IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 19, 2017, accepted April 9, 2017, date of publication April 28, 2017, date of current version June 7, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2697072

Kalman Filter With Dynamical Setting of Optimal Process Noise Covariance

GABRIEL F. BASSO, THULIO GUILHERME SILVA DE AMORIM, ALISSON V. BRITO, (Member, IEEE), AND TIAGO P. NASCIMENTO, (Member, IEEE)
LASER–Embedded Systems and Robotics Laboratory, Department of Computer Systems, Federal University of Paraiba, João Pessoa 58051-900, Brazil

Corresponding author: Gabriel F. Basso (gabriel.f.basso@gmail.com)

This work was supported in part by CAPES through a PNPD Scholarship and in part by CNPq through Edital Universal 2014.

ABSTRACT We propose a dynamical way to set the process error covariance matrix (**Q**) for a constant velocity (CV) model Kalman filter. We are able to achieve the best possible solution for the estimated state, in the sense of forecast error, while significantly reducing the convergence time at no significant computational cost. No assumptions regarding the statistical nature of the observed process are made and no prior knowledge of the system is required. To achieve this, we adopt a recently proposed performance index for the Kalman filter, we map the best **Q** for an ample range of model deviations (accelerations) and dynamically set the best possible **Q** for the CV filter by identifying the average acceleration of the measured signal online. We demonstrate our scheme ability by filtering simulated trajectories with low, medium, and high signal-to-noise ratios. We also track a real erratic target and compare our filter prediction with the best possible *a posteriori* CV filter.

INDEX TERMS Target motion prediction, Kalman filter (KF), target tracking, optimal filter.

I. INTRODUCTION

The Kalman Filter (KF) is arguably one of the most important state estimation techniques today, with a wide range of applications, such as target tracking [1]–[4], weather prediction [5], [6] and Neural Network training [7], [8]. Usually, the simple KF with second order state model alone, known as Constant Velocity (CV) model, is not accurate enough to effectively predict a target position, and many proposed tracker techniques rely on maneuvering detection and filter switching [9], [10], which effectively multiplies the complexity of the tracker by the number of filters, while also employing more complex models.

Possible methods to improve the Constant Velocity Kalman filter (CVKF) include optimally setting the filter parameter *a priori* or dynamically changing the parameters in an adaptive fashion. For the CVKF, the two free parameters are the measurement noise covariance (**R**) and the process error covariance (**Q**). The measurement noise is only dependent on the measuring apparatus and is usually known beforehand, whereas the process error covariance matrix represents a trade-off between estimated state noise/stability and model flexibility. An all zero **Q** will lead to minimal noise in the estimated state, but in the case of an ill suited choice of model, it will lead to a poor fit of observed *vs.* estimated state. In practice, the setting of **Q** values is done *a priori*, based on expectations.

Many techniques were proposed to overcome this limitation regarding the adaptive parameter setting of the KF, but these techniques rely on previous knowledge about the system, usually assumptions on the statistical nature of the process and/or measurement noise [11]–[14]. Some propose the inclusion of additional estimation blocks to the KF [15], while others are only valid for very specific application [16]. Also, many are computationally complex [17], which can be limiting when dealing with low computational capability, such as for low-power embedded systems. Nonetheless, a common point for almost all adaptive techniques is the computation of the optimal parameter inside the filter loop, which is usually the source of complexity. Another significant problem is the loose definition of optimality for the KF [18].

Another way to improve the CVKF is to optimally select the parameters *a priori*, but it has only casually been discussed [19], [20]. One interesting take on parameter selection, with a thoughtful discussion on optimal setting, is presented by [21], in which a performance index dependent on the process error matrix was defined with no assumptions about the process itself. Therefore, one could select an appropriate

parameter based on anticipations regarding the system and the desirable performance.

