

# Modified switched IMM estimator based on autoregressive extended Viterbi method for maneuvering target tracking

HADAE GH Mahmoudreza<sup>1</sup> and KHALOOZADEH Hamid<sup>2,\*</sup>

1. Department of Electrical & Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran 1477893855, Iran;

2. Department of Systems and Control, K. N. Toosi University of Technology, Tehran 1969764499, Iran

**Abstract:** In this paper, a new approach of maneuvering target tracking algorithm based on the autoregressive extended Viterbi (AREV) model is proposed. In contrast to weakness of traditional constant velocity (CV) and constant acceleration (CA) models to noise effect reduction, the autoregressive (AR) part of the new model which changes the structure of state space equations is proposed. Also using a dynamic form of the state transition matrix leads to improving the rate of convergence and decreasing the noise effects. Since AR will impose the load of overmodeling to the computations, the extended Viterbi (EV) method is incorporated to AR in two cases of EV1 and EV2. According to most probable paths in the interacting multiple model (IMM) during non-maneuvering and maneuvering parts of estimation, EV1 and EV2 respectively can decrease load of overmodeling computations and improve the AR performance. This new method is coupled with proposed detection schemes for maneuver occurrence and termination as well as for switching initializations. Appropriate design parameter values are derived for the detection schemes of maneuver occurrences and terminations. Finally, simulations demonstrate that the performance of the proposed model is better than the other older linear and also nonlinear algorithms in constant velocity motions and also in various types of maneuvers.

**Keywords:** interacting multiple model (IMM) filter, constant acceleration (CA), autoregressive (AR), extended Viterbi (EV), autoregressive extended Viterbi (AREV), extended Kalman filter (EKF).

**DOI:** 10.21629/JSEE.2018.06.04

## 1. Introduction

Maneuvering target tracking is one of the subjects that have been investigated during last decades. Regardless of what equipment should be used to receive target data, a suitable model selection has always been a main problem in target tracking for engineers. Surely what comes to mind is that the selected model should contain some optimizations to prove its advantage compared with other models. Of course it should be mentioned that all ideas of optimizations and

outperformances cannot be realized through a model but always based on problem structure, some important priorities of performance will be considered. Some of these optimizing factors of a model are tracking accuracy, reduction of noise effects and avoidance of unnecessary amount of computations during the target state tracking.

Our studies in last decades confront us with two general kinds of target estimating: First, the serial model methods [1–6] such as the variable-dimension (VD) filter and the input estimation (IE) filter [4–6]. In the VD filter, according to whether the maneuver is detected or not, the target state model changes in the frame of a decision based method. Constant-acceleration (CA) will be replaced with the constant-velocity (CV) model by detection of maneuver onset time and switched back to the CV model after the declaration of maneuver. However, complete reconstruction of state variables is the main problem in the VD filter during the model changing and will cause sharp discontinuities in the processing load [5], which in the IE approach may increase the tracking error [4–6]. In this method, the tracking filter starts in normal mode and detects the maneuver onset time. Then the unknown input (i.e., the target acceleration) is estimated by the least-squares estimation with the data in a sliding window. In the IE method one of the assumptions is that the target acceleration varies slowly, therefore this approach does not give acceptable performance if the acceleration changes rapidly. However, parallel model methods show better performance and the most significant one is the interacting multiple model (IMM) [7–15]. The IMM method consists of a first-degree model for non-maneuvering parts of motion and one or two models for the target maneuvering parts of motion and different process noise for different models. In IMM, weighted estimations of two or three models are combined with each other to make the final and optimal estimation for the IMM filter. In other words, they combine with different probabilities none of which are zero.

Regardless of IMM advantages to serial model methods, two drawbacks can be enumerated for this method: first, overmodeling during non-maneuvering motions which reduce the performance of the algorithm, and second, weakness of traditional CV and CA models to noise effect reduction. CV and CA models can only predict the target states but they cannot reduce the noise appropriately. Numerous improving methods such as reweighted IMM (RIMM), expectation and maximization (EM) Viterbi, particle filters (PF) and variable structure multiple model (VSMM) methods have been proposed in [6–10,16,17]. However, each of them has shortcomings that convince the users to give up the method, such as longer execution time and intensive computations compared with IMM and/or requiring some heuristic information about a problem in order to select an admissible model set to use at any given time (for VSMM model).

In contrast to weakness of traditional CV and CA models to noise effect reduction and disability of IMM to avoid load of overmodeling computations, autoregressive extended Viterbi (AREV) model as a new approach to improve IMM is proposed in this paper.

In the autoregressive (AR) part of this new model, with changing the structure of state space equations and using different forms of the state vector instead of its traditional form and also a dynamic form of the state transition matrix instead of its traditional static model, the rate of convergence can be increased and the noise effect is decreased. In AR, convex quadratic programming can be used to derive the exact solutions for AR model coefficients [18]. The degrees of freedom in the AR model, will reduce the noise by minimizing the mean-square error (MMSE) in addition to satisfaction of polynomial constraint of target motion. Also the filter convergence in the AR model is so rapid, since it can utilize more past data effectively, rather than only use the latest data as the traditional differential model [19].

Also the EV method that is derived from the parallel-type list Viterbi (LV) algorithm [20], is incorporated to AR to decrease the computation load of overmodeling. In EV with selecting distinct  $m$  potentially most likely paths instead of all the possible ones, the computation load of overmodeling will be decreased. In this method, it can work out  $m$  potentially most probable model paths at any time instant based on the EV algorithm and merge the information from these model paths into the state estimation. Moreover, an EV algorithm can be placed between the IMM and a VSMM design in the following sense. Such as IMM implementation, an EV algorithm propagates estimates obtained from all models at all times and like a VSMM design, an EV algorithm may combine and merge only subset of estimates received from all models at any given time to form

the state estimate of a system and thus potentially avoid the loss of accuracy due to overmodeling.

This paper is organized as follows. In Section 2, the traditional system model is introduced. AREV as the proposed model and related computations and properties are presented in Section 3. In Section 4, the switched AREV based tracking algorithm is introduced and finally, Section 5 includes the application of the proposed model and discussion of results and comparison to other linear and nonlinear models of tracking.

## 2. System model statements

We consider one dimensional (1D) discrete-time tracking model in the form

$$\mathbf{x}_{k+1} = \mathbf{A}^j \mathbf{x}_k + \mathbf{F}^j \mathbf{w}_k \quad (1)$$

$$\mathbf{z}_k = \mathbf{H}^j \mathbf{x}_k + \mathbf{D}^j \mathbf{v}_k \quad (2)$$

where  $\mathbf{x}_k$  is the target state value at time  $kT$ , and  $T$  denotes that the sampling time is omitted in relations and calculations for simplicity.  $\mathbf{F}^j \mathbf{w}_k$  and  $\mathbf{D}^j \mathbf{v}_k$  are process and measurement noises respectively, where  $\mathbf{v}_k$  and  $\mathbf{w}_k$  denote independent vector zero-mean white Gaussian processes.  $\mathbf{A}^j$  is the state transition matrix which in traditional system can be in two general forms of CV and CA models.  $\mathbf{H}^j = [1 \ 0 \ \cdots \ 0]_{1 \times k}$  is the measurement matrix and  $k$  is the dimension of target state vector. Index  $j$  is the number of parallel models constructing IMM procedure. Covariance matrices of process and measurement noises are  $\mathbf{Q}_k = E[(\mathbf{F}^j \mathbf{w}_k)(\mathbf{F}^j \mathbf{w}_k)']$  and  $\mathbf{R}_k = E[(\mathbf{D}^j \mathbf{v}_k)(\mathbf{D}^j \mathbf{v}_k)']$  respectively, where the notation  $E[\ ]$  denotes the expectation. The matrices  $\mathbf{Q}_k$  and  $\mathbf{R}_k$  are positive semidefinite and positive definite, respectively. The initial state  $\mathbf{x}_0$  is assumed to be independent of  $\mathbf{v}_k$  and  $\mathbf{w}_k$ . In the above system, neither the state  $\mathbf{x}_k$  vector nor the Markov jump process is observed. Instead only the measurement values can be observed. By the way, the model transition probabilities are given by

$$p_{ij} = p\{r_{k+1} = M_{k+1}^j | r_k = M_k^1\}, \quad \sum_{i=1}^n p_{ij} = 1. \quad (3)$$

## 3. AREV method in IMMs

In this section, AREV is introduced. In the study of known multiple model tracking algorithms such as generalized pseudo Bayesian (GPB) 1 & 2, IMM and VSMM, we can see two common properties. First, using static and time invariant structure for each submodel used in the algorithm, and second, a kind of target state vector that is related to only one step before  $k - 1$ . These traditional models have three general weaknesses.

First, the state transition matrix  $\mathbf{A}$  is one of the most important and effective elements in the performance of tracking and estimation. In the mentioned traditional models, all the parallel submodels for  $j = 1, 2, \dots, n$  ( $n$  is the number of submodels), have static and time invariant transient matrices  $\mathbf{A}^j$ . Although on the premise of the Kalman filter (KF) framework, this kind of transition matrix can have acceptable performance in tracking and estimation, but it suffers from disability to reduce the noise effect and cannot adjust to the process and measurement noise intensities adaptively.

Second, in these traditional models, the target state value in time instant  $k$  is related to only one step before  $k - 1$  while if we can use a wider window of last information, it will be possible to have a more reliable method of prediction and estimation.

Third, one of the problems that we encounter in multiple models target tracking such as GPB and IMM is the computation load of overmodeling that comes into the processing. An improving algorithm such as EV that uses the most probable hypothesis instead of all possible ones and decreases the load of computations will be so effective.

For these problems, new formats of the target state vector and the state transition matrix [21,22] are introduced as follows:

$$\mathbf{x}_k^{\text{AR}} = [x_k \ x_{k-1} \ \cdots \ x_{k-M+1}]^T \quad (4)$$

$$\mathbf{A}_{k+1/k}^{\text{AR}} = \begin{bmatrix} h_1 & h_2 & \cdots & h_M \\ 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 1 & 0 \end{bmatrix}. \quad (5)$$

As seen in (4), this vector is different from traditional target state vectors, since instead of position  $x$ , velocity  $\dot{x}$  and acceleration  $\ddot{x}$ , positions of  $x$  in  $M$  sequential time steps (from time  $k - M + 1$  to time  $k$ ) are used to make the state vector. Also in (5), the state transition matrix is composed of  $M$  dynamic parameters ( $h_m, m = 1, 2, \dots, M$ ) that should be optimized in every step of tracking and estimation. These optimizing parameters are the main difference between the new structure of matrix  $\mathbf{A}^{\text{AR}}$  and its traditional models. Because in traditional ones, the state transition matrix is constant during the tracking and estimation and there is no constraint to improve it during different parts of motion, but in  $\mathbf{A}^{\text{AR}}$ , in addition to the satisfaction of polynomial constraint of target motion [23,24], noise effect minimizing in the form of a specific cost function shall be satisfied.

