (i.e.,  $\mathbf{P}_0$ ,  $\mathbf{Q}$ , and  $\mathbf{R}$ ) are accurately known.

First, a group of terms are defined as follows. A fault mode,  $\mathbb{H}$ , hypothesizes the health status of each sensor. An example fault mode is given by

$$\mathbb{H}^{(j)} = \mathcal{M}_1\overline{\mathcal{M}}_2 \cdots \mathcal{M}_{n_s} \tag{12}$$

where  $j$  is the index of this fault mode,  $\mathcal{M}_i$  denotes the event that the  $i$ th sensor is healthy, and  $\overline{\mathcal{M}}_i$  represents the event that it is faulted.

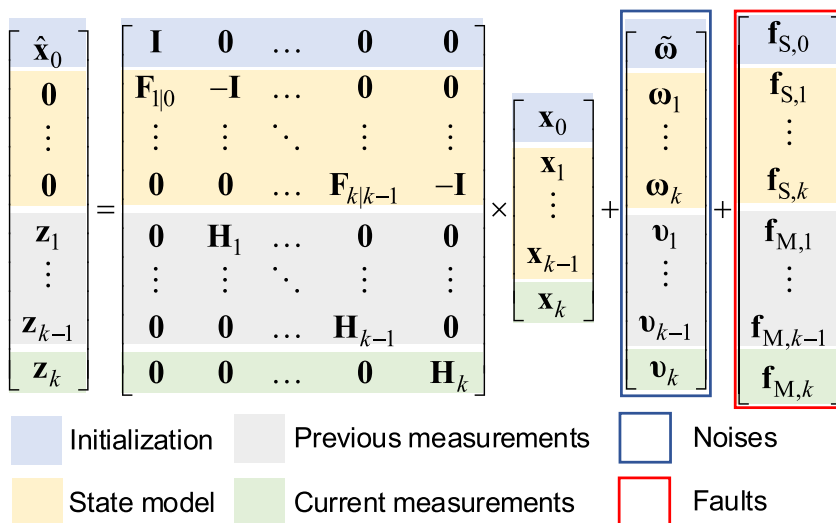
**Remark 1** The  $i$ th sensor is considered faulted once if it is faulted at any epoch during operation, i.e., the  $i$ th entry of  $\mathbf{f}_{M,k}$  is nonzero for any  $k$ . This is justified because (a) a fault usually lasts for a period and (b) prior faults can affect current estimates.

A fault mode is associated with a prior probability  $p$ , a sensor subset  $\mathbb{S}$ , and a filter  $F$ .  $p^{(j)}$  is the probability of  $\mathbb{H}^{(j)}$  being true,  $\mathbb{S}^{(j)}$  contains the healthy sensors under  $\mathbb{H}^{(j)}$ , and  $F^{(j)}$  uses the sensors in  $\mathbb{S}^{(j)}$  to perform state estimation.  $j = 0$  denotes the all-in-view mode; thus,  $F^{(0)}$  is the *main filter* and the others are *subfilters*.

The input parameters and key steps of the SS-KFIM algorithm are described below. The input parameters include two categories: Table 1 shows the parameters describing the error and fault characteristics, and Table 2 gives the navigation performance requirements. In Table 2,  $\mathbb{q}$  denotes the set of indexes that correspond to the states of interest.

Figure 2 presents the flowchart of the basic SS-KFIM algorithm. The first step is to determine the fault modes that need monitoring. One possible approach to accomplish this step is provided below.

**Fig. 1** Graphic illustration of the fault sources for a KF in a batch WLS form



**Table 1** Parameters describing the error and fault characteristics

| Category | Parameter            | Description                          |
|----------|----------------------|--------------------------------------|
| Noise    | <b>Q</b>             | Process noise covariance             |
|          | <b>R</b>             | Measurement noise covariance         |
|          | <b>P<sub>0</sub></b> | Initial error covariance             |
| Fault    | <b>p</b>             | Prior probability of each fault mode |

**Table 2** Parameters derived from the performance requirements

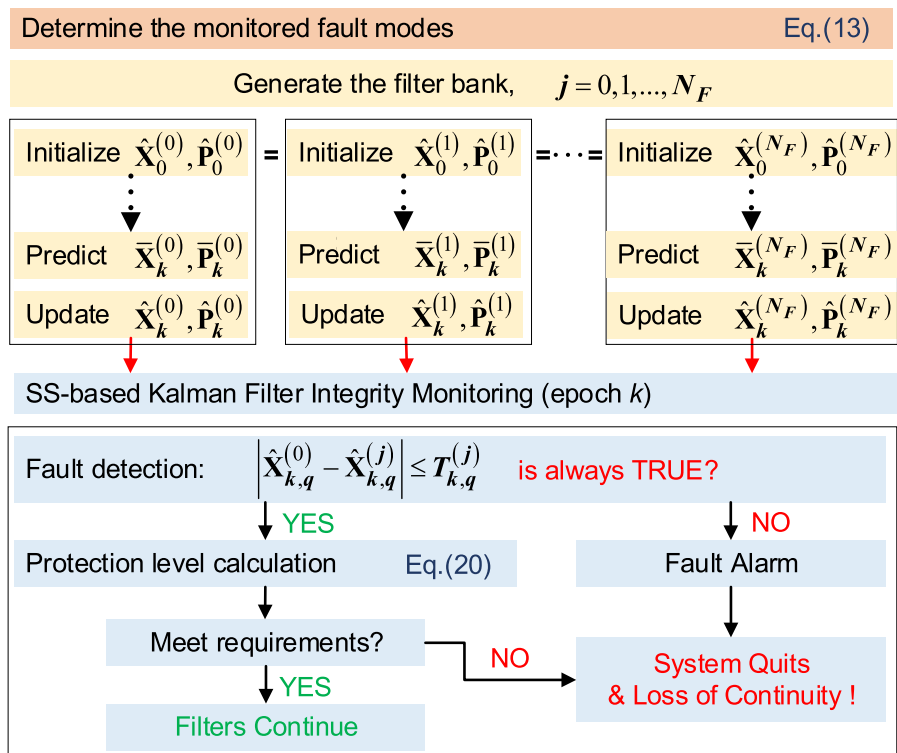
| Category   | Parameter                | Description  |
|------------|--------------------------|--|
| Integrity  | <i>P<sub>HMI</sub></i>   | Total integrity budget   |
|            | <i>P<sub>HML,q</sub></i> | Integrity budget for the <i>q</i> th state, <i>q</i> ∈ $\mathbb{Q}$      |
|            | <i>P<sub>THRES</sub></i> | Threshold for the integrity risk coming from the unmonitored fault modes |
| Continuity | <i>P<sub>FA</sub></i>    | Total continuity budget for false alarm                                  |
|            | <i>P<sub>FA,q</sub></i>  | Continuity risk for the <i>q</i> th state, <i>q</i> ∈ $\mathbb{Q}$       |

It is sufficient to only monitor a subset of the fault modes,  $\{\mathbb{H}^{(j)} | j = 0, 1, \dots, N_F\}$ , when we have:

$$P_{THRES} \geq P_{NM} = 1 - \sum_{j=0}^{N_F} p^{(j)} \tag{13}$$

where *j* indexes the fault mode in descending order of probability, and *P<sub>NM</sub>* is the probability of unmonitored modes. To make it clearer, Fig. 3 gives an example of the monitored fault modes and the unmonitored ones.

**Fig. 2** Flowchart of the existing solution separation-based KFIM algorithm



The second step is constructing the filter bank, which consists of the filters for each monitored fault mode, i.e.,  $\{\mathbb{F}^{(j)} | 0 \leq j \leq N_F\}$ . At each epoch, the FD test statistics are determined by:

$$\Delta \hat{\mathbf{x}}_q^{(j)} = \hat{\mathbf{x}}_q^{(0)} - \hat{\mathbf{x}}_q^{(j)}, \quad 1 \leq j \leq N_F, q \in \mathbb{Q} \tag{14}$$

where  $\hat{\mathbf{x}}_q^{(0)}$  and  $\hat{\mathbf{x}}_q^{(j)}$  are the estimates of the *q* th state from  $\mathbb{F}^{(0)}$  and  $\mathbb{F}^{(j)}$ , respectively. And their error standard deviations are computed by:

$$\sigma_q^{(0)} = \sqrt{\hat{\mathbf{P}}_{\langle q,q \rangle}^{(0)}}, \sigma_q^{(j)} = \sqrt{\hat{\mathbf{P}}_{\langle q,q \rangle}^{(j)}} \tag{15}$$

where  $\mathbf{A}_{\langle m,n \rangle}$  denotes the *m* th-row *n* th-column element of matrix **A**.

Under Assumption 2, the standard deviation for  $\Delta \hat{\mathbf{x}}_q^{(j)}$  is given by (Gunning et al. 2018):

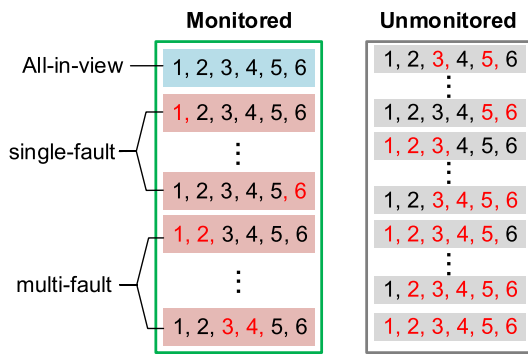
$$\sigma_{ss,q}^{(j)} = \sqrt{\hat{\mathbf{P}}_{\langle q,q \rangle}^{(j)} - \hat{\mathbf{P}}_{\langle q,q \rangle}^{(0)}} \tag{16}$$

This equation is derived based on the batch WLS form of a KF. Then the corresponding test threshold is computed as:

$$T_q^{(j)} = \mathcal{Q}(P_{FA,q}/2N_F) \cdot \sigma_{ss,q}^{(j)} \tag{17}$$

where  $\mathcal{Q}(p)$  is the (1-*p*) quantile of a zero-mean unit-variance Gaussian distribution.

The navigation system is declared to be healthy only if we have



**Fig. 3** Example of monitored and unmonitored fault modes. Each filled rectangle is a fault mode, where black numbers indicate used sensors and red numbers indicate excluded ones

$$|\Delta \hat{\mathbf{x}}_q^{(j)}| \leq T_q^{(j)}, \forall j \in \{j | 1 \leq j \leq N_F\}, \forall q \in \mathcal{Q} \quad (18)$$

If any test fails, the user will be warned that the navigation system is faulted.