We propose a dynamical way to set an optimal $\mathbf{Q}$ for a Constant Velocity model with regard to the prediction root-mean-squared error (RMSE), without relying on prior knowledge of the system. We show that it is possible to achieve the best possible final estimated state and reduce the convergence time, while adding only a marginal computational cost. In comparison with the usual adaptive parameter setting techniques, we identify only a simple control quantity inside the filter loop and update the parameter $\mathbf{Q}$ from a previously mapped file. To our knowledge, no one has mapped the optimal Kalman parameters to an identifiable quantity and applied this to dynamically select the appropriate parameters based on measured quantities. Objectively, we show the following.

- Our proposed algorithm is more effective in predicting a target state than the best possible CVKF using an Discrete-time Near Constant Velocity process error.
- The technique here suggested employs a process error matrix $\mathbf{Q}$ derived without any assumptions regarding the statistical nature of the process noise, therefore being suitable to a wide range of problems.
- Our method is simple to understand, implement and run, adding an average increment of only 7% in computational time compared to a standard CVKF. This is possible because we map $\mathbf{Q}$ beforehand, and thus only need to read the correct values inside the filter loop.
- Our algorithm has only one parameter that must be set *a priori*, the measurement noise covariance. Hence, its performance depends only on an usually already known parameter, and no further *guesses* are needed.
- Our approach is able to track maneuvering targets, as well as erratic ones. That is, we are able to satisfactorily track targets with both abrupt and incremental changes in its dynamics.

This paper is structured as follows. In Section II, we discuss the KF Tracker; in the subsequent section, we present our proposed algorithm; section IV presents the simulation results and a real object tracking experiment. Finally, our work conclusions are shown in Section V.

II. Kalman FILTER TRACKER

This section introduces the Kalman Filter for target tracking, defines the model used in this paper and discusses the influence of the error parameters on the tracking performance.

A. Kalman FILTER ALGORITHM

The KF is used to estimate a state vector of the target's parameters, based on a dynamic or measured model, in the presence of noise. The typical model is:

$$\mathbf{x}_k = \mathbf{\Phi}\mathbf{x}_{k-1} + \mathbf{w}_k, \tag{1}$$

where $\mathbf{x}_k$ denotes the true target state vector at observation k, $\mathbf{\Phi}$ is the state transition matrix from observation k to observation $(k+1)$, and $\mathbf{w}_k$ is the process noise with covariance matrix $\mathbf{Q}$. The measurement of the target is modeled as:

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k, \tag{2}$$

where $\mathbf{z}_k$ represents the measurement vector, $\mathbf{H}$ is the measurement matrix, and $\mathbf{v}_k$ is the measurement noise with covariance matrix $\mathbf{R}$.

The KF tracker for the above model sequentially estimates the state vectors by forecasting the future measurement, comparing the the predicted value with the current one and then correcting the model and the estimated state by minimizing the prediction error. This is accomplished by the following equations, known as the Kalman Filter Equations:

$$\tilde{\mathbf{x}}_k = \mathbf{\Phi}\hat{\mathbf{x}}_{k-1} \tag{3}$$
$$\tilde{\mathbf{P}}_k = \mathbf{\Phi}\hat{\mathbf{P}}_{k-1}\mathbf{\Phi}^\dagger + \mathbf{Q} \tag{4}$$
$$\mathbf{K}_k = \tilde{\mathbf{P}}_k\mathbf{H}^\dagger\,(\mathbf{H}\tilde{\mathbf{P}}_k\mathbf{H}^\dagger + \mathbf{R})^{-1} \tag{5}$$
$$\hat{\mathbf{P}}_k = \tilde{\mathbf{P}}_k - \mathbf{K}_k\mathbf{H}\tilde{\mathbf{P}}_k \tag{6}$$
$$\hat{\mathbf{x}}_k = \tilde{\mathbf{x}}_k + \mathbf{K}_k\,(\mathbf{z}_k - \mathbf{H}\tilde{\mathbf{x}}_k) \tag{7}$$

Here P is the covariance matrix of errors, and K is the gain that minimizes the errors in the estimated state vector. The superscripts $\tilde{}$ and $\hat{}$ denote the forecasts and the estimates, respectively.