Equations (6) and (7) show the polynomial constraint of target motion:

$$\mathbf{V}\mathbf{u}_k = \mathbf{b} \quad (6)$$

$$\mathbf{V} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & 2 & \cdots & M \\ 1 & 2^2 & \cdots & M^2 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 2^N & \cdots & M^N \end{bmatrix}_{(N+1) \times M} \quad (7)$$

where  $\mathbf{u}_k = [h_1 \ h_2 \ \cdots \ h_M]^T$ ,  $\mathbf{b} = [1 \ 0 \ \cdots \ 0]^T$  and  $\mathbf{V}$  is a Vandermonde matrix.  $N$  is the highest degree of polynomial used to approximate the range and  $M$  is the number of AR model coefficients. However, an important question is that how (6) can be derived.

It is well known that any continuous target trajectory can be approximated by a polynomial of a certain degree to an arbitrary accuracy. As such, it is possible to model target motion by the  $N$ th degree polynomial in the Cartesian coordinates.

$$\begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix} = \begin{bmatrix} a_0 & a_1 & \cdots & a_N \\ b_0 & b_1 & \cdots & b_N \\ c_0 & c_1 & \cdots & c_N \end{bmatrix} \begin{bmatrix} 1 \\ t \\ \vdots \\ t^N \end{bmatrix} + \begin{bmatrix} w_x(t) \\ w_y(t) \\ w_z(t) \end{bmatrix} \quad (8)$$

with a certain choice of the coefficients  $a_n, b_n, c_n$  ( $n = 0, 1, \dots, N$ ), where  $(x, y, z)$  are the position coordinates and  $(w_x, w_y, w_z)$  are the corresponding noise terms. Such an  $N$ th-degree polynomial amount to assume the  $N$ th time derivative of the position is nearly constant. The CV and CA models [21] are special cases (for  $N = 1, 2$ , respectively) of this general  $N$ th-degree model with white noise. For simplicity, considering the 1D situation without the noise term, after sampling uniformly, the range at time  $t_k$  can be depicted as

$$x_k = \sum_{n=0}^N a_n (t_k)^n \quad (9)$$

where  $t_k = kT$ . The AR model is used to predict the range at time  $k+1$  through the  $M$  latest samples  $x_k, x_{k-1}, \dots, x_{k+1-M}$  [19].

$$x_{k+1} = \sum_{m=1}^M h_m x_{k+1-m} \quad (10)$$

with the AR model coefficients  $h_m$  ( $m = 1, 2, \dots, M$ ) to be optimized in the sense of minimum mean square error (MMSE). According to (9),  $r_{k+1}$  and  $r_{k+1-m}$  in (10) can be written as follows:

$$\begin{aligned} x_{k+1} &= \sum_{n=0}^N a_n (k+1)^n T^n \\ x_{k+1-m} &= \sum_{n=0}^N a_n (k+1-m)^n T^n. \end{aligned} \quad (11)$$

Substituting (11) into (10), we obtain

$$\sum_{n=0}^N a_n(k+1)^n T^n = \sum_{m=1}^M h_m \sum_{n=0}^N a_n(k+1-m)^n T^n. \quad (12)$$

Rearranging the two summations on the right hand, (12) can be rewritten as

$$\sum_{n=0}^N a_n(k+1)^n T^n = \sum_{n=0}^N a_n \sum_{m=1}^M h_m(k+1-m)^n T^n. \quad (13)$$

Extracting the like terms (the terms having the same degree), we have

$$(k+1)^n = \sum_{m=1}^M h_m(k+1-m)^n, \quad n = 0, 1, \dots, N. \quad (14)$$

When  $n = 0$ , the zero degree constraint of the AR model coefficients is obtained from (14) as

$$\sum_{m=1}^M h_m = 1. \quad (15)$$

When  $n = 1$ , (14) is

$$k+1 = \sum_{m=1}^M h_m(k+1-m). \quad (16)$$

Using (15) in (16), we obtain the first-degree constraint of the AR model coefficients

$$\sum_{m=1}^M h_m m = 0. \quad (17)$$

When  $n = 2$ , (14) is

$$(k+1)^2 = \sum_{m=1}^M h_m(k+1-m)^2. \quad (18)$$

Using (15) and (17) in (18), we obtain the second-degree constraint

$$\sum_{m=1}^M h_m m^2 = 0. \quad (19)$$

In similarity, we can refer to the  $n$ th-degree constraint from (17) and (19) and obtain

$$\sum_{m=1}^M h_m m^n = 0. \quad (20)$$

Now we gather all the  $n$ th-degree constraints of the AR model coefficients in an equation

$$\begin{cases} \sum_{m=1}^M h_m = 1, & n = 0 \\ \sum_{m=1}^M h_m m^n = 0, & n = 1, 2, \dots, N \end{cases}. \quad (21)$$

The AR model coefficients are denoted by  $\mathbf{u}_k = [h_1 \ h_2 \ \dots \ h_M]^T$ ,  $\mathbf{b} = [1 \ 0 \ \dots \ 0]^T$  and

$$\mathbf{V} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 2 & \dots & M \\ 1 & 2^2 & \dots & M^2 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 2^N & \dots & M^N \end{bmatrix}_{(N+1) \times M}. \quad (22)$$

Thus (21) can be written as

$$\mathbf{V} \mathbf{u}_k = \mathbf{b}. \quad (23)$$

It can be seen that if  $M = N + 1$ , (23) is consistent [25] with the traditional discrete-time differential model, also the target motion can be depicted by the solution of  $\mathbf{u}_k = \mathbf{V}^{-1} \mathbf{b}$ . In other words, it means  $\mathbf{u}_k$  parameters derivation only depends on (23) but if  $M > N + 1$ , (23) cannot derive  $\mathbf{u}_k$  parameters singly and it is necessary to impose another constraint to have unique answers for  $\mathbf{u}_k$ . The best constraint is the noise effect reduction. It means adding the noise effect reduction constraint not only leads to satisfaction of polynomial constraint of target motion but also decreases the noise effect more than the traditional models. Noise effect reduction is related to minimizing the estimated covariance matrix  $\mathbf{P}_{k|k}$ . Thus two optimizing constraints to derive  $\mathbf{u}_k$  parameters are as follows:

$$\begin{cases} \min_{\mathbf{u}_k} (\mathbf{P}_{k|k}) \\ \text{s.t. } \mathbf{V} \mathbf{u}_k = \mathbf{b} \end{cases}. \quad (24)$$

And according to [21], minimizing of  $\mathbf{P}_{k|k}$  leads to minimizing  $\mathbf{u}_k^T \mathbf{P}_{k-1|k-1} \mathbf{u}_k$  on the base of the KF framework. Thus the final cost function as an optimization formula for deriving  $\mathbf{u}_k$  parameters is

$$\begin{cases} \min_{\mathbf{u}_k} \mathbf{u}_k^T \mathbf{P}_{k-1|k-1} \mathbf{u}_k \\ \text{s.t. } \mathbf{V} \mathbf{u}_k = \mathbf{b} \end{cases}. \quad (25)$$

The optimization formula (25) that is a kind of convex quadratic programming problem can be solved by Lagrange multiplier technique [18]. The closed form solutions of (25) in the cases of  $N = 1$ ,  $M = 4$  and  $N = 2$ ,  $M = 4$  are derived as follows. When  $N = 1$ ,  $M = 4$ , using the method of Lagrange multipliers [18], the Lagrange function can be written as

$$L(h_1, h_2, h_3, h_4, \lambda_0, \lambda_1) =$$

$$\sum_{i=1}^4 \sum_{j=1}^4 h_i (P_{k-1|k-1})_{(i,j)} h_j + \lambda_0 \left( \sum_{i=1}^4 h_i - 1 \right) + \lambda_1 \left( \sum_{i=1}^4 h_i i \right). \quad (26)$$

Setting the partial derivatives of  $L(h_1, h_2, h_3, h_4, \lambda_0, \lambda_1)$  with respect to all the arguments equal to zero, we obtain

$$\frac{\partial L}{\partial h_j} = 2 \sum_{i=1}^4 h_i (P_{k-1|k-1})_{(i,j)} + \lambda_0 + j \lambda_1 = 0, \quad j = 1, 2, 3, 4, \quad (27)$$

$$\frac{\partial L}{\partial \lambda_0} = \sum_{i=1}^4 h_i - 1 = 0 \quad (28)$$

$$\frac{\partial L}{\partial \lambda_1} = \sum_{i=1}^4 h_i i = 0. \quad (29)$$

The coefficients  $h_1$  and  $h_2$  are solved from (28) and (29) as

$$\begin{cases} h_1 = h_3 + 2h_4 + 2 \\ h_2 = -2h_3 - 3h_4 - 1 \end{cases}. \quad (30)$$

Substituting (30) to the partial derivatives  $\partial L / \partial h_j$  ( $j = 1, 2, 3, 4$ ), we have

$$h_3 = \frac{(2g_2^T P_{k-1|k-1} g_2)(g_1^T P_{k-1|k-1} g_3 + g_3^T P_{k-1|k-1} g_1) - (g_1^T P_{k-1|k-1} g_2 + g_2^T P_{k-1|k-1} g_1)(g_2^T P_{k-1|k-1} g_3 + g_3^T P_{k-1|k-1} g_2)}{(g_1^T P_{k-1|k-1} g_2 + g_2^T P_{k-1|k-1} g_1)^2 - 4g_1^T P_{k-1|k-1} g_1 \cdot g_2^T P_{k-1|k-1} g_2} \quad (31)$$

$$h_4 = \frac{(2g_1^T P_{k-1|k-1} g_1)(g_2^T P_{k-1|k-1} g_3 + g_3^T P_{k-1|k-1} g_2) - (g_1^T P_{k-1|k-1} g_2 + g_2^T P_{k-1|k-1} g_1)(g_1^T P_{k-1|k-1} g_3 + g_3^T P_{k-1|k-1} g_1)}{(g_1^T P_{k-1|k-1} g_2 + g_2^T P_{k-1|k-1} g_1)^2 - 4g_1^T P_{k-1|k-1} g_1 \cdot g_2^T P_{k-1|k-1} g_2} \quad (32)$$

where  $g_1 = [1 \ -2 \ 1 \ 0]^T$ ,  $g_2 = [2 \ -3 \ 0 \ 1]^T$  and  $g_3 = [2 \ -1 \ 0 \ 0]^T$ . Also using the same method, we obtain the solution of the optimization problem (25) when  $N = 2$ ,  $M = 4$  as

$$\begin{cases} h_4 = -\frac{g_1^T P_{k-1|k-1} g_2 + g_2^T P_{k-1|k-1} g_1}{2g_1^T P_{k-1|k-1} g_1} \\ h_1 = -h_4 + 3 \\ h_2 = 3h_4 - 3 \\ h_3 = -3h_4 + 1 \end{cases} \quad (33)$$

where  $g_1 = [-1 \ 3 \ -3 \ 1]^T$ ,  $g_2 = [3 \ -3 \ 1 \ 0]^T$ .

As mentioned in the beginning of this section, AREV is incorporated to IMM as an improving method, so it is necessary to review the IMM structure and see finally what will happen to IMM after improvement.