The final step is evaluating the PLs, which will be performed only if there is no fault alert. The PL is a statistical error bound computed to guarantee that the probability of the estimation error exceeding this number is smaller than the integrity risk requirement,  $P_{HMI}$ . Mathematically, the PL for the  $q$  th state ( $q \in \mathcal{Q}_M$ ) is defined as:

$$P_{HMI,q} = IR_q = \mathcal{P}\left(\left|\hat{\mathbf{x}}_q^{(0)} - \mathbf{x}_q\right| > PL_q, \bar{D}\right) \quad (19)$$

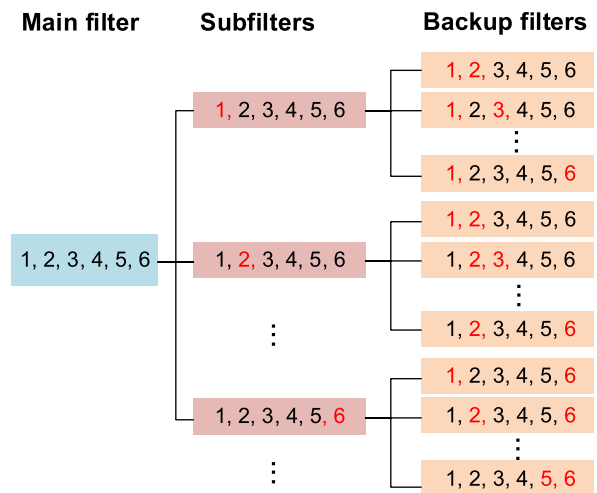
where  $\mathcal{P}(\blacksquare)$  is the probability of the event  $\blacksquare$  occurring, and  $\bar{D}$  denotes the event that there is no fault alert. According to ARAIM,  $PL_q$  can be computed by solving the following equation with a half-interval search (Blanch et al. 2015):

$$P_{HMI,q} - \frac{P_{HMI,q}}{P_{HMI}} P_{NM} = 2\mathcal{Q}\left(\frac{PL_q}{\sigma_q^{(0)}}\right) + \sum_{j=1}^{N_F} \mathcal{Q}\left(\frac{PL_q - T_q^{(j)}}{\sigma_q^{(j)}}\right) \cdot p^{(j)} \quad (20)$$

**Remark 2** Equation (16) is valid only if the all-in-view estimator is optimal in a least-squares sense, whereas the other equations in this subsection are independent of the estimator.

### Filter bank management for handling sensor-in and sensor-out events

Filter bank management, also called subset management, is a key function in the SS-KFIM algorithm. It aims at adding and/or removing filters from the filter bank if (a) the measurements of a new sensor arrive, called a sensor-in event, or (b) a sensor in use suddenly becomes unavailable, namely



**Fig. 4** Example structure of a filter bank for SS-KFIM FDE

a sensor-out event. Using GNSS/INS tight integration as an example, a sensor-in event is said to occur when a new satellite becomes visible, and conversely, the case that a satellite in view is suddenly blocked is called a sensor-out event.

Gunning et al. (2019a, 2019b) and Tanil et al. (2019) developed several subset management strategies for SS-KFIM FDE, in which a main filter, multiple subfilters, and many backup filters are run simultaneously, as shown in Fig. 4. However, if FE is not enabled, there is no need to always run a large number of backup filters. Therefore, a modified subset management strategy is presented below to adapt the FD-only SS-KFIM algorithm. This strategy can reduce the number of filters while maintaining low PLs.

### Filter bank management for a new sensor

When there is a new sensor, actions must be taken to monitor the fault modes associated with this sensor. In Fig. 5, we illustrate and compare three subset management strategies for this situation by simulating a sensor-in event at epoch  $k$ .

In *Strategy I* (Tanil et al. 2019), a “son” filter and a “daughter” filter are branched out from each of the original filters by inheriting its parameters. From epoch  $k$ , the “son” filter will use the new sensor while the “daughter” filter will not. This strategy ensures that  $P_{NM}$  will not change before and after epoch  $k$  because we have:

$$\mathcal{P}(\mathbb{H}^{(j)}) = \mathcal{P}(\mathbb{H}^{(j)} \mathcal{M}_{\text{new}}) + \mathcal{P}(\mathbb{H}^{(j)} \bar{\mathcal{M}}_{\text{new}}), \forall j > N_F \quad (21)$$

where  $\mathcal{M}_{\text{new}}$  denotes the event that the new sensor is healthy, and  $\mathbb{H}^{(j)}$  is the  $j$  th fault mode in the original filter bank. However, this strategy will double the number of filters.

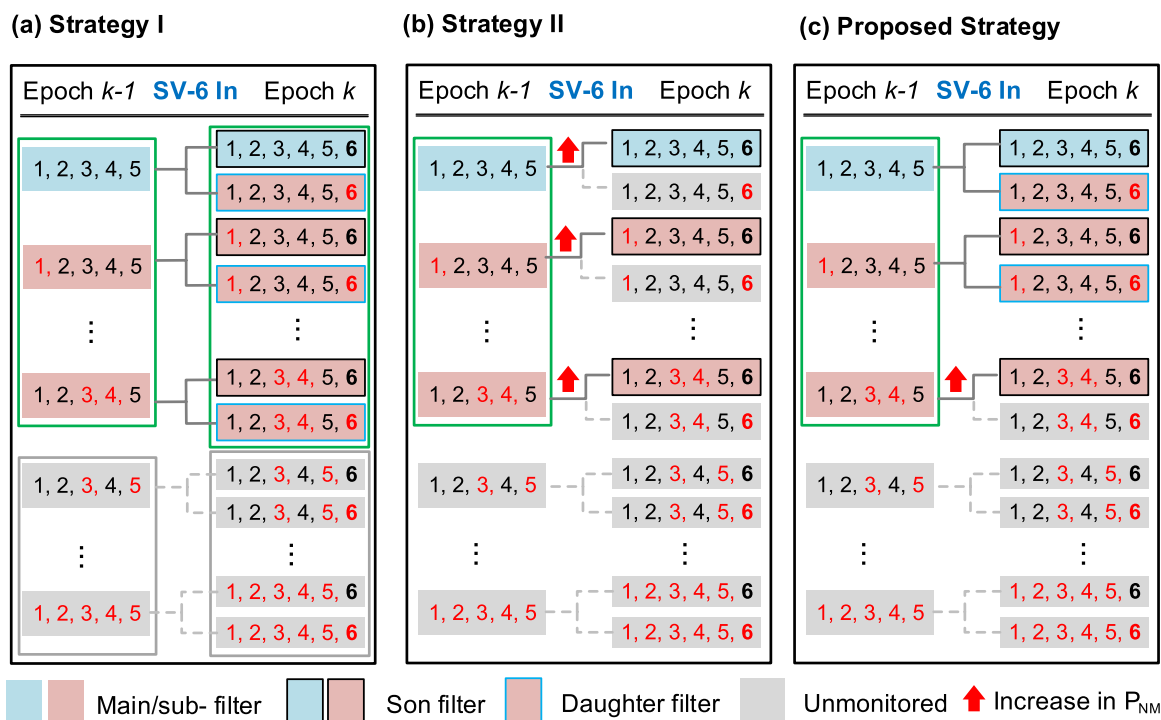


Fig. 5 Different filter bank management strategies to cope with a sensor-in event

Unlike *Strategy I*, *Strategy II* keeps the number of filters unchanged through simply adding the new sensor to each of the original filters. However, this will sharply increase  $P_{NM}$  because all the fault modes associated with the new sensor are unmonitored. Assuming that the health status of the new sensor is independent of those of other sensors, we can compute the increase in  $P_{NM}$  by:

$$\begin{aligned}
 P_{NM,add} &= \sum_{j=0}^{N_f} \mathcal{P}(\mathbb{H}^{(j)} \overline{\mathcal{M}}_{new}) \\
 &= \mathcal{P}(\overline{\mathcal{M}}_{new}) \times \sum_{j=0}^{N_f} \mathbb{P}^{(j)} > \mathcal{P}(\overline{\mathcal{M}}_{new}) \cdot \mathbb{P}^{(0)}
 \end{aligned}
 \tag{22}$$

Therefore, this strategy is usually not feasible because it may make  $P_{NM}$  exceed  $P_{THRES}$ .

The analysis above indicates that (a) running all the *daughter* filters is computationally expensive and (b) removing all the *daughter* filters may make  $P_{NM}$  exceed  $P_{THRES}$ . Therefore, a modified strategy is proposed to overcome these shortcomings. The basic idea is adding a subset of *daughter* filters to the filter bank given by *Strategy II* to ensure  $P_{NM}$  is smaller than  $P_{THRES}$ . Given that the fault modes are sorted in descending order of probability, it is sufficient to only add the *daughter* filters that correspond to the first  $N_d$  filters in the original filter bank when we have:

$$P_{NM} + P_{NM,add} = P_{NM} + \mathcal{P}(\overline{\mathcal{M}}_{new}) \times \sum_{j=N_d}^{N_f} \mathbb{P}^{(j)} < P_{THRES}
 \tag{23}$$

Compared with *Strategy I*, this strategy can reduce the number of filters by  $(N_f - N_d + 1)$ .

### Filter bank management for a sensor-out event

Subset management is also required when a previously used sensor becomes unavailable. Figure 6 shows an example of a sensor-out event and illustrates two subset management strategies for this event. After a sensor becomes unavailable, its previous measurements will still influence the filters that ever used it. Therefore, the fault modes associated with this sensor still need monitoring. An approach to achieve this would be to keep the filter bank unchanged, as *Strategy I* does. However, this strategy has a disadvantage in that the number of filters will never decrease even if a sensor has been unavailable for a long time.

To address this, we present a three-step subset management strategy through modifying the strategies proposed by Gunning et al. (2019a, 2019b) and Tanil et al. (2019). Compared with existing approaches, this strategy is more compatible with the FD-only SS-KFIM algorithm. Besides, it considers multi-fault cases and evaluates the effect of subset management on  $P_{NM}$ . Our strategy is illustrated as follows.



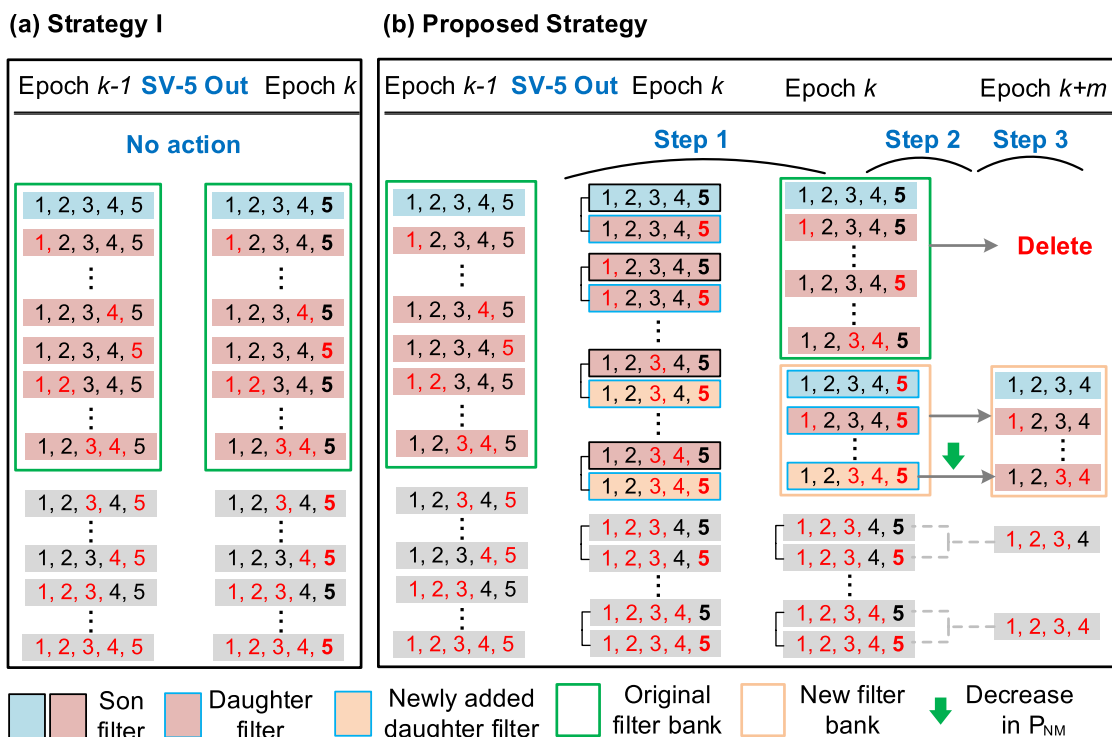


Fig. 6 Different filter bank management strategies to handle a sensor-out event

When a sensor becomes unavailable, the first step is adding some new filters and generating a new filter bank. For simplicity, the filters that ever used the lost sensor are called *son* filters and the others are *daughter* filters. In Fig. 6, the original filter bank consists of *son-daughter* filter pairs and individual *son* filters. For each individual *son* filter, we add the corresponding *daughter* filter through filter initialization. All the *daughter* filters form a new filter bank that is unaffected by the lost sensor. The new filter bank includes fewer filters and experiences lower  $P_{NM}$  than the original one. Specifically, the decrease in  $P_{NM}$  is given by:

