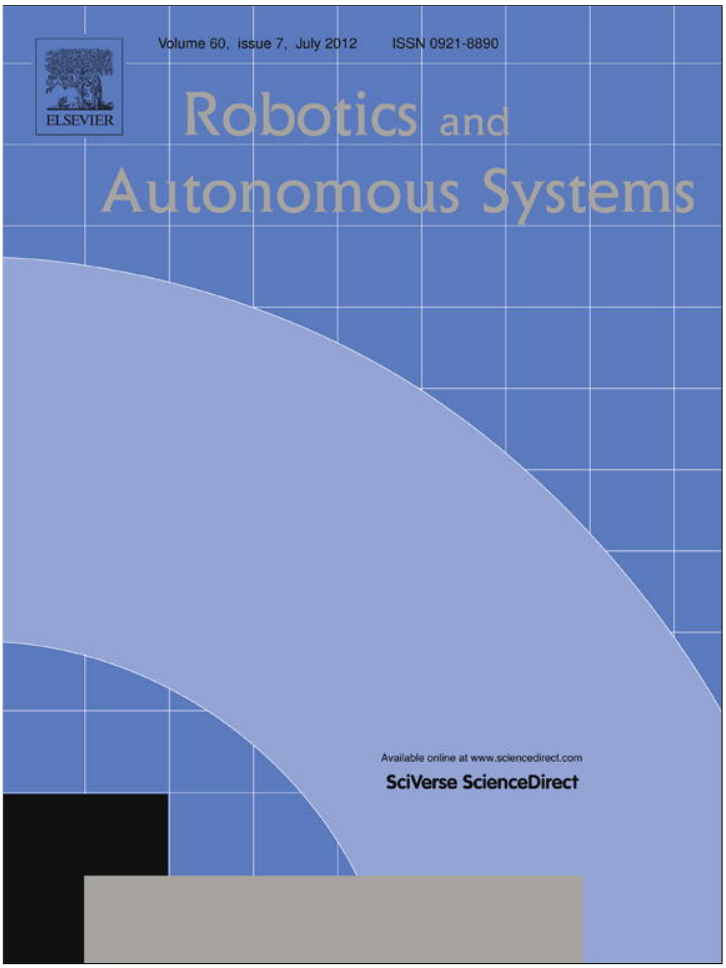
Nonlinear Kalman Filters and Particle Filters for integrated navigation of unmanned aerial vehicles

Article *in* Robotics and Autonomous Systems · July 2012

DOI: 10.1016/j.robot.2012.03.001

1 author:

Gerasimos Rigatos
Institute of Electrical and Electronics Engineers
165 PUBLICATIONS **2,428** CITATIONS



Robotics and Autonomous Systems 60 (2012) 978–995

Contents lists available at SciVerse ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Nonlinear Kalman Filters and Particle Filters for integrated navigation of unmanned aerial vehicles

Gerasimos G. Rigatos *,[1]

Department of Engineering, Harper Adams University College, TF10 8NB, Shropshire, UK

ARTICLE INFO

Article history:
Received 27 May 2010
Received in revised form
24 February 2012
Accepted 5 March 2012
Available online 16 March 2012

Keywords:
Nonlinear estimation
Extended Kalman Filtering
Sigma-point Kalman Filters
Particle Filter
Derivative-free nonlinear Kalman Filter
Sensor fusion
Differential flatness theory
Unmanned aerial vehicles

ABSTRACT

The paper studies and compares nonlinear Kalman Filtering methods and Particle Filtering methods for estimating the state vector of Unmanned Aerial Vehicles (UAVs) through the fusion of sensor measurements. Next, the paper proposes the use of the estimated state vector in a control loop for autonomous navigation and trajectory tracking by the UAVs. The proposed nonlinear controller is derived according to the flatness-based control theory. The estimation of the UAV's state vector is carried out with the use of (i) Extended Kalman Filtering (EKF), (ii) Sigma-Point Kalman Filtering (SPKF), (iii) Particle Filtering (PF), and (iv) a new nonlinear estimation method which is the Derivative-free nonlinear Kalman Filtering (DKF). The performance of the nonlinear control loop which is based on these nonlinear state estimation methods is evaluated through simulation tests. Comparing the aforementioned filtering methods in terms of estimation accuracy and computation speed, it is shown that the Sigma-Point Kalman Filtering is a reliable and computationally efficient approach to state estimation-based control, while Particle Filtering is well-suited to accommodate non-Gaussian measurements. Moreover, it is shown that the Derivative-free nonlinear Kalman Filter is faster than the rest of the nonlinear filters while also succeeding accurate, in terms of variance, state estimates.

1. Introduction

Nonlinear estimation based on probabilistic inference forms a core component in most modern guidance and navigation systems. The estimator fuses observations from multiple sensors with predictions from a nonlinear dynamic state-space model of the system under control. The most widely used algorithm for multi-sensor fusion is the Extended Kalman Filter (EKF); however this is based on linearization of the system dynamics, which results in a suboptimal application of the recursive estimation of the standard Kalman Filter [1,2]. Moreover, the EKF follows the assumption of Gaussian process/measurement noise which does not always hold. These can seriously affect the performance of the state estimation and even lead to divergence. Consequently, the performance of a control loop that uses an EKF-based estimate of the system's state vector can, in some cases, be unsatisfactory.

To overcome the EKF flaws, two different approaches to state estimation of nonlinear dynamical systems are proposed: (i) Sigma-Point Kalman Filters (SPKF) and particularly the Unscented Kalman Filter (UKF), and (ii) Particle Filters. SPKF methods have proven to be superior to EKF in a wide range of applications. Whereas the EKF can be viewed as a first order linearization method, the UKF achieves higher accuracy, without requiring additional computational effort. Furthermore, implementation of the UKF is substantially easier and unlike the EKF case it does not need any analytic derivation or computation of Jacobian matrices [3–7]. The state distribution in UKF is approximated by a Gaussian random variable, which is represented using a minimal set of suitably chosen weighted sample points. These sigma points are propagated through the true nonlinear system, thus generating the posterior sigma-point set, and the posterior statistics are calculated. The sample points progressively converge to the true mean and covariance of the Gaussian random variable.

The Particle Filter (PF) is a non-parametric state estimator which unlike the EKF does not make any assumption on the probability density function of the measurements [8–11]. The concept of particle filtering comes from Monte-Carlo methods. The Particle Filter has improved performance over the established nonlinear filtering approaches (e.g. the EKF), since it can provide optimal estimation in nonlinear non-Gaussian state-space models. Particle filters can approximate the system's state sufficiently when the number of particles (estimations of the state vectors which evolve in parallel) is large. The PF also avoids the calculations associated with the Jacobians which appear in the EKF equations [12]. The main stages of the PF are prediction (time update), correction (measurement update) and resampling for substituting

* Tel.: +44 0746 6851634; fax: +44 01743 614734.
E-mail addresses: grigat@ieee.org, ger9711@ath.forthnet.gr.
[1] Also with Unit of Industrial Automation, Industrial Systems Institute, 26504, Rion Patras, Greece.

0921-8890/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.robot.2012.03.001

the unsuccessful state vector estimates with those particles that have better approximated the real state vector. Comparing SPKF and PF methods, the latter require more sample points to approximate the state distribution. However, the PF is a non-parametric filter which can be applied to any kind of state distribution, while the SPKF state estimators are still based on the assumption of a Gaussian process and measurement noise.

Aiming also at finding more efficient implementations of nonlinear filtering, in this paper a Derivative-free approach to Kalman Filtering is introduced and applied to state estimation-based control of nonlinear dynamical systems, such as UAVs. In the Derivative-free nonlinear Kalman Filtering method (DKF) the system is first subject to a linearization transformation that is in accordance to the differential flatness theory and next state estimation is performed by applying the standard Kalman Filter recursion to the linearized model. The proposed method provides estimates of the state vector of the nonlinear system without the need for derivatives and Jacobians calculation. By avoiding linearization approximations, the proposed filtering method improves the accuracy of estimation of the system state variables, and results in smooth control signal variations and in minimization of the tracking error of the associated control loop. Moreover, the Derivative-free nonlinear Kalman Filter appears to be faster than the previously mentioned nonlinear filtering methods (i.e. EKF, UKF and PF) while also providing very accurate (in terms of variance) state estimates. The application of the Derivative-free nonlinear Kalman Filter to the UAV model confirms and extends the initial results about the filter's performance given in [13–15].

As a case study, the paper examines the application of the aforementioned nonlinear filtering methods to the problem of sensor fusion-based guidance and navigation of unmanned aerial vehicles (UAVs) [5]. Sensor fusion in land navigation systems has been studied in [16], while typical sensors in UAV navigation systems can be rate-gyros and accelerometers, barometric altimeters and magnetic compasses. GPS position and velocity measurements provide additional information about the UAV's motion. Measurement systems for UAV navigation have been analyzed in [17–20], while different control approaches for UAV flight control have been analyzed in [21–24]. The filtering approaches examined in this paper are used to fuse measurements coming from different sources (measurements from on-board UAV's sensors and/or GPS measurements), thus providing estimates of the state vector of the UAV. The implementation of UAV control through prior estimation of the UAV's state vector from distributed sensor measurements has been analyzed in [17]. Simulation experiments are carried out to evaluate the performance of the nonlinear filters and of the associated state estimation-based UAV control loops.

The structure of the paper is as follows: in Section 2 the principles of flatness-based control are explained and the application of flatness-based control to the UAV model is analyzed. In Section 3 the Extended Kalman Filter is introduced as a basic filtering approach for nonlinear dynamical systems. In Section 4 estimation of the UAV's state vector with the use of Sigma-Point Kalman Filters, when fusing measurements that come from different sensors, is presented. In Section 5 the application of Particle Filtering for estimation of the state vector of the UAV through fusion of distributed sensor measurements is analyzed. In Section 6 it is explained how Derivative-free nonlinear Kalman Filters can be designed in accordance to the differential flatness theory and how they are applied for sensor fusion and state estimation in the UAV case. In Section 7 simulation tests are presented to evaluate the performance of the filtering-based control loops when the UAV's state vector is estimated with the use of the aforementioned nonlinear filters. Finally in Section 8 concluding remarks are stated.

2. Flatness-based control for UAVs

2.1. Differential flatness theory for UAV control

Trajectory tracking by Unmanned Aerial Vehicles with the use of nonlinear control methods is examined first. Various approaches have been proposed for the UAV control among which Lyapunov functions-based control [20,21] and model-based predictive control [19,24]. In this paper it will be shown that flatness-based control is a suitable approach for implementing autonomous navigation of aerial vehicles. Flatness-based control theory stems from differential flatness and has been successfully applied to several nonlinear dynamical systems. Flatness-based control for a UAV helicopter-like model has been developed in [25], assuming that the UAV performs maneuvers at a constant altitude. The same kinematic model has been used in several studies on UAV trajectory tracking and control [22,23].

A dynamical system is considered to be differentially flat if the following properties hold: (i) the so-called flat output exists, i.e. a new variable which is expressed as a function of the system's state variables. The flat output and its derivatives should not be coupled in the form of an ordinary differential equation, (ii) the components of the system (i.e. state variables and control input) can be expressed as functions of the flat output and its derivatives [26,27]. Differential flatness is a property characterizing classes of systems. In certain cases expressing all system variables as functions of the flat output and its derivatives enables transformation to a linearized form for which the design of the controller becomes easier. In other cases by showing that a system is differentially flat one can easily design a reference trajectory as a function of the so-called flat output and can find a control law that assures tracking of this desirable trajectory [27–29].

Flatness-based control has been successfully applied for steering autonomous vehicles and particularly UAVs along desirable trajectories [27,28]. It this paper it is assumed that an helicopter-like UAV, performs maneuvers at a constant altitude. Then, one obtains the following UAV kinematics [25]

$$\begin{aligned} &\dot{x} = v\cos(\theta), \qquad \dot{y} = v\sin(\theta), \qquad \dot{\theta} = \omega = q_1 \\ &\dot{v} = q_2, \qquad \dot{h} = 0 \end{aligned} \tag{1}$$

where (x, y) is the desired inertial position of the UAV, θ is the UAV's heading (angle between the transversal axis of the UAV and axis OX), ω is the UAV's rate of change of the heading angle, v is the UAV's velocity, h is the UAV's altitude, and q_1, q_2 are control inputs constrained by the dynamic capability of the UAVs (namely the heading rate constraint and the acceleration constraint, respectively). There is an equivalence between the UAV's kinematic model and the model of a unicycle robot. The flat output is the cartesian position of the UAV's center of gravity, and is denoted as $\eta = (x, y)$. Then, the flatness-based dynamic compensator is

$$\begin{aligned} &\dot{\xi} = u_1\cos(\theta) + u_2\sin(\theta), \qquad v = \xi \\ &\omega = \frac{u_2\cos(\theta) - u_1\sin(\theta)}{\xi} \end{aligned} \tag{2}$$

where

$$\begin{aligned} u_1 &= \ddot{x}_d + k_{p_1}(x_d - x) + k_{d_1}(\dot{x}_d - \dot{x}) \\ u_2 &= \ddot{y}_d + k_{p_2}(y_d - y) + k_{d_2}(\dot{y}_d - \dot{y}). \end{aligned} \tag{3}$$

It has been shown (see [13,30]) that using the change of coordinates

$$\begin{aligned} &x_1 = x, \qquad x_2 = \dot{x} = \xi\cos(\theta) \\ &x_3 = y, \qquad x_4 = \dot{y} = \xi\sin(\theta) \end{aligned} \tag{4}$$

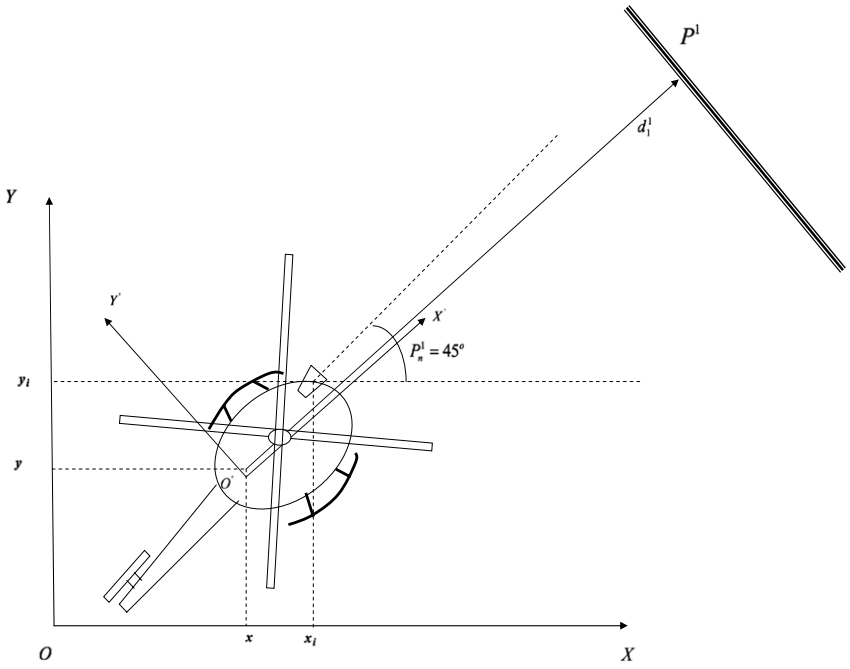


Fig. 1. Reference frames for the UAV.

the extended system becomes fully linearized and is described by the chains of integrators,

$$\ddot{x}_1 = u_1, \qquad \ddot{x}_3 = u_2. \tag{5}$$

The dynamic compensator of Eq. (2) has a potential singularity at $\xi = v = 0$, i.e. when the UAV is not moving, which is a case never met when the UAV is in flight [30]. On the equivalent and decoupled system of Eq. (5), one can easily design an exponentially stabilizing feedback, which has the form of Eq. (3). The proportional-derivative gains are chosen as $k_{p_1} > 0$ and $k_{d_1} > 0$ for $i = 1, 2$, so as to assure asymptotic elimination of the tracking error of Eq. (5).