One cycle of the IMM estimator has four basic steps: model conditioned reinitialization, model conditioned filtering, mode probability update and estimate fusion.

**Step 1** Calculation of transition matrix. This is a new step that should be incorporated to the IMM structure [19] to calculate state transition matrices. As we mentioned before, in every time step, convex quadratic programming (25) should be solved and obtain the transition matrices  $A^{AR}$  in all submodels.

**Step 2** Model conditioned reinitialization. The first part of Step 2 is calculating the mixing weight  $\mu_{k-1}^{j|i} = p_{ji} \mu_{k-1}^j / \mu_{k-1}^i$ . As seen, calculation of the mixing weight is on the base of predicted mode probability  $\mu_{k-1}^{(i)}$ . Also for calculating  $\mu_{k-1}^{(i)}$ , all the submodels from  $j = 1, 2, \dots, n$  are used, which means all the possible paths

with different probabilities shall be used to calculate this parameter. In other words, some of these paths may have no significant effects on this calculation and only increase the load of overmodeling, therefore it seems it is better to use most probable paths instead of all possible ones because this new idea not only avoids the influence of low probable submodels in the calculations but also increases the velocity of processing.

One of the most optimal algorithms introduced in [20,26] is LV. In this method, in any time  $k$ , only the  $m$  most probable paths ( $1 \leq m < n$ ) instead of all the possible ones that reach the submodel  $j$  ( $M^j$ ) are considered. According to this procedure, if  $w_k(i_{k-1}, j_k)$  be a probability weight of transition from submodel  $i$  in time  $k-1$  to submodel  $j$  in time  $k$ , and  $p^r = (i_0^r, i_1^r, \dots, i_N^r)$  be a path from time instant 0 to time instant  $N$  through the trellis, then the total probability weight of this path can be written as  $d_N(p^r) = \prod_{k=0}^{N-1} w_k(i_k^r, i_{k+1}^r)$ . Now we can define an optimized equation as follows:

$$\begin{cases} \max \sum_{l=1}^m d_N(p^l), \\ \text{s.t. } p^i \neq p^j, \quad i, j = 1, 2, \dots, m \end{cases}. \quad (34)$$

The goal of the above optimization problem is to find  $m$  distinct paths which obtain the largest sum of the  $m$  total probability weights. The LV algorithm was explained in [20,26] in details. However, it appears to us there is no functional mechanism of the LV algorithm to integrate with GPB and/or IMM algorithms for further hypothesis reduction. Implicitly, the fixed memory requirements

lead to the LV algorithm using  $n \times m \times k$  quantities to store the path history for each time instant  $k$ . Therefore instead of LV, a suboptimal algorithm called extended Viterbi (EV) [27] algorithm can be used. This algorithm can solve (34) too, while in contrast to LV, it can incorporate some functional mechanisms into the GPB and IMM algorithms for hypothesis reduction and also fewer quantities ( $n \times k$ ) to store the path history for each time instant  $k$ . Thereby a new method is yielded for efficiently computing a suboptimal state estimate of a discrete-time system with Markov switching parameters. The EV algorithm is shown as follows.

#### Initialization

$m$  and  $n$  are defined with  $1 \leq m \leq n$ ;  $Weight_0(j) = 1/n, j = 1, 2, \dots, n$ .

For  $k = 1, 2, \dots, N$ ,

#### Step 1 Recursion

For  $j = 1, 2, \dots, n, s = 1, 2, \dots, m$ ,

$$Weight_k(k) = \sum_{s=1}^m \max_{1 \leq i \leq n} \{w_k(i, j)\}.$$

$Weight_{k-1}(i)\}$ ,

$Apath_k^s(j) = \{Apath_{k-1}^1(i^*), j\}$ ,

where  $i^* = \arg\{\max_{1 \leq i \leq n} \{w_k(i, j)\}\}$ .

$Weight_{k-1}(i)\}$ ,

$index_k(j, s) = i^*$ .

#### Step 2 $m$ potentially best paths

For  $s = 1, 2, \dots, m$

$EV - Path_k(s) = Apath_k^1(i^*),$

where  $i^* = \arg\{\max_{1 \leq i \leq n} (Weight_k(i))\}$

And according to the EV algorithm, mixing weights can be derived as follows:

$$\begin{cases} l_{sj} = \arg\{\max_{1 \leq i \leq n} \{p_{ij}\mu_{k-1}(i)\}\} \\ \mu_{k-1}(l_{sj}|j) = \frac{\max_{1 \leq i \leq n} \{p_{ij}\mu_{k-1}(i)\}}{\sum_{s=1}^m \max_{1 \leq i \leq n} \{p_{ij}\mu_{k-1}(i)\}} \end{cases} \quad (35)$$

$n$  is the number of submodels and  $m$  is an integer ( $1 \leq m < n$ ). As seen in (35), in the EV algorithm not all the possible submodels but only the  $m$  most likely model paths are used to compute mixing weights.

The second and third parts of Step 2 are mixing estimate and mixing covariance computations. In the traditional IMM model, mixing estimate  $\bar{x}_{k-1|k-1}^i$  is related to state values and mixing weights of all the submodels with every probability and also mixing covariance  $P_{k-1|k-1}^i$  is related to the covariance matrix and mixing weights of all submodels. However, as we expect in the optimal EV algorithm, for computation of mixing estimate and covariance, only  $m$  probable submodel paths with their mixing

weights, state values and covariance matrices are used as follows:

$$\hat{x}_{k-1}^{0j} = \sum_{s=1}^m \hat{x}_{k-1}^{l_{sj}} \mu_{k-1}(l_{sj}|j) \quad (36)$$

$$P_{k-1}^{0j} = \sum_{s=1}^m \mu_{k-1}(l_{sj}|j) \{P_{k-1}^{l_{sj}} + [\hat{x}_{k-1}^{l_{sj}} - \hat{x}_{k-1}^{0j}] \cdot [\hat{x}_{k-1}^{l_{sj}} - \hat{x}_{k-1}^{0j}]'\}. \quad (37)$$

#### Step 3 Model conditioned filtering

This step is the KF framework in which the predicted states values and predicted covariance matrices and after that estimated states values and estimated covariance matrices are calculated for all the submodels.  $K_k$  is the filter gain matrix and  $S_k$  is the innovation covariance matrix. This step is introduced as follows [16]:

$$\hat{x}_{k|k-1}^j = A_{k|k-1}^j \hat{x}_{k-1|k-1}^{0j} \quad (38a)$$

$$P_{k|k-1}^j = A_{k|k-1}^j P_{k-1|k-1}^{0j} (A_{k|k-1}^j)^T + Q^j \quad (38b)$$

$$S_k^j = H_k^j P_{k|k-1}^j (H_k^j)^T + R_k^j \quad (38c)$$

$$K_k^j = P_{k|k-1}^j (H_k^j)^T (S_k^j)^{-1} \quad (38d)$$

$$\hat{x}_{k|k}^j = \hat{x}_{k|k-1}^j + K_k^j [z_j - H_k^j \hat{x}_{k|k-1}^j] \quad (38e)$$

$$P_{k|k}^j = [I^j - K_k^j H_k^j] P_{k|k-1}^j. \quad (38f)$$

KF frameworks for traditional IMM and AREV models are the same obviously, but substantially there are many differences between them, three major factors make these differences that cannot be seen obviously, state transition matrix  $A_{k|k-1}^j$ , mixing estimate  $\hat{x}_{k-1|k-1}^{0j}$  and mixing covariance  $P_{k-1|k-1}^{0j}$ . In traditional IMM, the state transition matrix is static and time invariant, which means it does not change during different times and different parts of motions but in the AREV model this matrix is dynamic and updated for every step time. As mentioned above, convex quadratic programming solve and derive dynamic parameters of this matrix. Also in IMM, mixing estimate and mixing covariance are calculated based on all possible submodels but in AREV these two parameters are derived only on the base of  $m$  most likely submodel paths and make more faster algorithm in convergence. Although these three major factors are entered in the first two relations (38a) and (38b) but according to the explanations they influence all other equations in the KF framework

#### Step 4 Mode probability update

After computing estimated state and covariance for each submodel, these values shall be combined to each other to make overall estimate and overall covariance respectively.

However, the weight of combination for each state and covariance is calculated in Step 4 as mode probability. Calculation of mode probability  $\mu_k^i$  is dependent of calculation of model likelihood  $\Lambda_k^i$  and both of them are calculated for all submodels, but it is notable that in the traditional IMM model, calculation of  $\mu$  and  $\Lambda$  is on the base of all possible model paths that reach the  $i$ th model but in the proposed AREV model,  $\mu$  and  $\Lambda$  will be computed on the base of  $m$  most probable paths that reach the  $i$ th model. In most cases  $m = 1$  for CV parts and  $m = 2$  for CA parts of motion. Then mode probability update for the proposed AREV model is as follows:

$$\begin{cases} \Lambda_k^j = N(d_k^j; 0, S_k^j) \\ d_k^j = [z_j - H_k^j \hat{x}_{k|k-1}^j] \\ \mu_k(j) = \frac{\Lambda_k(j) \sum_{s=1}^m \max_{1 \leq i \leq n}^{(s)} \{P_{ij} \mu_{k-1}(i)\}}{\sum_{j=1}^n \Lambda_k(j) \sum_{s=1}^m \max_{1 \leq i \leq n}^{(s)} \{P_{ij} \mu_{k-1}(i)\}} \end{cases} \quad (39)$$

#### Step 5 Calculation of $m$ largest mode probabilities

This step is specially for the proposed AREV model. In previous steps, mode probabilities for all submodels are calculated but since the overall estimate and covariance are calculated based on the  $m$  most likely estimations, we have to calculate  $m$  largest mode probabilities. Thus according to the EV model, the following equations are derived:

$$\begin{cases} \tilde{l}_r = \arg\{\max_{1 \leq j \leq n} \{\mu_k(j)\}\} \\ \tilde{\mu}_k(\tilde{l}_r) = \frac{\max_{1 \leq j \leq n} \{\mu_k(j)\}}{\sum_{r=1}^m \max_{1 \leq j \leq n} \{\mu_k(j)\}'} \end{cases} \quad (40)$$

#### Step 6 Estimate fusion

This is the last step for estimation and in this step overall estimation and covariance are calculated from the combination of the  $m$  most likely estimations derived from Steps 3–5.

$$\hat{x}_k = \sum_{r=1}^m \tilde{\mu}_k(\tilde{l}_r) \hat{x}_k^{\tilde{l}_r} \quad (41)$$

$$P_k = \sum_{r=1}^m \tilde{\mu}_k(\tilde{l}_r) \{P_k^{\tilde{l}_r} + [\hat{x}_k^{\tilde{l}_r} - \hat{x}_k][\hat{x}_k^{\tilde{l}_r} - \hat{x}_k]^T\}. \quad (42)$$

The structure of the proposed AREV model is presented in Table 1. As we know, the number of models  $n$  in most cases of IMM is 3 in the manner that the first submodel ( $M = 4$  &  $N = 1$ ) is suitable for CV parts of motion and the second and third submodels are used for CA parts of motion.  $m$  is smaller than  $n$  in the form of  $1 \leq m < n$ , it means that  $m = 1$  or  $m = 2$ . The choice for suitable  $m$  depends on two factors, load of computation and tracking

performance. However, according to results of [27], in the situation of non-maneuvering motion where the CV model in IMM models can be suitable for estimation and tracking, AREV( $m = 1$ ) with most probable paths will be enough and larger  $m$  increases the load of computation and leads to overmodeling. However, in the case of velocity changing, the most probable path is not reliable and merging estimates from two most probable models as a new estimate is more robust. Hence in non-maneuvering motion, AREV( $m = 1$ ) and in the case of maneuvering motion, AREV( $m = 2$ ) indicate better performances respectively. However, since in different cases of non-maneuvering and maneuvering motions, different models of AREV ( $m = 1$  or  $m = 2$ ) are suitable, therefore a switching logic is needed to use the proper model for any cases of motion.