$$P_{NM,dec} = \sum_{j \in \mathbb{J}_{ad}} \mathcal{P} \left( \mathbb{H}_n^{(j)} \overline{\mathcal{M}}_{out} \right) \tag{24}$$

where  $\mathbb{J}_{ad}$  includes the indexes of the newly added filters in the new filter bank,  $\mathbb{H}_n$  is the associated fault mode, and  $\overline{\mathcal{M}}_{out}$  denotes the event that the lost sensor was ever faulted before.

In the second step, two sets of PLs are, respectively, computed based on the original filter bank and the new one. Because the new filter bank includes some newly initialized filters, the corresponding PLs will be higher than those from the original filter bank. Therefore, at this stage, the original filter bank serves as the primary filter bank that outputs the PLs to the users. Finally, the third step is triggered when the two sets of PLs are sufficiently close. In this step, the original

filter bank is removed, and the new one becomes the primary filter bank.

Special attention is needed for the situations where another sensor-in or sensor-out event occurs before the third step is triggered. Preliminary strategies to address them are provided below. If a new sensor becomes available, we can apply the proposed strategy in Fig. 5 to both the original and the new filter banks. If the lost sensor recovers, directly removing the new filter bank is a feasible method. And for the situation where another sensor becomes unavailable, a straightforward approach would be to follow an event-by-event strategy, i.e., to cope with the new sensor-out event after finishing handling the current one (Tanil et al. 2019).

### Considering the faults in state initialization and propagation phases

The existing SS-KFIM algorithm only considers the faults in the measurement-update phase. However, the filter may also encounter the faults in state initialization and propagation phases. Therefore, the existing algorithm is modified as follows to accommodate all sources of faults.

First, the *separability* of the system model in (1) is discussed. A system model is called *separable* if it can be rewritten as:

$$\begin{bmatrix} \hat{\mathbf{x}}_{1,0} \\ \hat{\mathbf{x}}_{2,0} \\ \vdots \\ \hat{\mathbf{x}}_{c,0} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{1,0} \\ \mathbf{x}_{2,0} \\ \vdots \\ \mathbf{x}_{c,0} \end{bmatrix} + \begin{bmatrix} \tilde{\boldsymbol{\omega}}_1 \\ \tilde{\boldsymbol{\omega}}_2 \\ \vdots \\ \tilde{\boldsymbol{\omega}}_c \end{bmatrix} \tag{25}$$

$$\begin{bmatrix} \mathbf{x}_{1,k} \\ \mathbf{x}_{2,k} \\ \vdots \\ \mathbf{x}_{c,k} \end{bmatrix} = \begin{bmatrix} \mathbf{F}_{1,k|k-1} & & & \\ & \mathbf{F}_{2,k|k-1} & & \\ & & \ddots & \\ & & & \mathbf{F}_{c,k|k-1} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{1,k-1} \\ \mathbf{x}_{2,k-1} \\ \vdots \\ \mathbf{x}_{c,k-1} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\omega}_{1,k} \\ \boldsymbol{\omega}_{2,k} \\ \vdots \\ \boldsymbol{\omega}_{c,k} \end{bmatrix} \tag{26}$$

where  $c$  denotes the number of independent subsystems. If a system model is separable, a system fault may only affect one of its subsystems, and this kind of system fault is called a *subsystem fault*. Conversely, if the system model is inseparable (i.e.,  $c = 1$ ), all the states will be influenced by the system fault. Taking system faults into account, an example fault mode is given by:

$$\mathbb{H}^{(j)} = (\bar{\mathcal{S}}_1 \mathcal{S}_2 \cdots \mathcal{S}_c) (\mathcal{M}_1 \bar{\mathcal{M}}_2 \cdots \mathcal{M}_{n_z}) \tag{27}$$

where  $\mathcal{S}_i$  is the event that the  $i$ th subsystem is healthy and  $\bar{\mathcal{S}}_i$  is its opposite. Note that a subsystem is considered faulted once if it is faulted in state initialization and/or propagation phases at any epoch during operation.

For different fault modes, the associated state estimation models to compute the fault-tolerant solutions may vary. Figure 7 illustrates the state estimation models for various types of fault modes, and the basic rules are summarized as follows. If a measurement fault exists in  $\mathbb{H}^{(j)}$ , the faulted sensor(s) will be excluded by the subfilter  $\mathbb{F}^{(j)}$ ; if  $\mathbb{H}^{(j)}$  includes a (sub)system fault,  $\mathbb{F}^{(j)}$  will discard the corresponding (sub)system model.

Then we present a generalized KF architecture to accommodate different types of subfilters. First, let us rewrite the KF update equations as follows (Simon 2006):

$$\hat{\mathbf{P}}_k = \left( \bar{\mathbf{P}}_k^{-1} + \mathbf{H}_k^T \mathbf{R}_k^{-1} \mathbf{H}_k \right)^{-1} \tag{28}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{P}}_k \bar{\mathbf{P}}_k^{-1} \cdot \bar{\mathbf{x}}_k + \hat{\mathbf{P}}_k \mathbf{H}_k^T \mathbf{R}_k^{-1} \cdot \mathbf{z}_k \tag{29}$$

Based on this, the generalized subfilter equations are given by:

$$\bar{\mathbf{x}}_k^{(j)} = \mathbf{F}_{k|k-1} \hat{\mathbf{x}}_{k-1}^{(j)} \tag{30}$$

$$\bar{\mathbf{P}}_k^{(j)} = \mathbf{F}_{k|k-1} \hat{\mathbf{P}}_{k-1}^{(j)} \mathbf{F}_{k|k-1}^T + \mathbf{Q}_k \tag{31}$$

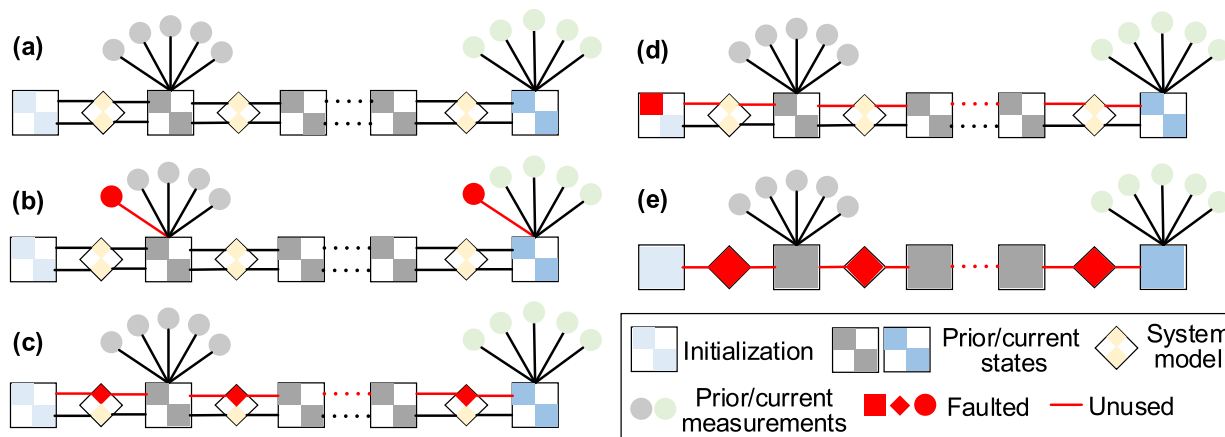
$$\hat{\mathbf{P}}_k^{(j)} = \left( \left( \mathbf{W}_S^{(j)} \bar{\mathbf{P}}_k^{(j)} \mathbf{W}_S^{(j)T} \right)^\dagger + \mathbf{H}_k^T \mathbf{W}_{M,k}^{(j)} \mathbf{H}_k \right)^\dagger \tag{32}$$

$$\hat{\mathbf{x}}_k^{(j)} = \hat{\mathbf{P}}_k^{(j)} \left( \mathbf{W}_S^{(j)} \bar{\mathbf{P}}_k^{(j)} \mathbf{W}_S^{(j)T} \right)^\dagger \cdot \bar{\mathbf{x}}_k^{(j)} + \hat{\mathbf{P}}_k^{(j)} \mathbf{H}_k^T \mathbf{W}_{M,k}^{(j)} \cdot \mathbf{z}_k \tag{33}$$

where  $\dagger$  represents the pseudoinverse operator (see Appendix B). Remark 3 explains why pseudoinverse operators are used here.  $\mathbf{W}_S^{(j)}$  is a diagonal matrix determined by:

$$\mathbf{W}_S^{(j)} = \mathbf{O} \left[ \mathbb{i}_{S,F}^{(j)}, n_x \right] \cdot \mathbf{I} \cdot \left( \mathbf{O} \left[ \mathbb{i}_{S,F}^{(j)}, n_x \right] \right)^T \tag{34}$$

where  $\mathbb{i}_{S,F}$  includes the indexes of those states associated with the faulted subsystems.  $\mathbf{O}[\mathbb{i}, n]$  is an operator for  $n$ -row matrices, and it sets all rows whose indexes belong to  $\mathbb{i}$  to  $\mathbf{0}$ . Thus,



**Fig. 7** Illustration of the state estimation models for different fault modes: **a** fault-free; **b** measurement fault, **c** subsystem fault in the propagation process; **d** subsystem fault caused by incorrect state initialization; **e** system fault in the propagation phase. Examples (a)~(d) correspond to a separable system model, while (e) is associated with an inseparable one



$$\begin{aligned} \left(\mathbf{W}_S^{(j)}\right)_{\langle q,q \rangle} &= 1, \text{ if the } q\text{th state belongs to a fault – free subsystem} \\ \left(\mathbf{W}_S^{(j)}\right)_{\langle q,q \rangle} &= 0, \text{ otherwise} \end{aligned} \tag{35}$$

Similarly,  $\mathbf{W}_M^{(j)}$  is determined by:

$$\mathbf{W}_M^{(j)} = \mathbf{O} \left[ \mathbb{i}_{M,F}^{(j)}, n_z \right] \cdot \mathbf{R}^{-1} \cdot \left( \mathbf{O} \left[ \mathbb{i}_{M,F}^{(j)}, n_z \right] \right)^T \tag{36}$$

where  $\mathbb{i}_{M,F}$  includes the indexes of the faulted sensors.