B. CONSTANT VELOCITY MODEL

In this work we use the most common second-order model, the Constant Velocity Model Kalman Filter (CVKF), with has the following estate vector and the transition matrices:

$$\mathbf{x} = (x \;\; v)^\dagger \tag{8}$$
$$\mathbf{\Phi} = \begin{pmatrix} 1 & \delta t \\ 0 & 1 \end{pmatrix} \tag{9}$$

where δt is the sampling interval, x denotes the position, and v the velocity of the target being tracked. This is the representation for a one-dimensional (1D) tracker, but applications to two or higher-dimensional problems are easily achieved by implementing one filter for each dimension.

C. ERROR PARAMETERS

The KF equations depend on the selection of the measurement and process covariance error matrices, which have to be defined a priori to best suit the modeled system.

This task is usually trivial for the measurement error, since normally one knows beforehand what is the measuring apparatus and how it behaves.

However, the same is not true of the process error. The process error covariance matrix ($\mathbf{Q}$) accounts for deviations in the observed system in relation to the selected model. A zero process error choice implies complete confidence in the match between observed and predefined model, so any deviation of the target behavior from the model selected will lead to catastrophic tracking error.

On the other hand, the choice of a comparatively large process error will lead the filter to accept big measured deviations as intrinsic to the process, thus reducing the effectiveness of

measurement noise reduction. It is important to notice that the process noise can accommodate both random perturbation in measurement and biased perturbations in transition, so $\mathbf{Q}$ represents a trade off between estimated state noise and tracking accuracy. Therefore, a good selection of $\mathbf{Q}$ is a *sine qua non* condition for effective noise filtering and efficient tracking.

D. USUAL Q MATRIX FOR CVKF

The most commonly used process noise is random acceleration, which in discrete time has the following covariance matrix

$$\mathbf{Q}_{DNCV} = \begin{pmatrix} \delta T^4/4 & \delta T^3/2 \\ \delta T^3/2 & \delta T^2 \end{pmatrix} \sigma_{ga}^2. \tag{10}$$

Here, σ_{ga}^2 is the variance of a white Gaussian acceleration. We refer to this model as a discrete-time nearly constant velocity (DNCV) and we adopt it in comparison with our technique due to the fact that not only it is widely used, but also because it produces the best performance over the alternatives when σ_{ga}^2 is correctly specified [21].

E. OPTIMAL Q MATRIX FOR CVKF

Recently, the problem of appropriately selecting the matrix $\mathbf{Q}$ was studied in detail [21], and the following Analytic Steady-State Performance Index was proposed to guide the choice of Q:

$$\mu = \frac{\sqrt{\sigma_p^2 + e_{fin}^2}}{\sigma_x} \tag{11}$$

where σ_x is the standard deviation of measurement errors; σ_p represents the prediction root-mean-squared error (RMSE) for the CVKF of a zero acceleration target, also known as *Smoothing Performance Index*; and e_{fin} represents the bias error for the CVKF of a target with a constant acceleration (a_c), also known as *Tracking Performance Index*.

In [21], it is shown that there is a direct link between $\mathbf{Q}$ and μ, and therefore one can select the elements of $\mathbf{Q}$ that minimize μ. Specifically, let $\mathbf{Q}$ be

$$\mathbf{Q} = \begin{pmatrix} q_1 & q_2 \\ q_2 & q_3 \end{pmatrix}, \tag{12}$$

then, we can write μ as

$$\mu = \sqrt{\frac{2\alpha^2 + 2\beta + \alpha\beta}{\alpha(4 - 2\alpha - \beta)} + \frac{a_c^2(\delta t)^4}{\sigma_x^2 \beta^2}}, \tag{13}$$

where

$$\alpha = 1 - D^2/16C \tag{14}$$
$$\beta = D/4 \tag{15}$$
$$D = C + D_1 - \sqrt{2(D_1^3 + D_2)} \tag{16}$$
$$D_1 = \sqrt{C(16 + 4A - 4B + C)} \tag{17}$$
$$D_2 = C(2A - 2B + C) \tag{18}$$
$$A = q_1/\sigma_x^2 \tag{19}$$
$$B = q_2\delta t/\sigma_x^2 \tag{20}$$
$$C = q_3\delta t^2/\sigma_x^2. \tag{21}$$