**Table 1** The proposed AREV model

<b>Initialization</b> (for $j = 1, 2, \dots, n$ )
$n$ : the number of parallel models in IMM.
$M$ : the number of AR model coefficients.
$N$ : the highest degree of polynomial used to approximate the range.
<b>Attention:</b> $M > N + 1$ .
$m$ : the most probable paths used in the EV algorithm.
<b>Attention:</b> $1 \leq m < n$ ( $m = 1$ in EV1 and $m = 2$ in EV2).
$\hat{x}_0^j$ and $P_0^j$ are initial estimate and initial covariance;
The process noise covariance: $Q = q_r T I$ ( $T$ is sampling time, and $I$ is identity matrix;
Initial model probability: $\mu_0(j) = 1/n$ ;
Model transition probability: $P = \{p_{ij}\}$ , $i, j = 1, 2, \dots, n$ .
<b>Step 1</b> Calculation of transition matrix (for $j = 1, 2, \dots, n$ )
In every time step, the convex quadratic programming problem (25) should be solved, and the transition matrix $A_{k k-1}^j$ is obtained by (5) in every AR model.
<b>Attention:</b> It is the special part of the proposed method in which the AR idea is incorporated.
<b>Step 2</b> Model-conditioned re-initialization (for $j = 1, 2, \dots, n$ & $s = 1, 2, \dots, m$ )
$\mu_{k-1}(l_{sj} j) = \frac{\max_{1 \leq i \leq n}^{(s)} \{p_{ij} \mu_{k-1}(i)\}}{\sum_{s=1}^m \max_{1 \leq i \leq n} \{p_{ij} \mu_{k-1}(i)\}}$
$l_{sj} = \arg\{\max_{1 \leq i \leq n}^{(s)} \{p_{ij} \mu_{k-1}(i)\}\}$
Mixing estimate: $\hat{x}_{k-1}^{0j} = \sum_{s=1}^m \hat{x}_{k-1}^{l_{sj}} \mu_{k-1}(l_{sj} j)$
Mixing covariance: $P_{k-1}^{0j} = \sum_{s=1}^m \mu_{k-1}(l_{sj} j) \{P_{k-1}^{l_{sj}} + [\hat{x}_{k-1}^{l_{sj}} - \hat{x}_{k-1}^{0j}] \times [\hat{x}_{k-1}^{l_{sj}} - \hat{x}_{k-1}^{0j}]^T\}$ .
<b>Step 3</b> Model-conditioned filtering (for $j = 1, 2, \dots, n$ )
Predicted state: $\hat{x}_{k k-1}^j = A_{k k-1}^j \hat{x}_{k-1 k-1}^{0j}$
Predicted covariance: $P_{k k-1}^j = A_{k k-1}^j P_{k-1 k-1}^{0j} (A_{k k-1}^j)^T + Q^j$
Filter gain: $K_k^j = P_{k k-1}^j (H_k^j)^T (S_k^j)^{-1}$
$S_k^j = H_k^j P_{k k-1}^j (H_k^j)^T + R_k^j$
Updated state: $\hat{x}_{k k}^j = \hat{x}_{k k-1}^j + K_k^j d_k^j$ , where $d_k^j = z_j - H_k^j \hat{x}_{k k-1}^j$
Updated covariance: $P_{k k}^j = [I^j - K_k^j H_k^j] P_{k k-1}^j$ .

Continued

**Step 4** Model probability update (for  $j = 1, 2, \dots, n$ )

$$\text{Model likelihood: } \Lambda_k^j = N(\mathbf{d}_k^j; 0, \mathbf{S}_k^j)$$

$$\text{Model probability: } \mu_k(j) = \frac{\Lambda_k \sum_{s=1}^m \max_{1 \leq i \leq n} \{p_{ij} \mu_{k-1}(i)\}}{\sum_{j=1}^n \Lambda_k(j) \sum_{s=1}^m \max_{1 \leq i \leq n} \{p_{ij} \mu_{k-1}(i)\}}$$

**Step 5** Calculation of  $m$  largest model probabilities (for  $r = 1, 2, \dots, m$ )

$$\tilde{\mu}_k(\tilde{l}_r) = \frac{\max_{1 \leq i \leq n} \{\mu_k(j)\}}{\sum_{r=1}^m \max_{1 \leq i \leq n} \{\mu_{k-1}(j)\}'}$$

$$\tilde{l}_r = \arg\{\max_{1 \leq i \leq n} \{\mu_k(j)\}'\}.$$

**Step 6** Estimate fusion

$$\text{Overall estimate: } \hat{\mathbf{x}}_k = \sum_{r=1}^m \tilde{\mu}_k(\tilde{l}_r) \hat{\mathbf{x}}_k^{\tilde{l}_r}.$$

#### 4. Switched IMM-AREV based tracking algorithm

In previous sections, it is shown since the selection of the state transition matrix in the AREV model is in dynamic form, the noise effect is reduced. In addition, for improvement of estimating and tracking, the state vector is in the manner that more previous information can be used to make the state value in time  $KT$ . Furthermore, in the AREV method only  $m$  most probable paths selected to be used in estimating and tracking against the traditional model that all possible paths are used. Also, according to the results of [27], different values for  $m$  lead to different performances in estimation. For non-maneuvering and constant velocity motions,  $m = 1$  makes better performance than  $m = 2$  but in maneuvering cases,  $m = 2$  is better to be used for acceptable performance. Now, because of the positive properties of AREV ( $m = 1$  &  $m = 2$ ), it comes to our mind that we can incorporate AREV to IMM and propose a switched IMM-AREV called S-AREV model for two cases of non-maneuvering and maneuvering motions so that for CV motions, AREV1 and for maneuvering cases, AREV2 will be used and switching happens between these two models according to the chi-square significance test model. Therefore, in this section we introduce the proposed model S-AREV. In the following subsections, the switching scheme from AREV1 to AREV2 and vice versa and changing the states from one case to other will be explained.

##### 4.1 Constant velocity tracking and maneuver onset time detection

As mentioned above, AREV1 makes a suitable model to have better performance in CV tracking and estimation but not in maneuvering motions, hence it means reasonable

criteria are necessary to be used for an important decision that if the maneuver occurs or not. This maneuver detector is chi-square significance test based. Since the maneuver occurrence leads to large innovations for AREV1, a fading memory average of innovations is utilized as the test statistic for maneuver detection [1,21,22,27]

$$\varepsilon_k = \phi(s) \sum_{i=k-s+1}^k \alpha^{s-i} \mathbf{v}_i^T \mathbf{S}_i^{-1} \mathbf{v}_i \quad (43)$$

where  $\phi(s) = (1 - \alpha)/(1 - \alpha^{s+1})$ , with  $0 < \alpha < 1$  as the forgetting factor,  $\mathbf{v}_i = \mathbf{z}_i - \mathbf{H} \mathbf{x}_{i|i-1}$  denotes the innovations from the quiescent model-based Kalman filtering of AREV1 at time  $i$  and  $\mathbf{S}_i$  stands for the corresponding covariance matrix which is given in (38c). Under the Gaussian assumption,  $\mathbf{v}_i^T \mathbf{S}_i^{-1} \mathbf{v}_i \sim \chi_{n_m}^2$  holds where  $n_m$  is the dimension of the measurement. The test statistics  $\varepsilon_k$  as a weighted sum of i.i.d Gaussian variables is not chi-square distributed but moment matching can change it approximately as a scaled version of a chi-square variable [1], that is

$$\varepsilon_k \sim \frac{1}{1 + \alpha_1} \chi_{s, n_z}^2 \quad (44)$$

where  $n_z = n_m(1 + \alpha)/(1 - \alpha)$ . However, the important and notable point is that what the relation between maneuver onset time detection and chi-square distribution is. As a response we can say that chi-square distribution test is a tool for evaluating occurrence probability for a special hypothesis. For example, if  $X_n^2(\lambda) < \varepsilon_0$ , it means that the occurrence probability for the special hypothesis is  $1 - \lambda$ , in other words, the confidence level for that hypothesis is  $1 - \lambda$ . According to (44), our hypothesis is the maneuver occurrence and we can consider  $1/(1 + \alpha_1) X_{s, n_z}^2 < \varepsilon_0$  ( $\lambda = 0.01$  or  $\lambda = 0.05$ ). It means with this value of  $\varepsilon_0$ , maneuver occurs with probability of 0.95 to 0.99. Therefore if according to (44),  $\varepsilon_k$  increases and gets more than  $\varepsilon_0$ , then the occurrence of a maneuver is accepted and we will have

$$\varepsilon_k > \frac{1}{1 + \alpha_1} X_{s, n_z}^2(\lambda) \quad (45)$$

and with probability 0.95 to 0.99 we can be sure that maneuver has occurred. As mentioned before,  $1 - \lambda$  as the level of confidence should be set quite high (95% or 99%). Now, there is a question that what the reasonable window length is. If the window length  $s$  is too large, then the modification of state estimates from AREV1 to AREV2 will take longer time so that real-time tracking requirements may not be satisfied and if the window length is too small, then the estimated maneuver onset time may not be reliable. For a fast and reasonable response to the maneuver



onset time, a relatively small length in terms of the forgetting factor  $\alpha_1$  is selected as (46) where  $\text{ceil}()$  means the nearest integer greater than or equal to the value in the parenthesis [2].

$$s_1 = \text{ceil} \left( \frac{1}{1 + \alpha_1} \right),$$

$$\arg \left\{ \min_{1 \leq i \leq L} (\mu_j^i) \right\} = 1, \quad j = k - s_1 + 1, \dots, k \quad (46)$$

where the sign '1' denotes the first AREV submodel.