The generalized KF architecture can also apply to the case where the system model is fault-free and the case where it is completely faulted. In the former case, (32) and (33) become:

$$\hat{\mathbf{P}}_k^{(j)} = \left( \left( \bar{\mathbf{P}}_k^{(j)} \right)^{-1} + \mathbf{H}_k^T \mathbf{W}_{M,k}^{(j)} \mathbf{H}_k \right)^{-1} \tag{37}$$

$$\hat{\mathbf{x}}_k^{(j)} = \hat{\mathbf{P}}_k^{(j)} \left( \bar{\mathbf{P}}_k^{(j)} \right)^{-1} \cdot \bar{\mathbf{x}}_k^{(j)} + \hat{\mathbf{P}}_k^{(j)} \mathbf{H}_k^T \mathbf{W}_{M,k}^{(j)} \cdot \mathbf{z}_k \tag{38}$$

In the latter case, the filter becomes a snapshot WLS:

$$\hat{\mathbf{P}}_k^{(j)} = \left( \mathbf{H}_k^T \mathbf{W}_{M,k}^{(j)} \mathbf{H}_k \right)^\dagger \tag{39}$$

$$\hat{\mathbf{x}}_k^{(j)} = \hat{\mathbf{P}}_k^{(j)} \mathbf{H}_k^T \mathbf{W}_{M,k}^{(j)} \cdot \mathbf{z}_k \tag{40}$$

**Remark 3 (Usage of pseudoinverse)** Pseudoinverse operators are applied to (32), (33) and (39) because the corresponding matrices are singular. Detailed explanations are given below.

If there is a (sub)system fault,  $\left( \mathbf{W}_S^{(j)} \bar{\mathbf{P}}_k^{(j)} \mathbf{W}_S^{(j)T} \right)$  will become singular. Without loss of generality, let us assume that the last subsystem is faulted. In this case,  $\left( \mathbf{W}_S^{(j)} \bar{\mathbf{P}}_k^{(j)} \mathbf{W}_S^{(j)T} \right)$  can be rewritten as:

$$\left( \mathbf{W}_S^{(j)} \bar{\mathbf{P}}_k^{(j)} \mathbf{W}_S^{(j)T} \right) = \begin{bmatrix} \mathbf{P}_h & \\ & 0 \end{bmatrix} \tag{41}$$

where  $\mathbf{P}_h$  is the covariance matrix of the states associated with the healthy subsystems. This yields:

$$\left( \mathbf{W}_S^{(j)} \bar{\mathbf{P}}_k^{(j)} \mathbf{W}_S^{(j)T} \right)^\dagger = \begin{bmatrix} \left( \mathbf{P}_h \right)^{-1} & \\ & 0 \end{bmatrix} \tag{42}$$

In this way, the state estimates from  $\mathbb{F}^{(j)}$  will not be affected by the subsystem fault.

Similarly,  $\left( \left( \mathbf{W}_S^{(j)} \bar{\mathbf{P}}_k^{(j)} \mathbf{W}_S^{(j)T} \right)^\dagger + \mathbf{H}_k^T \mathbf{W}_{M,k}^{(j)} \mathbf{H}_k \right)$  and  $\left( \mathbf{H}_k^T \mathbf{W}_{M,k}^{(j)} \mathbf{H}_k \right)$  may also be singular. This implies that some states cannot be estimated by  $\mathbb{F}^{(j)}$ , and thus a pseudoinverse

operator is used to obtain the covariance of the *estimable* states.

**Remark 4 (Estimability of the states)** For a subfilter, some states cannot be estimated. Specifically, the  $q$  th state is not estimable if the  $q$  th diagonal entry of  $\hat{\mathbf{P}}^{(j)}$  is zero.

**Remark 5 (Unobservable fault modes)** If any state of interest cannot be estimated by  $\mathbb{F}^{(j)}$ , one should remove  $\mathbb{H}^{(j)}$  from the list of monitored fault modes and add  $\mathbb{p}^{(j)}$  to  $P_{NM}$ .

### Addressing the situations where the all-in-view estimator is nonoptimal

According to Remark 2, the existing SS-KFIM algorithm is not applicable to the cases where the all-in-view estimator is not optimal in a least-squares sense. However, these cases may occur, such as in the following situations. (a) The error models used for continuity evaluation are different from those used for integrity. (b) A forgetting factor-based KF or an adaptive KF is used instead of the standard one. (c) Users adopt a self-tuning filter gain instead of the optimal one. Although ARAIM considers the first case (Blanch et al. 2015), the associated method only applies to snapshot WLS and cannot be used here.

To address these situations, we propose a new method to compute the solution separation covariance. For both optimal and nonoptimal filters, state prediction is performed based on (30), and state update is formulated as:

$$\hat{\mathbf{x}}_k^{(j)} = \mathbf{W}_S^{(j)} \bar{\mathbf{x}}_k^{(j)} + \mathbf{B}_k^{(j)} \cdot \left( \mathbf{z}_k - \mathbf{H}_k \mathbf{W}_S^{(j)} \bar{\mathbf{x}}_k^{(j)} \right), 0 \leq j \leq N_F \tag{43}$$

where  $\mathbf{W}_S^{(j)}$  is defined in (34), and  $\mathbf{B}$  is the filter gain. The determination of  $\mathbf{B}$  depends on the filter algorithm and the error models. Then the state errors are expressed by:

$$\bar{\boldsymbol{\epsilon}}_k^{(j)} = \mathbf{F}_{k|k-1} \hat{\boldsymbol{\epsilon}}_{k-1}^{(j)} + \boldsymbol{\omega}_k \tag{44}$$

$$\hat{\boldsymbol{\epsilon}}_k^{(j)} = \left( \mathbf{I} - \mathbf{B}_k^{(j)} \mathbf{H}_k \right) \mathbf{W}_S^{(j)} \bar{\boldsymbol{\epsilon}}_k^{(j)} + \mathbf{B}_k^{(j)} \boldsymbol{\nu}_k \triangleq \mathbf{A}_k^{(j)} \bar{\boldsymbol{\epsilon}}_k^{(j)} + \mathbf{B}_k^{(j)} \boldsymbol{\nu}_k \tag{45}$$

where  $\hat{\epsilon}$  and  $\bar{\epsilon}$  are the errors of  $\hat{\mathbf{x}}$  and  $\bar{\mathbf{x}}$ , respectively. Therefore, the state error covariances are recursively computed by:

$${}^c\hat{\mathbf{P}}_k^{(j)} = \mathbf{F}_{k|k-1} {}^c\hat{\mathbf{P}}_{k-1}^{(j)} \mathbf{F}_{k|k-1}^T + {}^c\mathbf{Q}_k \tag{46}$$

$${}^c\hat{\mathbf{P}}_k^{(j)} = \mathbf{A}_k^{(j)} {}^c\hat{\mathbf{P}}_k^{(j)} \mathbf{A}_k^{(j)T} + \mathbf{B}_k^{(j)} {}^c\mathbf{R}_k \mathbf{B}_k^{(j)T} \tag{47}$$

where the left superscript ‘‘c’’ indicates the process, measurement, and state error covariances for continuity evaluation. If the error models  $\{{}^c\mathbf{Q}_k, {}^c\mathbf{R}_k\}$  are not the same as  $\{\mathbf{Q}_k, \mathbf{R}_k\}$ , the state error covariances  $\{{}^c\hat{\mathbf{P}}_k, {}^c\hat{\mathbf{P}}_k\}$  will also be different from  $\{\hat{\mathbf{P}}_k, \hat{\mathbf{P}}_k\}$ .

**Remark 6** A filter is said to be nonoptimal if the gain  $\mathbf{B}_k$  does not minimize the unconditional state error covariance, i.e.,  ${}^c\hat{\mathbf{P}}_k$ , for any  $k$ .

The solution separation covariance is defined by:

$$\hat{\mathbf{P}}_{ss,k}^{(j)} \triangleq \text{cov}\left(\hat{\mathbf{x}}_k^{(0)} - \hat{\mathbf{x}}_k^{(j)}, \hat{\mathbf{x}}_k^{(0)} - \hat{\mathbf{x}}_k^{(j)}\right) \tag{48}$$

where cov denotes covariance. According to (45), we have:

$$\hat{\mathbf{x}}_k^{(0)} - \hat{\mathbf{x}}_k^{(j)} = \hat{\epsilon}_k^{(0)} - \hat{\epsilon}_k^{(j)} = \mathbf{A}_k^{(0)} \cdot \bar{\epsilon}_k^{(0)} - \mathbf{A}_k^{(j)} \cdot \bar{\epsilon}_k^{(j)} + (\mathbf{B}_k^{(0)} - \mathbf{B}_k^{(j)}) \cdot \nu_k \tag{49}$$

The covariance of  $\mathbf{f}(\bar{\epsilon}_k) \triangleq (\mathbf{A}_k^{(0)} \cdot \bar{\epsilon}_k^{(0)} - \mathbf{A}_k^{(j)} \cdot \bar{\epsilon}_k^{(j)})$  is computed by:

$$\begin{aligned} \text{cov}(\mathbf{f}(\bar{\epsilon}_k), \mathbf{f}(\bar{\epsilon}_k)) &= \mathbf{A}_k^{(0)} {}^c\hat{\mathbf{P}}_k^{(0)} \mathbf{A}_k^{(0)T} + \mathbf{A}_k^{(j)} {}^c\hat{\mathbf{P}}_k^{(j)} \mathbf{A}_k^{(j)T} \\ &\quad - \mathbf{A}_k^{(0)} \text{cov}(\bar{\epsilon}_k^{(0)}, \bar{\epsilon}_k^{(j)}) \mathbf{A}_k^{(j)T} - \mathbf{A}_k^{(j)} \text{cov}(\bar{\epsilon}_k^{(0)}, \bar{\epsilon}_k^{(j)}) \mathbf{A}_k^{(0)T} \end{aligned} \tag{50}$$

with

$$\begin{aligned} \text{cov}(\bar{\epsilon}_k^{(0)}, \bar{\epsilon}_k^{(j)}) &= \text{cov}\left(\mathbf{F}_{k|k-1} \hat{\epsilon}_{k-1}^{(0)} + \omega_k, \mathbf{F}_{k|k-1} \hat{\epsilon}_{k-1}^{(j)} + \omega_k\right) \\ &= \mathbf{F}_{k|k-1} \text{cov}\left(\hat{\epsilon}_{k-1}^{(0)}, \hat{\epsilon}_{k-1}^{(j)}\right) \mathbf{F}_{k|k-1}^T + {}^c\mathbf{Q}_k \end{aligned} \tag{51}$$

$$\text{cov}(\hat{\epsilon}_{k-1}^{(0)}, \hat{\epsilon}_{k-1}^{(j)}) = \frac{1}{2} \left( {}^c\hat{\mathbf{P}}_{k-1}^{(0)} + {}^c\hat{\mathbf{P}}_{k-1}^{(j)} - \hat{\mathbf{P}}_{ss,k-1}^{(j)} \right) \tag{52}$$