2.2. Sensor-Fusion for UAV navigation

To improve the accuracy of the estimation of the UAV's state vector, fusion of the measurements provided by on-board sensors can be performed (IMU, GPS, gyrocompasses, radar, vision sensors). The inertial coordinates system OXY is defined. Furthermore the coordinates system $O'X'Y'$ is considered (Fig. 1). $O'X'Y'$ results from OXY if it is rotated by an angle θ. The coordinates of the center of gravity of the UAV with respect to OXY are (x, y). It is assumed that the coordinates of a distance measuring sensor (e.g. vision sensor or radar) with respect to $O'X'Y'$ are x_i', y_i'. The orientation of the sensor with respect to $O'X'Y'$ is θ_i'. Thus the coordinates of the distance sensor with respect to OXY are (x_i, y_i) and its orientation is θ_i, and are given by:

$$\begin{aligned} x_i(k) &= x(k) + x_i' \sin(\theta(k)) + y_i' \cos(\theta(k)) \\ y_i(k) &= y(k) - x_i' \cos(\theta(k)) + y_i' \sin(\theta(k)) \\ \theta_i(k) &= \theta(k) + \theta_i'. \end{aligned} \tag{6}$$

For the localization of the UAV a first set of measurements comprises the UAV's cartesian coordinates and comes from an IMU or a GPS. A second set of measurements consists of the readings of the UAV's distance $d(k)$ from a reference surface P^j, and as mentioned before can be provided from a vision sensor or radar. A reference surface P^j in the UAV's 2D flight area can be represented by P_r^j and P_n^j (Fig. 1), where (i) P_r^j is the normal distance of the reference surface from the origin O, (ii) P_n^j is the angle between the normal line to the plane and the x-direction. The sensor providing this second set of measurements is at position $[x_i(k), y_i(k)]$ with respect to the inertial coordinates system OXY and its orientation is $\theta_i(k)$. Using the above notation, the distance measurement with reference to plane P^j is represented by P_r^j, P_n^j (see Fig. 1):

$$d_i^j(k) = P_r^j - x_i(k)\cos(P_n^j) - y_i(k)\sin(P_n^j). \tag{7}$$

In this paper it will be assumed that the measurements vector is given by $\gamma(x(k)) = [x(k), y(k), d(k)]$, where $[x(k), y(k)]$ are the UAV's cartesian coordinates, and $d(k)$ is the distance measurement.

3. Kalman and Extended Kalman Filtering

To enable navigation of the UAV when estimating its state vector by fusing measurements from on-board sensors, Kalman Filtering is proposed. In the discrete-time case, a linear dynamical system is assumed to be expressed in the form of a discrete-time state model [13,31–33]:

$$\begin{aligned} x(k+1) &= A(k)x(k) + L(k)u(k) + w(k) \\ z(k) &= Cx(k) + v(k) \end{aligned} \tag{8}$$

where the state $x(k)$ is a m-vector, $w(k)$ is a m-element process noise vector and A is a $m \times m$ real matrix. Moreover the output measurement $z(k)$ is a p-vector, C is an $p \times m$-matrix of real numbers, and $v(k)$ is the measurement noise. It is assumed that the process noise $w(k)$ and the measurement noise $v(k)$ are uncorrelated. Now the problem is to estimate the state $x(k)$ based on the sequence of output measurements $z(1), z(2), \ldots, z(k)$. For the initialization of the covariance matrix P one can set $\hat{P}(0) = \lambda I$, with $\lambda > 0$. For the linear model of Eq. (8), the standard Kalman filter equations are

Measurement update:

$$\begin{aligned} K(k) &= P^-(k)C^T[C \cdot P^-(k)C^T + R]^{-1} \\ \hat{x}(k) &= \hat{x}^-(k) + K(k)[z(k) - C\hat{x}^-(k)] \\ P(k) &= P^-(k) - K(k)CP^-(k) \end{aligned} \tag{9}$$

Time update:

$$P^-(k+1) = A(k)P(k)A^T(k) + Q(k)$$
$$\hat{x}^-(k+1) = A(k)\hat{x}(k) + L(k)u(k). \quad (10)$$

In the case of nonlinear dynamical systems, such as UAVs, state estimation can be performed using the Extended Kalman Filter (EKF). The following nonlinear state model is considered [13,15]:

$$x(k+1) = \phi(x(k)) + L(k)u(k) + w(k)$$
$$z(k) = \gamma(x(k)) + v(k) \quad (11)$$

where $x \in R^{m\times 1}$ is the system's state vector and $z \in R^{p\times 1}$ is the system's output, while $w(k)$ and $v(k)$ are uncorrelated, zero-mean, Gaussian noise processes with covariance matrices $Q(k)$ and $R(k)$ respectively. The operators $\phi(x)$ and $\gamma(x)$ are vectors defined as $\phi(x) = [\phi_1(x), \phi_2(x), \ldots, \phi_m(x)]^T$, and $\gamma(x) = [\gamma_1(x), \gamma_2(x), \ldots, \gamma_p(x)]^T$, respectively. Following a linearization procedure, ϕ is expanded into Taylor series about $\hat{x}$ and the Jacobian $J_\phi(x)$ is calculated at $\hat{x}(k)$. Similarly, following a linearization procedure, γ is expanded into Taylor series about $\hat{x}$ and the Jacobian $J_\gamma(x)$ is calculated at $\hat{x}(k)$. Now, the EKF recursion is as follows:

- *Measurement update*. Acquire $z(k)$ and compute:

$$K(k) = P^-(k)J_\gamma^T(\hat{x}^-(k)) \cdot [J_\gamma(\hat{x}^-(k))P^-(k)J_\gamma^T(\hat{x}^-(k)) + R(k)]^{-1}$$
$$\hat{x}(k) = \hat{x}^-(k) + K(k)[z(k) - \gamma(\hat{x}^-(k))] \quad (12)$$
$$P(k) = P^-(k) - K(k)J_\gamma(\hat{x}^-(k))P^-(k).$$

- *Time update*. Compute:

$$P^-(k+1) = J_\phi(\hat{x}(k))P(k)J_\phi^T(\hat{x}(k)) + Q(k)$$
$$\hat{x}^-(k+1) = \phi(\hat{x}(k)) + L(k)u(k). \quad (13)$$

4. Unscented Kalman Filter in state estimation of UAVs

For sensor fusion-based navigation of the UAV the Unscented Kalman Filter can be also used. The Unscented Kalman Filter (UKF) is a special case of Sigma-Point Kalman Filters that computes approximate solutions to the filtering problem of nonlinear systems of the form [34]

$$x_k = f(x_{k-1}, u_{k-1}) + q_{k-1}$$
$$z_k = h(x_k) + r_k \quad (14)$$

where $x_k \in R^n$ is the state, $y_k \in R^m$ is the measurement, $q_{k-1} \in R^n$ is a Gaussian process noise $q_{k-1} \sim N(0, Q_{k-1})$, and $r_k \in R^m$ is a Gaussian measurement noise $r_k \sim N(0, R_k)$. The mean and covariance of the initial state x_0 are m_0 and P_0, respectively. Some basic operations performed in the UKF algorithm (*Unscented Transform*) are as follows:

(1) Denoting the current state mean as $\hat{x}$, a set of $2n+1$ sigma points is taken from the columns of the $n \times n$ matrix $\sqrt{(n+\lambda)P_{xx}}$ as follows: $x^0 = \hat{x}$, $x^i = \hat{x} + [\sqrt{(n+\lambda)P_{xx}}]$, $x^i = \hat{x} - [\sqrt{(n+\lambda)P_{xx}}]$ and the associated weights are computed: $W_0 = \frac{\lambda}{(n+\lambda)}$, $W_i = \frac{1}{2(n+\lambda)}$, $i = 1, \ldots, 2n$. Matrix P_{xx} is the covariance matrix of the state x. The parameter λ is a scaling parameter.
(2) Transform of each one of the sigma points which are defined as $z^i = h(x^i)$ $i = 0, \ldots, 2n$.
(3) Mean and covariance estimates for z can be computed as $\hat{z} \simeq \sum_{i=0}^{2n} W_i z^i$, $P_{zz} = \sum_{i=0}^{2n} W_i(z^i - \hat{z})(z^i - \hat{z})^T$.
(4) The cross-covariance of x and z is estimated as $P_{xz} = \simeq \sum_{i=0}^{2n} W_i(x^i - \hat{x})(z^i - \hat{z})^T$.

The matrix square root of positive definite matrix P_{xx} means a matrix $A = \sqrt{P_{xx}}$ such that $P_{xx} = AA^T$ and a possible way for calculation is SVD [33]. Next the basic stages of the *Unscented Kalman Filter* are given: The Unscented Kalman Filter also consists of prediction stage (time update) and correction stage (measurement update) [34].

Time update: Compute the predicted state mean $\hat{x}_k^-$ and the predicted covariance $P_{xx_k}^-$ as

$$[\hat{x}_{k^-}, P_{xxk^-}] = UT(f_d, \hat{x}_{k-1}, P_{xxk-1})$$
$$P_{xxk^-} = P_{xxk-1} + Q_{k-1}. \quad (15)$$

Measurement update: Obtain the new output measurement z_k and compute the predicted mean $\hat{z}_k$ and covariance of the measurement P_{zzk}, and the cross covariance of the state and measurement P_{xzk}

$$[\hat{z}_k, P_{zzk}, P_{xzk}] = UT(h_d, \hat{x}_{k^-}, P_{xxk^-})$$
$$P_{zzk} = P_{zzk} + R_k. \quad (16)$$

Then compute the filter gain K_k, the state mean $\hat{x}_k$ and the covariance P_{xxk}, conditional to the measurement y_k

$$K_k = P_{xzk}{P_{zzk}}^{-1}$$
$$\hat{x}_k = \hat{x}_k^- + K_k[z_k - \hat{z}_k] \quad (17)$$
$$P_{xxk} = P_{xxk^-} - K_k P_{zzk} K_k^T.$$

The filter starts from the initial mean m_0 and covariance P_{xx0}.

5. Particle Filter in state estimation of UAVs

5.1. The prediction stage

To enable navigation of the UAV when estimating its state vector by fusing measurements from distributed sensors, Particle Filtering will be also examined. Particle Filtering is a method for state estimation that is not dependent on the probability density function of the measurements. The particles filter consists also of a measurement update (correction stage) and a time update (prediction stage) [35]. The prediction stage calculates $p(x(k)|Z^-)$ where $Z^- = [z(1), z(2), z(k-1)]$ are output measurements up to time instant $k-1$. It holds that $p(x(k-1)|Z^-) = \sum_{i=1}^N w_{k-1}^i \delta_{\xi_{k-1}^i}(x(k-1))$, while from Bayes formula it holds $p(x(k)|Z^-) = \int p(x(k)|x(k-1))p(x(k-1)|Z^-)dx$. From the above one finally obtains

$$p(x(k)|Z^-) = \sum_{i=1}^N w_{k-1}^i \delta_{\xi_{k^-}^i}(x(k)) \quad (18)$$

with $\xi_{k^-}^i \sim p(x(k)|x(k-1) = \xi_{k-1}^i)$.

The meaning of Eq. (18) is as follows: the state equation of the system is executed N times, starting from the N previous values of the state vectors $x(k-1) = \xi_{k-1}^i$.

Thus estimations of the current value of the state vector are obtained $\hat{x}(k)$, and consequently the mean value of the state vector will be given from Eq. (18). This means that the value of the state vector which is calculated in the prediction stage is the result of the weighted averaging of the state vectors which were calculated after running the state equation, starting from the N previous values of the state vectors ξ_{k-1}^i.