#### 4.2 Modification of state estimates

Suppose that a maneuver is detected at time  $k$ , then the maneuver onset time is estimated as  $\hat{k} = k - s_1 - 1$  and switching from AREV1 to AREV2 should be activated at time  $\hat{k}$ . In addition, updated state estimates and covariance matrices of the AREV2 are initialized as  $\mathbf{x}_{k|\hat{k}}^j = \mathbf{x}_{\hat{k}|\hat{k}}^j$  and  $\mathbf{P}_{k|\hat{k}}^j = \mathbf{P}_{\hat{k}|\hat{k}}^j$  respectively for  $j = 1, 2, \dots, n$ .  $n$  is the number of submodels in both cases of AREV1 and AREV2,  $\mathbf{x}_{\hat{k}|\hat{k}}$  and  $\mathbf{P}_{\hat{k}|\hat{k}}$  are the combined state estimate and the state error covariance matrix at time  $\hat{k}$  from the AREV1. Also, since each model is equally important, model probabilities of the AREV2 are  $\mu_{\hat{k}}(j) = 1/n$  for  $j = 1, 2, \dots, n$ . State estimates from time  $\hat{k}$  to  $k$  which are calculated by AREV1 should be regenerated in the case of AREV2.

#### 4.3 Maneuver tracking and detection of maneuver termination

The AREV2 algorithm is proposed for maneuver tracking and the detection of maneuver termination is undertaken simultaneously. For detection of maneuver termination that makes large innovations in AREV2, a fading memory average of innovations as the test statistic can be used over a sliding window. The function of this fading memory is the same as the case of maneuver onset time detection given in (43). Also innovation is made by subtracting the measurement and the predicted state of the AREV2, and for reduction of false alarms of maneuver termination, a wider window length will be chosen [9] as follows:

$$s_2 = \text{ceil} \left[ \frac{\ln(1 - \alpha_2) - \ln n_m}{\ln \alpha_2} - 1 \right],$$

$$\arg \left\{ \min_{1 \leq i \leq L} (\mu_j^i) \right\} = 1, \quad j = k - s_2 + 1, \dots, k. \quad (47)$$

Therefore, if  $s = s_2$ , then  $\varepsilon_k \sim (1/(1 + \alpha_2))\mathbf{X}_{s_2 n_2}^2$ . In addition to the test statistic, termination of maneuver is accepted if the effective probability of the first AREV model ( $N = 1$ ) in three parallel models is the largest for  $s_2$  sampling intervals and sign '1' denotes the first AREV model ( $N = 1$ ). In this case when the maneuver termination is declared, the algorithm switches to AREV1 and initial values of the estimates and covariance matrices are the val-

ues in the first step time of the maneuver termination window. The switching scheme between these two algorithms is shown in Fig. 1.

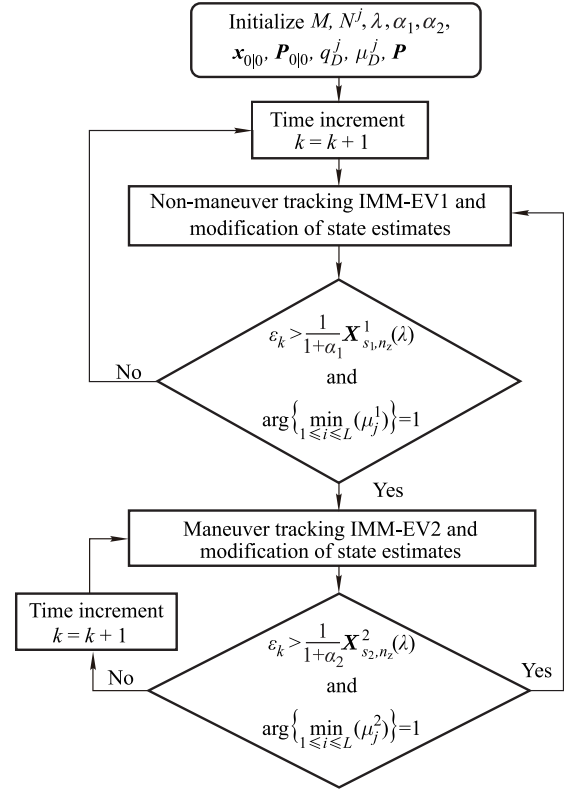


Fig. 1 Flowchart of proposed switching algorithm

### 5. Application of proposed model in target tracking problem

In this section, the proposed model S-AREV is implemented in a tracking example and its performance compared to three older algorithms, IMM-CA, IMM-AR and IMM-EV2. As we know, in the EV algorithm the parameter  $m$  denotes the most probable paths and should be selected  $1 \leq m < n$ . In this example, three parallel models are used ( $n = 3$ ) and the value of  $m$  is 2 in older model IMM-EV2. At the end of the example it is shown that the proposed algorithm is superior to the others. To obtain a fair comparison of simulations, we implement the same computer programs for similar algorithmic steps shared by algorithms (e.g., for a KF bank shared by all the algorithms compared to each other). Meanwhile, the number of Monte Carlo simulations is  $N = 100$ . The performance comparison for different tracking algorithms is according to the root-mean-square (RMS) position and velocity errors. The position and velocity RMS errors in time  $k$  and the  $i$ th run are calculated respectively as follows:

$$\delta_p^i(k) = \sqrt{(x^i(k) - \hat{x}^i(k))^2 + (y^i(k) - \hat{y}^i(k))^2} \quad (48)$$

$$\delta_v^i(k) = \sqrt{(\dot{x}^i(k) - \dot{\hat{x}}^i(k))^2 + (\dot{y}^i(k) - \dot{\hat{y}}^i(k))^2}. \quad (49)$$

Therefore, the position and velocity RMS errors in time  $k$  and for  $N$  runs of Monte Carlo simulations are calculated respectively as follows:

$$\delta_p(k) = \sqrt{\left(\frac{1}{N}\right) \sum_{i=1}^N (\delta_p^i(k))^2} \quad (50)$$

$$\delta_v(k) = \sqrt{\left(\frac{1}{N}\right) \sum_{i=1}^N (\delta_v^i(k))^2}. \quad (51)$$

Based on statistical significance analysis [21], the performance comparison among different algorithms can be done. At first we need some notations and definitions. The position RMS errors for IMM-CA, IMM-AR and IMM-EV2 in the  $i$ th run are denoted as  $\delta_{p-OM}^i(j)$ . ‘OM:oldmethod’ is used instead of CA, AR and EV2 and  $\delta_{p-SAREV}^i(j)$  for S-AREV position RMS error. The mean of differences between the squares of sample  $x$ - $y$  position errors of S-AREV and other three algorithms from the  $i$ th run over an interval  $[t_i, t_f]$  is defined as

$$\Delta_{p-OM,SAREV}^i(t_i, t_f) = \frac{1}{t_f - t_i + 1} \sum_{j=t_i}^{t_f} [\delta_{p-OM}^i(j)^2 - \delta_{p-SAREV}^i(j)^2]. \quad (52)$$

Then, the mean of  $\Delta_{p-OM,SAREV}^i(j)$  from  $N$  runs is given by

$$\bar{\Delta}_{p-OM,SAREV}(t_i, t_f) = \frac{1}{N} \sum_{i=1}^N \Delta_{p-OM,SAREV}^i(t_i, t_f). \quad (53)$$

Furthermore, the standard deviation of  $\Delta_{p-OM,SAREV}^i$  from  $N$  runs are given by

$$\sigma_{\bar{\Delta}_{p-OM,SAREV}}(t_i, t_f) = \frac{1}{N} \sqrt{\sum_{i=1}^N [\Delta_{p-OM,SAREV}^i(t_i, t_f) - \bar{\Delta}_{p-OM,SAREV}(t_i, t_f)]^2}. \quad (54)$$

The test statistic formula in (55) [21,22] as a comparison formula is a useful criterion to show that if the S-IMMAREV algorithm is superior to other algorithms or not over the sampling interval  $[t_i, t_f]$ ,

$$CMP_{p-SAREV \text{ vs OM}} = \frac{\bar{\Delta}_{p-OM,SAREV}(t_i, t_f)}{\sigma_{\bar{\Delta}_{p-OM,SAREV}}(t_i, t_f)} \geq 1.65. \quad (55)$$

It means the significance level of decision is at least 95%. Likewise, we can define the above parameters for velocity estimation among the algorithms by replacing the subscript ‘ $p$ ’ with the subscript ‘ $v$ ’. In the same way, the

decision that the  $x$  or  $y$  velocity tracking performance of the S-AREV algorithm is superior to other algorithms is accepted if comparison formula is satisfied.

$$CMP_{v-SAREV \text{ vs OM}} = \frac{\bar{\Delta}_{v-OMSAREV}(t_i, t_f)}{\sigma_{\bar{\Delta}_{v-OMSAREV}}(t_i, t_f)} \geq 1.65 \quad (56)$$

## 5.1 Linear models simulation and comparison

In this section, the four mentioned linear algorithms are compared with each other. In the IMM-CA and IMM-EV2 algorithms, the traditional models of target state vector and transient state matrix are used but in IMM-AR and S-AREV algorithms, the AR model of target state vector and transient state matrix introduced in Section 3 should be used, thus

$$\mathbf{x}_{CA,EV2} = [x(t) \quad \dot{x}(t) \quad \ddot{x}(t)] \quad (57)$$

$$\mathbf{x}_{AR,SAREV} = [x(t) \quad x(t-1) \quad x(t-2) \quad x(t-3)]. \quad (58)$$

The initial target state vector at  $t = 0$  is  $[2 \ 000 \ 15 \text{ m/s} \ 0 \text{ m/s}^2 \ 45 \ 000 \text{ m} \ -40 \text{ m/s} \ 0 \text{ m/s}^2]$ . The target position is sampled at every 10 s. The measurement noise variance of the  $x$  and  $y$  axes position is  $R = 100 \text{ m}^2$ . During the 20th to 40th sampling periods, the target moves with relatively smooth accelerations in both dimensions ( $a_x = a_y = 0.25 \text{ m/s}^2$ ) and during the 60th to 80th sampling periods, the target moves with fast accelerations ( $a_x = 0.6 \text{ m/s}^2$ ,  $a_y = -0.6 \text{ m/s}^2$ ). In all algorithms, three submodels are used without control input, the first submodel is a CV model with process noise intensity  $q_1 = 0.01$  and the other two models are CA models with process noise intensities of  $q_2 = 0.1$  and  $q_3 = 1$  respectively. The reason of considering two CA models is that the first one with low noise variance can be used for low accelerated motions and the second with higher noise variance can be used for high accelerated motions. Since the difference between observed target  $\mathbf{Z}_k$  and predicted state  $\mathbf{x}_{k|k-1}$  increases in maneuver parts of motion, the noise variance will be increased too (in comparison with CV motion) and in this way, more maneuver will cause more noise variance. Therefore process noise covariance matrices for CA and EV algorithms are

$$\begin{cases} \mathbf{Q}_{k(1)}^{CA,EV2} = q_1 T \begin{bmatrix} 1 & 1/T \\ 1/T & 2/T^2 \end{bmatrix} \\ \mathbf{Q}_{k(2)}^{CA,EV2} = q_2 T \begin{bmatrix} 1 & 1/T & 1/T^2 \\ 1/T & 2/T^2 & 3/T^3 \\ 1/T^2 & 3/T^3 & 6/T^4 \end{bmatrix} \\ \mathbf{Q}_{k(3)}^{CA,EV2} = q_3 T \begin{bmatrix} 1 & 1/T & 1/T^2 \\ 1/T & 2/T^2 & 3/T^3 \\ 1/T^2 & 3/T^3 & 6/T^4 \end{bmatrix} \end{cases} \quad (59)$$