Setting the values in $\mathbf{Q}$ - (q_1, q_2, q_3) - that minimize μ, in the case of constant acceleration a_c, will lead to the best possible tracking with the CVKF in this case, which corresponds to the minimal prediction error. Although not evident in the above equations, one important property of the Q matrix selected this way is that this Q does not rely on any assumptions regarding process noise [21], and therefore is suitable for random (Gaussian or otherwise), pseudo-random or biased process perturbations.

III. OUR TRACKING METHOD

Notice that an important limitation from this proposed matrix is that an appropriate a_c must still be set *a priori*, or a sub-optimal solution will be generated. Our proposed method addresses this problem by dynamically identifying the target acceleration and imbuing the CVKF with the optimal $\mathbf{Q}$ for that particular instant. We call our technique the *Dynamical Process Noise Covariance Kalman Filter* - DQKF.

However, minimizing μ with respect to (q_1, q_2, q_3, a_c) is not computationally trivial and takes a few seconds - usually less than 10s - for each value of a_c (using an Intel(R) Core(TM) i5-3470 CPU @ 3.20GHz), which prevents an online minimization for most applications.

To solve this problem, we mapped the optimal $\mathbf{Q}$ for 100 different a_c, ranging from 0.01 m/s^2 to 100 m/s^2 in a logarithm scale, so each time the identified target a_c changed significantly, we just read $\mathbf{Q}_{opt}(a_c)$ from memory and introduced it to the CVKF. Details can be found on section III-A.

In order to avoid excessive intervention in the CVKF via constant update of $\mathbf{Q}$, which would decrease the overall performance, we used a fading memory average of the acceleration, measured by the innovations of the estimated velocity, in the form of

$$a_c^{(k)} = \gamma a_c^{(k-1)} + (1 - \gamma)(\hat{\mathbf{x}}_k(2) - \hat{\mathbf{x}}_{k-1}(2))/\delta t \tag{22}$$

where γ is a decaying factor. This way, the estimated acceleration smooths out some of the identified state noise.

In our experiments, we observed that $\gamma = 0.75$ produced a good balance between sensitivity to acceleration variation and noise smoothing and no significant loss in performance was identified by varying γ between 0.5 and 0.9, whereas values smaller than 0.5 produced sporadic excess noise due to the frequent switching of $\mathbf{Q}$, and values bigger than 0.9 generated a large tracking error. Fine-tunning γ is not fundamental for the proposed technique, therefore we fixed its value at 0.75. Although the initial value of a_c can be anywhere within the available interval, we found that the biggest available value ensured a somehow faster initial convergence.

The processing time of our technique implementation, summarized in Algorithm 1, was only 7% longer, on average, than that of a standard CVKF with a fixed and predefined $\mathbf{Q}$, with no further code optimization.

Algorithm 1 Algorithm for the Kalman Filter with dynamically switched **Q** - DQKF.

Initialization
Given the measurement covariance error **R**;
Load the mapped $\mathbf{Q}_{opt}$;
Adjust the values of $\mathbf{Q}_{opt}$ with the selected **R** and δt;
Initial state: $\mathbf{x_0} = (0 \;\; 0)^{\dagger}$;
Initial estimated acceleration: $a_c = 100$;
Initial process covariance matrix: $\mathbf{Q}_{opt}(a_c)$;
Initial covariance matrix $\mathbf{P_0} = \mathbf{I} \cdot \mathbf{R}$;
for $k = 1, 2, \ldots$ **do**
  measure: $\mathbf{z}_k$
  apply filter: $[\hat{\mathbf{x}}_k, \hat{\mathbf{P}}_k] = CVKF[\mathbf{z}_k, \hat{\mathbf{x}}_{k-1}, \hat{\mathbf{P}}_{k-1}, \mathbf{Q}_{opt}]$
  find trailing acceleration:
  $a_c = 0.75\, a_c + 0.25(\hat{\mathbf{x}}_k(2) - \hat{\mathbf{x}}_{k-1}(2))/\delta t$
  update $\mathbf{Q}_{opt}$ to the closest mapped a_c available.
end

