Unmanned Systems, Vol. 10, No. 1 (2022) 69–91

DOI: 10.1142/S2301385022500042

UNMANNED SYSTEMS
https://www.worldscientific.com/worldscinet/us

Fault Diagnosis and Reconfiguration for Mobile Robot Localization Based on Multi-Sensors Data Fusion

Linda Hachemi*, Mohamed Guiatni, Abedlkrim Nemra

Intelligent Autonomous Systems Research Unit,
Ecole Militaire Polytechnique, Bordj E Bahri, Algiers, Algeria

In this paper, we propose a new approach for fault tolerant localization using multi-sensors data fusion for a unicycle-type mobile robot. The main contribution of this paper is a new architecture proposal for fault diagnosis and reconfiguration for mobile robot localization using multi-sensors data fusion and the duplication/comparison approach. Four different sensors usually embedded in mobile robots (Camera, IMU, GPS, and Odometer) are considered, while six different sensors couples combinations are used for sensor data fusion and the duplication of the localization and estimation system. In order to reach this aim, three different filters (EKF, SVSF, and ASVSF) have been proposed and compared. For each selected filter, a comparison mechanism is then introduced to compute different residuals by comparing the estimated robot position for each sensor couples separately. Faults are then detected using the structural residual diagnosis method. This approach assumes the occurrence of a single fault at a given time. A reconfiguration mechanism is then applied by selected the healthy sensors couple and their corresponding fusion filter. Several scenarios are considered for navigation-based fault tolerant localization approaches. Simulation results are presented to illustrate the advantage and performance of the proposed architecture. The proposed solutions are implemented and validated successfully using the V-REP simulator.

Keywords: Unicycle-type mobile robot; Diagnosis; localization; multi-sensors data fusion; V-REP.

US

1. Introduction

We want to build robots that are able to operate in different kinds of environments. Some environments are rather static, in the sense that layout changes are not frequent (e.g. new walls, furniture movement, etc.). However, the majority of environments conditions are inherently dynamic due different variations such as the movement of people and objects, or the change of illumination conditions. Robots must execute the assigned tasks correctly under different conditions. Consequently, they must be able to determine their position accurately and robustly at all times. This task, which is known as mobile robot localization, is a crucial ability to the successful deployment of any autonomous mobile robot. The availability of localization sensors of various modalities such as the Global Positioning System (GPS), inertial measurement unit (IMU), Vision, odometer, Sonars, Lidars, provides an estimate of the robot location based on information either from an external source or from the internal state of the robot [1]. However, data perceived by such sensors are often complex and subject to significant uncertainties, inaccuracies and faults [2].

To overcome the drawbacks of using a single localization sensor, the multi-sensors approach takes data from multiple, complementary sensors, and uses their redundancy to detect faults, filter noise, eliminate some aberrant data, increase the precision of perception, and extract complex knowledge about the environment.

Much work has been performed in the field of localization making sensor fusion techniques useful for merging different types of localization sensors in order to provide high precision estimates at high update rates. This task has been a subject of continuous research. Different popular algorithms have been used for sensor fusion for robot localization such as Kalman Filtering and its derivatives [3–5],

Received 26 November 2020; Revised 10 May 2021; Accepted 10 May 2021; Published 10 June 2021. This paper was recommended for publication in its revised form by editorial board member, Youmin Zhang.
Email Address: *salyhachemi@hotmail.fr*

Smooth Variable Structure Filter (SVSF) [6,7]. In [8], authors present a novel navigation method designed to support a real-time indoor robot localization using IMU sensors and dual Kalman filter (DKF) algorithm. Their results indicate that the method achieves the levels of accuracy location comparable with that of the IMU/Stereo vision fusion algorithm. In [9], authors investigate the moving robot localization problem using a Doppler–Azimuth radar array. The solution is formulated of nonlinear non-Gaussian estimation in the framework using a particle filter and a Random Finite Set (RFS) model of measurements. In [10], the localization problem is addressed by means of a tailored Extended H Filter (EHF) and fusing odometer and gyroscope data with position and heading measurements based on quick response (QR) code landmark recognition. Also, the proposed approach ensures a good trade-off in terms of computational burden, convergence time, and deployment complexity. Authors of paper [11] present a Hierarchical Sensor Fusion (HSF) method using artificial neural network (ANN) for robot self-localization with sonars octagon, digital compass, and wireless network signal strength measure. Whereas in [12], the authors evaluated the robustness of localization using three different sensors: a laser rangefinder, a depth camera, and WiFi signal strength. The three sensors have been merged for the localization for different areas of a map. In [13], the authors propose a camera-odometer fusion method based on the EKF filter and marker-pair. The proposed approach improves the localization accuracy for map building in visual Simultaneous Localization and Map (SLAM). In [14], authors propose a real-time outdoor positioning algorithm combining GPS and 3D Lidar-based odometer. In [15], the authors present a fusion algorithm that combines four different sensors information (IMU, joints encoder, camera and LIDAR) for quadruped robot state estimation using the Normal Distributions Transform (NDT) and the EKF Filter. While in [16], the authors propose an enhanced multi-sensor fusion methodology to fuse the information from low-cost GPS, MEMS IMU, and digital compass using the Empirical Mode Decomposition (EMD) and interval threshold filter. Their results indicate that the proposed methodology can achieve remarkable enhancement in positioning accuracy in GPS-denied environments. In [17], the authors present a deep learning-based approach for a mobile robot localization using a 2D laser and an IMU using Recurrent Convolutional Neural Network (RCNN)-based architecture. In [18], authors compare different sensor fusion techniques and evaluate their accuracy depending on the environment. It has been shown that LIDAR scan matching combined with other sensor data can be used to improve the accuracy of the robot localization and, thus, its navigation performance. They also propose a strategy to reduce the impact of a change in the environment renders map data or part of the available map is corrupted.

Nevertheless, increasing the number of sensors and the underlying data fusion algorithms increases the risks of hardware and software faults [5]. Thus, the occurrence of faults in the robot during its operation can have severe consequences and ultimately could lead to failure of robot mission. Fault diagnosis and reconfiguration for mobile robot localization using multi-sensors data fusion can help robots to independently detect and isolate internal failures and use the remaining functional capabilities to automatically overcome the limitations imposed by failures. Several researches have been performed in the field of fault detection and isolation of mobile robots. For example, in [15, 19], the authors propose a multi-sensor and multi-level FDI approach including pre-processing, local-data fusion, behavioral analysis, change detection, credibility computation, decision-level information fusion for mobile robots. Whereas in [19], a multi-sensor data fusion-based fault detection and isolation (FDI) method for robotic system is proposed. The proposed approach is independent of analytical model, which suits dynamic, uncertain and unstructured nature of the robot-operating environment. The authors in [20] propose a strategy that combines fault tolerant multi-sensor data fusion with robust controller scheme, which has been applied to a multi-robot mobile system tracking using the Kalman filter. In [19], the authors propose an adaptive multi-sensor data fusion method based on deep convolutional neural networks (DCNN) for fault diagnosis. A fault detection and reconfiguration approach is proposed in [21], where the fusion method is dynamically reconfigured upon the sensor fault detection using simple concepts of Information Theory. In [22], an FTC based on fault hiding approach is proposed for a nonholonomic mobile robot. First, a Sliding Mode Controller (SMC) is designed to control the robot and to cope with modeling uncertainty. Later on, it is enhanced to take into account actuator faults leading to a fault hiding approach for the sliding mode fault-tolerant control of the robot. Consistency is examined using a log likelihood ratio between the information innovations of each sensor for fault detection. In [23], an adaptive SMC (ASMC) scheme is proposed to accommodate system uncertainties caused by actuator faults. The authors in [2] propose an architecture based on duplication/comparison for detecting and diagnosing faults in a data fusion mechanism for autonomous robots. The proposed strategy uses the Kalman filter and offers detection and recovery services. The authors in [24] propose two fault tolerant control approaches (passive FTC, and active FTC) a quadcopter unmanned aerial vehicle (UAV).

Although the important number of works deals with fault detection, isolation and accommodation for robot localization, there are still some issues to be solved to improve the robot reliability, fault tolerance ability, and safety in autonomous robotics. In [25], the authors present a leader-follower type of fault-tolerant formation control (FTFC) methodology with application to multiple UAVs in the presence of actuator failures and potential collisions. Their proposed FTFC scheme consists of both outer-loop and inner-loop controllers. First, a leader-follower control scheme with integration of a collision avoidance mechanism is designed as the outer-loop controller for guaranteeing UAVs to keep the desired formation while avoiding the approaching obstacles. Then, an active FTC strategy for counteracting the actuator failures and also for preventing the healthy actuators from saturation is synthesized as the inner-loop controller.

In order to improve the efficiently of fault tolerant control of mobile robots, this work aims to propose a contribution in this field throw the proposal of a new architecture for fault diagnosis and reconfiguration for mobile robot localization using multi-sensors data fusion and the duplication/comparison approach. In contrast to single-sensor data-driven FDI methods, incorporation of multi-sensor fusion architecture makes the proposed FDI scheme more robust and reliable and reduce the uncertainty in the diagnosis. Four different sensors usually embedded in mobile robots (Camera, IMU, GPS, and Odometer) are considered, while six different sensors couples combinations are used for sensor data fusion and the duplication of the localization and estimation system.

Among the most used fusion filter, the Extended Kalman Filter (EKF) is the most famous. However, the EKF requires model linearization which makes it inaccurate when the process and observation models are highly nonlinear. Furthermore, the characteristics of process and observation noise should be known for the EKF. In addition, linearization using Jacobians can sometimes cause instabilities when implementing the EKF [26]. To overcome these weaknesses, we proposed two other filters: the Smooth Variable Structure Filter (SVSF) and the Adaptive SVSF filter (ASVSF). SVSF is a relatively new filter. It is a predictor–correct estimator based on sliding mode approach estimation. In its current form, the SVSF is not a classical filter in the sense that it does not have a covariance matrix [27]. The SVSF has the robustness to modeling errors and uncertainties and guaranteed stability using internal model of the system, either linear or nonlinear [26]. The boundary layer width (ϕ) of the filter reflects the levels of uncertainty in the model parameters and disturbance characteristics, where large values of (ϕ) leads to robustness without optimality and small values of (ϕ) provide optimality with poor robustness [7]. In the SVSF filter, the statistical noise parameter is always constant for all the processes and considered to be well known at the start. However, in the real application, it is impossible to precisely define these parameters being partially known or even unknown. This could degrade the filtering performance. For these reasons, the adaptive SVSF algorithm is proposed in this paper. Unlike the SVSF standard, the SVSF with Adaptive Layer Width (ASVSF) has covariance matrices, which can be used to determine the optimal gain value. ASVSF is very robust estimator against modeling errors and noises. Keeps a compromise between robustness and accuracy [28].

In order to reach the aim of this work, the three filters described above (EKF, SVSF and ASVSF) have been proposed and compared. For each selected filter, a comparison mechanism is then introduced to compute different residuals by comparing the estimated robot position for each sensor couples separately. Faults are then detected using the structural residual diagnosis method. This approach assumes the occurrence of a single fault at a given time. A reconfiguration mechanism is then applied by selected the healthy sensors couple and their corresponding fusion filter. The present configuration allows to tolerate hardware faults in sensors, and also software faults in data fusion algorithms. The rest of the paper is organized as follows. Section 2 describes the mathematical model of a unicycle-type mobile robot. Section 3 describes Localization architecture of the mobile robot. In Sec. 4, we present the multi-sensor data fusion, which uses three, filters: (FKE, SVSF and ASVSF). In Sec. 5, localization results are illustrated and compared two types of faults are considered: hardware fault in the GPS sensor, and one software fault in the filters. Both faults are validated is simulation using MATLAB/V-REP environment. Finally, in Sec. 6, conclusion and future work are presented.

2. System and Sensors Modeling

We consider in this work a unicycle mobile robot equipped with four different sensors: encoders, IMU, camera and GPS. However, the unicycle robot model includes many known differential drive robots and represents in many configurations even the four wheeled cars.

2.1. *Kinematic model of unicycle mobile robot*

A unicycle mobile robot is a nonholonomic system. It describes in general robots having usually two parallel-driven independent wheels, one mounted of each side of their center, as shown in (Fig. 1).

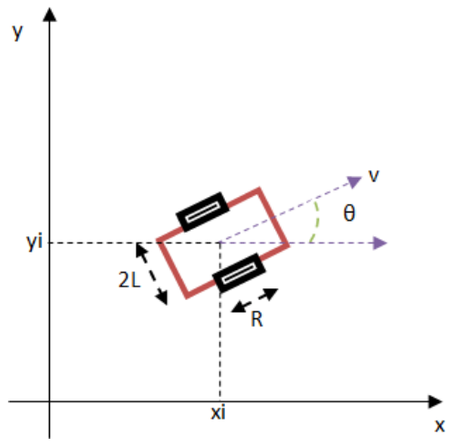

Fig. 1. Uni-cycle type mobile robot.