5.2. The correction stage

The a-posteriori probability density is found using Eq. (18). Now, a new position measurement $z(k)$ is obtained and the

objective is to calculate the corrected probability density $p(x(k)|Z)$, where $Z = \{z(1), z(2), \ldots, z(k)\}$. From Bayes law it holds that $p(x(k)|Z) = \frac{p(Z|x(k))p(x(k))}{p(Z)}$ which can be also written as

$$p(x(k)|Z) = \frac{p(z(k)|x(k))p(x(k)|Z^-)}{\int p(z(k)|x(k), Z^-)p(x(k)|Z^-)dx}. \quad (19)$$

Substituting Eq. (18) into Eq. (19) and after intermediate calculations one finally obtains

$$p(x(k)|Z) = \sum_{i=1}^{N} w_k^i \delta_{\xi_{k^-}^i}(x(k))$$
$$\text{where } w_k^i = \frac{w_{k^-}^i p(z(k)|x(k) = \xi_{k^-}^i)}{\sum_{j=1}^{N} w_{k^-}^j p(z(k)|x(k) = \xi_{k^-}^j)}. \quad (20)$$

Eq. (20) denotes the corrected value for the state vector. The recursion of the PF proceeds as follows:

- *Measurement update* (correction stage): Acquire $z(k)$ and compute the new value of the state vector

$$p(x(k)|Z) = \sum_{i=1}^{N} w_k^i \delta_{\xi_{k^-}^i}(x(k))$$
$$\text{with corrected weights} \quad (21)$$
$$w_k^i = \frac{w_{k^-}^i p(z(k)|x(k) = \xi_{k^-}^i)}{\sum_{j=1}^{N} w_{k^-}^i p(z(k)|x(k) = \xi_{k^-})^i} \quad \text{and} \quad \xi_k^i = \xi_{k^-}^i.$$

  Resampling for substitution of the degenerated particles. The particles of low weight factors are removed and their place is occupied by duplicates of the particles with high weight factors [13,35].
- *Time update* (prediction stage): compute state vector $x(k+1)$ according to the probability density function (pdf)

$$p(x(k+1)|Z) = \sum_{i=1}^{N} w_k^i \delta_{\xi_k^i}(x(k)) \quad (22)$$
$$\text{where } \xi_k^i \sim p(x(k+1)|x(k) = \xi_k^i).$$

5.3. Resampling issues in Particle Filtering

The algorithm of Particle Filtering which is described through Eqs. (18) and (20) has a significant drawback: after a certain number of iterations k, almost all the weights w_k^i become 0. In the ideal case all the weights should converge to the value $\frac{1}{N}$, i.e. the particles should have the same significance. The criterion used to define a sufficient number of particles is $N_k^{\text{eff}} = \frac{1}{\sum_{i=1}^{N} w_k^{i\,2}} \in [1, N]$. When N_k^{eff} is close to value N then all particles have almost the same significance. However using the algorithm of Eqs. (18) and (20) results in $N_k^{\text{eff}} \to 1$, which means that the particles are degenerated, i.e. they lose their effectiveness. Therefore, it is necessary to modify the algorithm so as to assure that degeneration of the particles will not take place [36–38]. The particles of low weight factors are removed and their place is occupied by duplicates of the particles with high weight factors. The total number of particles remains unchanged (equal to N) and therefore this procedure can be viewed as a "resampling" or "redistribution" of the particles set. There are improved versions of resampling which improve the speed of the algorithm [9]. A first choice is to perform a multinomial resampling. N particles are chosen between $\{\xi_k^1, \ldots, \xi_k^N\}$ and the corresponding weights are $w_k^1, \ldots, w_k^N$. The number of times each particle is selected is given by $[j_1, \ldots, j_n]$. Thus a set of N particles is again created, the elements of which are chosen after sampling with the discrete distribution $\sum_{i=1}^{N} w_k^i \delta_{\xi_k^i}(x)$. The particles $\{\xi_k^1, \ldots, \xi_k^N\}$ are chosen according to the probabilities $\{w_k^1, \ldots, w_k^N\}$. The selected particles are given equal weights $\frac{1}{N}$.

6. Differential flatness theory and canonical forms

6.1. Conditions for applying the differential flatness theory

Next, a new filter will be developed, in accordance to the differential flatness theory. It will be shown that the filter can be efficiently used in the problem of integrated UAV navigation. First, the generic class of nonlinear systems $\dot{x} = f(x, u)$ (including MIMO systems) is considered. Such systems can be transformed to the form of an affine in-the-input system by adding an integrator to each input [29]

$$\dot{x} = f(x) + \sum_{i=1}^{m} g_i(x)u_i. \quad (23)$$

The following definitions are now used [15]:

(i) Lie derivative: $L_f h(x)$ stands for the Lie derivative $L_f h(x) = (\nabla h)f$ and the repeated Lie derivatives are recursively defined as $L_f^0 h = h$ for $i = 0$, $L_f^i h = L_f L_f^{i-1} h = \nabla L_f^{i-1} hf$ for $i = 1, 2, \ldots$.
(ii) Lie Bracket: $ad_f^i g$ stands for a Lie Bracket which is defined recursively as $ad_f^i g = [f, ad_f^{i-1}]g$ with $ad_f^0 g = g$ and $ad_f g = [f, g] = \nabla gf - \nabla fg$.

If the system of Eq. (23) can be linearized by a diffeomorphism $z = \phi(x)$ and a static state feedback $u = \alpha(x) + \beta(x)v$ into the following form

$$\dot{z}_{i,j} = z_{i+1,j} \quad \text{for } 1 \le j \le m \text{ and } 1 \le i \le v_j - 1$$
$$\dot{z}_{v_{i,j}} = v_j \quad (24)$$

with $\sum_{j=1}^{m} v_j = n$, then $y_j = z_{1,j}$ for $1 \le j \le m$ are the 0-flat outputs which can be written as functions of only the elements of the state vector x. To define conditions for transforming the system of Eq. (23) into the canonical form described in Eq. (24) the following theorem holds [29]

Theorem. *For nonlinear systems described by Eq. (23) the following variables are defined:* (i) $G_0 = span[g_1, \ldots, g_m]$, (ii) $G_1 = span[g_1, \ldots, g_m, ad_f g_1, \ldots, ad_f g_m], \ldots$, (k) $G_k = span\{ad_f^j g_i$ *for* $0 \le j \le k, 1 \le i \le m\}$. *Then, the linearization problem for the system of Eq. (23) can be solved if and only if:* (1) *The dimension of* G_i, $i = 1, \ldots, k$ *is constant for* $x \in X \subseteq R^n$ *and for* $1 \le i \le n-1$, (2) *The dimension of* G_{n-1} *if of order n,* (3) *The distribution* G_k *is involutive for each* $1 \le k \le n-2$.

6.2. Transformation of MIMO systems into canonical forms

It is assumed now that after defining the flat outputs of the initial MIMO nonlinear system (this approach will be also shown to hold for the UAV case), and after expressing the system state variables and control inputs as functions of the flat output and of the associated derivatives, the system can be transformed in the Brunovsky canonical form:

$$\dot{x}_1 = x_2$$
$$\ldots$$
$$\dot{x}_{r_1 - 1} = x_{r_1}$$

$$\begin{aligned}
\dot{x}_{r_1} &= f_1(x) + \sum_{j=1}^{p} g_{1_j}(x)u_j + d_1 \\
\dot{x}_{r_1+1} &= x_{r_1+2} \\
&\cdots \\
\dot{x}_{p-1} &= x_p \\
\dot{x}_p &= f_p(x) + \sum_{j=1}^{p} g_{p_j}(x)u_j + d_p \\
y_1 &= x_1 \\
&\cdots \\
y_p &= x_{n-r_p+1}
\end{aligned} \tag{25}$$

where $x = [x_1, \ldots, x_n]^T$ is the state vector of the transformed system (according to the differential flatness formulation), $u = [u_1, \ldots, u_p]^T$ is the set of control inputs, $y = [y_1, \ldots, y_p]^T$ is the output vector, f_i are the drift functions and $g_{i,j}$, $i, j = 1, 2, \ldots, p$ are smooth functions corresponding to the control input gains, while d_j is a variable associated to external disturbances. It holds that $r_1 + r_2 + \cdots + r_p = n$. Having written the initial nonlinear system into the canonical (Brunovsky) form it holds

$$y_i^{(r_i)} = f_i(x) + \sum_{j=1}^{p} g_{ij}(x)u_j + d_j. \tag{26}$$

Next the following vectors and matrices can be defined: $f(x) = [f_1(x), \ldots, f_n(x)]^T$, $g(x) = [g_1(x), \ldots, g_n(x)]^T$, with $g_i(x) = [g_{1i}(x), \ldots, g_{pi}(x)]^T$, $A = \mathrm{diag}[A_1, \ldots, A_p]$, $B = \mathrm{diag}[B_1, \ldots, B_p]$, $C^T = \mathrm{diag}[C_1, \ldots, C_p]$, $d = [d_1, \ldots, d_p]^T$, where matrix A has the MIMO canonical form, i.e. with block-diagonal elements

$$\begin{aligned}
A_i &= \begin{pmatrix} 0 & 1 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ldots & \vdots \\ 0 & 0 & \cdots & 1 \\ 0 & 0 & \cdots & 0 \end{pmatrix}_{r_i \times r_i} \\
B_i^T &= \begin{pmatrix} 0 & 0 & \cdots 0 & 1 \end{pmatrix}_{1 \times r_i} \\
C_i^T &= \begin{pmatrix} 1 & 0 & \cdots 0 & 0 \end{pmatrix}_{1 \times r_i}.
\end{aligned} \tag{27}$$

Thus, Eq. (26) can be written in state-space form

$$\begin{aligned}
\dot{x} &= Ax + Bv + B\tilde{d} \\
y &= Cx
\end{aligned} \tag{28}$$

where the control input is written as $v = f(x) + g(x)u$. The system of Eqs. (27) and (28) is in controller and observer canonical form.

6.3. Canonical forms for the UAV model

Using the differential flatness theory of Sections 6.1 and 6.2 for transforming the initial UAV's model (described in Section 2.1) into a linearized equivalent that is finally written in the Brunovsky form, one has Eq. (5) which means $\ddot{x} = u_1$ and $\ddot{y} = u_2$. Next, the state variables $x_1 = x$, $x_2 = \dot{x}$, $x_3 = y$ and $x_4 = \dot{y}$ are used. Considering the state vector $x \in R^{4\times 1}$, the following matrices are also defined

$$\begin{aligned}
A &= \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \qquad B = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \\
C &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}.
\end{aligned} \tag{29}$$

Using the matrices of Eq. (29) one obtains the Brunovsky form of the MIMO model of the UAV:

$$\begin{aligned}
\dot{x} &= Ax + Bv \\
y &= Cx
\end{aligned} \tag{30}$$

where the new input v is given by $v = [u_1(x, t), u_2(x, t)]^T$. For the system of Eqs. (29) and (30), state estimation is possible by applying the standard Kalman Filter. The system is first turned into discrete-time form using common discretization methods and then the recursion of the linear Kalman Filter described in Eqs. (9) and (10) is applied.

7. Simulation tests

7.1. EKF Filtering-based Sensor Fusion for UAV navigation

Assuming a constant sampling period $\Delta t_k = T$ the measurement equation for the UAV is $z(k) = \gamma(x(k)) + v(k)$, where $z(k)$ is the vector containing the sequence of measurements of the cartesian coordinates of the vehicle and $v(k)$ is the measurement noise vector. As mentioned in Section 2, apart from the coordinates of the UAV which are provided by the IMU or GPS one can also consider a distance measurement $d(k)$ from a landmark surface, as shown in Fig. 1. Thus, the measurements vector becomes

$$\begin{aligned}
&z(k) = [x(k) + v_1(k), y(k) + v_2(k), d(k) + v_3(k)], \\
&\quad k = 1, 2, 3 \cdots.
\end{aligned} \tag{31}$$

To obtain the Extended Kalman Filter (EKF), the kinematic model of the UAV is linearized about the estimates $\hat{x}(k)$ and $\hat{x}^-(k)$, and the control input $U(k)$ is applied. The EKF recursion consists of the measurement update part and of the time update part as described in Eqs. (12) and (13) respectively. The input gain matrix $L(k)$ for the UAV model can be written in the form

$$L(k) = \begin{pmatrix} T\cos(\theta(k)) & 0 \\ T\sin(\theta(k)) & 0 \\ 0 & T \end{pmatrix} \tag{32}$$

while the Jacobian matrices $J_\phi(\hat{x}(k))$ and $J_\gamma^T(\hat{x}^-(k))$ are given by

$$J_\phi(\hat{x}(k)) = \begin{pmatrix} 1 & 0 & -v(k)\sin(\hat{\theta}(k))T \\ 0 & 1 & v(k)\cos(\hat{\theta}(k))T \\ 0 & 0 & 1 \end{pmatrix} \tag{33}$$

$$J_\gamma(\hat{x}^-(k)) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\cos(P_n) & -\sin(P_n) & -x_i\cos(\hat{\theta}(k) - P_n) + y_i\sin(\hat{\theta}(k) - P_n) \end{pmatrix}. \tag{34}$$

For the elements of the process noise covariance matrix $Q(k) = \mathrm{diag}[\sigma^2(k), \sigma^2(k), \sigma^2(k)]$ indicative values can be $\sigma^2(k) = 10^{-3}$. Using the estimated state vector, function $\gamma(x)$ appearing in the measurements equations part of the vehicle's kinematic model becomes $\gamma(\hat{x}(k)) = [\hat{x}(k), \hat{y}(k), \hat{d}(k)]$.

In the simulation experiments it has been considered that the coordinates of the distance measuring sensor (x_i, y_i) coincide with the UAV's center of gravity (x, y). Indicative results about tracking of the circular reference trajectory with use of the Extended Kalman Filter are shown in Figs. 2 to 5 and 8 to 10. In Fig. 2 one can notice the accuracy of tracking of the reference trajectory, succeeded by the proposed state estimation-based control scheme. In Figs. 3 and 4, the associated x and y axis tracking errors are shown. In Figs. 5 and 8 the accuracy of tracking of the x and y axis position setpoints is depicted. Finally, in Figs. 9 and 10 the convergence to the x and y axis velocity setpoints is given.