A. MAPPING THE OPTIMAL Q

Minimizing μ with respect to (q_1, q_2, q_3, a_c) can be done by many optimization algorithms, with the first choice usually being gradient descent. We opted for a modified Particle Swarm Optimization algorithm, as proposed in [22], for its variable step size, which can reach arbitrarily small steps, and consequently lock in the actual minimum.[1]

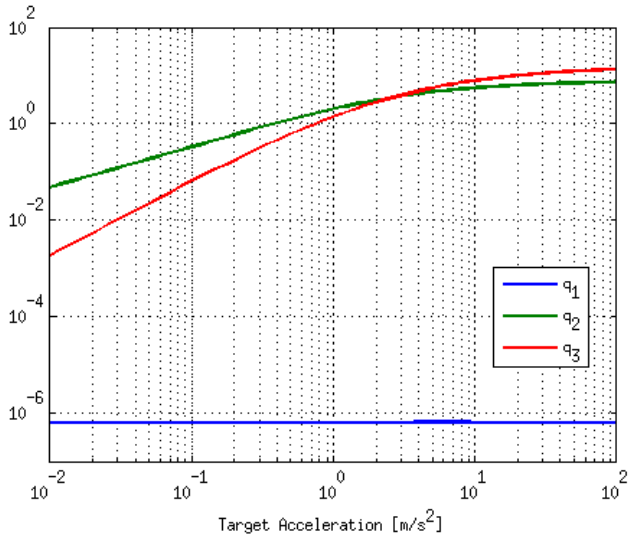

FIGURE 1. Mapped values of (q_1, q_2, q_3) for $a_c = [0.01 \ldots 100\ m/s^2]$, in the case of $\delta t = 1s$ and $\sigma_x = 1\ m$. Note the *log*×*log* scale.

Specifically, we minimized Eq. 13 with respect to the parameters (A, B, C), which is equivalent to minimizing (q_1, q_2, q_3) when $\delta t = 1$ and $\sigma_x = 1\, m$. For each a_c, we started the algorithm with 100 particles and ran 1000 iterations for 100 different initialization conditions. We performed the calculations for the optimal **Q** for 100 different a_c, ranging from 0.01 m/s^2 to 100 m/s^2 in a logarithm scale and saved the best values of (A, B, C) in a `csv` file. Saving it in `double` precision, we have only 2400 Bytes of data, which can be easily loaded to memory for fast access, and adjusted to the desired δt and σ_x.

[1] Within the numerical error.

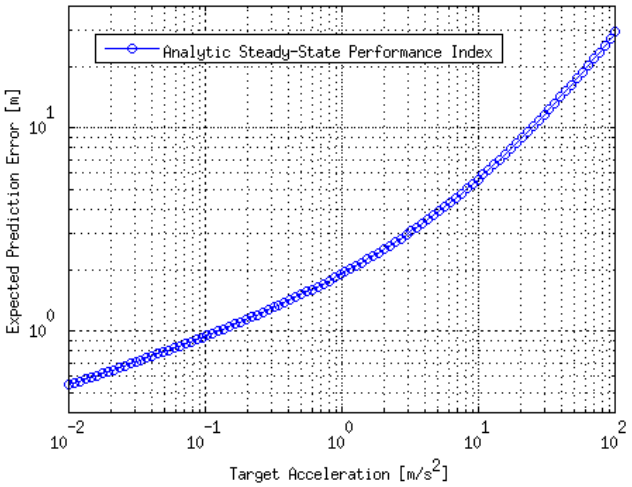

FIGURE 2. Expected values of prediction error of a target with constant acceleration, within the range $a_c = [0.01 \ldots 100\ m/s^2]$, in the case of $\delta t = 1s$ and $\sigma_x = 1\ m$.