Note that (52) can be proved by:

$$\begin{aligned} \hat{\mathbf{P}}_{ss,k-1}^{(j)} &= \text{cov}\left(\hat{\epsilon}_{k-1}^{(0)} - \hat{\epsilon}_{k-1}^{(j)}, \hat{\epsilon}_{k-1}^{(0)} - \hat{\epsilon}_{k-1}^{(j)}\right) \\ &= {}^c\hat{\mathbf{P}}_{k-1}^{(0)} + {}^c\hat{\mathbf{P}}_{k-1}^{(j)} - 2\text{cov}\left(\hat{\epsilon}_{k-1}^{(0)}, \hat{\epsilon}_{k-1}^{(j)}\right) \end{aligned} \tag{53}$$

Then, the covariance of  $\mathbf{g}(\nu_k) \triangleq (\mathbf{B}_k^{(0)} - \mathbf{B}_k^{(j)}) \cdot \nu_k$  is given by:

$$\text{cov}(\mathbf{g}(\nu_k), \mathbf{g}(\nu_k)) = (\mathbf{B}_k^{(0)} - \mathbf{B}_k^{(j)}) {}^c\mathbf{R}_k (\mathbf{B}_k^{(0)} - \mathbf{B}_k^{(j)})^T \tag{54}$$

Given that  $\bar{\epsilon}_k$  and  $\nu_k$  are independent of each other,  $\hat{\mathbf{P}}_{ss,k}^{(j)}$  can be finally computed by:

$$\begin{aligned} \hat{\mathbf{P}}_{ss,k}^{(j)} &= \text{cov}(\mathbf{f}(\bar{\epsilon}_k), \mathbf{f}(\bar{\epsilon}_k)) + \text{cov}(\mathbf{g}(\nu_k), \mathbf{g}(\nu_k)) \\ &= {}^c\hat{\mathbf{P}}_k^{(0)} + {}^c\hat{\mathbf{P}}_k^{(j)} - \mathbf{A}_k^{(0)} \text{cov}(\bar{\epsilon}_k^{(0)}, \bar{\epsilon}_k^{(j)}) \mathbf{A}_k^{(j)T} \\ &\quad - \mathbf{A}_k^{(j)} \text{cov}(\bar{\epsilon}_k^{(0)}, \bar{\epsilon}_k^{(j)}) \mathbf{A}_k^{(0)T} - \mathbf{B}_k^{(0)} {}^c\mathbf{R}_k \mathbf{B}_k^{(j)T} - \mathbf{B}_k^{(j)} {}^c\mathbf{R}_k \mathbf{B}_k^{(0)T} \end{aligned} \tag{55}$$

Together with (51) and (52), this equation gives a recursive method to compute the solution separation covariance when the all-in-view filter is nonoptimal.

### Performance analysis with an illustrative example

Simulations are conducted around an example application of precision navigation for UAM approach to demonstrate our implementation of the SS-KFIM algorithm. Protection level evaluation is critical in this application. Besides, this application poses stringent navigation requirements that are difficult to satisfy using snapshot GNSS positioning. Therefore, this example uses tight integration of INS, code-differential GNSS, and Visual Odometry (VO).

### Simulation set-up

An error-state Kalman filter is used to fuse (a) IMU measurements, (b) GNSS pseudoranges, and (c) VO-derived body-frame velocity information. These sensors are installed in a forward-right-down way, and the geometry offsets among them are accurately compensated. The state vector consists of (a) INS errors in the East-North-Up (ENU) frame, (b) IMU sensor biases, and (c) receiver clock offsets. The state transition model is identical to that of GNSS/INS tight integration (Groves 2013), which is not restated here for brevity. The measurement model is obtained by augmenting the measurement model of GNSS/INS tight integration (Groves 2013) with the VO measurement model. Specifically, the VO measurement matrix is given by  $\mathbf{H}_v = [\mathbf{0}_{3 \times 3}, \mathbf{C}_v, \mathbf{0}_{3 \times 11}]$  where  $\mathbf{C}_v$  is the body-to-ENU transformation matrix.

The simulated flight trajectory is depicted in Fig. 8, and the simulation parameters are given in Table 3. In this table, the IMU is a tactical-grade one, and the VO velocity accuracy is preliminarily set for evaluation purposes only. Besides, sensor fault probabilities are set to different values to conduct sensitivity analyses. The GNSS elevation mask

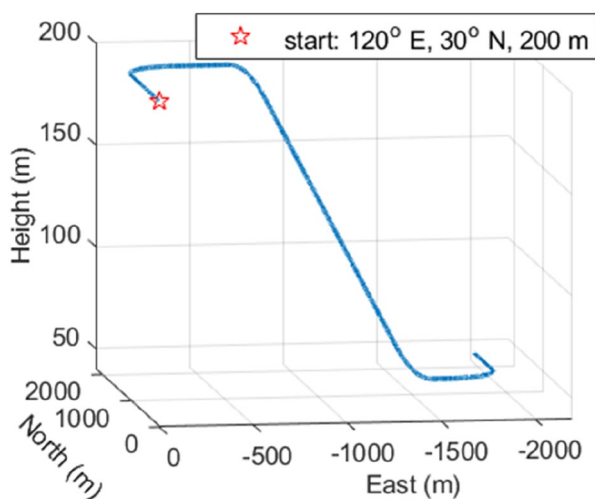


Fig. 8 Simulated flight trajectory of a UAM approach phase

angle is set to 30°, and Fig. 9 presents the skyplot of the visible satellites.

Finally, Table 4 shows the navigation performance requirements, which are referring to civil aviation (Blanch et al. 2015) and can preliminarily represent the case in UAM tasks. The states of interest include the east, north, and up positions, and they are, respectively, represented by  $q=1, 2,$  and 3 in this table. The associated PLs are denoted by EPL, NPL, and UPL in turn.

### Evaluating the protection levels in different cases

The PLs of the example multi-sensor navigation system are evaluated in different cases. First, Fig. 10 presents the PLs

of GNSS/INS tight integration under different satellite fault probabilities ( $P_{sat}$ ). In this figure, the PLs increase significantly as  $P_{sat}$  increases from  $10^{-5}$  to  $10^{-4}$ . This is as expected because only single-fault events are monitored for  $P_{sat}=10^{-5}$ , whereas multi-fault events also need monitoring when  $P_{sat}$  becomes  $10^{-4}$ . Compared with a single-fault event, a multi-fault event usually contributes more to the integrity risk.

Then Fig. 11 compares the PLs between GNSS/INS and GNSS/INS/VO, and it proves that employing VO measurements can bring a significant performance improvement. Next, Fig. 12 demonstrates the sensitivity of the PLs over the VO fault probability ( $P_{vo}$ ). The result shows that the PLs are similar for  $P_{vo}$  being  $10^{-8}$  and  $10^{-5}$  but experience a noticeable increase when  $P_{vo}$  becomes  $10^{-3}$ . The increase is caused by the newly-monitored events, especially those where VO and a GNSS constellation are faulted simultaneously.

In contrast to the previous figures that focus on measurement faults, Fig. 13 reveals the effect of a system fault on the PLs by setting the IMU fault probability to  $10^{-5}$ . The result suggests that the PLs increase dramatically when the fault modes associated with the IMU are monitored. This figure also compares the PLs between GNSS/INS and GNSS standalone, and it can be seen that GNSS/INS has better integrity performance than GNSS standalone even if IMU faults are considered.

### Demonstrating the filter bank management strategies

Simulations are carried out based on GNSS/INS tight integration to demonstrate the filter bank management strategies, in which  $P_{sat}$  is set to  $10^{-4}$  and  $P_{ins}$  is  $10^{-8}$ . By setting the satellite C01 to be invisible before  $t=80$  s, Fig. 14 compares

Table 3 Basic simulation parameters

| Sensor       | Parameter                        | Value  |                                       |
|--------------|----------------------------------|--|---------------------------------------|
| IMU (100 Hz) | White noise ( $1 \sigma$ )       | G: $0.2^\circ/\sqrt{\text{hr}}$  | A: $120 \mu\text{g}/\sqrt{\text{Hz}}$ |
|              | In-run instability               | G: $2^\circ/\text{hr}$   | A: $15 \mu\text{g}$                   |
|              | Correlation time                 | G: 500 s   | A: 500 s                              |
|              | Initial constant bias            | G: $10^\circ/\text{hr}$  | A: 200 $\mu\text{g}$                  |
|              | Fault probability                | $10^{-8}$ (default), $10^{-5}$ , or $10^{-4}$                          |                                       |
| GNSS (1 Hz)  | Pseudorange noise ( $1 \sigma$ ) | 1.0 m, white noise   |                                       |
|              | Clock bias error                 | noise ( $1 \sigma$ ): 1.0 m  |                                       |
|              | Clock drift error                | noise ( $1 \sigma$ ): 0.02 m/s; time constant: 200 s                   |                                       |
|              | Initial clock error              | bias: 10 m; drift: 0.1 m/s   |                                       |
|              | Constellation                    | GPS (G01-G32) and BDS-2 (C01-C14)                                      |                                       |
|              | Fault probability                | satellite: $10^{-5}$ (default) or $10^{-4}$ ; constellation: $10^{-5}$ |                                       |
|              | Ephemeris                        | brdm0010.19p   |                                       |
| Start epoch  | 00:00:00, 2019/01/01             |  |                                       |
| VO (1 Hz)    | Velocity noise ( $1 \sigma$ )    | 0.3 m/s for each direction   |                                       |
|              | Fault probability                | $10^{-8}$ (default), $10^{-5}$ , or $10^{-3}$                          |                                       |

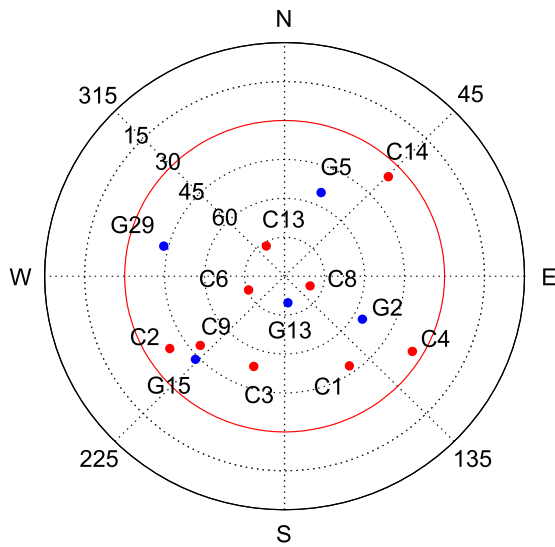


Fig. 9 Skyplot of the visible satellites

Table 4 Navigation performance requirements on integrity and continuity

| Parameter   | Value (preliminary)  |
|-------------|--|
| $P_{HMI}$   | $10^{-7}$  |
| $P_{HMI,q}$ | $P_{HMI,1} = P_{HMI,2} = 10^{-9};$<br>$P_{HMI,3} = 9.8 \times 10^{-8}$         |
| $P_{THRES}$ | $8 \times 10^{-8}$   |
| $P_{FA}$    | $3.99 \times 10^{-6}$  |
| $P_{FA,q}$  | $P_{FA,1} = P_{FA,2} = 4.5 \times 10^{-8};$<br>$P_{FA,3} = 3.9 \times 10^{-6}$ |