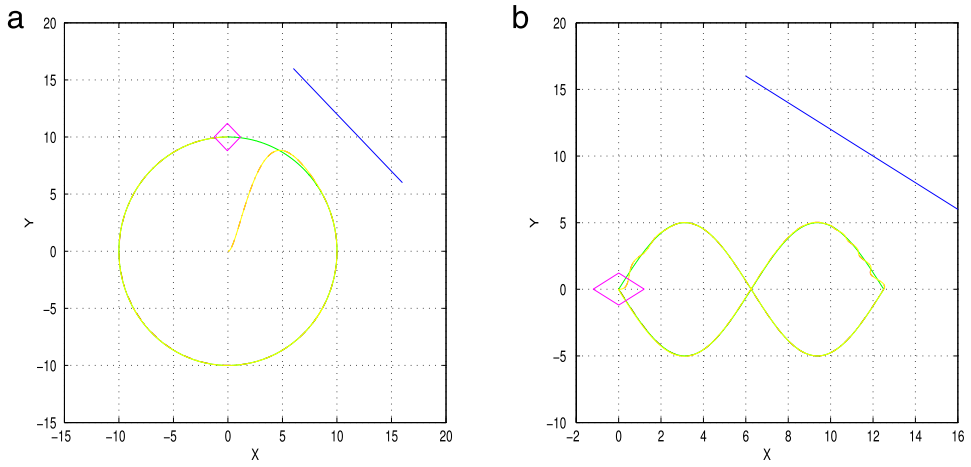


Fig. 2. The trajectory of the UAV (dashed red line) when the UAVs state vector is estimated with the use of Extended Kalman Filtering and (a) the UAV tracks the reference circular path (green line) (b) the UAV tracks the reference eight-shaped path (green line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

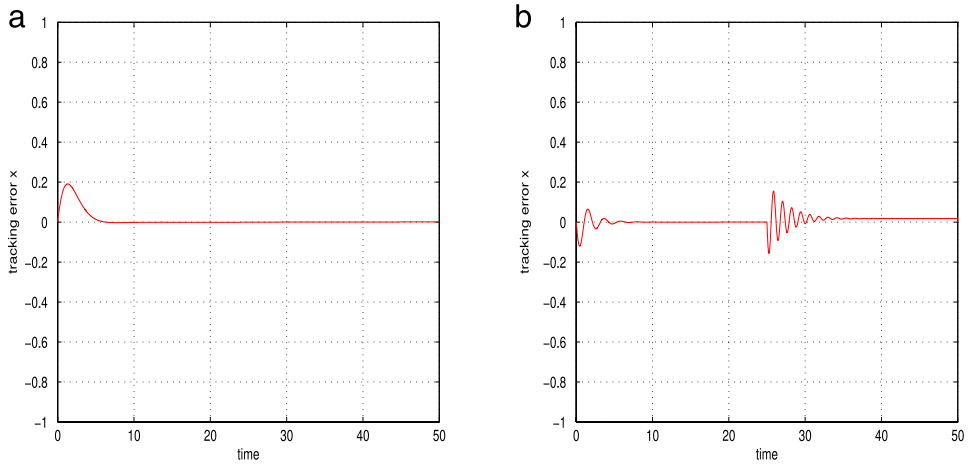


Fig. 3. The x-axis trajectory tracking error of the UAV when the UAVs state vector is estimated with the use of Extended Kalman Filtering and (a) the UAV tracks the reference circular path (b) the UAV tracks the reference eight-shaped path.

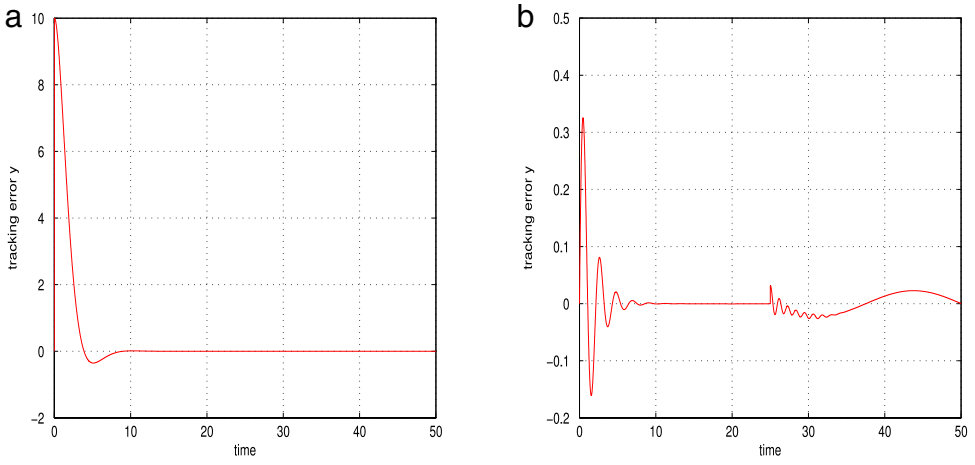


Fig. 4. The y-axis trajectory tracking error of the UAV when the UAVs state vector is estimated with the use of Extended Kalman Filtering and (a) the UAV tracks the reference circular path (b) the UAV tracks the reference eight-shaped path.

7.2. Unscented Kalman Filtering-based Sensor Fusion for UAV navigation

The Unscented Kalman Filter is used next to obtain a solution for the problem of integrated navigation of the UAV. By definition of the measurement vector one has again that the output function $\gamma(x(k))$ is given by $\gamma(x(k)) = [x(k), y(k), d(k)]$. The UAV's state is $[x(k), y(k), \theta(k)]^T$ and the control input is denoted by $U(k) = [u(k), \omega(k)]^T$.

The *time update* of the UKF is

$$x_k^i = \phi(x_{k-1}^i) + L(k-1)U(k-1), \quad i = 0, 1, \ldots, 2n$$

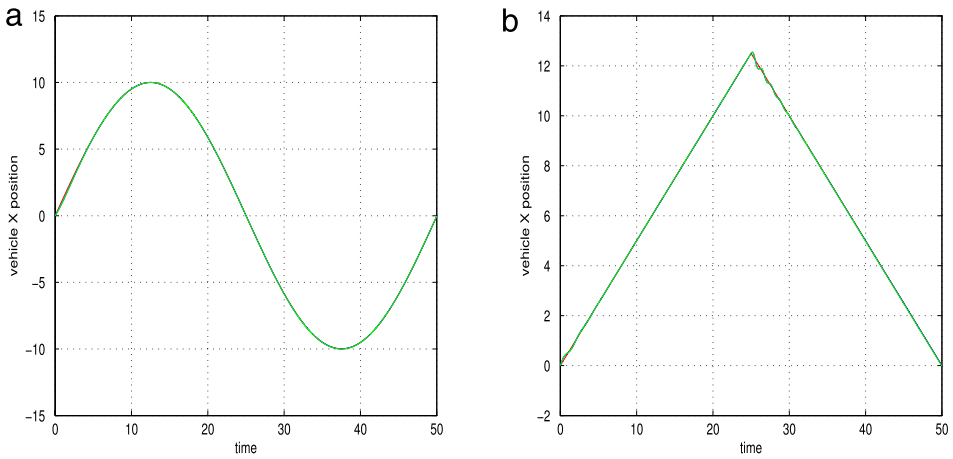


Fig. 5. The *x*-axis position of the UAV (green line) when the UAVs state vector is estimated with the use of Extended Kalman Filtering and (a) the UAV tracks the reference circular path (red line) (b) the UAV tracks the reference eight-shaped path (red line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

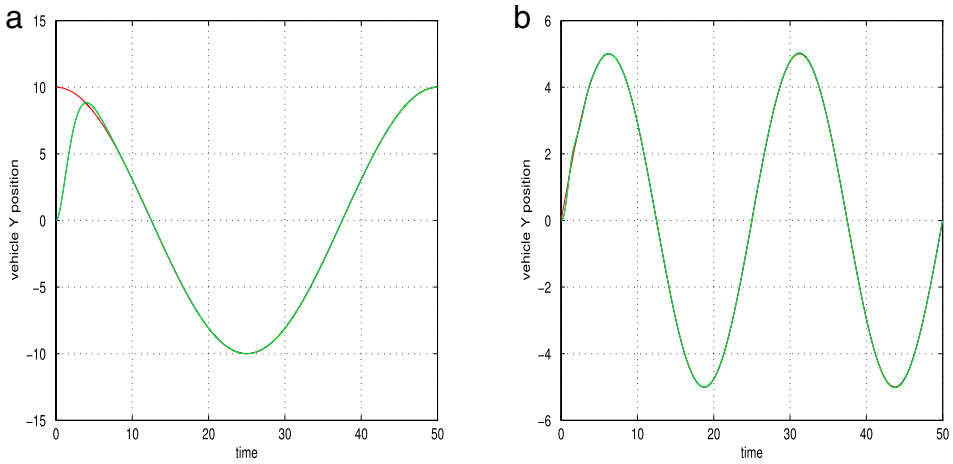


Fig. 6. The *y*-axis position of the UAV (green line) when the UAVs state vector is estimated with the use of Extended Kalman Filtering and (a) the UAV tracks the reference circular path (red line) (b) the UAV tracks the reference eight-shaped path (red line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

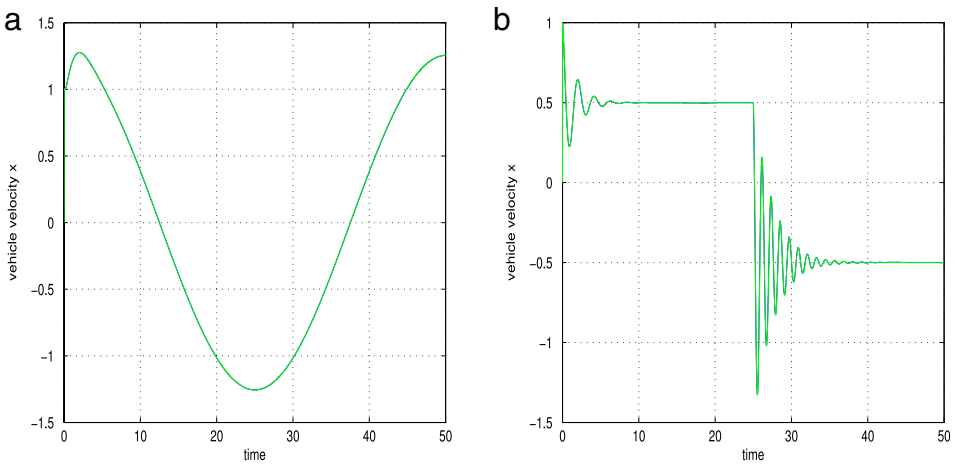


Fig. 7. The *x*-axis velocity of the UAV (green line) when the UAVs state vector is estimated with the use of Extended Kalman Filtering and (a) the UAV tracks the circular path velocity setpoint (blue line) (b) the UAV tracks the eight-shaped path velocity setpoint (blue line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

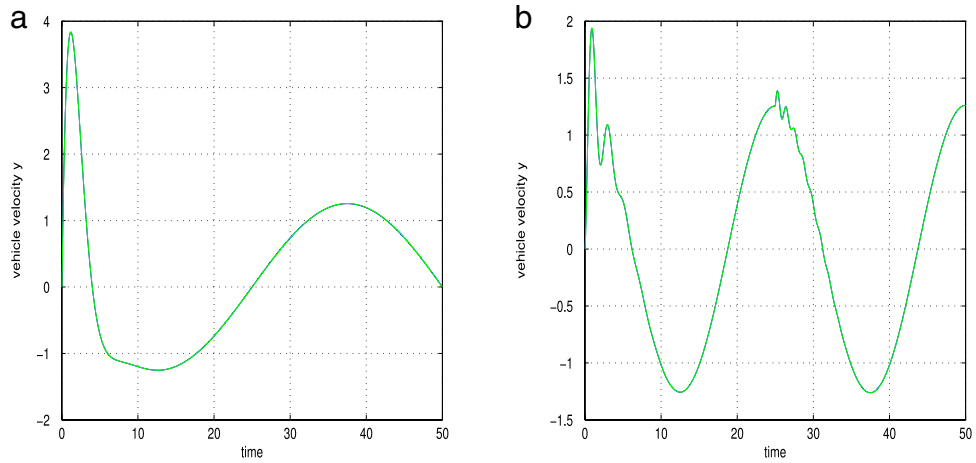


Fig. 8. The *y*-axis velocity of the UAV (green line) when the UAVs state vector is estimated with the use of Extended Kalman Filtering and (a) the UAV tracks the circular path velocity setpoint (blue line) (b) the UAV tracks the eight-shaped path velocity setpoint (blue line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

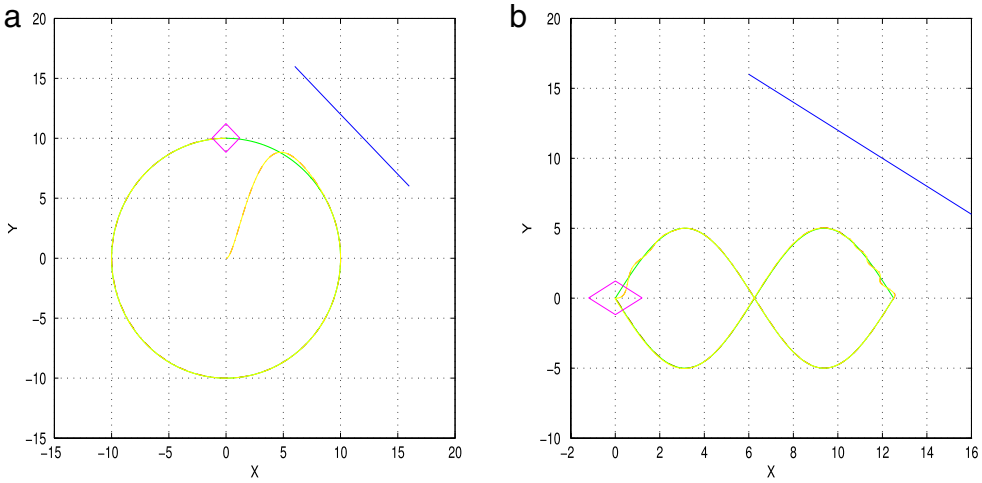


Fig. 9. The trajectory of the UAV (dashed red line) when the UAVs state vector is estimated with the use of Unscented Kalman Filtering and (a) the UAV tracks the reference circular path (green line) (b) the UAV tracks the reference eight-shaped path (green line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

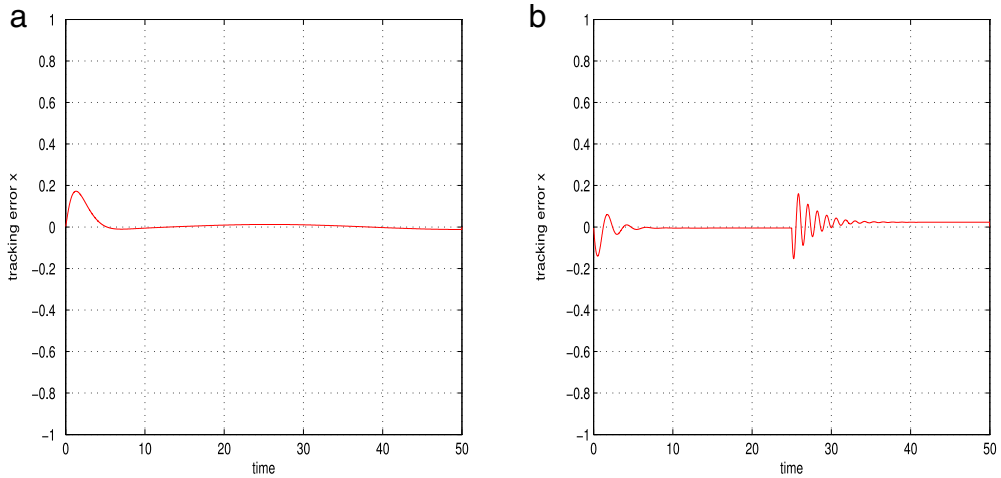


Fig. 10. The *x*-axis trajectory tracking error of the UAV when the UAVs state vector is estimated with the use of Unscented Kalman Filtering and (a) the UAV tracks the reference circular path (b) the UAV tracks the reference eight-shaped path.