The simplified nonlinear kinematic model of the robot is given as follows:

$$\begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{1}$$

where v is linear velocity, is the orientation of the robot, is the rotational velocity which is the difference of the wheel velocities divided by the radius of rotation $L.2L$ is defined as the distance between the two wheels (x_c, y_c). Are the position coordinates of the center of mobile robot in 2D space. By considering R the radius of wheels, is computed as a function of the right and the left wheels velocity v_r, v_l, respectively

$$\omega = \frac{R}{L}(v_r - v_l). \tag{2}$$

2.2. *Odometer modeling*

We consider that the two wheels of unicycle mobile robot are equipped with encoders. Encoders measure the wheel's velocity by counting the pulses per sampling period. Therefore, it will give the information about the displacement of the robot relative to its last known position. In our case, we give the wheel's velocity in V-REP using the odometer model presented in Eq. (1), where $(x = x_c)$ and $(y = y_c)$ represent the odometer outputs. The odometer model is given as follows [3]:

$$X_{K+1} = f(X_K, U_K) + W_K, \tag{3}$$

where $X_K = [x/y/]^T$ is the state vector and W_K is a Gaussian noise for the process model with covariance matrix Q_K. Consequently, the discrete model of the odometer can be written as follows:

$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_{K+1} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_K + \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ w \end{bmatrix} T_e + W_K, \tag{4}$$

where

$$f(X_K, U_K) = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_K + \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ w \end{bmatrix} T_e. \tag{5}$$

T_e represents the sampling rate. The Jacobian matrix of f with respect to X^{K+1} is given as follows:

$$F = \frac{\partial f(x,u)}{\partial x} = \begin{bmatrix} 1 & 0 & -T_e v\sin\theta \\ 0 & 1 & T_e v\cos\theta \\ 0 & 0 & 1 \end{bmatrix}. \tag{6}$$

With

$$U = [v \quad w]^T. \tag{7}$$

The covariance matrix of W_K can be calculated as follows:

$$Q_K = BQB^T, \tag{8}$$

where Q is a diagonal matrix defined as follows:

$$Q_K = \begin{bmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_w^2 \end{bmatrix}, \tag{9}$$

where σ_v^2 and σ_w^2 represent the noise variance of and, respectively, and B is defined as follows:

$$B = \frac{\partial f(X,U)}{\partial U} = \begin{bmatrix} T_e v\cos\theta & 0 \\ T_e v\sin\theta & 0 \\ 0 & T_e \end{bmatrix}. \tag{10}$$

2.3. *GPS modeling*

GPS measures the position of the robot. We assume that its measurement vector Z_K is given as follows [32]:

$$\begin{aligned} Z_K &= [x_{\text{gps}} \quad y_{\text{gps}} \quad z_{\text{gps}} \quad V_{x\text{gps}} \quad V_{y\text{gps}} \quad V_{z\text{gps}}]_K^T \\ &= h(X_K) + V_K, \end{aligned} \tag{11}$$

where

$$h = \begin{bmatrix} x \\ y \\ z \\ Uc\theta s\psi - Vc\phi s\psi + Vs\phi c\psi + W(s\phi c\psi + c\phi s\theta c\psi) \\ Uc\theta s\psi + Vc\phi c\psi + Vc\phi s\theta c\psi - W(s\phi c\psi + c\phi s\theta c\psi) \\ -Us\theta + Vs\phi c\theta + Wc\phi c\theta \end{bmatrix}, \tag{12}$$

where U, V and W represent the velocities in the body frame. ϕ, θ and ψ represent the Euler angles and $c\alpha = \cos\alpha, s\alpha = \sin\alpha, \alpha \in \{\phi, \theta, \psi\}$. V_K are Gaussian noises for the measurement model with covariance R.

$$R = \begin{bmatrix} \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_z^2 \end{bmatrix} & 0 \\ 0 & \begin{bmatrix} \sigma_{v_x}^2 & 0 & 0 \\ 0 & \sigma_{v_y}^2 & 0 \\ 0 & 0 & \sigma_{v_z}^2 \end{bmatrix} \end{bmatrix}, \tag{13}$$

where σ_x^2, σ_y^2, σ_z^2, $\sigma_{v_x}^2$, $\sigma_{v_y}^2$ and $\sigma_{v_z}^2$, represent the noise variance of the GPS position and his velocity, respectively. The Jacobian matrix of h with respect to X is

$$H = \frac{\partial h(x, v)}{\partial x} = \begin{bmatrix} I^{3\times3} & 0^{3\times3} & 0^{3\times3} \\ 0^{3\times3} & C_{bn} & 0^{3\times3} \end{bmatrix}. \tag{14}$$

C_{bn} is the direct Co-sine transform matrix from body frame to the navigation frame. It is defined as follows:

$$C_{bn} = [A_1 \quad A_2 \quad A_3], \tag{15}$$

where

$$A_1 = \begin{bmatrix} c\theta c\psi \\ s\theta s\phi c\psi - s\psi c\phi \\ s\theta c\phi c\psi + s\psi s\phi \end{bmatrix}, \tag{16}$$

$$A_2 = \begin{bmatrix} c\theta s\psi \\ s\psi s\theta s\phi + c\psi c\phi \\ s\psi s\theta c\phi - c\psi c\theta \end{bmatrix}, \tag{17}$$

$$A_3 = \begin{bmatrix} -s\theta \\ s\phi c\phi \\ s\phi c\theta \end{bmatrix}. \tag{18}$$

2.4. *Inertial navigation system modeling*

An Inertial Navigation System (INS) consists of 3-axis gyroscope, which provide roll, pitch and yaw body angle and a 3-axis accelerometer, which provide accelerations along the three body axes. There are two basic inertial mechanisms, which are used to derive the Euler angles from the rate gyro: stable platform and strap-down INS [31]. We would focus on strap-down INS, where gyros and accelerometers are strapped-down to the aircraft body frame. The INS uses an IMU that measures the accelerations (a_x, a_y, a_z) and the rotation rates (p, q, r) of the robot platform with high update rates, which can then be transformed and processed to provide its position (x, y, z), velocity (u, v, w) and attitude (ϕ, θ, ψ) resulting in an INS. Let us present the INS with following nonlinear model:

$$\dot{X}(t) = f(X(t), U(t), t) + W_K, \tag{19}$$

$$y(t) = h(X(t), U(t), t), \tag{20}$$

where is the state vector, which contains the position, velocity and Euler angles, represents the IMU outputs (angular rates and accelerations) defined as follows:

$$X = [[x \quad y \quad z \quad u \quad v \quad w \quad \phi \quad \theta \quad \psi]]^T, \tag{21}$$

$$U = [p \quad q \quad r \quad a_x \quad a_y \quad a_z]^T. \tag{22}$$

W_K is Gaussian noise. Navigation equation requires the definition of at least two frames, one for the body/inertial representation, and one for the navigation frame representation. Then, the equations of motion can be given by a simple integrations and frame transformations. The Euler angles ϕ, θ, ψ can be calculated as follows:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & s\phi s\theta & c\phi \sec\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{23}$$

Assuming that IMU is mounted on the vehicle center of gravity, the true vehicle acceleration in the body frame is expressed by

$$\begin{bmatrix} \dot{\mu} \\ \dot{\vartheta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} a_x + r - q + gs\theta \\ a_y - r + p - gc\theta c\phi \\ a_z + q - p - gc\phi c\theta \end{bmatrix}. \tag{24}$$

a_x, a_y and a_z: are accelerations. The resulting acceleration vector is integrated with respect to time to obtain the velocity of the vehicle in the body frame:

$$\begin{bmatrix} \mu \\ \vartheta \\ \omega \end{bmatrix} = \int \begin{bmatrix} \dot{\mu} \\ \dot{\vartheta} \\ \dot{\omega} \end{bmatrix} dt. \tag{25}$$

The velocity vector is then integrated to estimate the position of the vehicle in the body frame. If the velocity is transformed down to navigation frame and integrated, we get the position vector $[x \quad y \quad z]$ in the navigation frame as follows:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \int C_{bn(\phi,\theta,\psi)}^T \begin{bmatrix} \mu \\ \vartheta \\ \omega \end{bmatrix} dt. \tag{26}$$

Then, $f(X(t), U(t), t)$ is the nonlinear transition state function given as follows [32,33]:

$$f(X,U) = \begin{bmatrix} C_{bn}^T \begin{bmatrix} u \\ v \\ w \end{bmatrix} \\ \begin{bmatrix} ax + vr - wq + gs\theta \\ ay - ur + wp - gc\theta c\phi \\ az + uq - vp - gc\phi c\theta \end{bmatrix} \\ \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & s\phi \sec\theta & c\phi \sec\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \end{bmatrix}. \quad (27)$$

f is linearized, then the Jacobean matrix is computed with respect to X. It gives

$$\frac{\partial f}{\partial X} = \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} & \frac{\partial f}{\partial z} & \frac{\partial f}{\partial u} & \frac{\partial f}{\partial v} & \frac{\partial f}{\partial w} & \frac{\partial f}{\partial \phi} & \frac{\partial f}{\partial \theta} & \frac{\partial f}{\partial \psi} \end{bmatrix}, \quad (28)$$

$$\begin{bmatrix} \frac{\partial f}{\partial u}, \frac{\partial f}{\partial v}, \frac{\partial f}{\partial w} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 0 \\ -r \\ q \end{bmatrix} & \begin{bmatrix} C_{bn}^T \\ r \\ 0 \end{bmatrix} & \begin{bmatrix} -q \\ p \\ 0 \end{bmatrix} \\ & \begin{bmatrix} -p \\ 0^{2\times 2} \end{bmatrix} & \end{bmatrix}. \quad (29)$$

We consider Q_K to be the covariance matrix of W_K defined as follows:

$$Q_K = BQB^T, \quad (30)$$

where

$$B = \frac{\partial f}{\partial u} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\omega & 1 & 1 & 0 & 0 \\ \omega & 0 & -u & 0 & 1 & 0 \\ -v & u & 0 & 0 & 0 & 1 \\ 1 & s\phi t\theta & c\phi t\theta & 0 & 0 & 0 \\ 0 & c\phi & -s\phi & 0 & 0 & 0 \\ 0 & s\phi \sec\theta & rc\phi \sec\theta & 0 & 0 & 0 \end{bmatrix}, \quad (31)$$

$$Q = \begin{bmatrix} \begin{bmatrix} \sigma_p^2 & 0 & 0 \\ 0 & \sigma_q^2 & 0 \\ 0 & 0 & \sigma_r^2 \end{bmatrix} & \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix} \\ \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix} & \begin{bmatrix} \sigma_{a_x}^2 & 0 & 0 \\ 0 & \sigma_{a_y}^2 & 0 \\ 0 & 0 & \sigma_{a_z}^2 \end{bmatrix} \end{bmatrix}, \quad (32)$$

where σ_p^2, σ_q^2, σ_r^2, $\sigma_{a_x}^2$, $\sigma_{a_y}^2$ and $\sigma_{a_z}^2$ represent the noise variance of the IMU angle rates and accelerations, respectively.

2.5. *Camera modeling*

We assume that the camera is installed in the gravity center of the mobile robot. The position of the camera as well as the mobile robot is estimated using visual odometry algorithm [34, 35]. Then the measurement visual odometer matrix is given by

$$Z_K = [\, x_{\text{cam}} \quad y_{\text{cam}} \quad z_{\text{cam}} \,] = h(X_K) + V_K, \quad (33)$$

where V_K is a Gaussian noise for the measurement model with covariance R.

$$R = \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_z^2 \end{bmatrix}, \quad (34)$$

where σ_x^2, σ_y^2, σ_z^2 represent the noise variance of the camera position measurement. The Jacobian matrix of h with respect to X_{K+1} is given by

$$H_K = \frac{\partial h(x,v)}{\partial x} = I_3. \quad (35)$$

3. Multi Sensor Data Fusion for Robot Localization

The multi-sensor fusion approach merges data from different sensors, and uses their complementarity and redundancy to increase the precision of the localization process. In our architecture, the four sensors presented above are gathered in two couples. The first couple uses data from an INS and an GPS while the second one uses data from odometer (ODO) and the Camera (CAM). Three different fusion filters are proposed in our architecture for robot localization: EKF, SVSF and ASVSF.

3.1. *Extended kalman filter (EKF)*

The extended Kalman filter can be viewed as a nonlinear version of the Kalman filter, it linearizes the process and measurement models around the current estimate.

$$X_K = f(X_{K-1}, U_{K-1}) + W_K, \quad (36)$$

$$Z_K = h(X_K) + V_K. \quad (37)$$

The model and implementation equations for extended Kalman filter are defined as follows: Prediction:

$$\hat{X}_{K/K-1} = f(\hat{X}_K), \quad (38)$$

$$P_{K/K-1} = F(\hat{X}_K) P_K F^T(\hat{X}_K) + Q. \quad (39)$$

Update:

$$K_K = P_{K/K-1} H^T (H P_{K/K-1} H^T + R)^{-1}, \quad (40)$$

$$\hat{X}_K = \hat{X}_{K/K-1} + K_K[Z_K - h(\hat{X}_{K/K-1})], \quad (41)$$

$$P_K = P_{K/K-1} - K_K H P_{K/K-1}, \quad (42)$$

where represents the predicted state vector and $X - K/K - 1$ denotes the estimated state vector, K is the Kalman gain matrix, represents the covariance matrix; F is the Jacobian matrix of the state equation f with respect to the state vector $X - K + 1$ and the noise variable $W - K$, respectively; H is the Jacobian matrix of the measurement h with respect to the statevector $X - K + 1$ and the noise variable $V - K$, respectively [36]. The measurement model describes what the sensors measure. GPS and camera measure the position of the robot, encoder measures the wheel's velocity, and the INS measures the angular velocity and acceleration rate in gyroscope and accelerometer, respectively [4, 32].