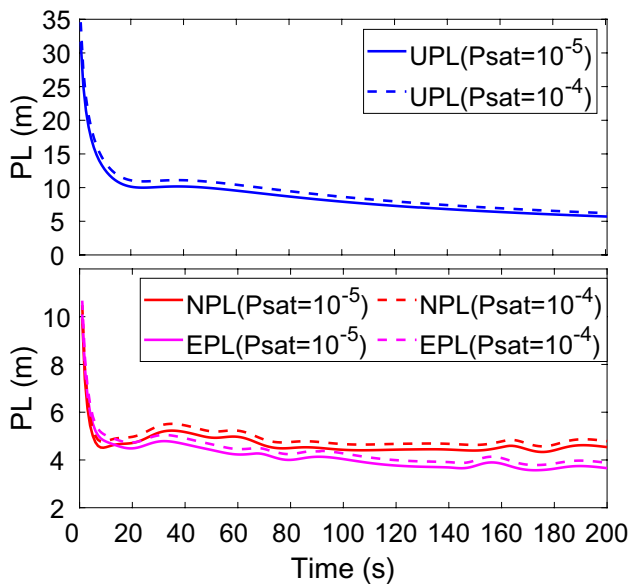


Fig. 10 PLs of GNSS/INS tight integration under different satellite fault probabilities (Psat)

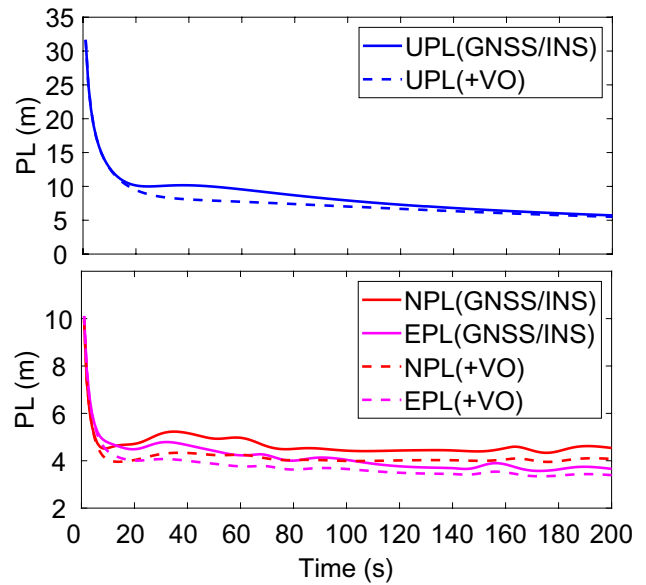


Fig. 11 Comparison between the PLs of GNSS/INS and GNSS/INS/VO

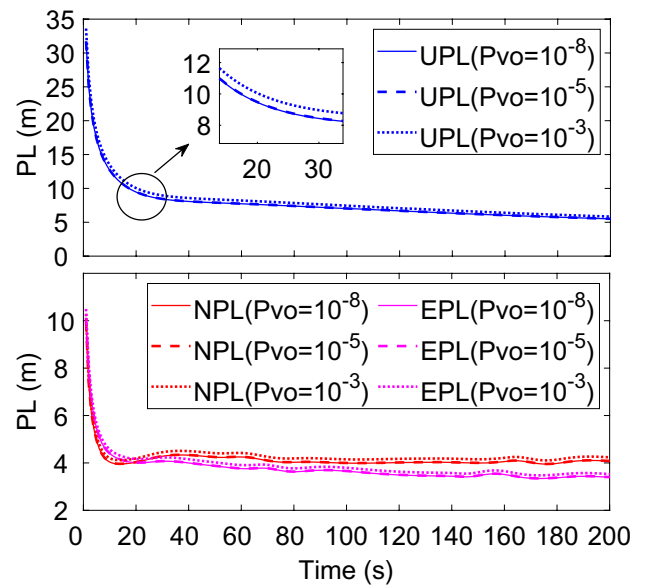
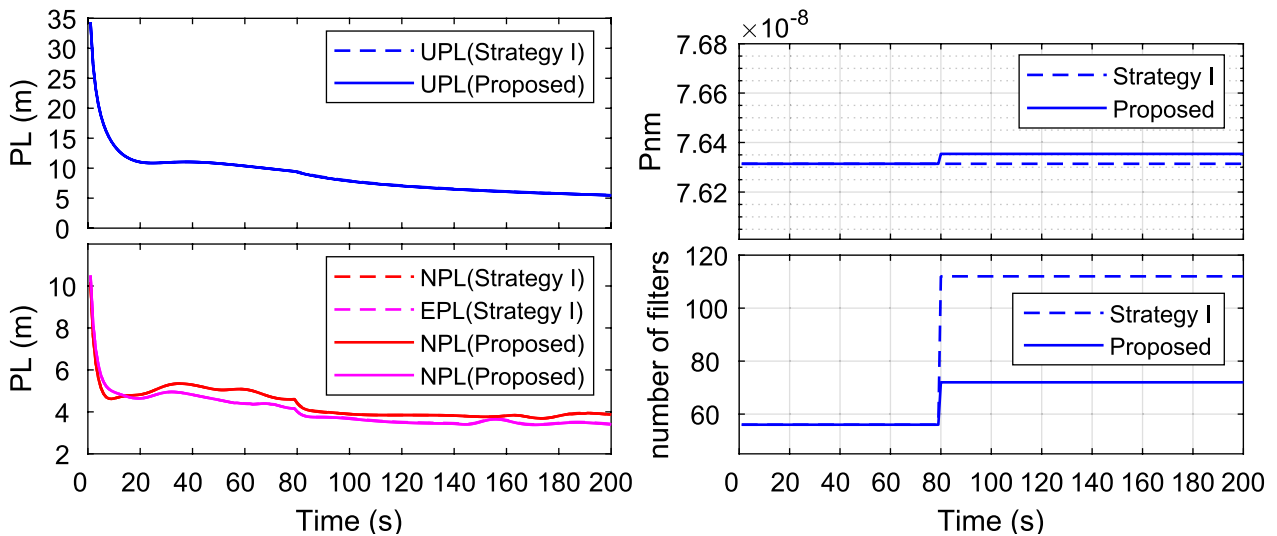
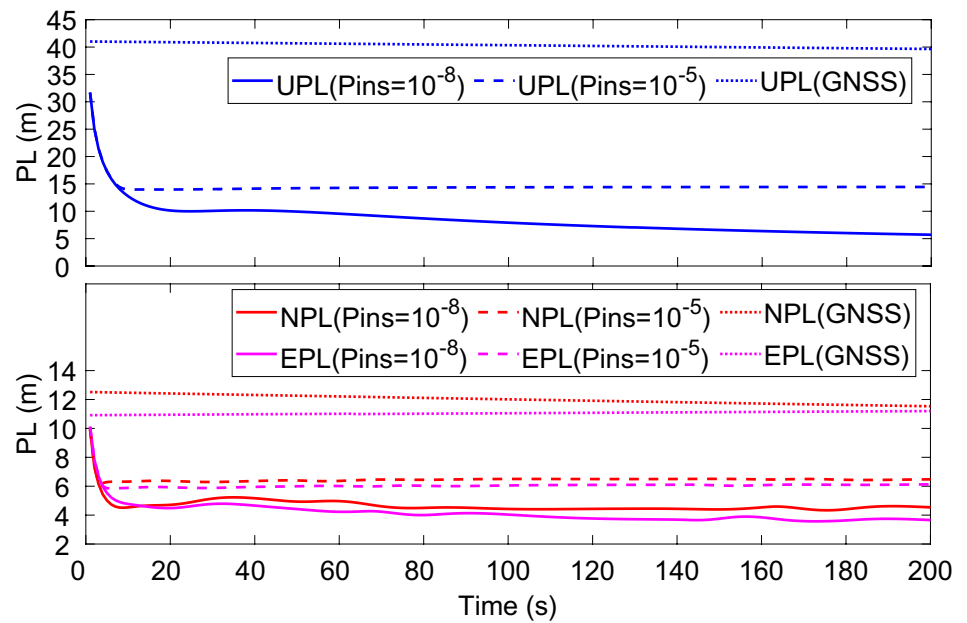


Fig. 12 PLs of GNSS/INS/VO under different VO fault probabilities (Pvo)

the performance of the proposed strategy and Strategy I for a sensor-in event. The result indicates that as compared to Strategy I, the proposed strategy produces similar PLs while requiring significantly fewer filters. In other words, the proposed strategy requires less computational load, because the computational cost is approximately proportional to the number of filters. Note that the PLs cannot be computed with Strategy II because it makes the probability of unmonitored events exceed the target integrity risk.

**Fig. 13** PLs of GNSS/INS tight integration under different IMU fault probabilities (Pins)



**Fig. 14** Performance of the proposed strategy and Strategy I when a sensor-in event occurs at  $t=80$  s: **a** protection levels; **b** probability of unmonitored events (“Pnm”) and the number of filters

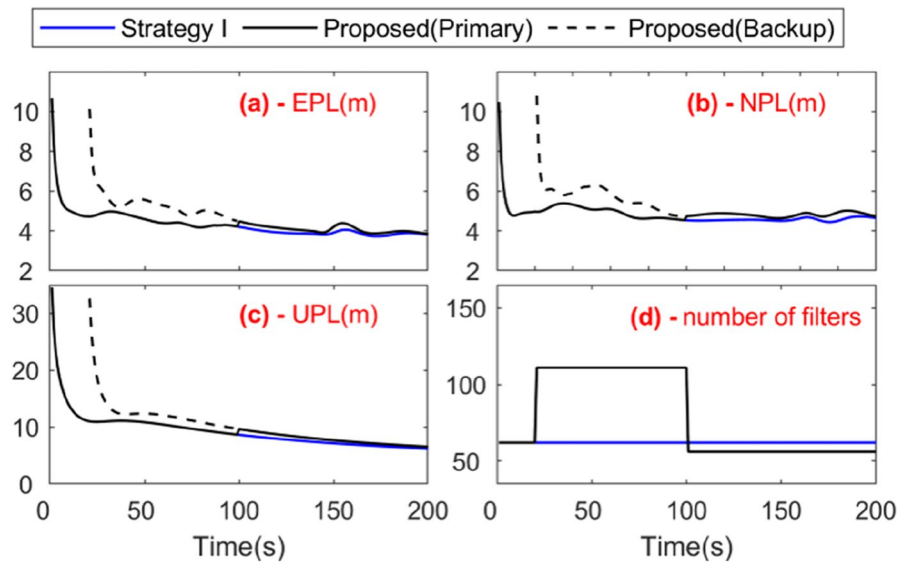
Similarly, Fig. 15 shows the performance of the proposed strategy and Strategy I for a sensor-out event by setting C01 to be invisible after  $t=20$  s. This figure helps illustrate the procedures of the proposed strategy: first ( $t=20$  s), a new filter bank is constructed with 49 newly-initialized filters; then ( $20 \leq t < 100$  s), the original filter bank is the primary filter bank that outputs the PLs to users, and the new one is maintained as the backup; finally ( $t=100$  s), when the backup filter bank produces similar PLs to the primary one, the original filter bank is deleted and the new filter bank becomes the primary. Compared with Strategy I, which takes no action after a sensor-out event occurs, the proposed

strategy reduces the number of filters, i.e., lightening the computational load, at the cost of a slight increase in the PLs after the final step.