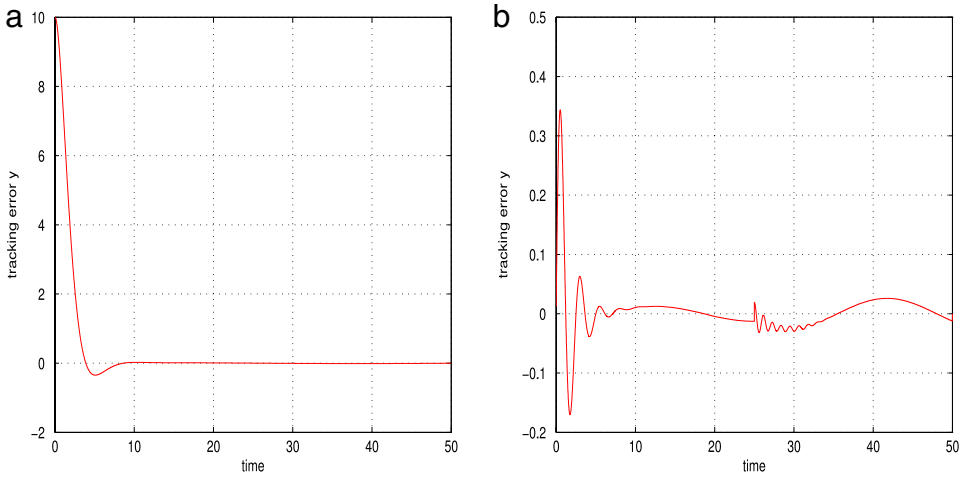


Fig. 11. The *y*-axis trajectory tracking error of the UAV when the UAVs state vector is estimated with the use of Unscented Kalman Filtering and (a) the UAV tracks the reference circular path (b) the UAV tracks the reference eight-shaped path.

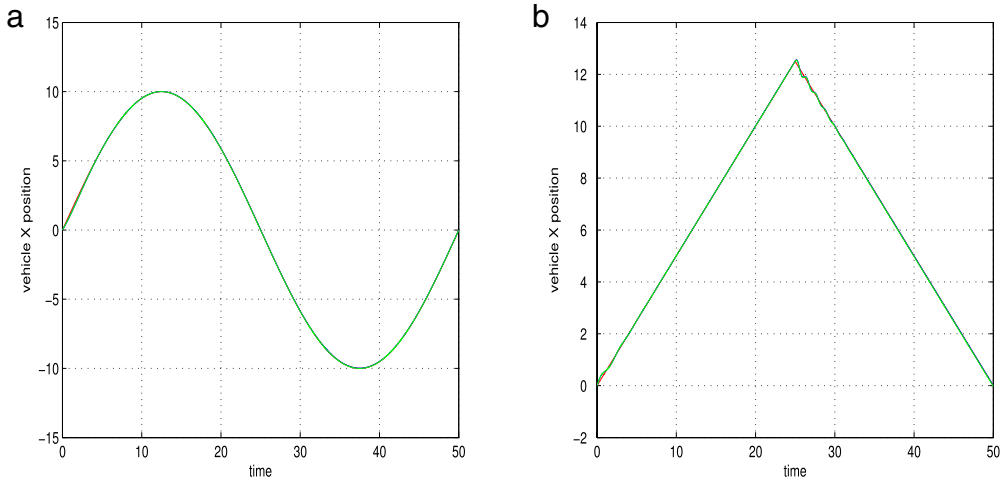


Fig. 12. The *x*-axis position of the UAV (green line) when the UAVs state vector is estimated with the use of Unscented Kalman Filtering and (a) the UAV tracks the reference circular path (red line) (b) the UAV tracks the reference eight-shaped path (red line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

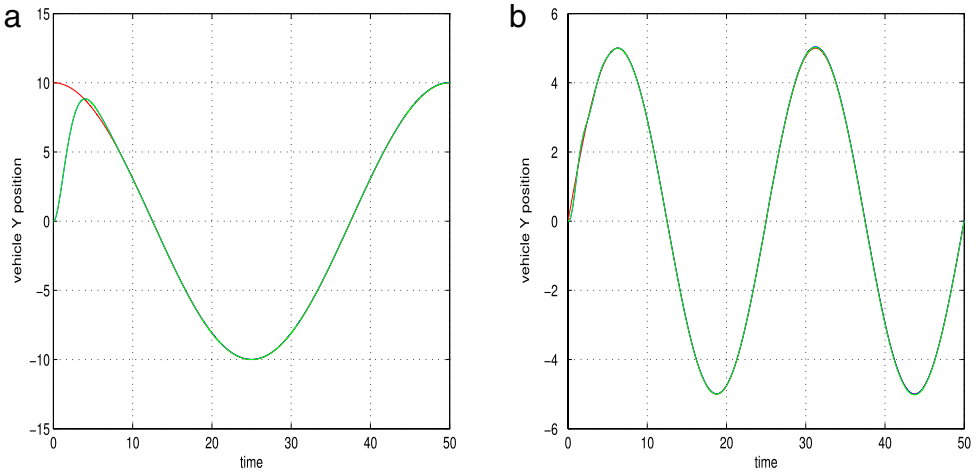


Fig. 13. The *y*-axis position of the UAV (green line) when the UAVs state vector is estimated with the use of Unscented Kalman Filtering and (a) the UAV tracks the reference circular path (red line) (b) the UAV tracks the reference eight-shaped path (red line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

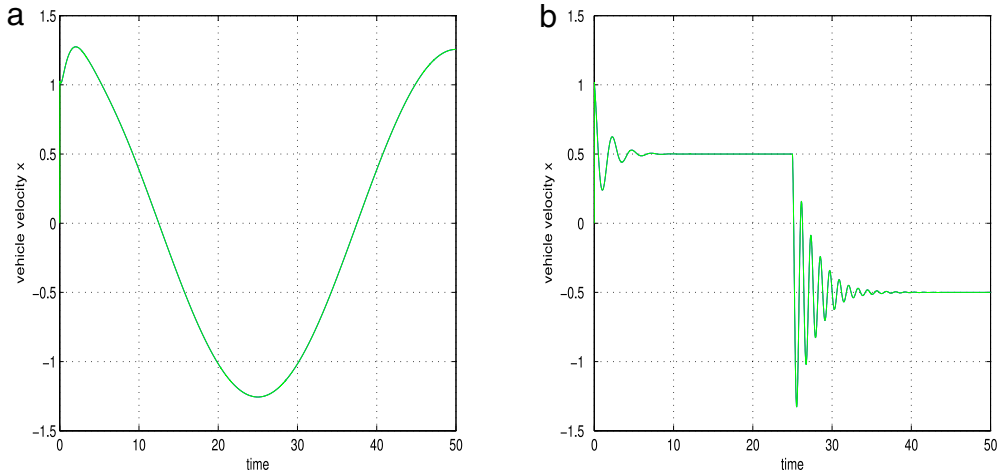


Fig. 14. The *x*-axis velocity of the UAV (green line) when the UAVs state vector is estimated with the use of Unscented Kalman Filtering and (a) the UAV tracks the circular path velocity setpoint (red line) (b) the UAV tracks the eight-shaped path velocity setpoint (red line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

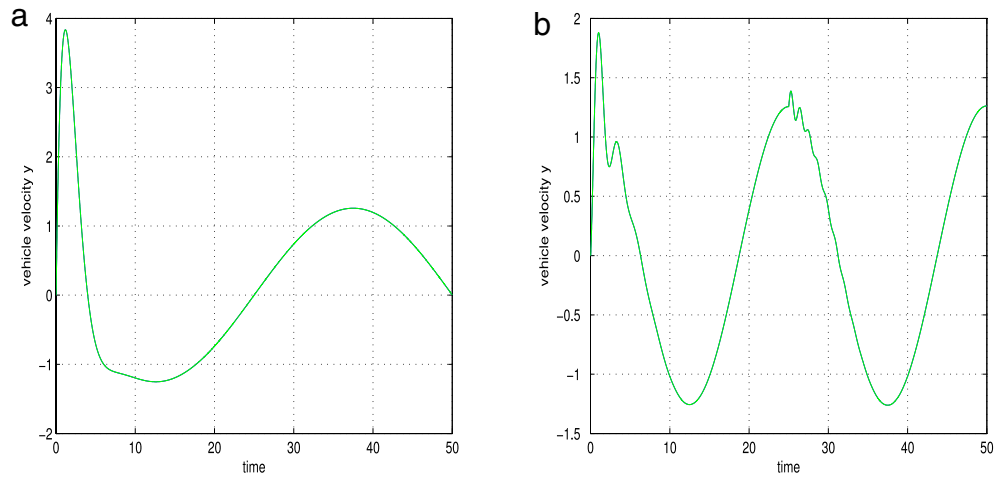


Fig. 15. The y-axis velocity of the UAV (green line) when the UAVs state vector is estimated with the use of Unscented Kalman Filtering and (a) the UAV tracks the circular path velocity setpoint (blue line) (b) the UAV tracks the eight-shaped path velocity setpoint (blue line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

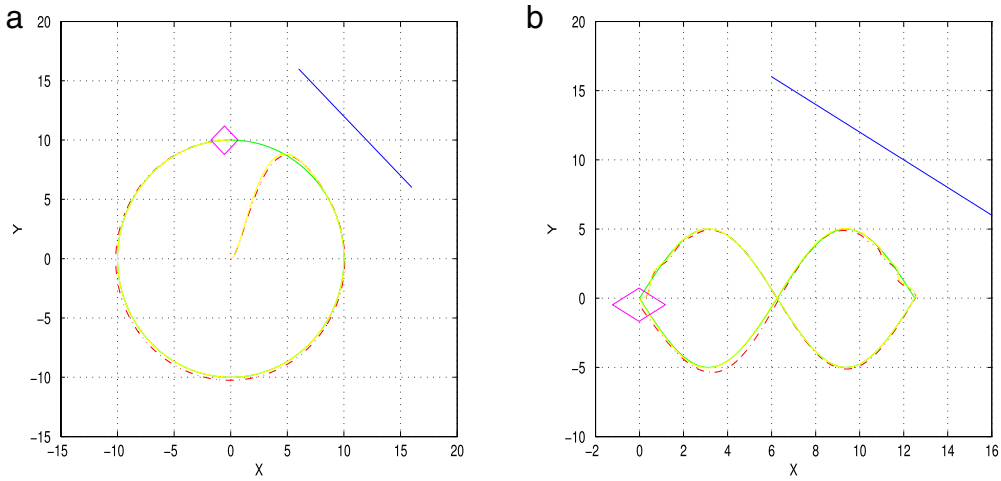


Fig. 16. The trajectory of the UAV (dashed red line) when the UAVs state vector is estimated with the use of Particle Filtering and (a) the UAV tracks the reference circular path (green line) (b) the UAV tracks the reference eight-shaped path (green line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

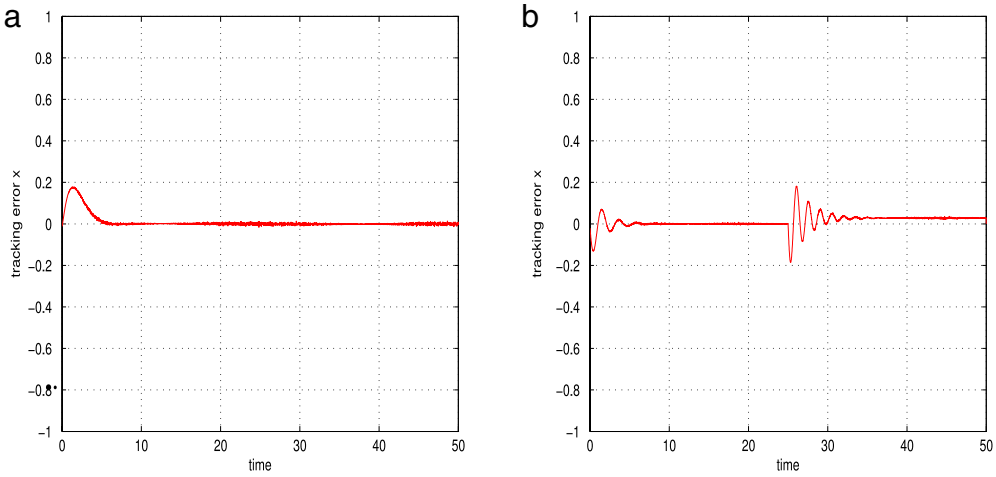


Fig. 17. The x-axis trajectory tracking error of the UAV when the UAVs state vector is estimated with the use of Particle Filtering and (a) the UAV tracks the reference circular path (b) the UAV tracks the reference eight-shaped path.

$$\hat{x}_k^- = \sum_{i=0}^{2n} w_i x_{k^-}^i$$

$$P_{xxk^-} = P_{xxk-1} + Q_k.$$

The *measurement update* of the UKF is

$$z_k^i = h(x_{k^-}^i, u_k) + r_k, \quad i = 0, 1, \ldots, 2n$$

$$\hat{z}_k = \sum_{i=0}^{2n} w_i z_k^i$$

$$P_{zz_k} = \sum_{i=0}^{2n} w_i [z_k^i - \hat{z}_k][z_k^i - \hat{z}_k]^T + R_k$$

$$P_{xz_k} = \sum_{i=0}^{2n} w_i [x_{k^-}^i - \hat{x_{k^-}}][z_k^i - \hat{z}_k]^T$$

$$K_k = P_{xz_k} {P_{zz_k}}^{-1}$$

$$\hat{x}_k = \hat{x}_{k^-} + K_k[z_k - \hat{z}_k]$$

$$P_{xxk} = P_{k^-} - K_k P_{zzk} K_k^T$$

where matrix $L(k)$ is as defined in the EKF case, while $Q(k) = \text{diag}[\sigma^2(k), \sigma^2(k), \sigma^2(k)]$, and $R(k) = \text{diag}[\sigma^2(k), \sigma^2(k), \sigma^2(k)]$, with $\sigma^2(k) = 10^{-3}$. The following parameters were considered in the conducted simulation experiments: state noise $w(k) = 0$, state covariance matrix $P_{xx} \epsilon R^{3\times3}$, $P_{zz} \epsilon R^{3\times3}$, $P_{xz} \epsilon R^{3\times3}$, and UKF Gain $K \epsilon R^{3\times3}$.