For IMM-AR and S-AREV algorithms, three submodels are considered too that the first one with  $N = 1$  is for CV motion and two other models with  $N = 2$  are for CA motions. The number of coefficients in all three submodels is  $M = 4$ . For consistency with the process noise intensity in (59), the process noise covariance matrices of AR models are

$$\begin{cases} Q_{k(1)}^{\text{AR,SAREV}} = q_1 T l \\ Q_{k(2)}^{\text{AR,SAREV}} = q_2 T l \\ Q_{k(3)}^{\text{AR,SAREV}} = q_3 T l \end{cases} \quad (60)$$

The transition probability matrix of four algorithms is as follows:

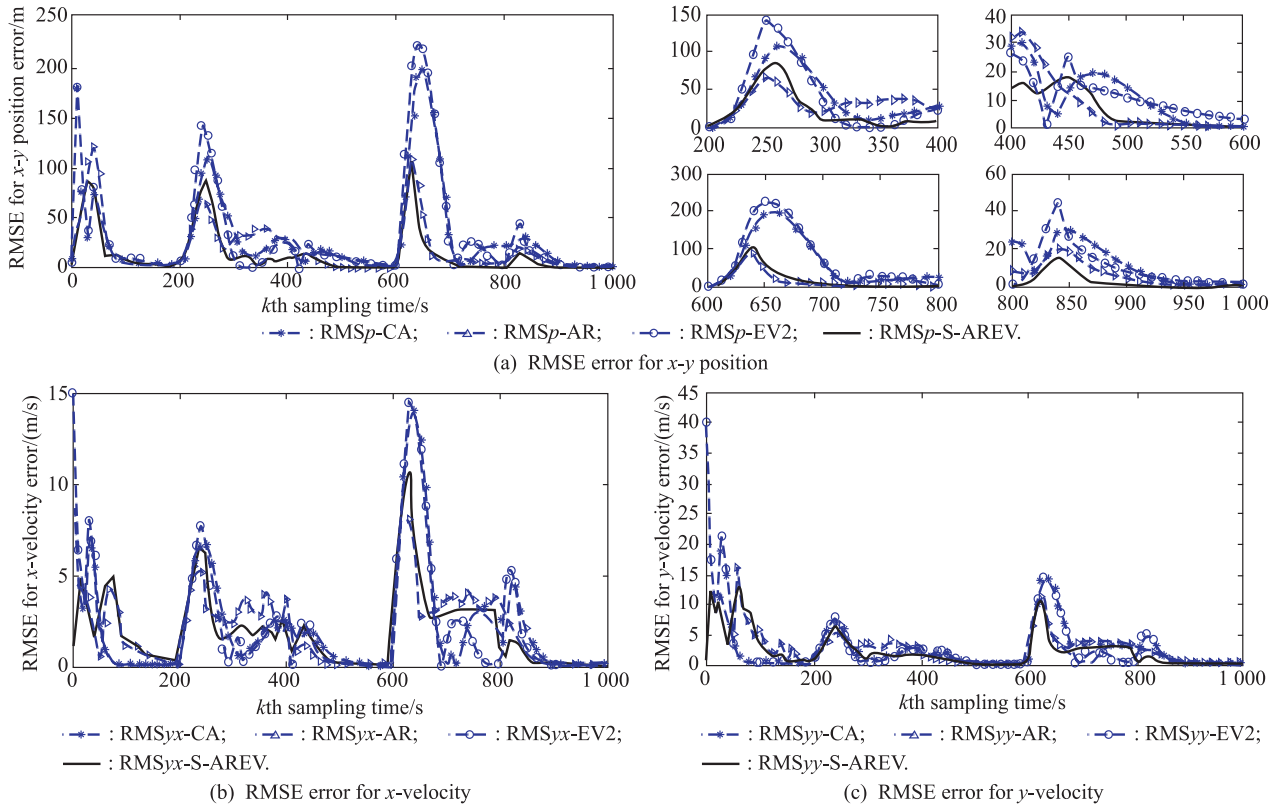
$$P_{\text{tr}} = \begin{bmatrix} 0.7 & 0.3 & 0 \\ 0.3 & 0.4 & 0.3 \\ 0 & 0.3 & 0.7 \end{bmatrix}. \quad (61)$$

The Monte Carlo simulations with 100 runs are carried out for a period of 1 000 s. Fig. 2, Fig. 3 and Table 2 show the results of the older and proposed performances clearly.

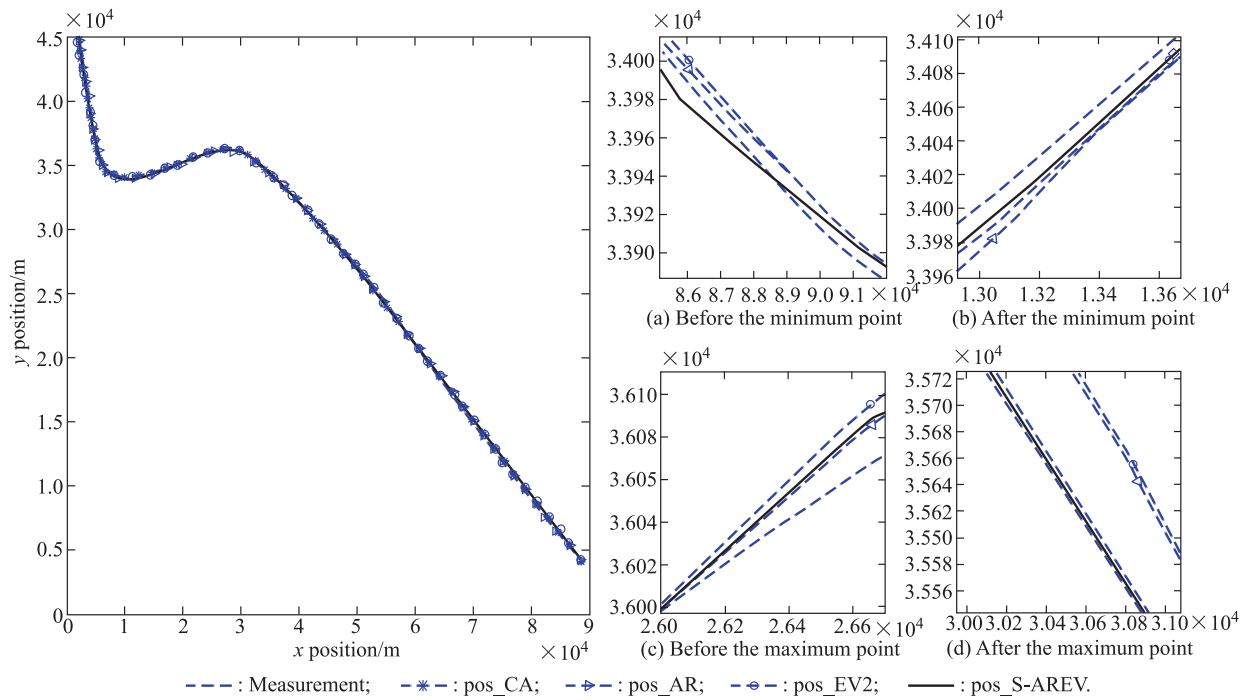
In Fig. 2(a), RMS position error comparison for three older algorithms and the new proposed one is illustrated and for better observation, Fig. 2(a) is divided to four maneuvering and non-maneuvering subplots in different intervals. Totally in all five parts, S-AREV as the proposed

method has better performance in position estimation compared to older methods specially in maneuvering parts of motion and it can be said that in 0 – 200 s, EV2 and CA have better performances than AR and EV2 is approximately the same as CA. In 200 – 400 s, S-AREV acts like AR till 300 s and acts better after that time to 400 s. AR and EV2 have approximately the same performances but EV2 is better than CA in this part time. In 400 – 600 s, EV2 is approximately the same as CA and AR is better than EV2 and CA specially after 500 s. In 600 – 800 s, S-AREV acts approximately the same as AR, but AR is better than both of EV2 and CA, and EV2 is a little better than CA, and finally in 800 – 1 000 s, AR is better than EV2 and CA, but EV2 has the same performance like CA.

In the same way, in Fig. 2(b) and Fig. 2(c), comparison of velocity estimations are illustrated. It can be seen that the results in both  $x$  and  $y$  directions are the same and totally in all the part times, S-AREV has better performance compared to other methods but for comparison among older methods, it can be said that in 0 – 200 s, S-AREV has the same performance like other methods, some parts better and some parts not better, AR is a little worse than S-AREV, and EV2 and CA have the same performance like each other.



**Fig. 2** RMS  $x$ - $y$  position and velocity errors yielded by IMM-CA, IMM-AR, IMM-EV( $m = 2$ ) and proposed algorithm S-AREV from 100 Monte Carlo runs



**Fig. 3** Target moving and predicted trajectories yielded by IMM-CA, IMM-AR, IMM-EV( $m = 2$ ) and proposed algorithm S-AREV from 100 Monte Carlo runs

**Table 2** Comparison of position and velocity estimations for proposed and older algorithms

Comparison of	Time	$CMP_{p-S-AREV \text{ vs CA}}$	$CMP_{p-S-AREV \text{ vs AR}}$	$CMP_{p-S-AREV \text{ vs EV2}}$	$CMP_{p-AR \text{ vs CA}}$	$CMP_{p-EV2 \text{ vs CA}}$
	interval/s					
position estimations for different algorithms according to (56)	0–200	5.423	7.558	8.165	0.978	1.235
	200–400	12.165	10.125	19.267	13.836	2.030 3
	400–600	11.433	2.158 7	9.346 9	2.899	1.469
	600–800	66.229	1.785	85.719	24.919	2.443
	800–1 000	23.154	3.458	35.746	19.743	1.584
Comparison of	Time	$CMP_{vx-S-AREV \text{ vs CA}}$	$CMP_{vx-S-AREV \text{ vs AR}}$	$CMP_{vx-S-AREV \text{ vs EV2}}$	$CMP_{vx-AR \text{ vs CA}}$	$CMP_{vx-EV2 \text{ vs CA}}$
	interval/s					
$x$ -velocity estimations for different algorithms according to (57)	0–200	1.068	1.584	1.021	0.877	1.005
	200–400	1.055	1.615	1.051	0.915	1.154
	400–600	5.401	1.159	5.491	2.651	1.166
	600–800	1.255	1.687	1.548	1.054	1.114
	800–1 000	9.762	8.746	9.154	7.119	7.256
Comparison of	Time	$CMP_{vy-S-AREV \text{ vs CA}}$	$CMP_{vy-S-AREV \text{ vs AR}}$	$CMP_{vy-S-AREV \text{ vs EV2}}$	$CMP_{vy-AR \text{ vs CA}}$	$CMP_{vy-EV2 \text{ vs CA}}$
	interval/s					
$y$ -velocity estimations for different algorithms according to (57)	0–200	1.178	1.072	0.987	0.824	0.912
	200–400	1.101	1.856	1.142	0.826	1.151
	400–600	6.455	1.106	9.852	2.954	1.002
	600–800	1.167	2.065	1.507	1.007	1.523
	800–1 000	10.875	8.166	8.462	8.463	5.451

In 200 – 400 s, all the methods approximately act like each other, but AR is worse than them. In 400 – 600 s, S-AREV and AR are a little better than CA and EV2, and CA and EV2 act like each other approximately. In 600 – 800 s, S-AREV and AR are better in initial times but after 700 s, EV2 and CA have better performance, and finally in 800 – 1 000 s, S-AREV is better than all and AR

is better than CA and EV2.