**Evaluating the protection levels when the all-in-view filter is nonoptimal**

Our implementation of SS-KFIM can address the situations where the all-in-view filter is nonoptimal. To test this function, we simulate two cases of GNSS/INS integrated systems, namely “Default” and “Nonoptimal” cases. In both cases, the error models for integrity are

**Fig. 15** Performance of the proposed strategy and Strategy I when a sensor-out event occurs at  $t=20$  s. Note that for these two strategies, their PL curves are coincident before  $t=100$  s

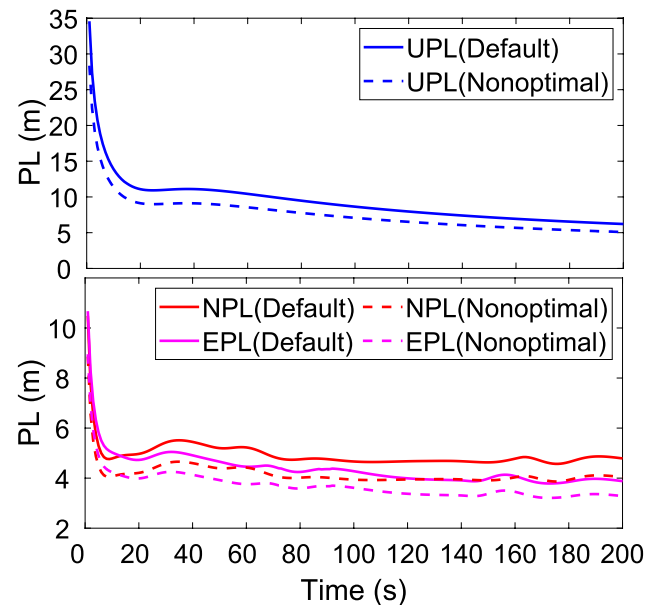


derived from Table 3, and we set  $P_{sat}$  to  $10^{-4}$  and  $P_{ins}$  to  $10^{-8}$ . In the “Default” case, the error models for continuity are the same as those for integrity, whereas in the “Nonoptimal” case, the process and measurement error covariances for continuity evaluation are half the corresponding error covariances for integrity.

Figure 16 presents the PLs in these two cases. The PLs in the “Default” case are separately computed using the existing SS-KFIM algorithm and our implementation, and their outputs agree with each other. In the “Nonoptimal” case, the PLs are only calculated with our implementation because the existing approach is not applicable. This suggests that our implementation extends the applicability of the existing SS-KFIM algorithm. If the users who adopt the existing approach hope to compute the PLs in the “Nonoptimal” case, they have to replace the error models for continuity with those for integrity. The PLs they obtained in this way are the same as those in the “Default” case, which are obviously larger than those output by our implementation. This indicates the better performance of our implementation than the existing one in the “Nonoptimal” case and other similar situations.

### Validating the fault detection capability

Finally, the fault detection capability of the SS-KFIM algorithm is evaluated under the simulated fault scenarios given in Table 5. The following figures only show the FD test that corresponds to the true fault event, because this test is most likely to detect the fault first. Also, as a reminder, the system will issue an alarm and quit once a fault is detected.



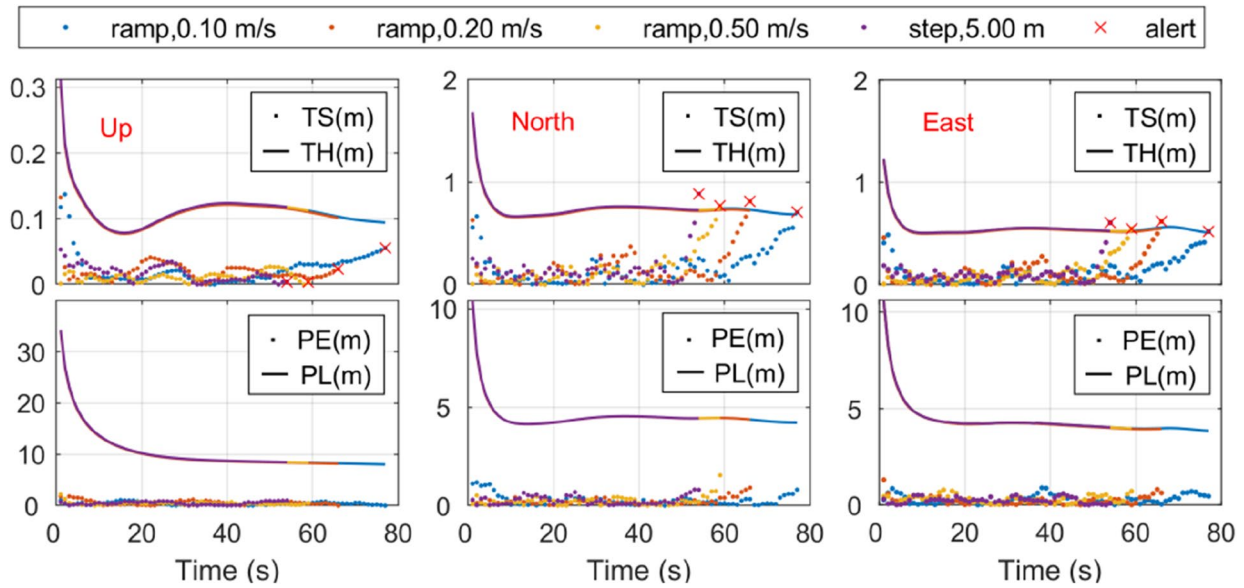
**Fig. 16** PLs in the “Default” and “Nonoptimal” cases

Figures 17 and 18 demonstrate the FD performance against a single-satellite fault and a dual-satellite fault, respectively. The results suggest that the fault detector can effectively detect GNSS faults in both single- and multi-fault cases. Also, they show that the time-to-detect value (i.e., the time elapsed from when a fault occurs to when it is detected) varies with fault types and magnitudes. Specifically, the detection time becomes shorter as the fault

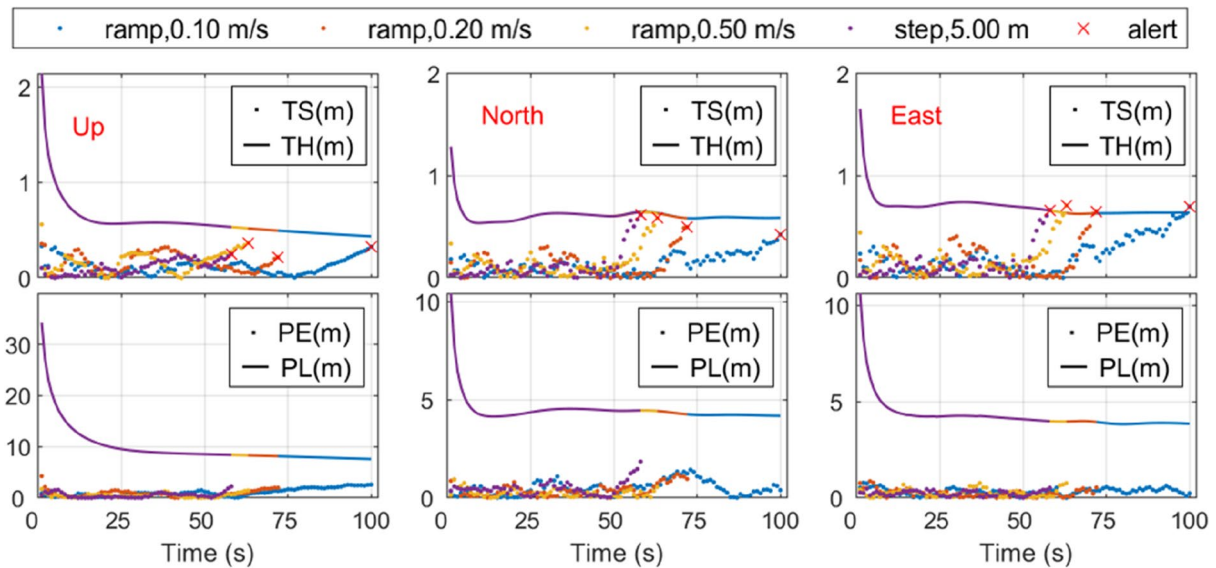


**Table 5** Simulation configurations in different cases

| Case      | System      | Probability   | Fault location           | Duration      |
|-----------|-------------|---|--------------------------|---------------|
| Figure 17 | GNSS/INS/VO | $P_{\text{sat}}=10^{-4}, P_{\text{vo}}=10^{-5}, P_{\text{ins}}=10^{-8}$ | C01                      | $t \geq 50$ s |
| Figure 18 |             |   | C09 and G13              |               |
| Figure 19 |             |   | VO, z-axis               |               |
| Figure 20 | GNSS/INS    | $P_{\text{sat}}=10^{-4}, P_{\text{ins}}=10^{-4}$                        | IMU, z-axis acceleration |               |



**Fig. 17** Fault detection performance in the presence of a single GNSS fault. “TH” and “TS” separately represent thresholds and test statistics. The time-to-detect values are 27 s, 16 s, 9 s, and 4 s, for the four fault cases, respectively



**Fig. 18** Fault detection performance in the presence of two simultaneous GNSS faults. For the four fault cases, the time-to-detect values are 50 s, 22 s, 13 s, and 8 s, respectively

rate magnitude increases. More importantly, the results prove that the PLs safely bound the Position Errors (PEs) before a fault is detected, which indicates the effectiveness of the PL evaluation method.

Then the FD capability in a VO-faulted case is evaluated in Fig. 19. Since the pitch angle of the air vehicle is almost zero, the injected fault mainly affects the vertical position component. Therefore, Fig. 19 only presents the result of this component, and it proves that the SS-KFIM algorithm can also protect the GNSS/INS/VO system against VO faults.

Finally, Fig. 20 shows the behavior of the fault detector in the presence of system faults. In this figure, the system faults are caused by IMU failures. Specifically, the step fault reflects a sudden change in the accelerometer bias, and the ramp fault represents an event that there is an anomalous bias drift. As shown in this figure, IMU faults could cause significant navigation errors, and this highlights the importance of protecting multi-sensor navigation systems against system faults. Also, the results suggest that our implementation of SS-KFIM is effective in detecting system faults and quantifying its effect on navigation integrity.