Fig. 1 shows the optimal values found for (q_1, q_2, q_3), and Fig. 2 shows the expected prediction error of a target with constant acceleration, both in the case of $\delta t = 1s$ and $\sigma_x = 1\, m$ and for $a_c = [0.01 \ldots 100\ m/s^2]$.

IV. RESULTS

In order to evaluate the performance of the proposed technique, we performed a series of simulations and one experiment. First, we performed the tracking of a target with variable acceleration under different signal-to-noise ratios (SNR). Then, we investigated the DQKF response to sharp turns by tracking the trajectory of a real erratic object based on a log previously acquired with sudden velocity inversion and also under different SNR.

TABLE 1. Comparative results from CVKF and DQKF - Prediction RMSE.

SNR	σ_x	CVKF	DQKF	%
40 dB	10^3	2.5×10^3	5.2×10^2	21 %
20 dB	10^4	1.9×10^4	3.7×10^3	20 %
0 dB	10^5	6.3×10^4	2.7×10^4	42 %

TABLE 2. Comparative results from CVKF and DQKF - tracking bias.

SNR	σ_x	CVKF	DQKF	%
40 dB	10^3	3.1×10^3	1.4×10^2	4.8 %
20 dB	10^4	2.7×10^4	3.7×10^2	1.4 %
0 dB	10^5	1.1×10^5	9.9×10^2	0.9 %