3.2. *The smooth variable structure filter (SVSF)*

The smooth variable structure filter is a predictor–corrector estimator based on sliding mode concepts (see Fig. 2). It uses a switching gain to converge to the estimates within a boundary of the true state values. The method is model based and may be applied to differentiable linear or nonlinear dynamic equation [37].

The estimation process is iterative and may be summarized by the following set of equations [38, 39]:

The predicted state:

$$X_{K+1/K} = f(X_{k/k}, U_K). \quad (43)$$

The corresponding predicted measurement:

$$Z_{K+1/K} = h(X_{k+1/k}). \quad (44)$$

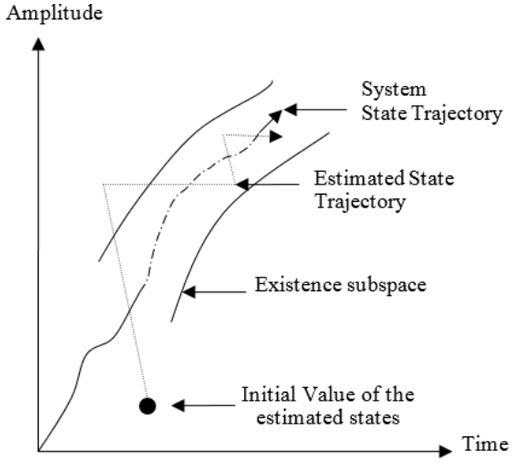

Fig. 2. SVSF estimation concept [35].

The *a priori* measurement error:

$$E_{K+1/K} = Z_{k+1} - Z_{K+1/K}. \quad (45)$$

The (SVSF) gain:

$$K = H + [(|E_{K+1/K}| + \gamma|E_{K/K}|)^{\circ} sate(\varphi^{-1}E_{K+1/K}), \quad (46)$$

where

H: linearized measurement matrix. γ: the convergence rate matrix with $(0 \leq \gamma_{ii} \leq 1)$. φ: the smoothing boundary layer width.

with:

$$\text{sate}(\varphi^{-1}E_{K+1/K}) = \begin{cases} 1 & \varphi_i^{-1}E_{K+1/K}, \\ \varphi_i^{-1}E_{K+1/K} & -1 < \varphi_i^{-1}E_{K+1/K} < 1, \\ -1 & \varphi_i^{-1}E_{K+1/K} \leq -1. \end{cases} \quad (47)$$

The updated state estimate:

$$X_{K+1/K+1} = X_{K+1/K} + K_{K+1}. \quad (48)$$

The updated measurement estimate:

$$Z_{K+1/K+1} = h(X_{K+1/K+1}). \quad (49)$$

The posterior measurement error:

$$Z_{K+1/K+1} = h(X_{K+1/K+1}). \quad (50)$$

3.3. *The adaptive smooth variable structure filter (ASVSF)*

Unlike the standard SVSF, the SVSF with an adaptive boundary layer width (ASVSF) has a covariance matrix, which could be used to determine the optimal gain value presented in [41]. The ASVSF is defined by the following equations:

Initialization:

- The original pose of the coordinate system: X_0.
- Covariance matrix: P_0.
- The posteriori measurement error vector: E_0.

Prediction:

The predicted state:

$$X_{K+1/K} = f(X_{K/K}, U_K). \quad (51)$$

The predicted covariance matrix:

$$P_{K+1/K} = F_x P_{K/K} F_x^T + F_u Q F_u^T, \quad (52)$$

where F_X and F_U are the Jacobian matrix of f with respect to X and U. Q is the Covariance matrix. The *a priori* measurement error:

$$E_{K+1/K} = Z_{K+1} - h(X_{K+1/K}). \quad (53)$$

The optimal width of the boundary layer:

$$\varphi^{\text{opt}} = (H_{K+1}P_{K+1}H_{K+1}^{T} + R)(H_{K+1}P_{K+1}H_{K+1}^{T})^{-1} \times [\text{diag}(|E_{K+1/K}| + \gamma|E_{K/K}|], \quad (54)$$

where

R is the Covariance matrix. H is the Jacobian matrix.

The (ASVSF) gain:

$$K_{K+1} = H^{+}\text{diag}[(|E_{K+1/K}| + \gamma|E_{K/K}|)^{\circ} \times \text{sate}(\varphi^{\text{opt}-1}E_{K+1/K})][\text{diag}(E_{K+1/K})]^{-1}. \quad (55)$$

The updated state estimate:

$$X_{K+1/K+1} = X_{K+1/K} + K_{K+1}E_{K+1/K}. \quad (56)$$

The predicted covariance matrix:

$$P_{K+1/K+1} = (I - K_{K+1}H_{K+1})P_{K+1/K}(I - K_{K+1}H_{K+1})^{T} + K_{K+1}RK_{K+1}. \quad (57)$$

The posterior measurement error:

$$E_{K+1/K+1} = Z_{K+1} - h(X_{K+1/K+1}). \quad (58)$$

4. Fault Tolerant Localization Approach

The proposed architecture for fault tolerant localization is based on multiple sensors data fusion, duplication and comparison approaches. The three different filters mentioned above have been evaluated separately in order to diagnosis and recover to an operational mode following the occurrence of some hardware (sensor) or software faults. The conventional duplication/comparison method has been designed by implementing simultaneously four-dependent parallel branches for data fusion using heterogeneous sensors data as inputs, to obtain the operation safety system. The dependency between the branches in term of sensor allows to solve the FDI problem based on structured residuals method. This dependency is also used to minimize the number of sensors for the robot navigation. Therefore, it induces minimization of hardware faults. Indeed, our approach is applied for the fault tolerant localization of a unicycle mobile robot type Pioneer 3DX. In our approach, we assume the case of a single fault while the multiple simultaneous faults were not considered. The present configuration allows to tolerate hardware faults in sensors and also software faults in data fusion algorithms. The detailed FDI and accommodation architecture is presented in Fig. 3, and the adopted configuration follows the four steps summarized in what follows.

4.1. *Detection*

The faults detection algorithm follows three steps:

- Step 1: Data acquisition from the Pioneer 3DX sensors (ODO, GPS, INS, CAM) simulated under the V-REP software;
- Step 2: Data fusion based on EKF in the first architecture, SVSF in the second and ASVSF in the third architecture in order to overcome the weakness of each sensor. It is

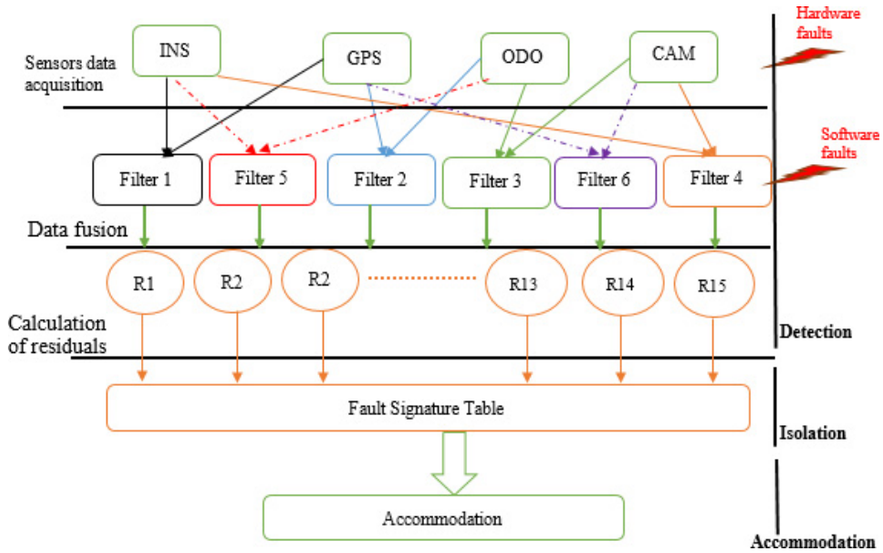

Fig. 3. Synoptic diagram: Fault detection based on multi sensor data fusion.

based essentially on six-dependent parallel data fusion branches:

- Branch F1: It receives data from an INS sensor (prediction model) and from the GPS sensor (correction model);
- Branch F2: It receives data from an Odometer sensor (prediction model) and from the GPS sensor (correction model);
- Branch F3: It receives data from an Odometer sensor (prediction model) and from the Camera sensor (correction model);
- Branch F4: It receives data from an INS sensor (prediction model) and from the Camera sensor (correction model);
- Branch F5: complementary filter between INS sensor and Odometer sensor;
- Branch F6: complementary filter between GPS sensor and Camera sensor.

- Step 3: Residuals calculation and comparison based on the distances calculation between the four branches outputs as follows:

$$R_i = \sqrt{(x_j - x_k)^2 + (y_j - y_k)^2}, \tag{59}$$

where i, j and k are defined in Table 1.

This means that the deviations between the branches are used for the FDI. The Euclidean distance is chosen as the method of calculating these distances since it is the most commonly used distance and the most useful method to deal with fault detection. If a residual exceeds its predefined threshold, this implies that a significant difference is produced between its two corresponding branches, which reveals the presence of a fault in the system. Fixed thresholds Thrdi are defined by considering the nominal operation mode using the three-sigma method proposed in [41] and used in [42] for each residual. This method is based on the calculation of the average mu_i and σ the standard deviation of each residual signal R_i, $i = 1, 2, \ldots, 15$, as follows:

$$\mathrm{Thrd}_i = \mu_i + 3\sigma_i, \tag{60}$$

which guarantees that the probability to Ri falling away its mean by more than 3 is at most 5% in nominal operation mode and consequently 95% in faulty mode.

$$\Pr(|R_i - \mu_i| > 3\sigma_i). \tag{61}$$

4.2. *Isolation*

When a fault is detected, we proceed to the fault isolation, in order to identify the faulty component in the decision making and to deduce the correct output of the system. For this purpose, the residual vector must have a number of properties to uniquely characterize each fault. The residuals are designed to be affected by a subset of faults and robust

Table 1. Advantages and weakness of filters.

	Strengths	Weakness
EKF	Extension of the Kalman Filter;	Requires model linearization which likely yield inaccurate and may cause the filter to diverge when the process and observation models are highly nonlinear [29];
	Suitable for systems that are mildly nonlinear;	Unreliable estimates when the characteristics of process and observation noise are not centered Gaussian noise;
	First-order Taylor series expansions;	Linearization using Jacobians can sometimes cause instabilities when implementing the filter;
	Suboptimal filter for nonlinear models [26].	Difficult to implement and tune.
SVSF	Robustness to modeling errors and uncertainties;	Limited to bounded uncertainties;
	Forces the state estimates to vary within a boundary around the true state trajectory;	Requires a trade-off between robustness to modeling uncertainties and estimation accuracy [29];
	More robust to modeling errors and uncertainties, and stable compared with the Kalman Filter [29];	Requires an accurate system model and known noise statistics to approximate the posterior state;
	Does not require covariance matrix;	A constant and typically large boundary layer width may the filter to lose accuracy.
	Internal model of the system, either linear or nonlinear [30]	
ASVSF	Enhanced optimality by calculating an optimal time-varying smoothing boundary layer;	Its covariance is complicatedly formulated;
	Very robust estimator against modeling errors and noises;	It might cause a negative definite symmetric matrix [2].
	Keeps a compromise between robustness and accuracy [28];	

Table 2. Residuals indices.

i	1	2	3	4	5	6	7	8
(j,k)	(1,2)	(2,3)	(1,3)	(4,3)	(4,2)	(4,1)	(5,1)	(5,2)
i	9	10	11	12	13	14	15	
(j,k)	(5,3)	(5,4)	(5,6)	(6,1)	(6,2)	(6,3)	(6,4)	

to the rest of faults. Thus, one subset of residuals reacts when a fault occurs. The signature table contains the sensitivity and robustness information of the residuals. It is defined as follows:

$$\begin{cases} 1 & \text{if } R_i > \text{Thrd}_i, \\ 0 & \text{if } R_i < \text{Thrd}_i. \end{cases} \quad (62)$$

In this paper, the columns of the signature table are presented in Table 2, corresponding to the faults (sensor and software faults) while the rows correspond to the residuals. Considering only one defective sensor or software at a time, the fault can be isolated (identified) using the signature table, because each sensor or software fault signature (column vector) is different from the other fault signatures.

4.3. *Accommodation*

The accommodation which aims to ensure the fault recovery consists to switch from the faulty to the fault-free branch.

- If a sensor fault is detected in the INS, the system output will be the average of the second and the third output filters.
- If a sensor fault is detected in the GPS, the system output will be the average of the third and fourth branches.
- If a sensor fault is detected in the ODO, the system output will be the average of the first and the fourth branches.
- If a sensor fault is detected in the CAM, the system output will be the average of the first and the second branches.
- If a software is detected in the first filter, the system output will be the average of the second, the third and the fourth branches.
- If a software is detected in the second filter, the system output will be the average of the first, the third and the fourth branches.
- If a software is detected in the third filter, the system output will be the average of the first, the second and the fourth branches.
- If a software is detected in the fourth filter, the system output will be the average of the first, the second and the third branches.

5. Simulation and Results

In this section, we present a set of simulations in order to assess the architectures described above in presence of faults. This section contains three parts: at first, the software environment will be presented, while in the second part we describe two types of faults considered in our evaluation. The third part is dedicated to faults injection and the obtained results.