## 5.2 S-AREV vs nonlinear extended Kalman filter (EKF)

Four mentioned models compared to each other are linear. Now, as the last comparison, we want to check the performance of the proposed IMM-AREV to a nonlinear

model like interacting multiple model extended Kalman filter (IMM\_EKF) and finally in Fig. 4, Fig. 5 and Table 3, we will see the performance of AREV vs EKF. In fact, like the older linear models, we consider three nonlinear models parallel to each or in other words, three EKFs are paralleled with each other to estimate different parts of motion like CV, CA with low acceleration and CA with high acceleration. Then like AREV we consider three values for covariance  $q_1$ ,  $q_2$  and  $q_3$ . Also the state and measurement equations are described by the following non-linear discrete-time model [28,29].

$$\begin{cases} \mathbf{X}(k+1) = \mathbf{\Psi}(\mathbf{X}(k)) + \mathbf{G}(k)\boldsymbol{\omega}(k) \\ \mathbf{Z}(k) = \mathbf{H}(k)\mathbf{X}(k) + \mathbf{v}(k) \end{cases} \quad (62)$$

where

$$\mathbf{\Psi}(\mathbf{X}(k)) = \mathbf{F}(k)\mathbf{X}(k) + \mathbf{G}(k)(\mathbf{X}(k)). \quad (63)$$

The state vector is the same as (33) and  $\mathbf{F}(k)$ ,  $\mathbf{G}(k)$ ,  $\mathbf{H}(k)$  and  $\mathbf{C}(k)$  are defined as follows:

$$\begin{aligned} \mathbf{F} &= \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} \frac{T^2}{2} & T & 0 & 0 \\ 0 & 0 & \frac{T^2}{2} & T \end{bmatrix}^T, \\ \mathbf{H} &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} \frac{T^2}{2} & T & 0 & 0 \\ 0 & 0 & \frac{T^2}{2} & T \end{bmatrix}^T. \end{aligned} \quad (64)$$

Like linear models,  $T$  is the sampling time between successive measurements. The drag is a force directed in opposition to the target speed and with an intensity equal to  $(1/2)((g/\beta))\rho v^2$  where  $g$  is the gravity acceleration,  $\beta$  is the ballistic coefficient and equal to  $40\,000\text{ kg}\cdot\text{m}^{-1}\text{s}^{-2}$  and  $\rho$  is the air density (typically it is an exponentially decaying function of height),  $\rho = c_1 e^{-c_2 y}$  where  $c_1 = 1.227$  and  $c_2 = 1.093 \times 10^{-4}$  for  $y < 9\,144\text{ m}$  and  $c_1 = 1.754$ ,  $c_2 = 1.491 \times 10^{-4}$  for  $y > 9\,144\text{ m}$  and  $v$  is the module of target velocity. In terms of state vector components, the drag is

$$\begin{aligned} \mathbf{f}(\mathbf{X}(k)) &= -\frac{1}{2}\frac{g}{\beta}\rho(y(k))(\dot{x}^2 + \dot{y}^2) \cdot \\ &\quad \begin{bmatrix} \cos\left(\arctan\left(\frac{y(x)}{x(k)}\right)\right) \\ \sin\left(\arctan\left(\frac{y(x)}{x(k)}\right)\right) \end{bmatrix} \end{aligned} \quad (65)$$

$$\begin{cases} \cos\left(\arctan\left(\frac{y(x)}{x(k)}\right)\right) = \frac{x}{\sqrt{x^2 + y^2}} \\ \sin\left(\arctan\left(\frac{y(x)}{x(k)}\right)\right) = \frac{y}{\sqrt{x^2 + y^2}} \end{cases}. \quad (66)$$

Then

$$\mathbf{f}(\mathbf{X}(k)) = -\frac{1}{2}\frac{g}{\beta}\rho(y(k))\sqrt{\dot{x}^2 + \dot{y}^2(k)} \begin{bmatrix} \dot{x}^2(k) \\ \dot{y}^2(k) \end{bmatrix}. \quad (67)$$

At first we have to calculate the Jacobian matrix. Using the Taylor series expansion around  $\hat{\mathbf{X}}(k|k)$

$$\mathbf{F}_j(k) = [\boldsymbol{\Delta}_{\mathbf{X}(k)}(\mathbf{f}^T(\mathbf{X}(k)))] \quad (68)$$

$$F[1, 1] = 0, \quad F[2, 1] = 0 \quad (69a)$$

$$F[1, 2] = -0.5\frac{g}{\beta}\rho(y(k))\frac{2\dot{x}^2(k) + \dot{y}^2(k)}{\sqrt{\dot{x}^2(k) + \dot{y}^2(k)}} \quad (69b)$$

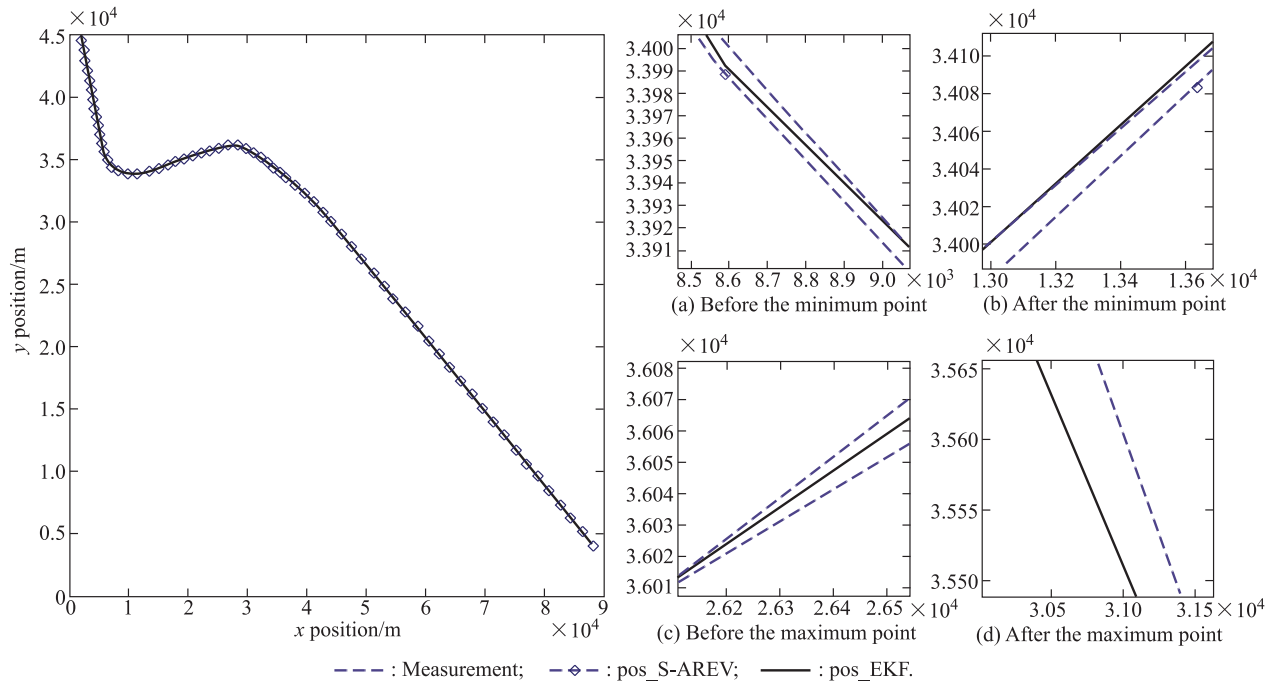
$$F[1, 3] = 0.5\frac{g}{\beta}c_2\rho(y(k))\dot{x}(k)\sqrt{\dot{x}^2(k) + \dot{y}^2(k)} \quad (69c)$$

$$F[1, 4] = F[2, 2] = 0.5\frac{g}{\beta}\rho(y(k))\frac{\dot{x}(k)\dot{y}^2(k)}{\sqrt{\dot{x}^2(k) + \dot{y}^2(k)}} \quad (69d)$$

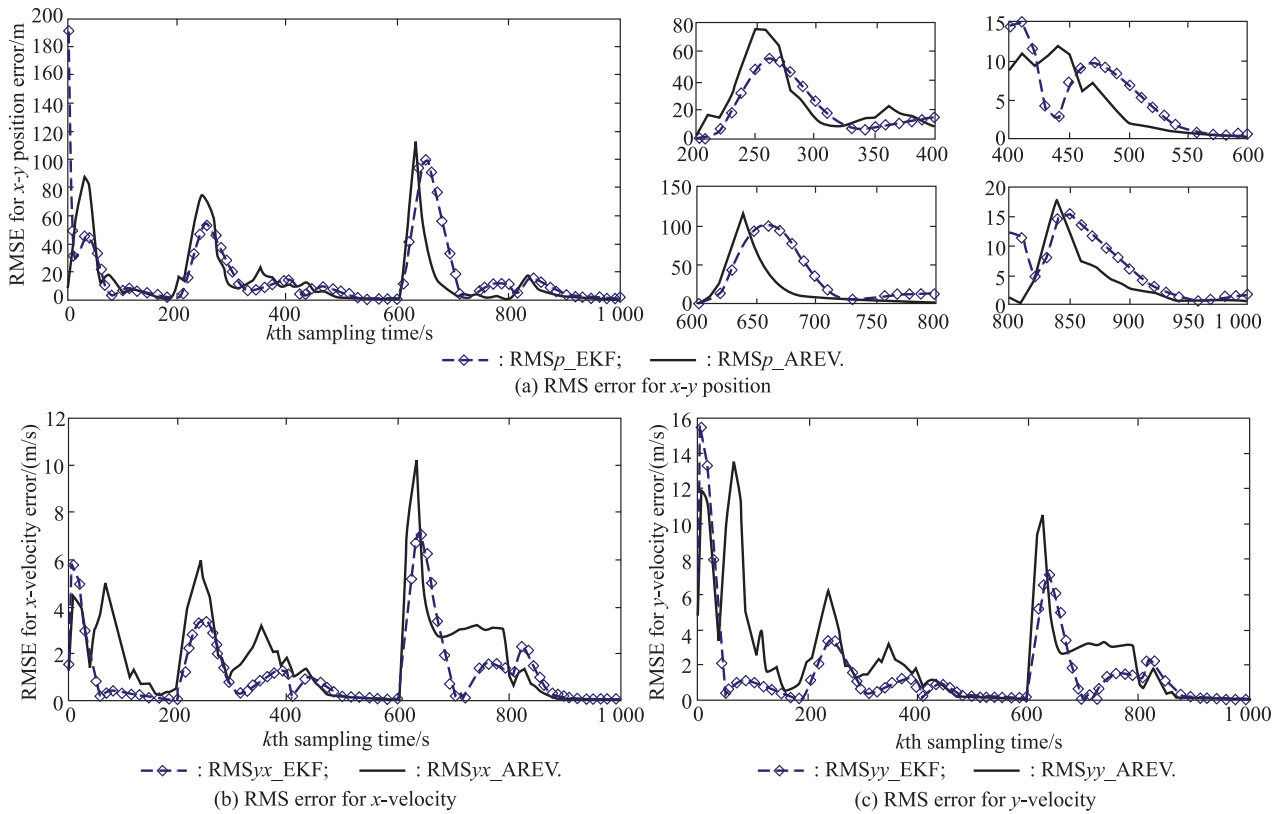
$$F[2, 3] = 0.5\frac{g}{\beta}c_2\rho(y(k))\dot{y}(k)\sqrt{\dot{x}^2(k) + \dot{y}^2(k)} \quad (69e)$$

$$F[2, 4] = -0.5\frac{g}{\beta}\rho(y(k))\frac{\dot{x}^2(k) + 2\dot{y}^2(k)}{\sqrt{\dot{x}^2(k) + \dot{y}^2(k)}}. \quad (69f)$$