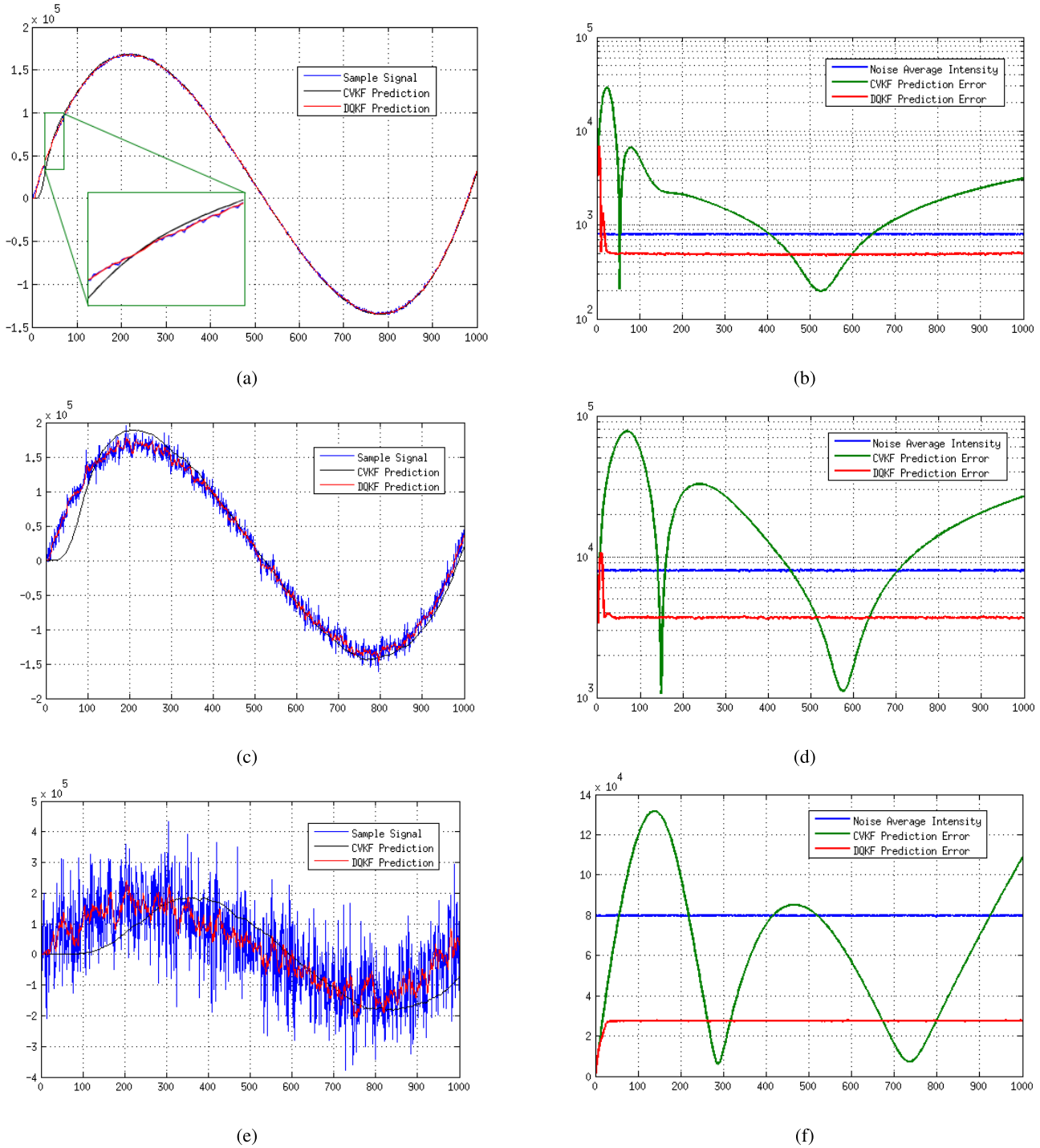

FIGURE 3. **Comparative results of the DQKF and the standard CVKF, for the low noise, medium noise and high noise cases. Figs. 3a, 3c and 3e are samples from the Monte-Carlo simulation, showing the predicted trajectory for both the DQKF and the CVKF, with the observed trajectory. Horizontal axes are time (*s*) for all figure. Vertical axes are measured position (*m*) for Figs. 3a, 3c and 3e, and average prediction RSME (*m*) for Figs. 3b, 3d and 3f. (a) Sample signal - 40 *dB*. (b) Trajectory prediction RMSE - 40 *dB*. (c) Sample signal - 20 *dB*. (d) Trajectory prediction RMSE - 20 *dB*. (e) Sample signal - 0 *dB*. (f) Trajectory prediction RMSE - 0 *dB*.**

A. TRACKING MANEUVERING TARGETS

For the maneuvering target case, the proposed trajectory was defined by

$$x_0 = 0m \tag{23}$$

$$v_0 = 1.7 \cdot 10^3 m/s \tag{24}$$

$$a_0 = -10m/s^2 \tag{25}$$

$$\frac{\partial a}{\partial t} = 0.02m/s^3, \tag{26}$$

from $t = 1\ s$ to $t = 1000s$, sampled with $\delta t = 1\ s$.

Noise was added with standard deviations $\sigma_x = 10^3\ m$, for the low-noise case; $\sigma_x = 10^4\ m$ for the medium-noise case; and $\sigma_x = 10^5\ m$ for the high-noise case. This is equivalent to a SNR of approximately $40dB$, $20dB$ and $0dB$, respectively. We performed 10^5 Monte-Carlo simulations for each case and measured both the prediction root mean squared error (RMSE) and the tracking bias. The prediction RMSE was calculated based on the real, noise-free signal, and the tracking bias was calculated as the average difference between the predicted position and the real (noise-free) position in the last predicted point, at t = 1000s.

We compared the performance of the DQKF with a standard CVKF, in the discrete time near-continuous velocity model, with the process error matrix defined in section II-D and $\sigma_{ga}^2 = 33.3\ m$, in line with the standard recommendation [19].

Table 1 shows the RMSE of predicting the next position, and Table 2 shows the tracking bias.

It is interesting to notice that the prediction RMSE was significantly smaller for the DQKF, being a fifth of the CVKF for the low and medium-noise cases. Most impressive, though, is that the tracking bias performance of the DQKF is much less sensitive to noise than the CVKF, being more than $100\times$ better for the high-noise case.