5.1. *Software environment*

In our study, we use the Virtual Robot Experimentation Platform (V-REP), as physical simulator, which provides an easy and intuitive environment to create a virtual platform and to include some popular robots, objects, structures, actuators and sensors. V-REP is a product of Coppelia Robotics [43] developed for general purpose robot simulation. For simulator in the loop (SIL) configuration tests, V-REP allows to control the simulation from outside by external implied controller algorithm. The controller communicates with the simulation scene using a communication socket and approximately one hundred functions that can be called from Matlab program. In our application the command and controller part are developed in Matlab while sensors and physical interaction are created and simulated in V-REP. A snapshot of the V-REP environment is shown in Fig. 4.

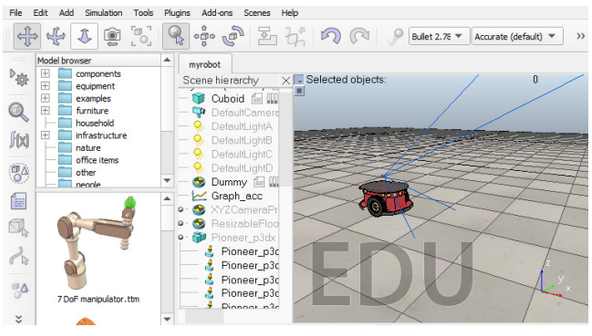

Fig. 4. V-REP software platform.

5.2. *Faults*

Hardware faults are simulated by altering and modifying sensor output values, while software faults are simulated by modifying models and gains of the data fusion algorithms.

- Hardware faults: The sensors embedded in our mobile robot may be subject to three categories of hardware faults: jump faults, frozen data faults, and null faults, these faults are modeled and detailed in [3].
- Software faults: These faults are in general software development faults, generally caused by code implementation errors. Classical examples of software faults include: incorrect assignment of a value; checking absence or incorrect validation of data or loops or conditions; incorrect algorithm or missing implementation; incorrect function or missing parameters. In our experiments, we consider the software of incorrect assignment of the noise covariance matrices.

5.3. *Simulations, results and discussion*

Simulations have been performed by considering a model of the pioneer 3DX mobile robot. In order to evaluate the proposed approach, two parameters are measured:

- Fault detection delay, defined as follows:

$$D_{\text{Det}} = t_{\text{Det}} - t_{\text{inj}}, \tag{63}$$

  where t_{inj} and t_{Det} represent the fault injection time and the fault detection time, respectively
- Fault isolation delay, defined as follows:

$$D_{\text{Isol}} = t_{\text{Det}} - t_{\text{Isol}}, \tag{64}$$

  where t_{Isol} represents the fault isolation time t_{Iso}.

5.3.1. *Scenario* 1: *Nominal behavior*

First, we perform a set of tests without faults in order to characterize the nominal behavior of the localization system for a circular trajectory of the robot. The obtained results enabled us to set the different faults detection thresholds. These results served also as reference sample for comparison with the obtained behaviors in presence of faults. The estimated position obtained from the three proposed architectures are shown in (Fig. 5) and the different branches are shown in (Figs. 6–8). These figures show that the three proposed fusion filters allow to estimate the robot localization. The mean square errors (MSE) are reported in Table 4. These results show the superiority of the ASVSF filter which corroborate with the literature.

We have observed that our data fusion algorithms (EKF blocks, SVSF blocks and ASVSF blocks) reach well results in terms of the robot-estimated localization.

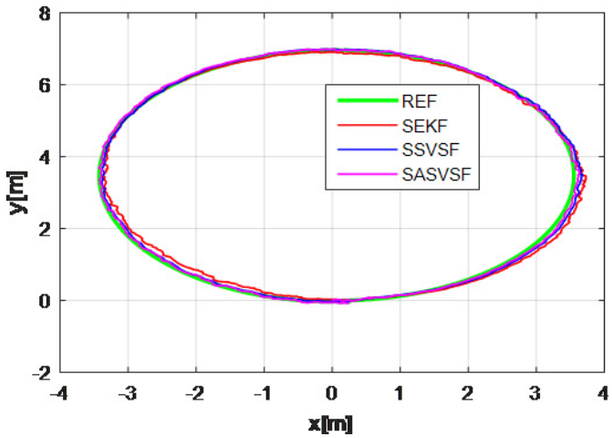

Fig. 5. Nominal behavior: Position estimated by the three systems outputs and the reference trajectory.

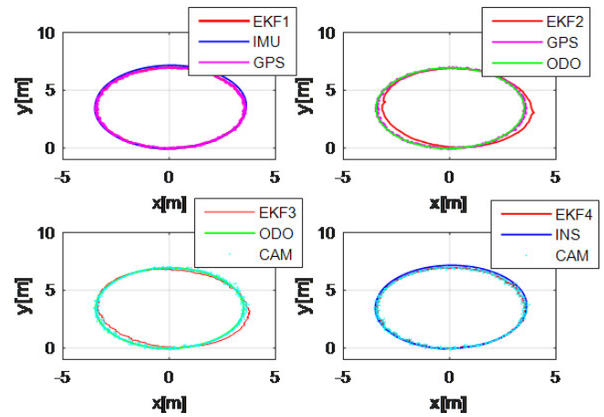

Fig. 6. Nominal behavior: Position estimated by the fourth EKF branches outputs and the sensors outputs.

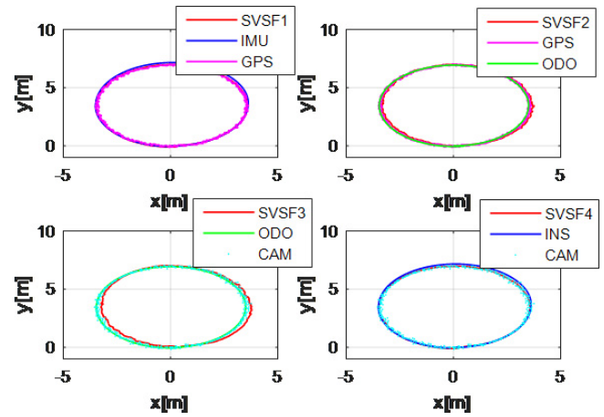

Fig. 7. Nominal behavior: Position estimated by the fourth SVSF branches outputs and the sensors outputs.

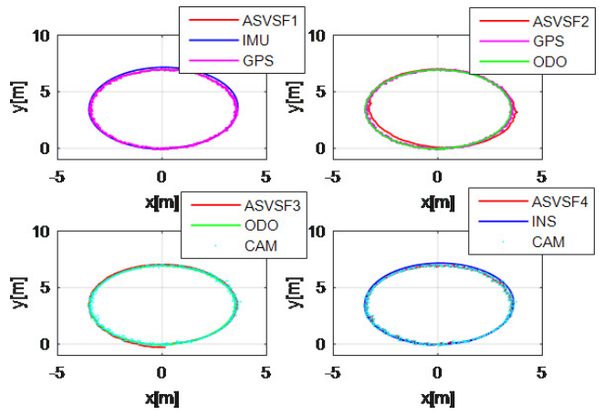

Fig. 8. Nominal behavior: Position estimated by the fourth ASVSF branches outputs and the sensors outputs.

Table 3. Signature table.

Residus	INS	GPS	ODO	CAM	F1	F2	F3	F4	F5	F6
R1	1	1	1	0	1	1	0	0	0	0
R2	0	1	1	1	0	1	1	0	0	0
R3	1	1	1	1	1	0	1	0	0	0
R4	1	0	1	1	0	0	1	1	0	0
R5	1	1	1	1	0	1	0	1	0	0
R6	1	1	0	1	1	0	0	1	0	0
R7	1	1	1	0	1	0	0	0	1	0
R8	1	1	1	0	0	1	0	0	1	0
R9	1	0	1	1	0	0	1	0	1	0
R10	1	0	1	1	0	0	0	1	1	0
R11	1	1	1	1	0	0	0	0	1	1
R12	1	1	0	1	1	0	0	0	0	1
R13	0	1	1	1	0	1	0	0	0	1
R14	0	1	1	1	0	0	1	0	0	1
R15	1	1	0	1	0	0	0	1	0	1

Figures 9–11 show the different distances (Residuals signals) from three architectures (Rekfi, Rsvsfi and Rasvsfi) which are below the corresponding fault detection thresholds (Thrdi, Thrdsi and Thrdasi) We noted: All thresholds have been chosen to be higher than the observed gaps between the different sensors and filters in the nominal behavior.

Table 4. MSE of the proposed fusion filters in nominal conditions.

Architectures	Branche1	Branche2	Branche3	Branche4
MSE 1 (EKF)	0.0028	0.4268	0.4198	0.0025
MSE 2 (SVSF)	0.0015	0.4107	0.3867	0.0020
MSE 3 (ASVSF)	8.947 10-4	0.0066	0.0187	0.011

5.3.2. *Scenario* 2: *Additive faults on GPS*

GPS suffers from several errors due to environmental factors involved in determining a position of a given user. These factors cause jumps in the position provided by the GPS. We added the output of the GPS sensor at distinct instants, in order to reveal the influence of the jump on the estimate given by the filters: EKF SVSF and ASVSF, by studying the behavior of the system and reaction to these different jumps.

$$X_{\text{GPS}} = X_{\text{GPS}} + \text{Jump}. \tag{65}$$

We simulated tow different jumps (5, 15) injected in both directions (X, Y), respectively.

Case 1: Fault injection at time $t_{\text{inj}} = 280\,\text{s}$.

- For the first architecture, as it can be seen from the residual signals presented in Fig. 12 and the estimated robot localization using the defective branches (INS/GPS) and (ODO/GPS) shown in Fig. 13, the fault detection was done when R_1 exceeds the threshold Trhd_1, at time $t_{\text{Det}} = 280\,\text{s}$ $((D_{\text{Det}} = 0\,\text{s}))$. We note that the fault isolation is not carried out, when we obtained signature vector does not correspond to any fault (hardware nor software fault).
- For the second architecture: as can be seen from the obtained residuals signals, presented in Fig. 14. The fault is detected, when the distance R_2 exceeds the threshold Thrds2 at time $t_{\text{Det}} = 280\,\text{s}$ and $D_{\text{det}} = 0\,\text{s}$. Based on the fault isolation method, we notice that the fault isolation is carried at time $t_{\text{Isol}} = 280\,\text{s}$, when we obtained signature vector is [0 1 1 0 1 1 1 1 0 0 1 1 1 1 1], corresponding to a fault in the GPS sensor (hardware fault). The estimated robot localization using the defective branches (INS/GPS) and (ODO/GPS) is shown in Fig. 15. Our fault recovery approach allows to switch to another fault-free branch, which does not use the faulty sensor (GPS) data as input. From the fault recovery results presented in Fig. 18, it can be seen that the injected fault is correctly detected, isolated and recovered.

  Third architecture: As can be seen from the obtained results (residual signals) presented in Fig. 16, The fault detection was done when R_3 exceeds the threshold Trhd_3, at time $t_{\text{Det}} = 280\,\text{s}$ $((D_{\text{Det}} = 0\,\text{s}))$. We note that the fault isolation is not carried the, when we obtained signature vector, do not corresponding to a fault in any sensor (hardware fault) or in any filters (software fault). So, the estimated robot localization using the defective branches (INS/GPS) and (ODO/GPS) is shown in Fig. 17. The not fault recovery is shown in Fig. 18.

  In this case, we noticed the high sensitivity of the SVSF to sudden disturbances compared to the two other filters, while we notice that the EKF and the ASVSF took more time to react to the same type of fault.

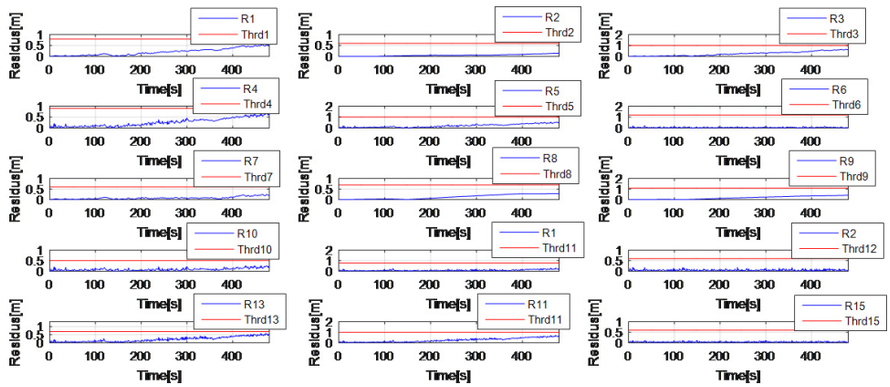

Fig. 9. Nominal behavior: Distance between the positions given by the six EKF-Filterss.