The use of UKF for fusing the data that come from the IMU or GPS sensor with the distance measurement $d(k)$ provides an estimation of the state vector $[\hat{x}(k), \hat{y}(k), \hat{\theta}(k)]$ and enables the successful application of the nonlinear steering control of Eq. (2). The obtained results are depicted in Fig. 11, for the case of tracking of a circular and an eight-shaped reference trajectory, respectively. In Fig. 11 one can notice the accuracy of tracking of the reference trajectory, succeed by the UKF-based control scheme. In Figs. 14 and 15 the precision of tracking of the x and y axis setpoints is depicted. In Figs. 12 and 13, the associated x and y axis tracking errors are shown. Moreover, in Figs. 6 and 7 one can observe the tracking of the x and y axis velocity setpoints.

From the simulation tests presented in Figs. 6, 7 and 11 to 15 (which were performed under the assumption of Gaussian measurement noise), it can be observed that the UKF succeeds good approximation of the UAV's state vector and subsequently results in good trajectory tracking. The number of sample points used by the UKF is small, which enables fast computation of the UAV's state vector.

7.3. Particle Filtering-based Sensor Fusion for UAV navigation

The Particle Filter can also provide solution to the problem of integrated UAV navigation. Fusion of measurements provided by the IMU or GPS and a distance sensor was again considered. The UAV model described in Eq. (1), and the control law given in Eq. (2) are used again. The number of particles was set to $N = 1000$. As the number of particles increases, the performance of the particle filter-based tracking algorithm also improves, but this happens at the demand of higher demand for computational resources. Control of the diversity of the particles through the tuning of the resampling procedure may also affect the performance of the algorithm. Results have been obtained again for the case of tracking of a circular and an eight-shaped trajectory, as shown in Fig. 16(a) and (b), respectively. In Figs. 17 and 18, the associated x and y axis tracking errors are shown. In Figs. 19 and 20 the accuracy of tracking of the x and y axis position setpoints is presented. Additionally, in Figs. 21 and 22 the tracking of the x and y axis velocity setpoints is depicted.

From the simulation experiments it can be also seen that the particle filter for a sufficiently large number of particles can have good performance, in the problem of estimation of the state vector of the UAV, without being subject to the constraint of Gaussian distribution for the obtained measurements. The number of particles influences the performance of the particle filter algorithm. The accuracy of the estimation succeeded by the PF algorithm improves as the number of particles increases. The initialization of the particles, (state vector estimates) may also affect the convergence of the PF towards the real value of the state vector of the monitored system. It should be also noted that the calculation time is a critical parameter for the suitability of the PF algorithm for real-time applications. When it is necessary to use more particles, improved hardware and some new technologies, such as making parallel processing available to embedded systems, enable the PF to be implemented in real-time systems [37].

7.4. Derivative-free Kalman Filtering-based Sensor Fusion for UAV navigation

The Derivative-free nonlinear Kalman Filter (DKF) is also used to solve the problem of integrated UAV's navigation. The DKF performs fusion of the UAV's coordinates measurements (x_i, y_i) with the distance d_i of the UAV from a reference surface, as follows:

The system's state vector that was written in the observer canonical form described by Eqs. (29) and (30) is extended

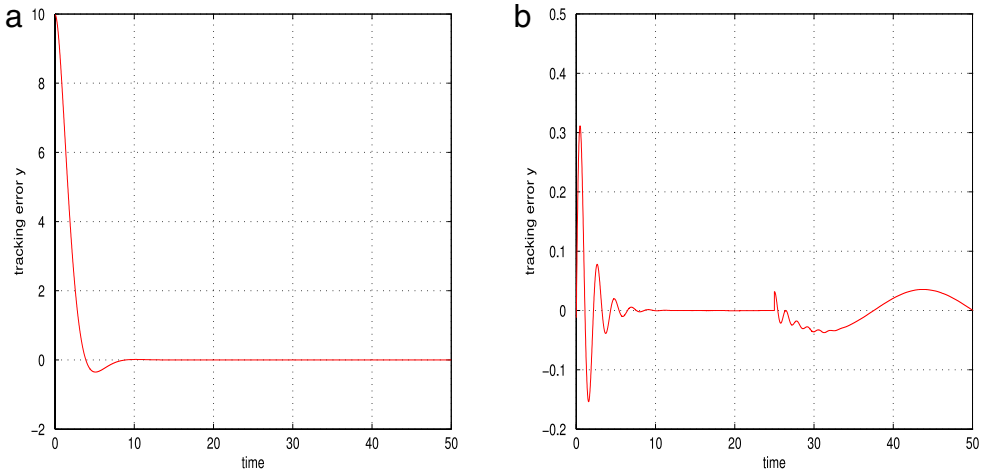


Fig. 18. The y-axis trajectory tracking error of the UAV when the UAVs state vector is estimated with the use of Particle Filtering and (a) the UAV tracks the reference circular path (b) the UAV tracks the reference eight-shaped path.

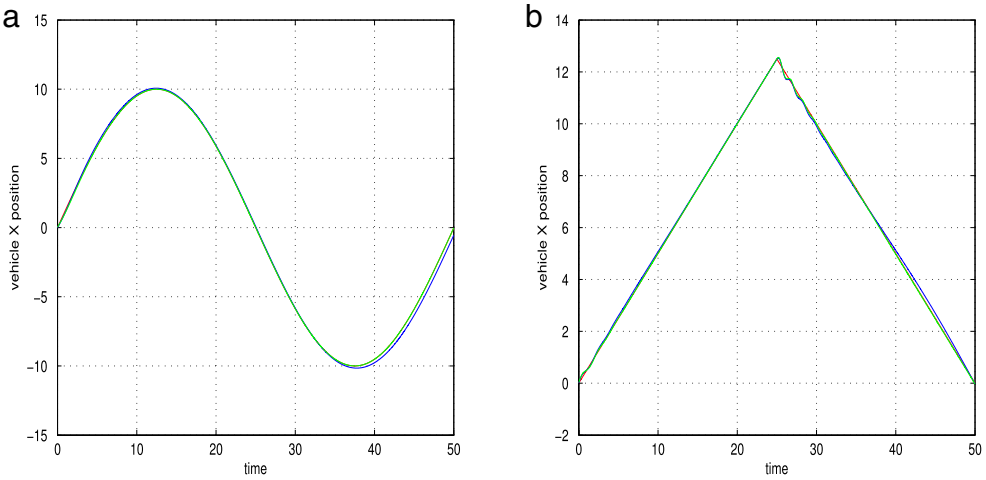


Fig. 19. The x-axis position of the UAV (green line) when the UAVs state vector is estimated with the use of Particle Filtering and (a) the UAV tracks the reference circular path (red line) (b) the UAV tracks the reference eight-shaped path (red line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

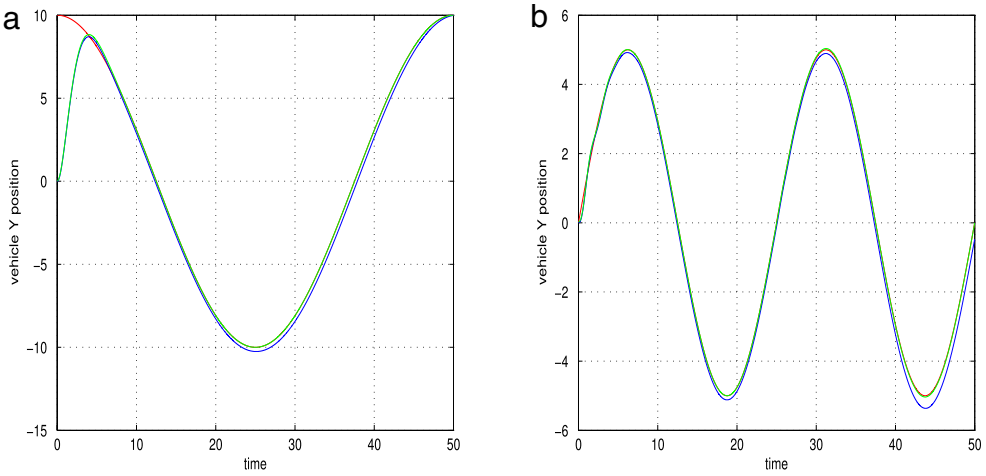


Fig. 20. The y-axis position of the UAV (green line) when the UAVs state vector is estimated with the use of Particle Filtering and (a) the UAV tracks the reference circular path (red line) (b) the UAV tracks the reference eight-shaped path (red line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

through the inclusion of additional state variables that describe the dynamics of the distance measurement d with respect to the reference surface P^j. Thus the extended state vector of the system now becomes $x_e = [x_1, x_2, x_3, x_4, x_5, x_6]^T$ with $x_1 = x$, $x_2 = \dot{x}$, $x_3 = y$, $x_4 = \dot{y}$, $x_5 = d$ and $x_6 = \dot{d}$. The extended output vector is written as $y_e = [y_1, y_2, y_3]^T$, with $y_1 = x$, $y_2 = y$ and $y_3 = d$

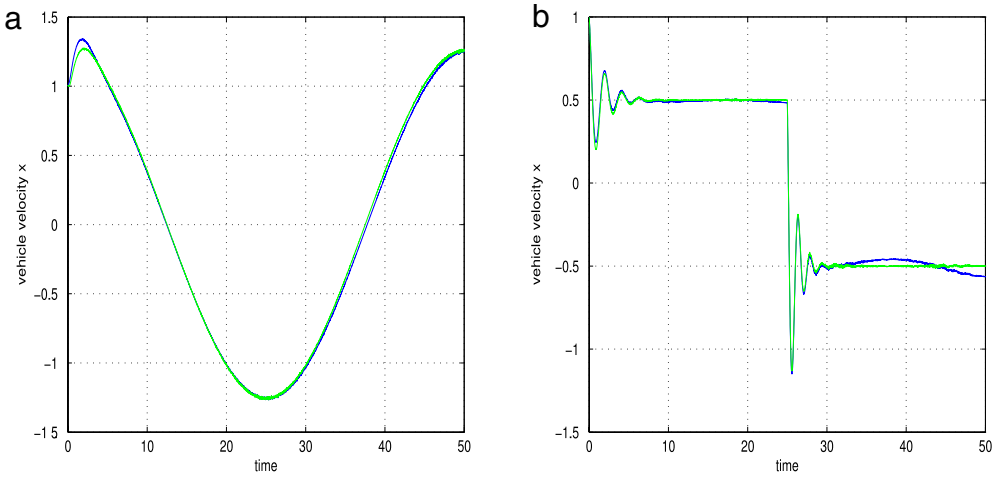


Fig. 21. The x-axis velocity of the UAV (green line) when the UAVs state vector is estimated with the use of Particle Filtering and (a) the UAV tracks the circular path velocity setpoint (blue line) (b) the UAV tracks the eight-shaped path velocity setpoint (blue line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

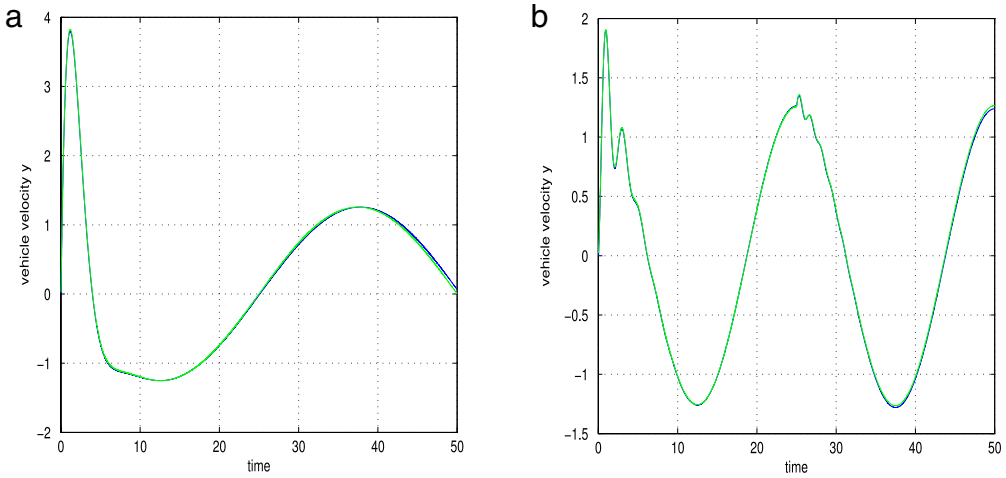


Fig. 22. The y-axis velocity of the UAV (green line) when the UAVs state vector is estimated with the use of Particle Filtering and (a) the UAV tracks the circular path velocity setpoint (blue line) (b) the UAV tracks the eight-shaped path velocity setpoint (blue line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

which means that measurements of the UAV cartesian coordinates (x, y) and of the UAV distance d from the reference surface can be obtained. The distance measuring sensor is taken to coincide with the point defining the cartesian coordinates of the UAV (e.g. center of gravity). In that case, from Eq. (6) one has $x_i = x$ and $y_i = y$. The UAV system is written in the new state-space form which is also an observer canonical form:

$$\begin{aligned} \dot{x}_e &= A_e x_e + B_e v_e \\ y_e &= C_e x_e \end{aligned} \tag{35}$$

while the associated state-space matrices are

$$A_e = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad B_e = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{36}$$

$$C_e = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \tag{37}$$

while the extended inputs vector is defined as $u_e = [u_1, u_2, u_3]^T$ where u_1 and u_2 were defined in Eq. (3). Assuming that the incidence angle P_n does not vary in time (see Fig. 1), one has

$$u_3 = -\ddot{x}_1 \cos(P_n) - \ddot{x}_3 \sin(P_n). \tag{38}$$

It is noted that knowing the orientation of the landmark surface in a cartesian coordinates system, the coordinates of the UAV at time instant $t = kT_s$ and the coordinates of a reference point on the landmark surface, it is always possible to compute the incidence angle P_n. Indicative results are presented again for the case of tracking of a circular and an eight-shaped trajectory, as shown in Fig. 23(a) and (b), respectively. In Figs. 24 and 25, the tracking errors for the setpoints of the x and y axis are given. In Figs. 26 and 27 the precision of tracking for position setpoints of the x and y axis is presented. Additionally, in Figs. 28 and 29, convergence to the x and y axis velocity setpoints is shown.