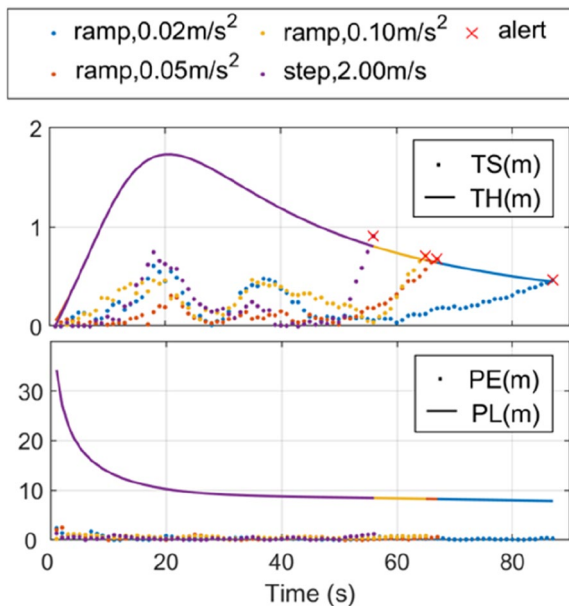


Fig.19 Fault detection results in the vertical direction in various VO-faulted cases. The time-to-detect values are 37 s, 17 s, 15 s, and 6 s for the four fault cases, respectively

### Conclusions

We implement the Solution Separation-based Kalman Filter Integrity Monitoring (SS-KFIM) technique to realize fault detection and Protection Level (PL) evaluation for multi-sensor

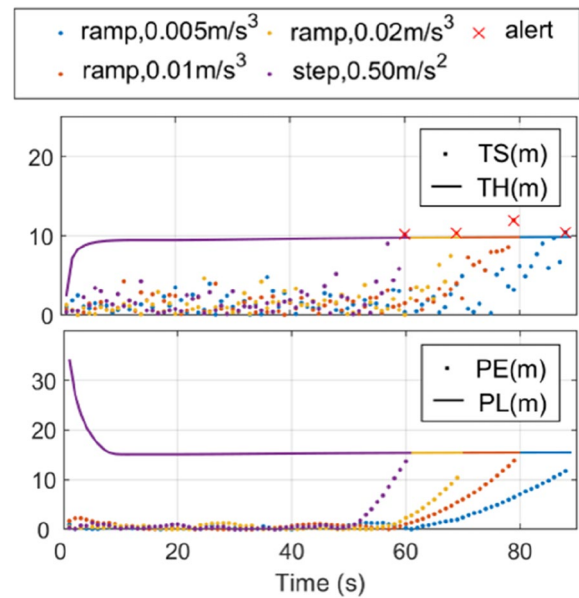


Fig.20 Fault detection results in the vertical direction in various IMU-faulted cases. For the four fault cases, the time-to-detect values are 38 s, 29 s, 19 s, and 10 s, respectively

navigation systems. In our implementation, the filter bank management strategies to handle sensor-in and sensor-out events are discussed. Besides, we consider the faults in state initialization and propagation phases aside from those at the measurement-update stage. Furthermore, our implementation can address the situations where the all-in-view filter is not optimal. Simulations are conducted for an urban air mobility navigation application where global navigation satellite systems, an inertial navigation system, and visual odometry are tightly coupled using a Kalman filter. The results suggest that (a) the modified filter bank management strategy is beneficial to reducing the number of filters while maintaining low PLs, (b) our implementation is effective in protecting multi-sensor navigation systems against all sources of faults, and (c) it extends the applicability of the existing SS-KFIM algorithm through enabling the handling of nonoptimal all-in-view filters. Future work will focus on implementing the SS-KFIM technique with an efficient fault exclusion scheme and mitigating the computational issue.

### Appendix A : Quantitative analysis on the IMU fault probability

An IMU may be faulted if any one of its components is in an abnormal condition (Bhatti et al. 2007). IMU failures have led to several civil aviation accidents (Crispoltoni et al. 2018). Compared with civil aircraft, future autonomous systems are more vulnerable to IMU failures because they usually employ low-cost IMUs. By showing the

**Table 6** MTBF values of five tactical-grade IMUs (the data are available on the Internet)

| No | IMU      | Company          | MTBF (hours) |
|----|----------|------------------|--------------|
| 1  | STIM 210 | Sensoror         | 70,000       |
| 2  | HG 4930  | Honeywell        | 59,000       |
| 3  | LN 200   | Northrop Grumman | 20,000       |
| 4  | VN 110E  | VectorNav        | 45,000       |
| 5  | ISA-100C | Northrop Grumman | 46,100       |

Mean-Time-Between-Faults (MTBF) values of five tactical-grade IMUs, Table 6 implies that their fault probabilities are about  $10^{-4} \sim 10^{-5}$  per hour. Therefore, it is necessary to monitor IMU faults in safety-critical applications.

## Appendix B: Implementation of the pseudoinverse operator

A real symmetric singular matrix  $\mathbf{M}$  can be factorized as:

$$\mathbf{M} = \mathbf{U}\mathbf{A}\mathbf{U}^T = \begin{bmatrix} \mathbf{U}_\lambda & \mathbf{U}_0 \end{bmatrix} \begin{bmatrix} \lambda & \\ & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{U}_\lambda^T \\ \mathbf{U}_0^T \end{bmatrix} \quad (56)$$

Then the pseudoinverse of  $\mathbf{M}$  is given by:

$$\mathbf{M}^\dagger = \mathbf{U}_\lambda \lambda^{-1} \mathbf{U}_\lambda^T \quad (57)$$

(56) and (57) can be implemented using the *pinv* function in MATLAB.

**Acknowledgements** This work was supported by the National Key Research and Development Program of China (2021YFB3901501) and the National Natural Science Foundation of China (62103274).

**Data Availability** The simulation data are available from the corresponding author upon reasonable request.

## References

- Arana GD, Hafez OA, Joerger M, and Spenko M, (2019) Recursive integrity monitoring for mobile robot localization safety. In: Proceedings ICRA 2019, Montreal, Canada, May 20–24, pp. 305–311
- Bhatti UI, Ochieng WY, Feng S (2007) Integrity of an integrated GPS/INS system in the presence of slowly growing errors Part I: a critical review. *GPS Sol* 11(3):173–181
- Bhatti UI, Ochieng WY, Feng S (2012) Performance of rate detector algorithms for an integrated GPS/INS system in the presence of slowly growing error. *GPS Sol* 16(3):293–301
- Blanch J, Walter T, Enge P (2010) RAIM with optimal integrity and continuity allocations under multiple failures. *IEEE Trans Aerosp Electron Syst* 46(3):1235–1247
- Blanch J et al (2015) Baseline advanced RAIM user algorithm and possible improvements. *IEEE Trans Aerosp Electron Syst* 51(1):713–732
- Blanch J, Gunning K, Walter T, de Groot L, Norman L (2019) Reducing computational load in solution separation for Kalman filters and an application to PPP integrity. In: Proceedings ION ITM 2019, Institute of Navigation, Reston, Virginia, USA, January 28–31, pp. 720–729
- Blanch J, Walter T (2021) Fast protection levels for fault detection with an application to advanced RAIM. *IEEE Trans Aerosp Electron Syst* 57(1):55–65
- Brown RG (1992) A baseline GPS RAIM scheme and a note on the equivalence of three RAIM methods. *Navigation* 39(3):301–316
- Crispoltoni M, Fravolini ML, Balzano F, D’Urso S, Napolitano MR (2018) Interval fuzzy model for robust aircraft IMU sensors fault detection. *Sensors* 18(8):2488
- Groves P (2013) Principles of GNSS, inertial and multi-sensor integrated navigation systems(2nd Edition). Artech House, London
- Gunning K, Blanch J, Walter T, de Groot L, and Norman L (2018) Design and Evaluation of Integrity Algorithms for PPP in Kinematic Applications. In: Proceedings of the 31st International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2018), Miami, Florida, September 24–28, pp. 1910–1939
- Gunning K, Blanch J, Walter T, de Groot L, Norman L (2019a) Integrity for tightly coupled PPP and IMU. In: Proceedings ION GNSS+ 2019a, Institute of Navigation, Miami, Florida, September 16–20, pp. 3066–3078
- Gunning K, Blanch J, Walter T (2019b) SBAS corrections for PPP integrity with solution separation. In: Proceedings ION ITM 2019b, Institute of Navigation, Reston, Virginia, January 28 – 31, pp. 707–719
- Guo S et al. (2019) Fault tolerant multi-sensor federated filter for AUV integrated navigation. In: Proceedings 2019 IEEE underwater technology, Kaohsiung, Taiwan, April 16–19, pp. 1–4
- International Civil Aviation Organization (ICAO) (2009) Annex 10, Aeronautical telecommunications, volume 1 (radio navigation aids), amendment 84, Montreal, QC, Canada
- Joerger M, Pervan B (2013) Kalman filter-based integrity monitoring against sensor faults. *J Guid Control Dyn* 36(2):349–361
- Joerger M, Chan FC, Pervan B (2014) Solution separation versus residual-based RAIM. *Navigation* 61(4):273–291
- Jurado J, Raquet J, Schubert Kabban CM, Gipson J (2020) Residual-based multi-filter methodology for all-source fault detection, exclusion, and performance monitoring. *Navigation* 67(3):493–510
- Lee J, Kim M, Lee J, and Pullen S (2018) Integrity assurance of Kalman-filter based GNSS/IMU integrated systems against IMU faults for UAV applications. In: Proceedings ION GNSS+ 2018, Institute of Navigation, Miami, Florida, USA, September 24–28, pp. 2484–2500
- Simon D (2006) Optimal state estimation: Kalman, H $\infty$ , and nonlinear approaches. Wiley, Hoboken, NJ, USA
- Tanil C, Khanafseh S, Joerger M, Pervan B (2018a) An INS monitor to detect GNSS spoofers capable of tracking vehicle position. *IEEE Trans Aerosp Electron Syst* 54(1):131–143
- Tanil C, Khanafseh S, Joerger M, and Pervan B (2018b) Sequential integrity monitoring for Kalman filter innovations-based detectors. In: Proceedings ION GNSS+ 2018b, Institute of Navigation, Miami, Florida, USA, September 24–28, pp. 2440–2455
- Tanil C, Khanafseh S, Joerger M, Kujur B, Kruger B, de Groot L, Pervan B (2019) Optimal INS/GNSS coupling for autonomous car positioning integrity. In: Proceedings ION GNSS+ 2019, Institute of Navigation, Miami, Florida, USA, September 16–20, pp. 3123–3140

- Wang R, Xiong Z, Liu J, Xu J, Shi L (2016) Chi-square and SPRT combined fault detection for multisensor navigation. *IEEE Trans Aerosp Electron Syst* 52(3):1352–1365
- Wang S, Zhan X, Pan W (2018) GNSS/INS tightly coupling system integrity monitoring by robust estimation. *J Aeronaut Astronaut Aviat* 50(1):61–80
- Wang S, Zhan X, Zhai Y, Liu B (2020) Fault detection and exclusion for tightly coupled GNSS/INS system considering fault in state prediction. *Sensors* 20(3):590
- Zhai Y, Joerger M, Pervan B (2018) Fault exclusion in multi-constellation global navigation satellite systems. *J Navig* 71(6):1281–1298
- Zhong L, Liu J, Li R, Wang R (2017) Approach for detecting soft faults in GPS/INS integrated navigation based on LS-SVM and AIME. *J Navig* 70(3):561–579

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.



**Shizhuang Wang** received his B.S. in Aerospace Engineering from Shanghai Jiao Tong University (SJTU), China, in 2018. He is currently a Ph.D. candidate at the School of Aeronautics and Astronautics, SJTU. His research interests focus on navigation integrity and multi-sensor integration.



**Yawei Zhai** obtained his Ph.D. in Mechanical and Aerospace Engineering from the Illinois Institute of Technology, USA, in 2018. He is currently a post-doctoral researcher at Shanghai Jiao Tong University. His research focuses on Advanced Receiver Autonomous Integrity Monitoring (ARAIM) and multi-sensor integration.



**Xingqun Zhan** received his Ph.D. from Harbin Institute of Technology, China, in 1999. He is currently a professor and the associate dean of the School of Aeronautics and Astronautics at Shanghai Jiao Tong University. He is an associate editor of *Aerospace Science & Technology* and an AIAA associate fellow. His research interests focus on navigation integrity multi-sensor integration, and cooperative navigation.