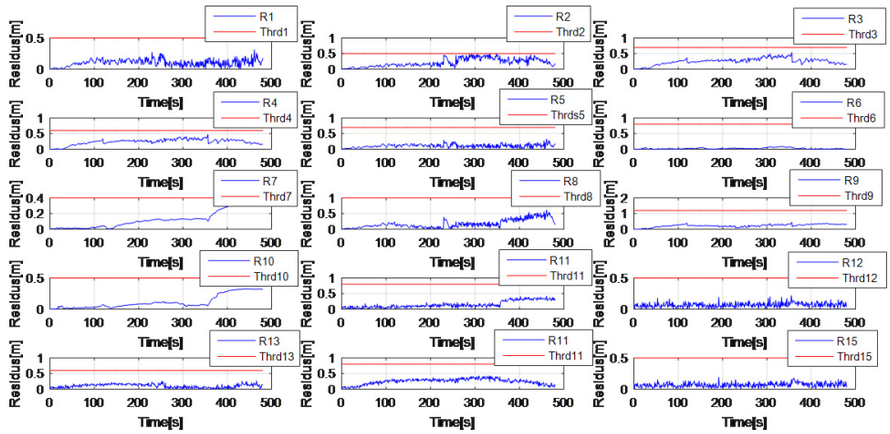

Fig. 10. Nominal behavior: Distance between the positions given by the six SVSF-Filters.

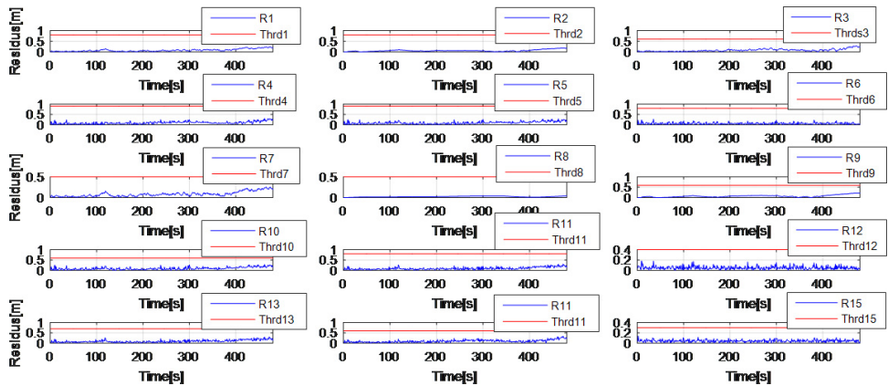

Fig. 11. Nominal behavior: Distance between the positions given by the six ASVSF-Filters.

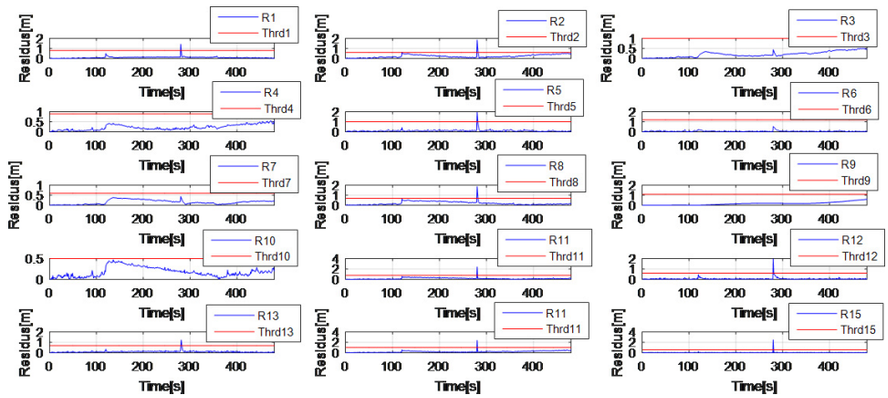

Fig. 12. Additive faults on GPS: Distance between the positions given by the six EKF-Filters.

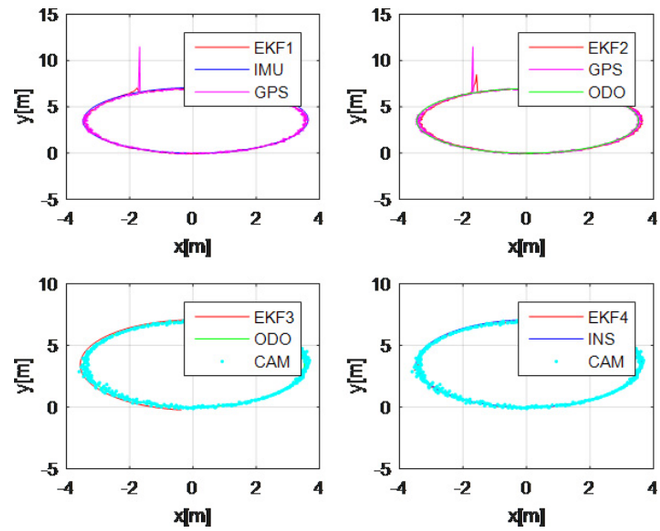

Fig. 13. Additive faults on GPS: Position estimated by the fourth EKF branches outputs and the sensors outputs.

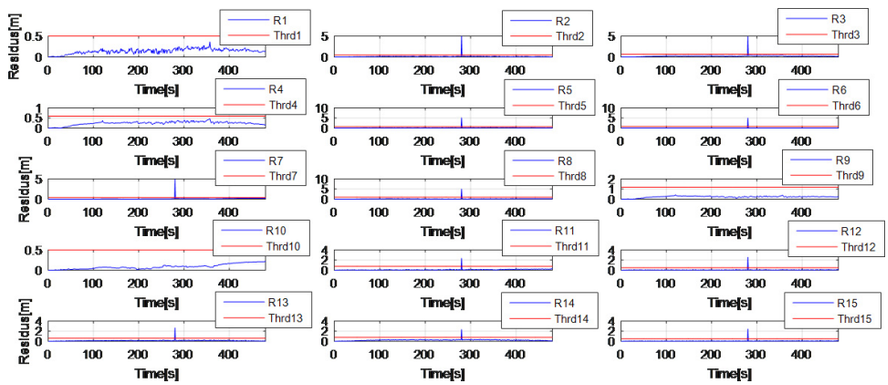

Fig. 14. Additive faults on GPS: Distance between the positions given by the six SVSF-Filters.

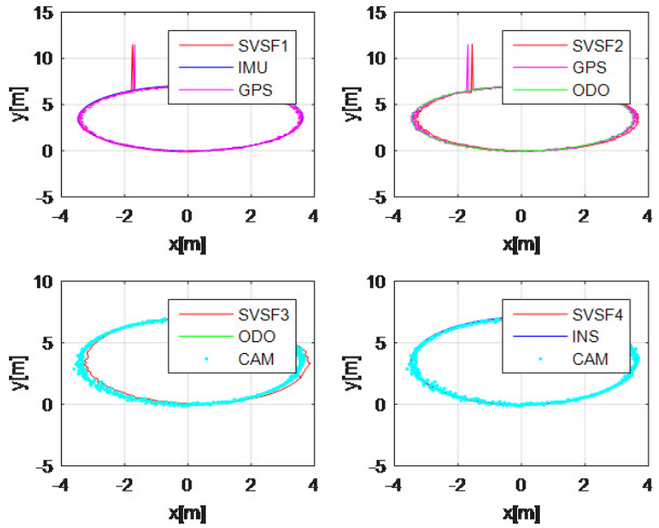

Fig. 15. Additive faults on GPS: Position estimated by the fourth SVSF branches outputs and the sensors outputs.

Case 2: For the second case where the fault has been injected at time $t_{\text{inj}} = 180\,\text{s}$, the fault is detected when the distance R_1 exceeds the threshold Thrd_1 (Figs. 19–21) at time $t_{\text{Det}} = 180\,\text{s}$ then $D_{\text{Det}} = 0\,\text{s}$. Based on the fault isolation method, we notice that the fault isolation is carried at time $t_{\text{Isol}} = 180\,\text{s}$ the $D_{\text{Isol}} = 0\,\text{s}$, when we obtained signature vector is [1 1 1 0 1 1 1 1 0 0 1 1 1 1 1], corresponding to a fault in the GPS sensor (hardware fault). As shown Figs. 22–24, which represent the estimated robot localization using the defective branches (INS/GPS) and (ODO/GPS, our fault recovery approach allows to switch to another fault-free branch, which does not use the faulty sensor (GPS) data as input. From the fault recovery results presented in Fig. 25, it can be seen that the injected fault is correctly detected, isolated and recovered.

5.3.3. *Scenario* 3: *The blocked frame fault in camera sensor*

For third scenario, we have injected in the camera sensor the blocked frame fault, mentioned in (Sec. 5.2). For blocked frame faults, the sensor always sends the same output from a precise time t_{inj}.

$$S_{\text{CAM}}(t_k) = S_{\text{CAM}}(t_{\text{inj}}) \quad t_k > t_{\text{inj}}. \tag{66}$$

This type of fault is frequent in measuring devices, following any incident, a sensor can be stuck on its last output at any time in its life cycle. A typical example of this fault is the blocking of a camera on the last image acquired. The third injection is a permanent fault in camera sensor. At time $t_{\text{inj}} = 240\,\text{s}$, the camera is blocked at the last position providing the same data until the end of the robot mission.

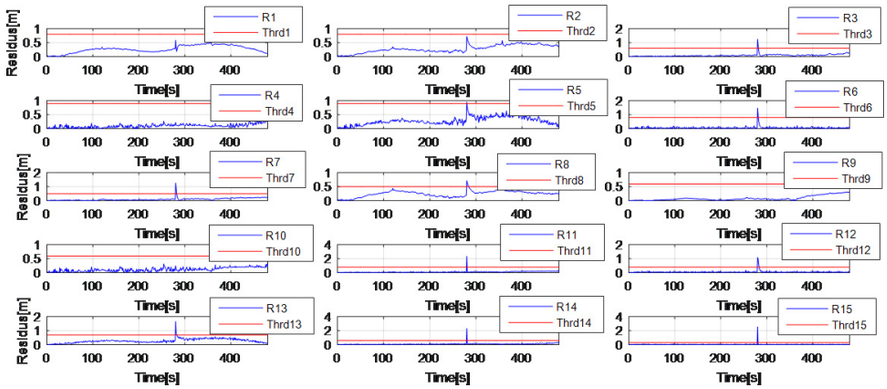

Fig. 16. Additive faults on GPS: Distance between the positions given by the six ASVSF-Filters.

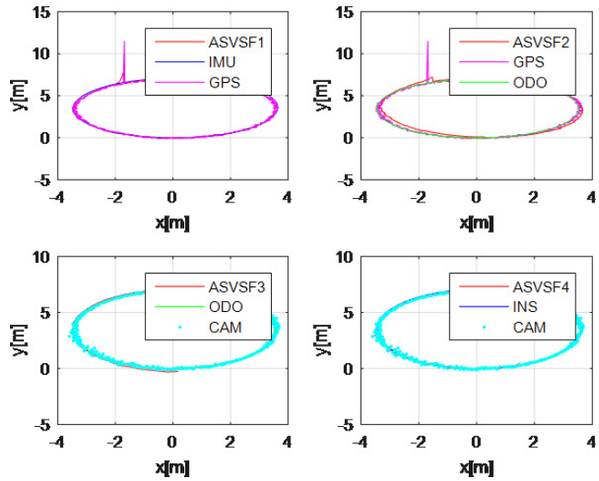

Fig. 17. Additive faults on GPS: Position estimated by the fourth ASVSF branches outputs and the sensors outputs.

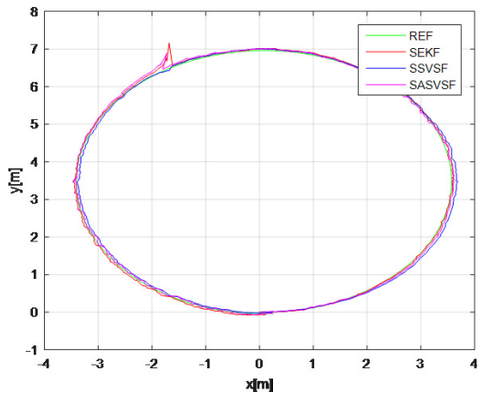

Fig. 18. Additive faults on GPS: Position estimated by the three systems outputs and the reference trajectory.

- First architecture: The distances Ri are presented in Fig. 26, indicating a fault in the localization system. The fault is detected at time $t_{\text{Det}} = 250\,\text{s}$ when R_{10} exceeds the threshold $\text{Thrd}_{10}(D_{\text{Det}} = 10\,\text{s})$. The system identified the camera as a faulty sensor when R3 exceeds the threshold Thd3 at time $t_{\text{Iso}} = 310\,\text{s}(D_{\text{Isol}} = 60\,\text{s})$. Figure 27 shows the robot localization given by the defective sensor (F3–F4). The fault recovery is shown in Fig. 32, using the fault-free branch.
- Second architecture: The residuals Ri are presented in Fig. 28, indicating a fault in the localization system. The fault is detected at time $t_{\text{Det}} = 246\,\text{s}$ when R_2 exceeds the threshold $\text{Thd}_2(D_{\text{Det}} = 6\,\text{s})$. The system identified the camera as a faulty sensor when R_{13} exceeds the threshold Thrd_{13} at time $t_{\text{Iso}} = 270\,\text{s}(D_{\text{Isol}} = 24\,\text{s})$. Figure 29 shows the robot localization given by the defective sensor (F3–F4). The fault recovery is shown in Fig. 32, using the fault-free branch for the remaining robot mission.
- Third architecture: The distances Ri are presented in Fig. 30, indicating a fault in the localization system. The fault is detected at time $t_{\text{Det}} = 250\,\text{s}$ when R_{10} exceeds the threshold $\text{Thrd}_{10}(D_{\text{Det}} = 10\,\text{s})$. The system identified the camera as a faulty sensor when Rasvsf13 exceeds the threshold Thd13 at time $t_{\text{Iso}} = 276\,\text{s}(D_{\text{Isol}} = 26\,\text{s})$. Figure 31 shows the robot localization given by the defective sensor (F3–F4). The fault recovery is shown in Fig. 32, using the fault-free branch for the remaining robot mission.