Indicative results about the accuracy of estimation provided by the considered nonlinear filtering algorithms, as well as about the accuracy of tracking succeeded by the associated state estimation-based control loop are given in Table 1. It can be noticed that the Derivative-free nonlinear Kalman Filter is significantly more accurate and robust than the Extended Kalman Filter. Its accuracy is comparable to the one of the Unscented Kalman Filter. More over, its accuracy is also comparable to the one succeeded by the Particle Filter for a moderate number of particles (e.g. $N = 1000$).

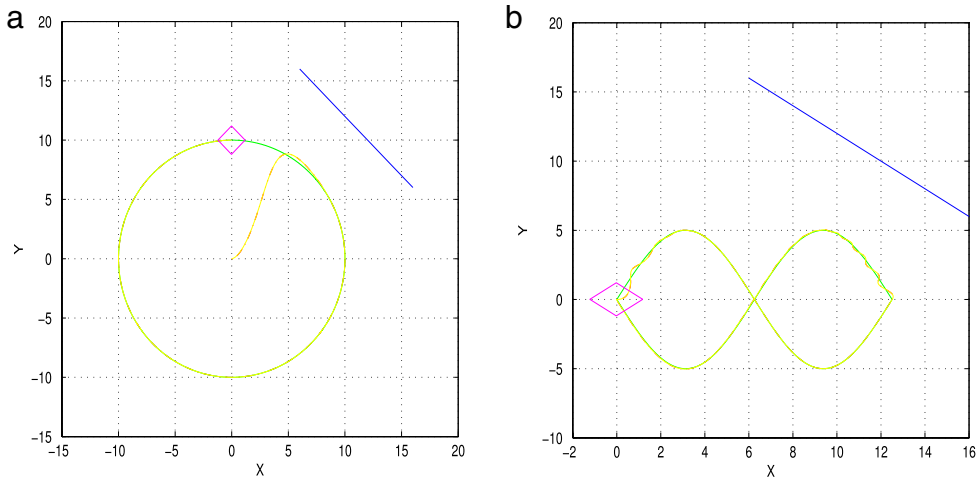


Fig. 23. The trajectory of the UAV (dashed red line) when the UAVs state vector is estimated with the use of Derivative-free nonlinear Kalman Filtering and (a) the UAV tracks the reference circular path (green line) (b) the UAV tracks the reference eight-shaped path (green line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

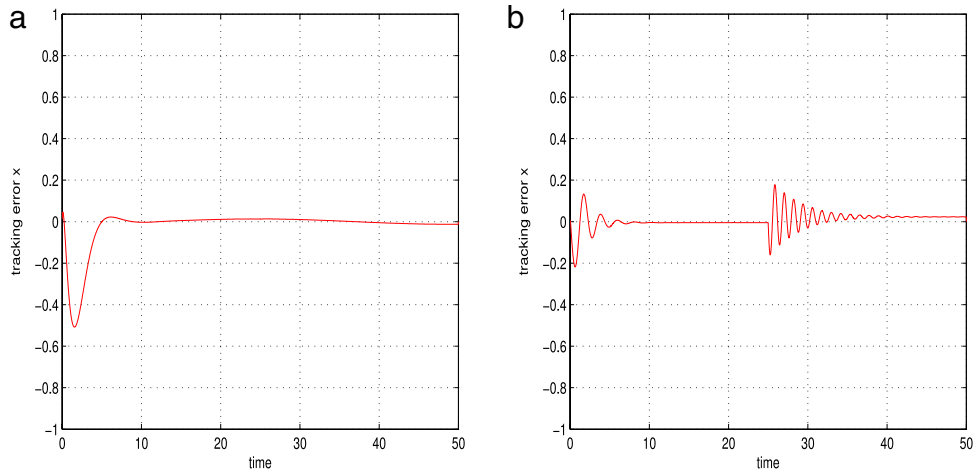


Fig. 24. The *x*-axis trajectory tracking error of the UAV when the UAVs state vector is estimated with the use of Derivative-free nonlinear Kalman Filtering and (a) the UAV tracks the reference circular path (b) the UAV tracks the reference eight-shaped path.

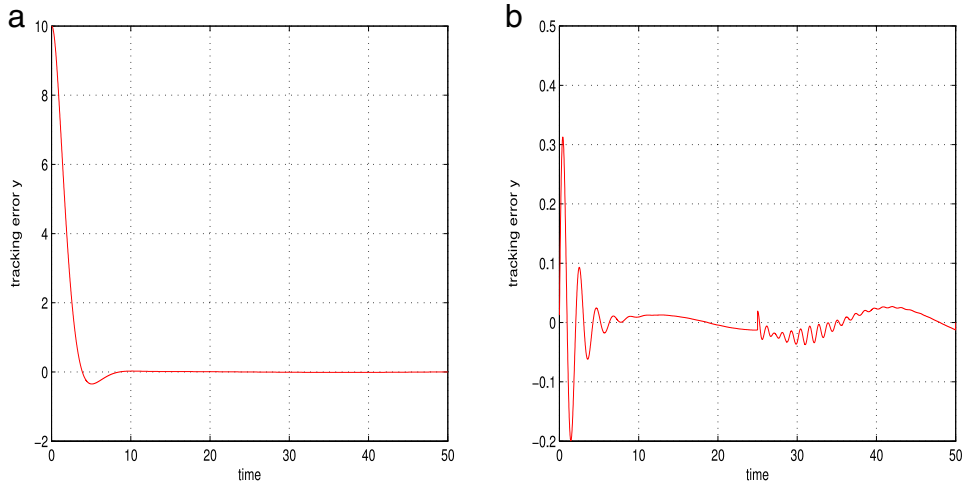


Fig. 25. The *y*-axis trajectory tracking error of the UAV when the UAVs state vector is estimated with the use of Derivative-free nonlinear Kalman Filtering and (a) the UAV tracks the reference circular path (b) the UAV tracks the reference eight-shaped path.

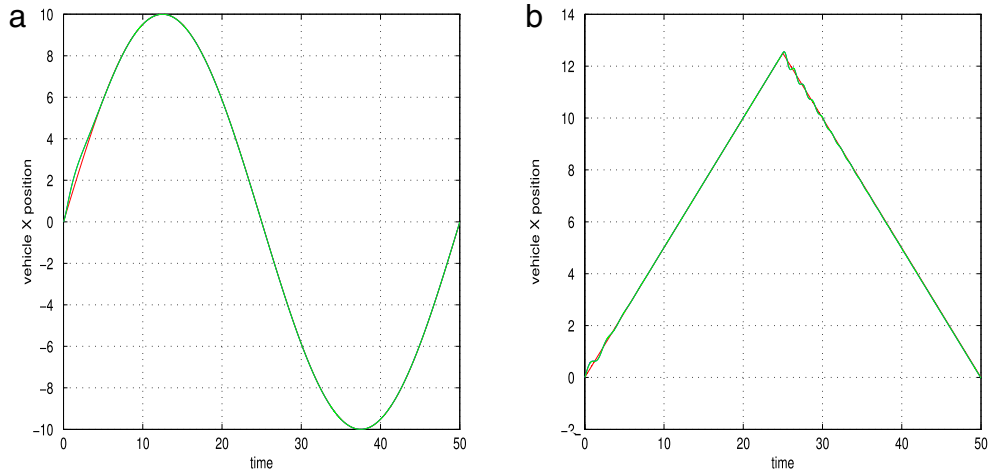


Fig. 26. The *x*-axis position of the UAV (green line) when the UAVs state vector is estimated with the use of Derivative-free nonlinear Kalman Filtering and (a) the UAV tracks the reference circular path (red line) (b) the UAV tracks the reference eight-shaped path (red line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

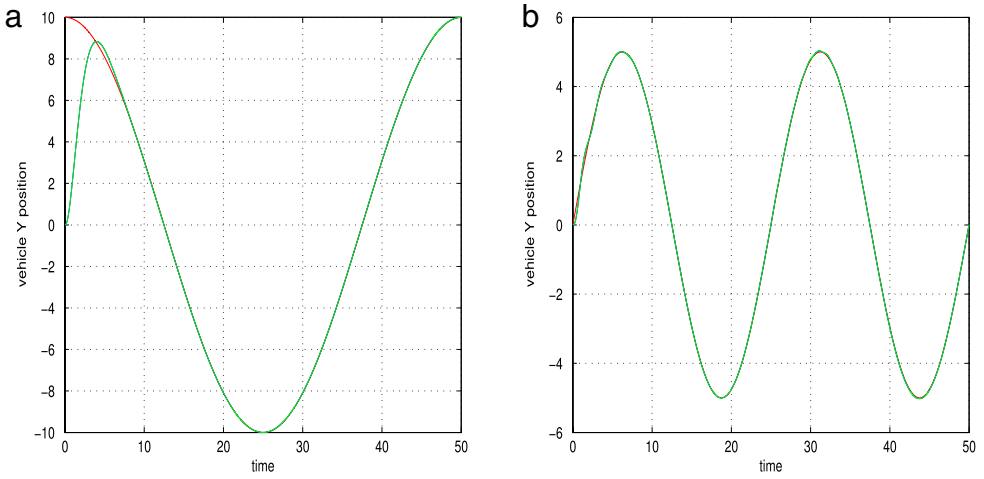


Fig. 27. The *y*-axis position of the UAV (green line) when the UAVs state vector is estimated with the use of Derivative-free nonlinear Kalman Filtering and (a) the UAV tracks the reference circular path (red line) (b) the UAV tracks the reference eight-shaped path (red line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

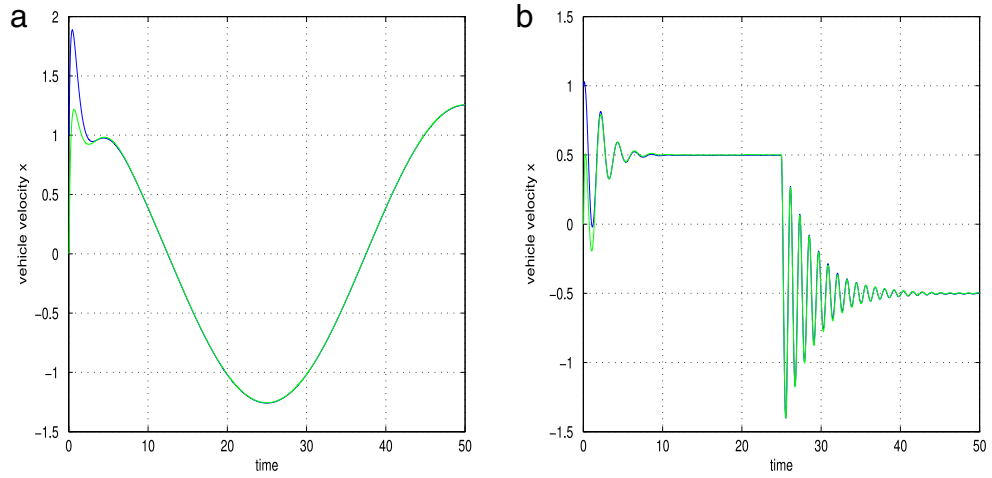


Fig. 28. The *x*-axis velocity of the UAV (green line) when the UAVs state vector is estimated with the use of Derivative-free nonlinear Kalman Filtering and (a) the UAV tracks the circular path velocity setpoint (blue line) (b) the UAV tracks the eight-shaped path velocity setpoint (blue line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

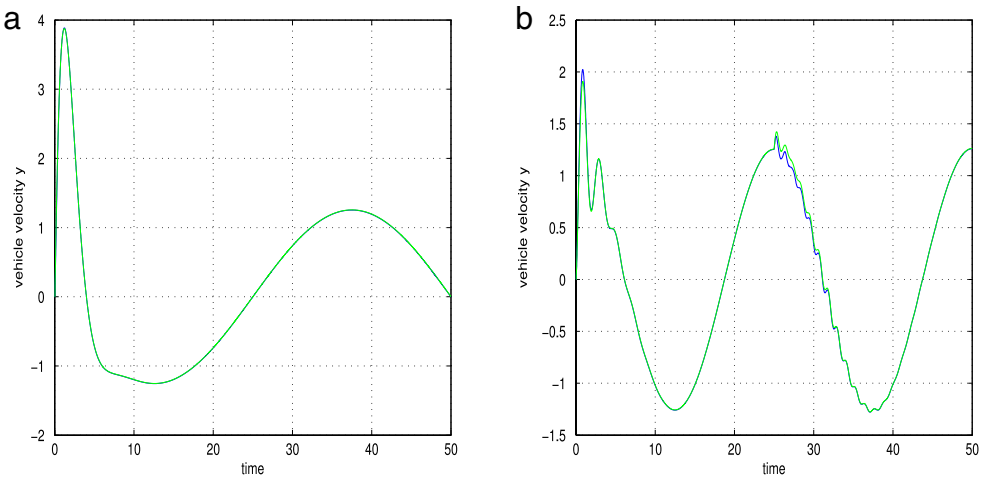


Fig. 29. The y-axis velocity of the UAV (red line) when the UAVs state vector is estimated with the use of Derivative-free nonlinear Kalman Filtering and (a) the UAV tracks the circular path velocity setpoint (blue line) (b) the UAV tracks the eight-shaped path velocity setpoint (blue line). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 1
RMSE of tracking with nonlinear filtering (Gaussian noise).