The process noise covariance matrix for the EKF model is the same as the CA and EV2 model, it means  $\mathbf{Q}^{\text{EKF}} = \mathbf{Q}^{\text{CA, EV2}}$  for  $q_1$ ,  $q_2$  and  $q_3$  and also measurement noise covariance is the same as linear models. The comparison results between the proposed AREV model and nonlinear EKF are depicted in Fig. 4 and Fig. 5 and also Table 3. Figs. 4(a) to 4(d) are comparisons of tracking models before minimum point, after minimum point, before maximum point and after maximum point. As seen, in all times the performance of position estimation in AREV is better than EKF but for velocity estimation, it seems EKF has better performance than AREV. However, totally we can consider some disadvantages for EKF, unlike its linear counterpart, the EKF in general is not an optimal estimator (of course it is optimal if the measurement and the state transition model are both linear, as in that case the EKF is identical to the regular one). In addition, if the initial estimate of the state is wrong, or if the process is modeled incorrectly, the filter may quickly diverge, owing to its linearization. Another problem with the EKF is that the estimated covariance matrix tends to underestimate the true covariance matrix and therefore risks becoming inconsistent in the statistical sense without the addition of “stabilizing noise”.



**Fig. 4** Target moving and predicted trajectories yielded by nonlinear model EKF and proposed algorithm S-AREV from 100 Monte Carlo runs



**Fig. 5** RMS  $x$ - $y$  position and velocity errors yielded by nonlinear model EKF and proposed algorithm S-AREV from 100 Monte Carlo runs

**Table 3** Comparison of position and velocity estimations for proposed AREV and nonlinear EKF

Time interval/s	$CMP_{p-S-AREV}$ vs EKF	$CMP_{vx-S-AREV}$ vs EKF	$CMP_{vy-S-AREV}$ vs EKF
0–200	36.32	–5.760	–7.180
200–400	43.06	1.013	0.963
400–600	52.20	12.02	19.490
600–800	67.16	0.879	0.388
800–1000	85.18	7.020	3.260

## 6. Conclusions

In this paper, a new S-AREV is presented. The AREV algorithm is incorporated to IMM framework and relatively changes its structure. This incorporation has three advantages. First, change of the transition matrix from static to dynamic form leads to noise effect reduction in addition to satisfying target motion constraint. Second, reformation of the state vector can use more information from older target positions, which makes the prediction and estimation more reliable. And third, one of the traditional IMM problems is the load of unnecessary computation that happens in these models, in AREV instead of all history paths, and only most probable ones are used to prevent the overmodeling problems. In AREV1, only the first most probable path is used and this model is so useful for non-maneuvering motions, but AREV2 is appropriate for maneuvering motions because not only the first path but two most probable paths are used to increase the performance of tracking and estimation. In IMM-EKF, despite of its better performance in velocity tracking, because of its divergence for wrong initial estimate or incorrect process modeling and also other problems that we encounter in EKF, we prefer to use AREV rather than CA, AR, EV and even EKF. In future work we can apply AR models to nonlinear systems and consider multiplicative noises in system and also consider inputs into the systems and use simultaneously input estimation algorithm and AR models for nonlinear systems.

## References

- [1] LI X R, JILKOV V P. A survey of maneuvering target tracking. Part IV: decision-based methods. Proc. of SPIE Conference on Signal and Data Processing of Small Targets, 2002: 4728–4760.
- [2] BAR-SHALOM Y, BIRMIWAL K. Variable dimension filter for maneuvering target tracking. IEEE Trans. on Aerospace Electronics Systems, 1982, 18(5): 621–629.
- [3] ZHANG Z, WU Y, SUN W. Modeling and adaptive motion/force tracking for vertical wheel on rotating table. Journal of Systems Engineering and Electronics, 2015, 26(5): 1060–1069.
- [4] WANG S, DA X, LI M, et al. Adaptive back-tracking search optimization algorithm with pattern search for numerical optimization. Journal of Systems Engineering and Electronics, 2016, 27(2): 395–406.
- [5] YANG J, LI P, LI Z, et al. Multiple extended target tracking algorithm based on Gaussian surface matrix. Journal of Systems Engineering and Electronics, 2016, 27(2): 279–289.
- [6] KHALOOZADEH H, KARSAZ A. Modified input estimation technique for tracking maneuvering targets. IET Radar Sonar & Navigation, 2009, 3(1): 30–41.
- [7] MAZOR E, AVERBUCH A, BAR-SHALOM Y, et al. Interacting multiple model methods in target tracking: a survey. IEEE Trans. on Aerospace Electronics Systems, 1998, 34(1): 103–123.
- [8] FU X, JIA Y, DU J, et al. New interacting multiple model algorithms for the tracking of the maneuvering target. IET Control Theory Applications, 2010, 4(10): 2184–2194.
- [9] HO T J. A switched IMM-extended Viterbi estimator-based algorithm for maneuvering target tracking. Automatica, 2011, 47: 92–98.
- [10] FOO P H, NG G W. Combining the interacting multiple model method with particle filters for maneuvering target tracking. IET Radar Sonar & Navigation, 2011, 5(3): 234–255.
- [11] BOERS Y, DRIESSEN J N. Interacting multiple model particle filter. IEE Proceeding of Radar, Sonar and Navigation, 2003, 150(5): 344–349.
- [12] LANEUVILLE D, BAR-SHALOM Y. Maneuvering target tracking: a Gaussian mixture based IMM estimator. Proc. of IEEE/AIAA Aerospace Conference, 2012: 1–12.
- [13] LAN J, LI X R, JILKOV V P, et al. Second-order Markov chain based multiple-model algorithm for maneuvering target tracking. IEEE Trans. on Aerospace Electronics Systems, 2013, 49(1): 3–19.
- [14] LI X R. Multiple-model estimation with variable structure. Part II: model-set adaptation. IEEE Trans. on Automatic Control, 2000, 45(11): 2047–2060.
- [15] YU C H, CHOI J W. Interacting multiple model filter-based distributed target tracking algorithm in underwater wireless sensor networks. International Journal of Control, Automation and Systems, 2014, 12(3): 618–627.
- [16] LI X R, JILKOV V P. Survey of maneuvering target tracking. Part V: multiple-model methods. IEEE Trans. on Aerospace Electronics Systems, 2005, 41(4): 1255–1321.
- [17] JING L, VADAKKEPAT P. Interacting MCMC particle filter for tracking maneuvering target. Digital Signal Processing, 2010, 20: 561–574.
- [18] SINGIRESU S R. Engineering optimization: theory and practice. 4th ed. New York: Wiley, 2009.
- [19] LI X R, JILKOV V P. Survey of maneuvering target tracking. Part I: dynamic models. IEEE Trans. on Aerospace Electronics Systems, 2003, 39(4): 1333–1364.
- [20] SESHADRI N, SUNDBERG C E W. List Viterbi decoding algorithms with applications. IEEE Trans. on Communications, 1994, 42(2/3/4): 313–323.

- [21] JIN B, JIU B, SU T, et al. Switched Kalman filter-interacting multiple model algorithm based on optimal autoregressive model for maneuvering target tracking. *IET Radar, Sonar & Navigation*, 2014, 9(2): 199–209.
- [22] HO T J. A switched IMM-extended Viterbi estimator-based algorithm for maneuvering target tracking. *Automatica*, 2011, 47(1): 92–98.
- [23] VÄLIVIITA S, OVASKA S J, VAINIO O. Polynomial predictive filtering in control instrumentation: a review. *IEEE Trans. on Industrial Electronics*, 1999, 46(5): 876–888.
- [24] XU L, LI X R, DUAN Z, et al. Modelling and state estimation for dynamic systems with linear equality constraints. *IEEE Trans. on Signal Processing*, 2013, 61(11): 2927–2939.
- [25] MEYER C D. *Matrix analysis and applied linear algebra*. Philadelphia: SIAM, 2000.
- [26] BAR-SHALOM Y, LI X R. *Estimation and tracking, principle, techniques and software*. Boston: Artech House, 1993.
- [27] HO T J, CHEN B S. Novel extended Viterbi-based multiple-model algorithms for state estimation of discrete-time systems with Markov jump parameters. *IEEE Trans. on Signal Processing*, 2006, 54(2): 393–404.
- [28] HERAB H M, KHALOOZADEH H. Extended input estimation method for tracking non-linear maneuvering targets with multiplicative noises. *IET Radar, Sonar & Navigation*, 2016, 10(9): 1683–1690.
- [29] RAHMATI H, KHALOOZADEH H, AYATI M. Novel approach for nonlinear maneuvering target tracking based on input estimation. *Proc. of the International Conference on Mechanical and Aerospace Engineering*, 2012: 4415–4423

## Biographies



**HADAEGH Mahmoudreza** received his B.S. degree in electronics engineering from Shiraz University, Shiraz, Iran, in 1999, M.S. degree in control engineering from K. N. Toosi University of Technology, Tehran, in 2001. He is currently a Ph.D. student in control engineering with the Department of Electrical and Computer Engineering, Tehran Science and Research Branch, Islamic

Azad University, Tehran, Iran. Now he is an instructor in Islamic Azad University and his interests include target tracking for single and multiple targets and wireless sensor networks.

E-mail: mr\_hadaegh@yahoo.com



**KHALOOZADEH Hamid** received his B.S. degree in control engineering from Sharif University of Technology, Tehran, Iran, in 1990, M.S. degree in control engineering from K. N. Toosi University of Technology, Tehran, in 1993, and Ph.D. degree in control engineering from Tarbiat Modarres University, Tehran, in 1998. He is currently a professor with the Department of Systems and Control, Faculty of Electrical Engineering at the K. N. Toosi University of Technology.

He is the director of the Industrial Control Center of Excellence (ICCE), at K. N. Toosi University of Technology. His interests include stochastic estimation and control, system identification, optimal control, and time series analysis.

E-mail: h.khaloozadeh@kntu.ac.ir