The reaction of the three filters is almost similar to this type of fault, since the three different filters diverge when the camera is blocked.

In the three architectures, based on these results, it can be concluded that the injected fault in camera output is detected, isolated and recovered.

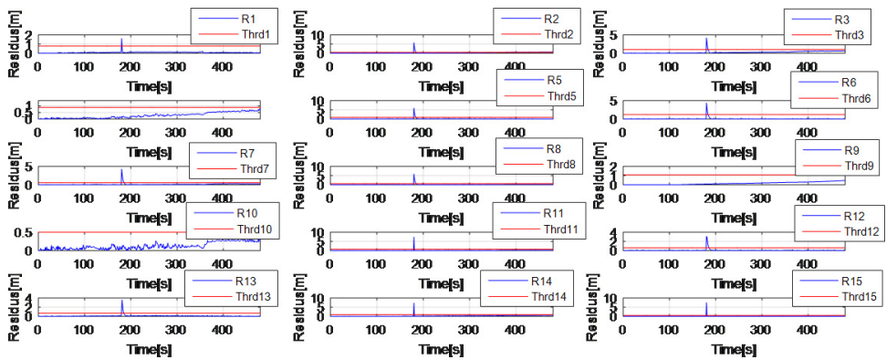

Fig. 19. Additive faults on GPS: Distance between the positions given by the six EKF-Filters.

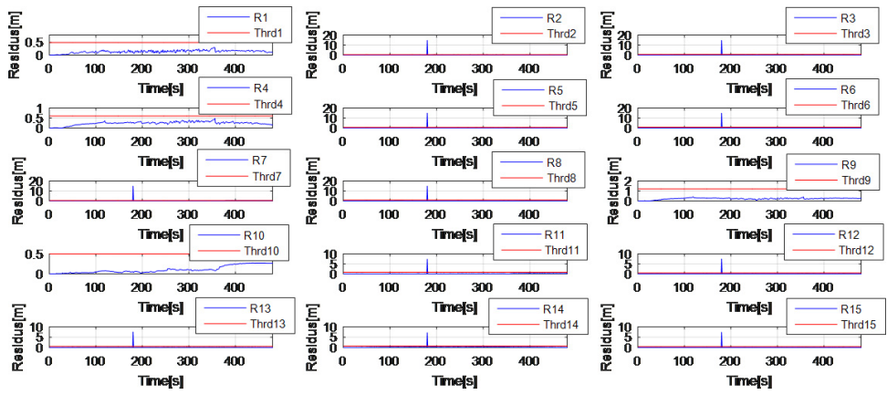

Fig. 20. Additive faults on GPS: Distance between the positions given by the six SVSF-Filters.

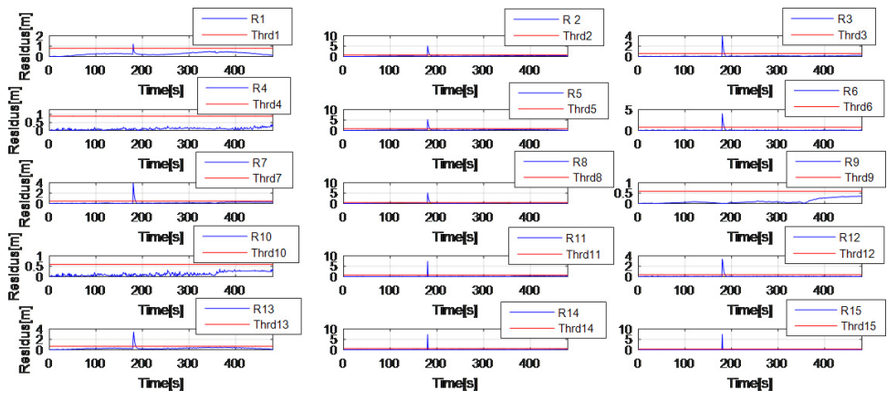

Fig. 21. Additive faults on GPS: Distance between the positions given by the six ASVSF-Filters.

5.3.4. *Scenario* 4: *The zeroing fault on odometer*

In the third scenario, we have injected the zeroing fault on odometer sensor, the sensor always delivers the same output (which we consider here zero).

$$S_{\mathrm{ODO}}(t_k) = 0 \quad t_k > t_{\mathrm{inj}}. \tag{67}$$

This fault does not differ much from the frame-blocked fault, but this time instead of the sensor getting stuck on its last measurement, here it is stuck on a zero output. This type of fault is representative of the skating or sliding of a mobile robot during its mission.

We inject the fault on the odometer at time $t_{\mathrm{Inj}} = 230\,\mathrm{s}$.

- First architecture: The obtained distances R_i are shown in

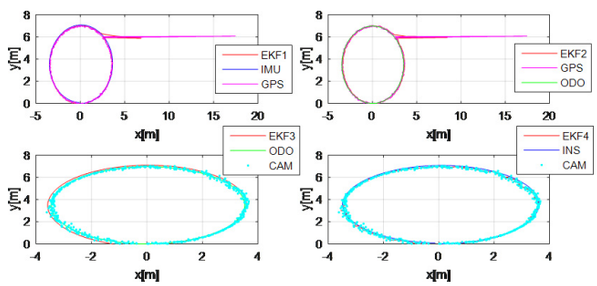

Fig. 22. Additive faults on GPS: Position estimated by the fourth EKF branches outputs and the sensors outputs.

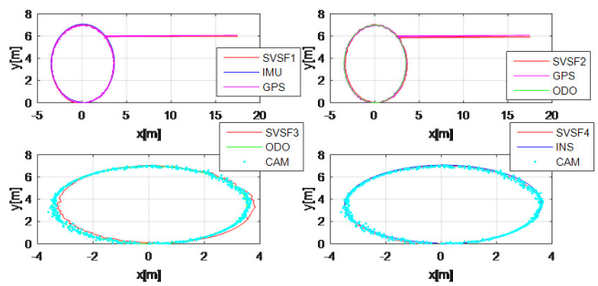

Fig. 23. Additive faults on GPS: Position estimated by the fourth SVSF branches outputs and the sensors outputs.

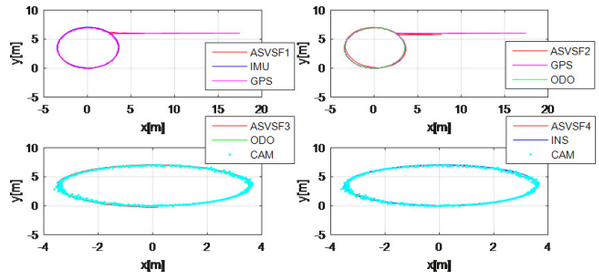

Fig. 24. Additive faults on GPS: Position estimated by the fourth ASVSF branches outputs and the sensors outputs.

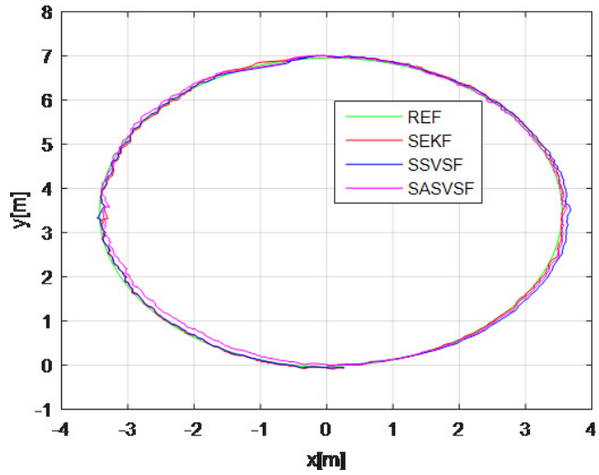

Fig. 25. Additive faults on GPS: Position estimated by the three systems outputs and the reference trajectory.

Fig. 33. The fault detection was done when R_1 exceeds the threshold Trhd_1, at time $t_{\text{Det}} = 230\,\text{s}(D_{\text{Det}} = 0\,\text{s})$. We note that the fault isolation is carried out at time $t_{\text{Iso}} = 230\,\text{s}(D_{\text{Isol}} = 0\,\text{s})$ when the rest of the thresholds which means the signature exceeds their own threshold at the same time. Indicating that the faulty sensor is the

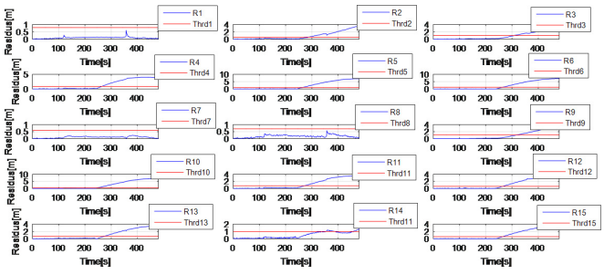

Fig. 26. The blocked frame fault in camera sensor: Distance between the positions given by the six EKF-Filters.

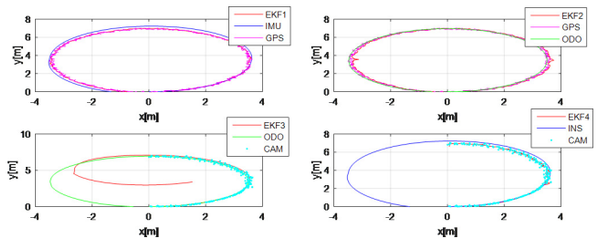

Fig. 27. The blocked frame fault in camera sensor: Position estimated by the fourth EKF branches outputs and the sensors outputs.

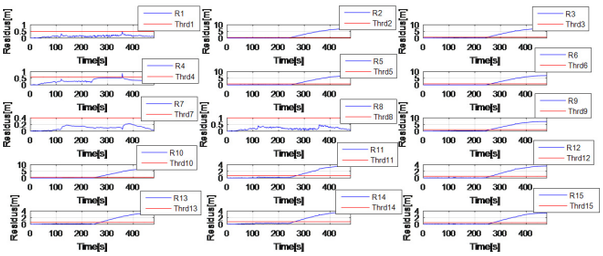

Fig. 28. The blocked frame fault in camera sensor: Distance between the positions given by the six SVSF-Filters.

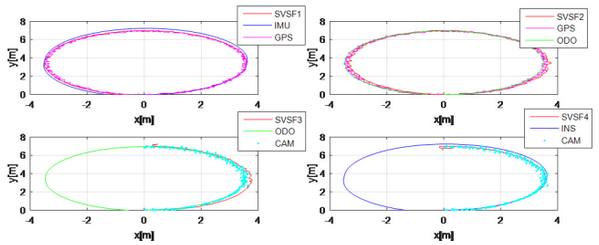

Fig. 29. The blocked frame fault in camera sensor: Position estimated by the fourth SVSF branches outputs and the sensors outputs.

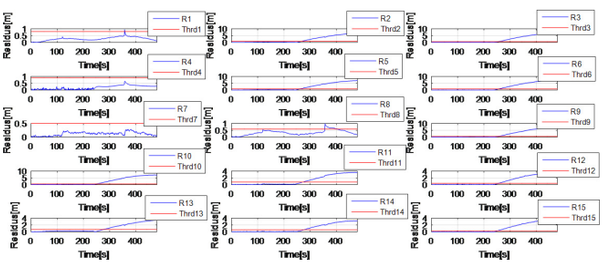

Fig. 30. The blocked frame fault in camera sensor: Distance between the positions given by the six ASVSF-Filters.

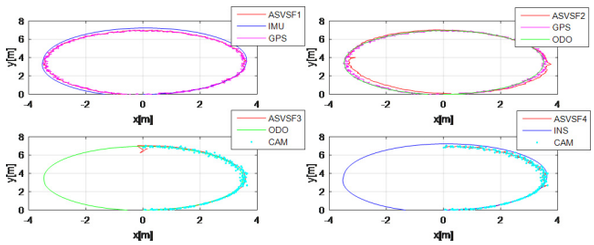

Fig. 31. The blocked frame fault in camera sensor: Position estimated by the fourth ASVSF branches outputs and the sensors outputs.

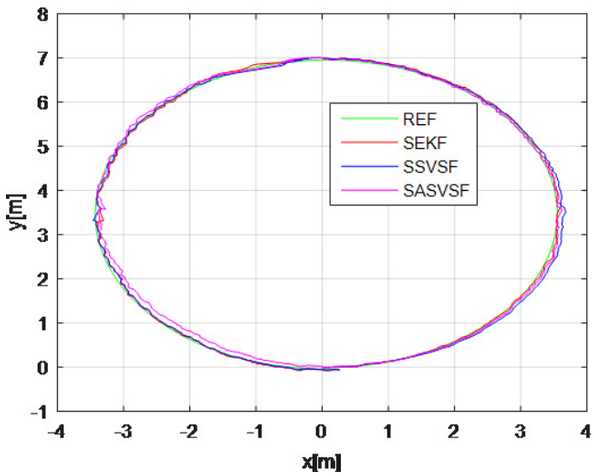

Fig. 32. The blocked frame fault in camera sensor: Position estimated by the three systems outputs and the reference trajectory.

odometer. This defective sensor is presented in Fig. 34. Using the fault-free branches, which does not use the odometer defective data, the fault recovery result is presented in Fig. 39.