	$RMSE_x$	$RMSE_y$	$RMSE_\theta$
SPKF	0.0146	0.0162	0.0012
PF	0.0060	0.0031	0.0230
EKF	0.0371	0.0294	0.1835
DKF	0.0086	0.0103	0.0004

Table 2
RMSE of tracking with nonlinear filtering (Rayleigh noise).

	$RMSE_x$	$RMSE_y$	$RMSE_\theta$
SPKF	0.0218	0.0238	0.0026
PF	0.0092	0.0049	0.0717
EKF	0.0475	0.0338	0.2445
DKF	0.0091	0.0102	0.0021

Table 3
Run time of nonlinear estimation algorithms.

	SPKF	PF	EKF	DKF
Total runtime (s)	111.14	623.92	104.39	89.69
Cycle time (s)	0.0222	0.1248	0.0209	0.0179

The Particle Filter can succeed to diminish further the variance of estimation however this requires a large number of particles (e.g. $N > 1500$) and induces additional computational cost (see Table 2).

The aggregate runtime and the cycle time (duration of each iteration of the filter's algorithm) for the Derivative-free nonlinear Kalman Filter, for the Extended Kalman Filter, for the Unscented Kalman Filter and for the Particle Filter (in case of $N = 1000$ particles), using the Matlab platform on a PC with a 1.6 GHz Intel i7 processor, is depicted in Table 3. It can be noticed that the Derivative-free nonlinear Kalman Filter is faster than the other nonlinear estimation algorithms. Actually, the Derivative-free nonlinear Kalman Filter (DKF) appears to be 20% faster than the Unscented Kalman Filter (UKF). The improved speed of the DKF can be more apparent in higher dimensional, multi-degree of freedom systems, where the computation of cross-covariance matrices used by the UKF will impose more numerical operations (see Table 3).

Remark. An additional issue in the performance of nonlinear filtering-based control schemes, is their robustness to model uncertainties and external disturbances. This can be resolved by adopting the disturbance observer concept or the unknown input observer concept in the design of the nonlinear filtering algorithms (see Ref. [13,39]).

8. Conclusions

The paper has studied trajectory tracking by unmanned aerial vehicles, when the UAV's state vector was first estimated by fusing measurements from distributed sensors (for instance IMU and GPS measurements with measurements from distance sensors) and subsequently was used in a nonlinear control loop. The proposed nonlinear control method was *flatness-based control* whereas the filtering methods were (i) Extended Kalman Filtering (EKF), (ii) Sigma-point Kalman Filtering and particularly the Unscented Kalman Filter (UKF), (iii) Particle Filtering (PF), and (iv) Derivative-free nonlinear Kalman Filtering (DKF).

The Extended Kalman Filter is a widely used estimation method in autonomous navigation systems however it is characterized by cumulative errors due to performing an approximate linearization of the system's dynamics. The Unscented Kalman Filter does not include any linearization of the system's dynamics and does not require the computation of partial derivatives and of Jacobian matrices. The UKF is based on a transformation of the system's state vector through its approximation by a set of weighted sample points (sigma-points) which are propagated through the system's nonlinear equation and which progressively converge to the true mean of the state vector's distribution. The Particle Filter makes no assumptions on the forms of the state vector and measurement probability densities. In the particle filter a set of weighted particles (state vector estimates evolving in parallel) is used to approximate the posterior distribution of the state vector. To succeed the convergence of the algorithm, at each iteration resampling takes place through which particles with low weights are substituted by particles of high weights. Finally, in the proposed Derivative-free Kalman Filtering method the system is first subject to a linearization transformation and next state estimation is performed by applying the standard Kalman Filter recursion to the linearized model. The resulting filtering recursion is computationally very efficient since it requires neither the computation of partial derivatives and Jacobians (as in the EKF case) nor the computation of cross-covariance matrices (as in the UKF case).

Simulation tests were used to evaluate the nonlinear control loops which are based on estimation of the UAV's state vector with the aforementioned nonlinear filtering methods. It was shown that the UKF is a reliable and computationally efficient

approach to state estimation-based control, while Particle Filtering is well-suited to accommodate non-Gaussian measurements. It was pointed that the UKF has improved estimation accuracy comparing to the EKF. Moreover, it was noted that in terms of computation speed the UKF is faster than the Particle Filter since it requires less sample points to approximate the state distribution. Both the UKF and PF provide reliable solutions to nonlinear estimation and control problems, with the Particle Filter to require less a-priori knowledge about the statistical characteristics of the measurements and of the system's state variables. Finally, it was shown that the Derivative-free nonlinear Kalman Filter is faster than the other nonlinear estimation algorithms (20% faster than UKF) while its estimation accuracy appears to be comparable and in some test cases improved with reference to the one of UKF.

References

[1] G.G. Rigatos, Particle Filtering for state estimation in industrial robotic systems, IMeche Journal of Systems and Control Engineering 222 (6) (2008) 437–455.
[2] G.G. Rigatos, S.G. Tzafestas, Extended Kalman Filtering for Fuzzy Modelling and Multi-Sensor Fusion, Mathematical and Computer Modelling of Dynamical Systems, vol. 13, Taylor & Francis, 2007, pp. 251–266.
[3] S. Julier, J. Uhlmann, H.F. Durrant-Whyte, A new method for the nonlinear transformations of means and covariances in filters and estimators, IEEE Transactions on Automatic Control 45 (3) (2000) 477–482.
[4] S.J. Julier, J.K. Uhlmann, Unscented Filtering and nonlinear estimation, in: Proceedings of the IEEE, vol. 92, pp. 401–422, 2004.
[5] R. van der Merwe, E.A. Wan, S.I. Julier, Sigma-Point Kalman Filters for nonlinear estimation and Sensor-Fusion applications to intergrated navigation, in: Proceedings of the AIAA Guidance, Navigation and Control Conference, Providence, RI, USA, August, 2004.
[6] S. Särrkä, On unscented Kalman Filtering for state estimation of continuous-time nonlinear systems, IEEE Transactions on Automatic Control 52 (9) (2007) 1631–1641.
[7] R. Kandepu, B. Foss, M. Imsland, Applying the unscented Kalman filter for nonlinear state estimation, Journal of Process Control 18 (2008) 753–768. Elsevier.
[8] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssel, Particle Filters for positioning, navigation and tracking, IEEE Transactions on Signal Processing 50 (2) (2005) 425–437. Special issue on Monte Carlo methods for statistical signal processing.
[9] S. Arulampalam, S.R. Maskell, N.J. Gordon, T. Clapp, A tutorial on particle filters for on-line nonlinear/non-Gaussian Bayesian tracking, IEEE Transactions on Signal Processing 50 (2002) 174–188.
[10] F. Campillo, Particulaire, Modèles de Markov Cachés, Master Course Notes Filtrage et traitment des donées, Université de Sud-Toulon Var, France, 2006.
[11] F. Caron, M. Davy, E. Duflos, P. Vanheeghe, Particle filtering for multi-sensor data fusion with switching observation models: applications to land vehicle positioning, IEEE Transactions on Signal Processing 55 (6) (2007) 2703–2719.
[12] G.G. Rigatos, Autonomous robots navigation using flatness-based control and multi-sensor fusion, in: A. Lazinica (Ed.), Robotics, Automation and Control, I-Tech Education and Publishing KG, Vienna Austria, 2008.
[13] G.G. Rigatos, Modelling and Control for Intelligent Industrial Systems: Adaptive Algorithms in Robotics and Industrial Engineering, Springer, 2011.
[14] G.G. Rigatos, Derivative-free nonlinear Kalman Filtering for MIMO dynamical systems: applications to multi-DOF robotic manipulators, International Journal of Advanced Robotic Systems, InTech (2011).
[15] G.G. Rigatos, A derivative-free Kalman Filtering approach to state estimation-based control of nonlinear dynamical systems, IEEE Transactions on Industrial Electronics (2011).
[16] W. Li, H. Leung, Simultaneous registration and fusion of multiple dissimilar sensors for cooperative driving, IEEE Transactions on Intelligent Transportation Systems 5 (2) (2004) 84–98.
[17] R. Karlsson, T.B. Schön, D. Törnquist, G. Conte, F. Gustafson, Utilizing model structure for efficient simultaneous localization and mapping for a UAV application, in: IEEE 2008 Aerospace Conference. 2008. http://dx.doi.org/10.1109/AERO.2008.4526442.
[18] N. Léchevin, C.A. Rabbath, Sampled-data control of a class of nonlinear flat systems with application to unicycle trajectory tracking, ASME Journal of Dynamical Systems, Measurement and Control 128 (2006) 722–728.
[19] L. Singh, J. Fuller, Trajectory generation for a UAV in urban terrain using nonlinear MPC, in: Proceedings American Control Conference, pp. 2301–2308, 2001.
[20] W. Ren, R.W. Beard, Trajectory tracking for unmanned air vehicles with velocity and heading rate constraints, IEEE Transactions on Control Systems Technology 12 (5) (2004) 706–716.
[21] R.W. Beard, T.W. McLain, M. Goodrich, E.P. Anderson, Coordinated target assignment and intercept for unmanned air vehicles, IEEE Transactions on Robotics and Automation 18 (2002) 911–922.
[22] A.W. Proud, M. Pachter, J.J. D'Azzo, Close formation flight control, in: AIAA Conference on Guidance, Navigation and Control, AIAA-99-4207, 1999.
[23] Y. Zhao, L. Bai, B.W. Gordon, Distributed simulation and virtual reality visualization of multi-robot distributed receding horizon control systems, in: Proceedings of the 2007 IEEE International Conference on Robotics and Biomimetics, Sanya, China, Dec. 2007.
[24] C.L. Cassilo, W. Moreno, K.P. Valavanis, Unmanned helicopter waypoint trajectory tracking using model predictive control, in: 2007 Mediterranean Conference on Control and Automation, Athens, Greece, July 2007.
[25] N. Léchevin, C.A. Rabbath, Sampled-data control of a class of nonlinear flat systems with application to unicycle trajectory tracking, ASME Journal of Dynamical Systems, Measurement and Control 128 (2006) 722–728.
[26] J. Lévine, On necessary and sufficient conditions for differential flatness, in: Applicable Algebra in Engineering, Communications and Computing, vol. 22, Springer, 2011, pp. 47–90.
[27] M. Fliess, H. Mounier, in: G. Picci, D.S. Gilliam (Eds.), Tracking Control and π-Freeness of Infinite Dimensional Linear Systems, in: Dynamical Systems, Control, Coding and Computer Vision, vol. 258, Birkhaüser, 1999, pp. 41–68.
[28] J. Villagra, B. d'Andrea-Novel, H. Mounier, M. Pengov, Flatness-based vehicle steering control strategy with SDRE feedback gains tuned via a sensitivity approach, IEEE Transactions on Control Systems Technology 15 (2007) 554–565.
[29] S. Bououden, D. Boutat, G. Zheng, J.P. Barbot, F. Kratz, A triangular canonical form for a class of 0-flat nonlinear systems, International Journal of Control 84 (2) (2011) 261–269. Taylor and Francis.
[30] G. Oriolo, A. De Luca, M. Vendittelli, WMR control via dynamic feedback linearization: design, implementation and experimental validation, IEEE Transactions on Control Systems Technology 10 (2002) 835–852.
[31] E.W. Kamen, J.K. Su, Introduction to Optimal Estimation, Springer, 1999.
[32] M. Bassevile, I. Nikiforov, Detection of Abrupt Changes: Theory and Applications, Prentice-Hall, 1993.
[33] G. Rigatos, Q. Zhang, Fuzzy Model Validation Using the Local Statistical Approach, Publication Interne IRISA No 1417, Rennes, France, 2001.
[34] S.J. Julier, J.K. Uhlmann, Unscented Filtering and Nonlinear Estimation, in: Proceedings of the IEEE, vol. 92, pp. 401–422, 2004.
[35] S. Thrun, W. Burgard, D. Fox, Probabilistic Robotics, MIT Press, 2005.
[36] G.G. Rigatos, Particle Filtering for State Estimation and Control of DC Motors, ISA Transactions, vol. 48, Elsevier, 2008, pp. 62–72. http://dx.doi.org/10.1016/j.isatra.2008.10.005. 2009.
[37] N. Yang, W.F. Tian, Z.H. Jin, C.B. Zhang, Particle Filter for Sensor Fusion in a Land Vehicle Navigation System, Measurement Science and Technology, vol. 16, Institute of Physics Publishing, 2005, pp. 677–681.
[38] D. Crisan, A. Doucet, A survey of convergence results on particle filtering methods for practitioners, IEEE Transactions on Signal Processing 50 (3) (2002) 736–746.
[39] Y. Xia, Z. Zhu, M. Fu, S. Wang, Attitude tracking of rigid spacecraft with bounded disturbances, IEEE Transactions on Industrial Electronics 58 (2) (2011) 647–659.

Dr. Gerasimos G. Rigatos is currently a Lecturer in "Mechatronics and Artificial Intelligence" at the Department of Engineering, of Harper-Adams University College, in Shropshire, UK. Dr. Rigatos obtained a diploma (1995) and a Ph.D. (2000) both from the Department of Electrical and Computer Engineering, of the National Technical University of Athens (NTUA), Greece. In 2001 he was a post-doctoral researcher at the Institut de Recherche en Informatique et Systèmes Aléatoires IRISA, in Rennes France. Since 2002 he also holds a researcher position at the Industrial Systems Institute (Greek Secretariat for Research and Technology), on the topic of Modeling and Automation of Industrial Systems. In 2007 he was an invited professor (maître des conférences) at Université Paris XI (Institut d' Electronique Fondamentale). He has been also an adjunct professor in Greek Universities where he has taught mostly automatic control courses. His research interests include the areas of mechatronics, robotics and control, computational intelligence and adaptive systems, optimization and fault diagnosis. He is a member of the IEEE, IET and IMACS.