- Second architecture: The obtained distances R_i are shown in Fig. 35. The fault detection was achieved when R_7 exceeds the threshold Trhd_7, at time time$t_{\text{Det}} = 230\,\text{s}(D_{\text{Det}} = 0\,\text{s})$. We note that the fault isolation is not carried, the signature of the resulting residues does not bind to any defective sensor, this defective sensor is presented in Fig. 36 when we see that the injected fault does not affect the branches.
- Third architecture: The obtained distances R_i are shown in Fig. 37. The fault detection was done when R_1 exceeds the threshold Trhdas1, at time $t_{\text{Det}} = 230\,\text{s}(D_{\text{Det}} = 0\,\text{s})$. We note that the fault isolation is carried out at time $t_{\text{Iso}} = 230\,\text{s}(D_{\text{Isol}} = 0\,\text{s}))$ when the rest of the thresholds which means the signature exceeds their own threshold at the same time indicating that the faulty sensor is the odometer. The corresponding results are presented in Fig. 38. The fault recovery result is presented in Fig. 39 using the fault-free branches, which does not use the odometer defective data.

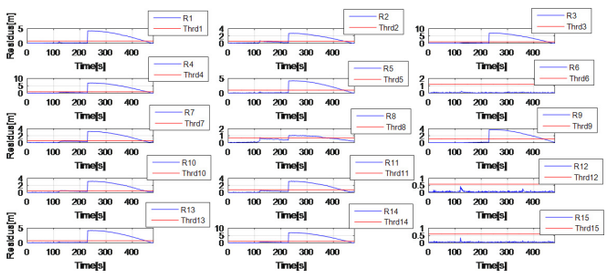

Fig. 33. The zeroing fault on odometer: Distance between the positions given by the six EKF-Filters.

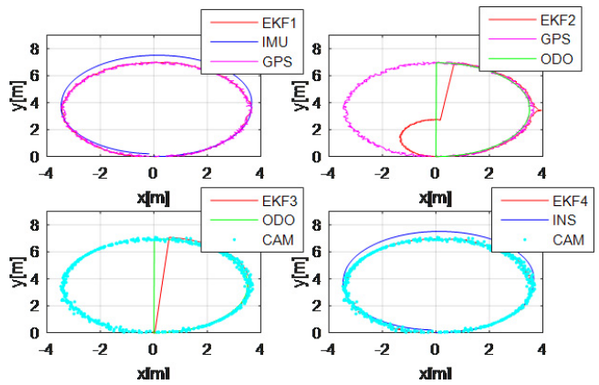

Fig. 34. The zeroing fault on odometer: Position estimated by the fourth EKF branches outputs and the sensors outputs.

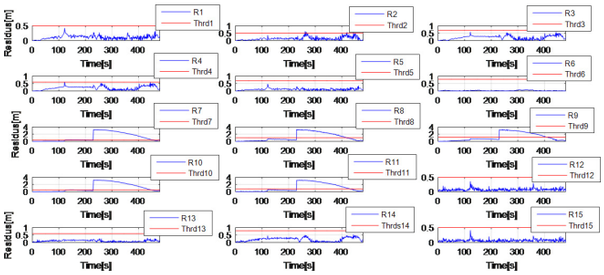

Fig. 35. The zeroing fault on odometer: Distance between the positions given by the six SVSF-Filters.

In this case, we notice that the estimated trajectory by the second filter of first and third architectures diverges, while that of the SVSF is too close to that of the reference trajectory despite the presence of the defect (Null fault in the odometer).

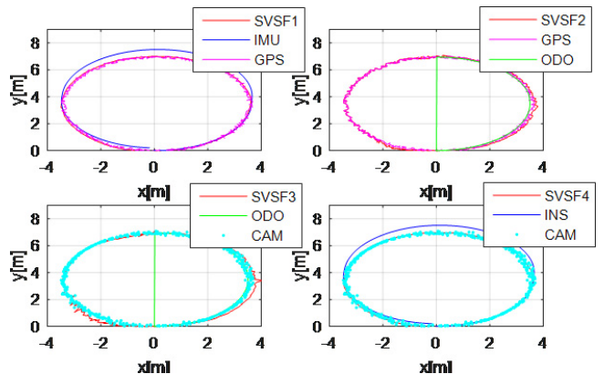

Fig. 36. The zeroing fault on odometer: Position estimated by the fourth SVSF branches outputs and the sensors outputs.

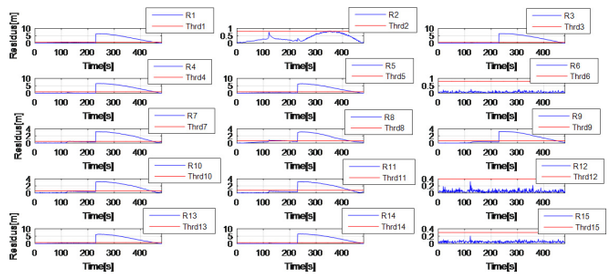

Fig. 37. The zeroing fault on odometer: Distance between the positions given by the six ASVSF-Filters.

It has been found that the detection, isolation and recovery of the encoder fault are reached even before the appearance of the fault impact on estimated localization result. This is due to the right selection of residuals used for FDI in order to minimize the fault detection and isolation delay.

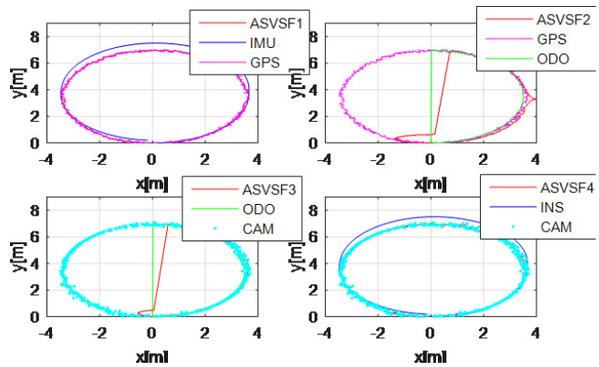

Fig. 38. The zeroing fault on odometer: Position estimated by the fourth ASVSF branches outputs and the sensors outputs.

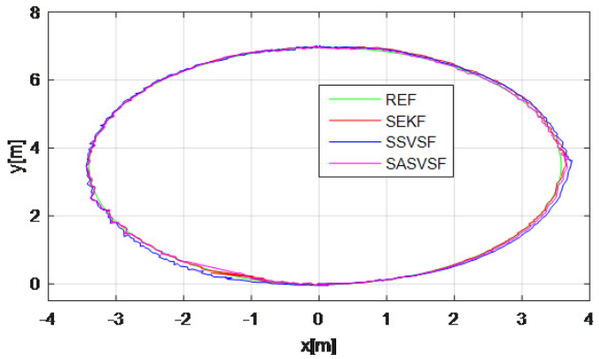

Fig. 39. The zeroing fault on odometer: Position estimated by the three systems outputs and the reference trajectory.

5.3.5. *Scenario* 5: *Software fault: modification in the covariance matrices of the first filter*

In this scenario, we consider the occurrence of a programming permanent fault introduced in the data fusion process F2. Note that both noise covariance matrices Q and R have an important impact on the output of the EKF and ASVSF filters which are generally determined empirically. In this scenario, the fault is simulated by multiplying the measurement noise covariance of the sensor ODO in the covariance matrix Q by a coefficient of 10^3, which results in the reduction of the GPS confidence in the data fusion F2. The fault is injected at the beginning of the mission.

- First architecture: from the obtained distances R_i presented in Fig. 40, it can be shown that a fault has occurred. So, the system detects the fault at time $t_{\text{Det}} = 170\,\text{s}$ when R_8 exceeds the threshold $\text{Thrd}_8(D_{\text{Det}} = 170\,\text{s})$. The isolation of fault is done when $R5$ exceeds the threshold Thrd5 at time $t_{\text{Iso}} = 220\,\text{s}(D_{\text{Isol}} = 50\,\text{s})$, indicating a software fault in the data fusion EKF(ODO/GPS). This defective branch causes system failure. So, the estimated robot localization is presented in Fig. 41. Figure 46 shows the results of our fault recovery approach using the free-fault branch.

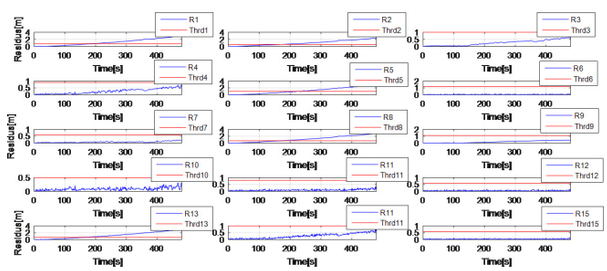

Fig. 40. Software fault in the second filter: Distance between the positions given by the six EKF-Filters.

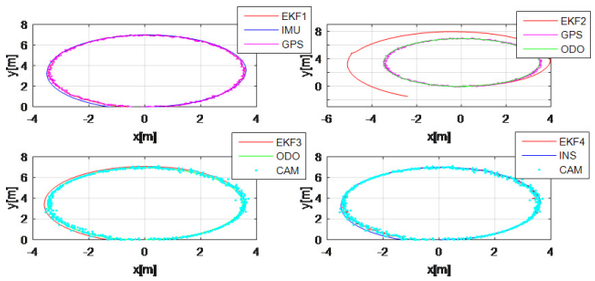

Fig. 41. Software fault in the second filter: Position estimated by the fourth EKF branches outputs and the sensors outputs.

- Second architecture: from the obtained distances R_i presented in Fig. 42, it can be shown that no faults have occurred. Figure 43 shows that all branches are free-fault. That mean the SVSF output remains insensitive to this fault.
- Third architecture: From the obtained distances R_i presented in Fig. 44, it can be shown that a fault has occurred. So, the system detects the fault at time $t_{\text{Det}} = 265\,\text{s} R_8$ exceeds the threshold $\text{Thrd}_8(D_{\text{Det}} = 265\,\text{s})$. The isolation of fault is done when R_5 exceeds the threshold Thrd_5 at time $t_{\text{Iso}} = 328\,\text{s}(D_{\text{Isol}} = 63\,\text{s})$, indicating a software fault in the data fusion EKF(ODO/GPS). The estimated robot localization is presented in Fig. 45, which show that the defective branch causes system failure. Figure 46 shows the results of our fault recovery approach using the free-fault branch.

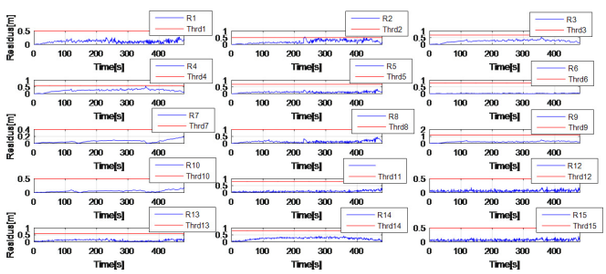

Fig. 42. Software fault in the second filter: Distance between the positions given by the six SVSF-Filters.

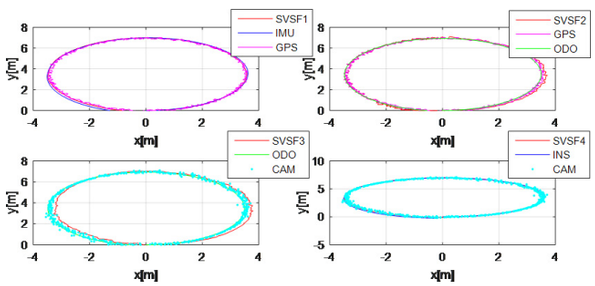

Fig. 43. Software fault in the second filter: Position estimated by the fourth SVSF branches outputs and the sensors outputs.

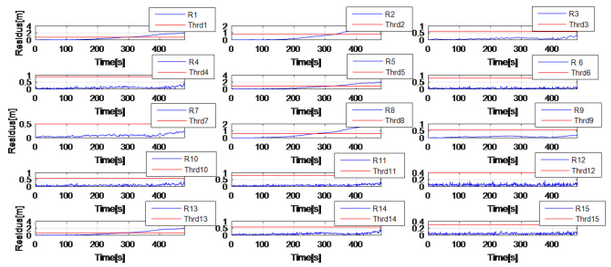

Fig. 44. Software fault in the second filter: Distance between the positions given by the six ASVSF-Filters.

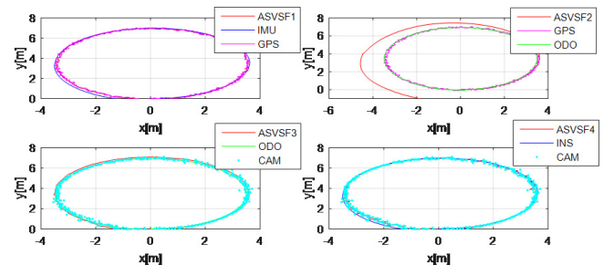

Fig. 45. Software fault in the second filter: Position estimated by the fourth ASVSF branches outputs and the sensors outputs.

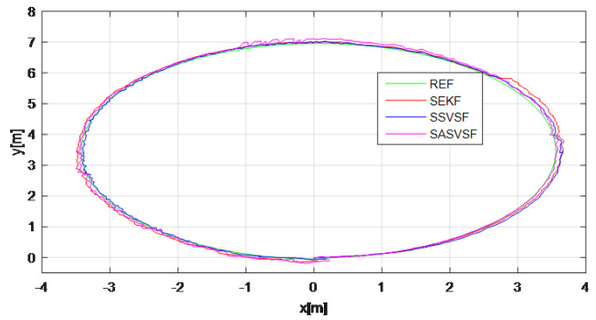

Fig. 46. Software fault in the second filter: Position estimated by the three systems outputs and the reference trajectory.

6. Conclusions and Future Works

We proposed in this work a fault tolerant strategy for mobile robot localization using multi-sensors data fusion and measurement duplication/comparison approach using

residuals. The comparison of six sensor pairs ((IMU/GPS), (Odometer/GPS), (Odometer/Camera), (INS/Camera), (IMU/Odometer) and (GPS/Camera)) outputs with advanced fusion filters (EKF, SVSF and ASVSF) allows to detect and identify any sensor or software faults. Several scenarios have simulated using the V-REP simulation environment. In the nominal case, the three filters give an efficient estimation of the robot trajectory with a light superiority for the ASVSF. It has been observed that in the case of sensor fault, the SVSF and the ASVSF are more sensitive than the EKF. However, in the case of software fault, the SVFS gives better detection results since it doesn't depend on the defected software parameters unlike the EKF and the ASVSF. Thanks to the duplication approach of filters and sensors, the reconfiguration of the localization process is guaranteed in presence of sensor or software faults by choosing the branch that does not contain faults. Future works will focus on the evaluate our architecture using real experimental data and real fault injection and, we intend to enrich this approach by adding other faults.

References

[1] A. Canedo-Rodríguez, V. lvarez-Santos, C. V. Regueiro, R. Iglesias, S. Barro and J. Presedo, Particle filter robot localisation through robust fusion of laser, WiFi, compass, and a network of external cameras, *Inform. Fus.* **27** (2016) 170–188.

[2] Y. Song, S. Nuske and S. Scherer, A multi-sensor fusion MAV state estimation from long-range stereo, IMU, GPS and barometric sensors, *Sensors* **17** (2017) 11.

[3] K. Bader, B. Lussier and W. Schon, A fault tolerant architecture for data fusion: A real application of Kalman filters for mobile robot localization, *Robot. Autonom. Syst.* **88** (2017) 11–23.

[4] E. I. Al Khatib, M. A. Jaradat, M. Abdel-Hafez and M. Roigari, Multiple sensor fusion for mobile robot localization and navigation using the Extended Kalman Filter, *10th Int. Symp. Mechatronics and its Applications (ISMA)* (IEEE, 2015).

[5] L. Hachemi, A. Nemra and M. Guiatni, Fault diagnosis and fault tolerant control for mobile robot based on multi-sensors data fusion, *The Electrical Engineering Int. Conf. EEIC*, Bejaia, Algeria, 2019.

[6] S. Yazdkhasti and J. Z. Sasiadek, Multi sensor fusion based on adaptive Kalman filtering, in *Advances in Aerospace Guidance*, eds. B. Dołęga, R. Głębocki, D. Kordos and M. Żugaj, Navigation and Control (Springer, Cham, 2018).

[7] F. Outamazirt, F. Li, L. Yan and A. Nemra, A new SINS/GPS sensor fusion scheme for UAV localization problem using nonlinear SVSF with covariance derivation and an adaptive boundary layer, *J. Aeronaut.* **29**(2) (2016) 424–440.

[8] L. Cheng, B. Song, Y. Dai, H. Wu and Y. Chen, Mobile robot indoor dual Kalman filter localisation based on inertial measurement and stereo vision, *CAAI Trans. Intell. Technol.* **2**(4) (2017) 173–181.

[9] R. Guan, B. Ristic, L. Wang and R. Evans, Monte Carlo. localisation of a mobile robot using a Doppler-Azimuth radar, *Automatica* **97** (2018) 161–166.

[10] P. Nazemzadeh, D. Fontanelli, D. Macii and L. Palopoli, Indoor localization of mobile robots through QR code detection and dead reckoning data fusion, *IEEE/ASME Trans. Mech.* **22**(6) (2017) 2588–2599.

[11] C. Magrin and E. Todt, Multi-sensor fusion method based on artificial neural network for mobile robot self-localization, *Latin American Robotics Symp. (LARS), Brazilian Symp. Robotics (SBR) and Workshop on Robotics in Education (WRE)* (IEEE, 2019).

[12] S. Behnke, J. Biswas, M. Veloso *et al.*, Multi-sensor mobile robot localization for diverse environments, in *RoboCup 2013*, Lecture Notes in Computer Science, Vol. 8371 (Springer-Verlag, Berlin, Heidelberg, 2014), pp. 468–479.

[13] Y. Li, L. He, X. Zhang, L. Zhu, H. Zhang and Y. Guan, Multi-sensor fusion localization of indoor mobile robot, *IEEE Int. Conf. Real-time Computing and Robotics (RCAR)*, Irkutsk, Russia, 2019, pp. 481–486.

[14] C. Li, H. Sun and P. Ye, Multi-sensor fusion localization algorithm for outdoor mobile robot, *Journal of Physics: Conf. Series, 2nd Int. Conf. Computer Information Science and Artificial Intelligence CISAI*, Vol. 1453, Xi'an, China, 2019, pp. 25–27.

[15] A. Abid, M. Tahir Khan, C. W. de Silva, Fault detection in mobile robots using sensor fusion, *The 10th Int. Conf. Computer Science and Education* (ICCSE Fitzwilliam College, Cambridge University, UK, 2015), pp. 22–24.

[16] X. Li, W. Chen, C. Chan, B. Li and X. Song, Multi-sensor fusion methodology for enhanced land vehicle positioning, *Inform. Fus.* **46** (2019) 51–62.

[17] C. Li, S. Wang, Y. Zhuang and F. Yan, Deep sensor fusion between 2D laser scanner and IMU for mobile robot localization, *IEEE Sens. J.* **21** (2019) 1.

[18] Q. Li, J. P. Queralta, T. N. Gia, Z. Zou and T. Westerlund, Multi-sensor fusion for navigation and mapping in autonomous vehicles: Accurate localization in urban environments, *Unmanned Syst.* **08**(3) (2020) 229–237.

[19] A. Abid and M. Tahir Khan, Multi-sensor multi-level data fusion and behavioral analysis based fault detection and isolation in mobile robots, *8th IEEE Annual Information Technology, Electronics and Mobile Communication Conf. (IEMCON)* (IEEE, 2017).

[20] B. Abci, J. A. Hage, M. E. Badaoui El Najjar and V. Cocquempot, Multi-robot autonomous navigation system using informational fault tolerant multi-sensor fusion with robust closed loop sliding mode control, *21st Int. Conf. Information Fusion (FUSION)* (IEEE, 2018).

[21] N. A. Tmazirte, M. E. E. Najjar, C. Smaili and D. Pomorski, Dynamical reconfiguration strategy of a multi sensor data fusion algorithm based on Information Theory, *IEEE Intelligent Vehicles Symp. (IV)* (IEEE, 2013).

[22] A. Stancu, E. Codres and V. Puig, A fault hiding approach for the sliding mode fault-tolerant control of a non-holonomic mobile robot, *3rd Conf. Control and Fault-Tolerant Systems (SysTol)* Barcelona, Spain, 2016.

[23] B. Wang and Y. Zhang, Adaptive sliding mode fault-tolerant control for an unmanned aerial vehicle, *Unmanned Syst.* **05**(4) (2017) 209–221.

[24] K. Hicham, Tolerance aux defauts via la methode backstepping des systemes non lineaires application: Systeme UAV de type quadrirotor, Ph.D. dissertation, University Ferhat Abbas de Setif (2012).

[25] Z. Liu, C. Yuan, X. Yu and Y. Zhang, Fault-tolerant formation control of unmanned aerial vehicles in the presence of actuator faults and obstacles, *Unmanned Syst.* **4**(3) (2016) 197–211.

[26] F. Outamazirt, L. Yan, F. Li and A. Nemra, Comparing between the performance of SVSF with EKF and NH for the autonomous airborne navigation problem, *2016 IEEE Aerospace Conf.* (IEEE, 2016), pp. 1–8.

[27] S. A. Gadsden and S. R. Habibi, A new form of the smooth variable structure filter with a covariance derivation, *49th IEEE Conf. Decision and Control* (IEEE, 2010).

[28] A. Allam, M. Tadjine, A. Nemra and E. Kobzili, Stereo vision as a sensor for SLAM based Smooth Variable Structure Filter with an adaptive Boundary Layer Width, *6th Int. Conf. Systems and Control (ICSC)* (IEEE, 2017), pp. 14–20.

[29] Z. Dang, T. Wang and F. Pang. Tightly-coupled data fusion of VINS and odometer based on wheel slip estimation, *2018 IEEE Int. Conf. Robotics and Biomimetics (ROBIO)* (IEEE, 2018), pp. 1613–1619.

[30] S. A. Gadsden, D. Dunne, S. R. Habibi and T. Kirubarajan, Comparison of extended and unscented Kalman, particle, and smooth variable structure filters on a bearing-only target tracking problem, *SPIE, Int. Society for Optics and Photonics, Signal and Data Processing of Small Targets*, San Diego, California, 2009, pp. 113–125.

[31] N. Abdelkrim, Robust airborne 3D visual simultaneous localisation and mapping, *Comput. Sci. Publ.* **55** (2009) 345–376.

[32] H. Zhao and Z. Wang, Motion measurement using inertial sensors, ultrasonic sensors, and magnetometers with extended Kalman filter for data fusion, *IEEE Sens. J.* **12**(5) (2012) 943–953.

[33] S. Gao, Y. Zhong, X. Zhang and B. Shirinzadeh, Multi-sensor optimal data fusion for INS/GPS/SAR integrated navigation system, *Aerosp. Sci. Technol.* **13**(4–5) (2009) 232.

[34] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox and N. Roy, Visual odometry and mapping for autonomous flight using an RGB-D camera, *Robot. Res.* **100** (2016) 235–252.

[35] Z. Zichao, H. Rebecq, C. Forster and D. Scaramuzza, Benefit of large field-of-view cameras for visual odometry, *IEEE Int. Conf. Robotics and Automation (ICRA)* (IEEE, 2016).

[36] F. Demim, A. Nemra, K. Louadj, M. Hamerlain and A. Bazoula, Cooperative SLAM for multiple UGVs navigation using SVSF filter, *Autom. J. Control Measur. Electron. Comput. commun.* **58** (2017) 119–129.

[37] L. Hachemi, A. Nemra and M. Guiateni, Fault tolerant control for unicycle mobile robot navigation: Active vs passive approaches, *ICCEE18: Int. Conf. on Communications and Electrical Engineering*, El-Oued, Algeria, 2018.

[38] S. A. Gadsden, D. Dunne, R. Tharmarasa, S. R. Habibi and T. Kirubarajan, Application of the smooth variable structure filter to a multi-target tracking problem, *Signal Processing, Sensor Fusion, and Target Recognition XX*, Orlando, Florida, 2011.

[39] A. Allam, M. Tadjine, A. Nemra and E. Kobzili, Stereo vision as a sensor for SLAM based smooth variable structure filter with an adaptive boundary layer width, *2017 6th International Conference on Systems and Control (ICSC)*, 2017, pp. 14–20, doi: 10.1109/ICoSC.2017.7958700.

[40] Y. Tian, H. Suwoyo, W. Wang and L. Li, Volume an ASVSF-SLAM algorithm with time-varying noise statistics based on MAP creation and weighted exponent, *Hindawi Math. Probl. Eng.* **2019** (2019) 2765731.

[41] F. Pukelsheim, The three sigma rule, *Amer. Statist.* **48** (1994) 88–91.

[42] S. Mellah, G. Graton, E. M. El Adel, M. Ouladsine and A. Planchais, Mobile robot additive fault diagnosis and accommodation, *8th Int. Conf. Systems and Control (ICSC)*, Marrakesh, Morocco, 2019.

[43] M. Freese, S. Singh, F. Ozaki and N. Matsuhira, Virtual robot experimentation platform V-REP: A versatile 3D robot simulator, *Simulation, Modeling, and Programming for Autonomous Robots - Second Int. Conf. SIMPAR*, Darmstadt, Germany, 2010, pp. 15–18.

Linda Hachemi was born in Algiers in October 1992. She obtained her Master's degree in Electronic Instrumentation Engineering, Faculty of Electronics and Computer Science, electrical option, at the Houari Boumédiéne University of Science and Technology in July 2016.

Mohamed Guiatni received B.E. and M.Sc. degrees from the Ecole Militaire Polytechnique (EMP), Algiers, Algeria, and Ph.D. degree from the University of Evry, Evry, France, in 2009. He is currently the Head of the Complex Systems Control and Simulation Laboratory and an Associate Professor with EMP. His research interests include haptic devices design and control, mechatronics, rehabilitation robotics, interactive simulation for military, and medical applications. (Based on document published on 13 August 2018).

Abdelkarim Nemra received his engineering degree from the Military Polytechnic School of Algiers (EMP), Algeria, in 1998 and the Ph.D degree on electrical engineering from Cranfield University United Kingdom (UK) in 2010. Since 2011, he is with the Department of Intelligent Autonomous Vehicles, EMP, Algeria.