Figs. 3a, 3c and 3e are samples from the Monte Carlo simulation, for the low-noise, medium-noise and high-noise cases, respectively. The **blue** line is the measured signal, the **black** line is the predicted signal from the CVKF, and the **red** line is the predicted signal from the DQKF. Both predictions are for times $t = 2$ to $t = 1000$.

Figs. 3b, 3d and 3f are the prediction RMSE, calculated for each point in the trajectory, from $t = 2$ to $t = 1000$, for the low-noise, medium-noise and high-noise cases, respectively. The **blue** line is the average noise intensity, the **green** line is the prediction RMSE for the CVKF, and the **red** line is the prediction RMSE for the DQKF.

B. REAL ERRATIC TRAJECTORY

An interesting case is the sharp-turn, in which the target being tracked completely inverts its direction. Even though it is an extreme case, a good performance of a tracking filter under these conditions is indicative of how well it will perform in less extreme cases. As such, we performed an experiment using a red ball tracked by a RGB-D camera (Fig. 4a) with a frame rate of 30 frames-per-second (fps).

The camera and tracking algorithms used to acquire the log from the ball's movement had a short latency due to environment conditions and computational power (Fig. 4). Nevertheless, there was some noise from the distance between the ball and the camera (the range of the camera is up to 5m only), from the variation on the rate of frame capture and from the spacial segmentation. This noise caused the target's position to jump from one point to another with different velocity and direction.

The tracking algorithm was robust to partial occlusion and light variations, so the log could be acquired without external

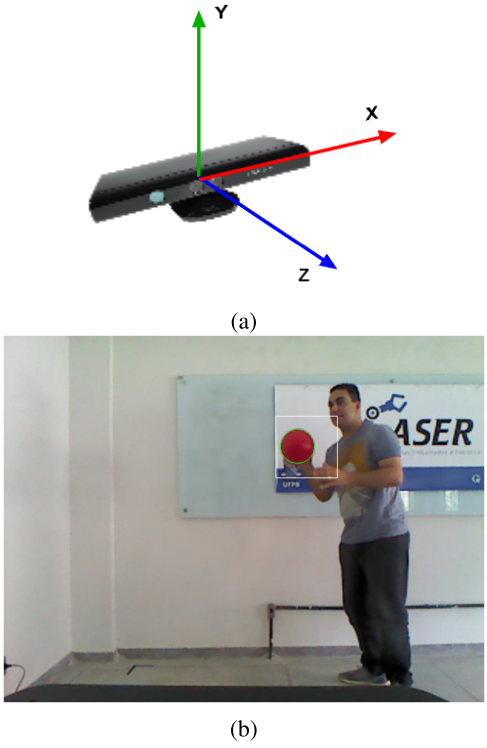

FIGURE 4. (a) RGB-D Camera. (b) A sample frame from the tracking experiment.

influence. The objective of this experiment was to see the response in tracking a real object with an erratic trajectory subject to aleatory change in direction. The tracked object moved in highly dynamic fashion with a stochastic, adiabatic and chaotic trajectory. Fig. 5 presents the tracking results from the comparison between the CVKF approach, the CVKF approach with best possible σ_{ga} and our DQKF proposed approach. We found the best possible σ_{ga} for the CVKF by trial and error, using a PSO algorithm in the fashion of [22], and minimizing the prediction error *a posteriori*. As such, it represents the actual minimum prediction RMSE that could be achieved with a CVKF.

We used Algorithm 1 with $a_c = 100$. It was observed that changing the values of γ and of the initial a_c had a small impact in the RMSE, including possible small gains that would reduce the difference from the best possible solution. However, we decided to keep the previously used values in order to remain faithful to our proposal of no prior knowledge or 'smart guesses'. With this decision, our technique (Fig. 5e) approaches the CVKF with the best possible value of σ_{ga} (Fig. 5c), being just 2.5% worse than the best possible solution. This difference is probably due to the considerable initial error, which is a consequence of the large initial a_c, but, in contrast, we do not know the trajectory *a priori*! Our approach also had a mean prediction error 5 times better than the classic CVKF, as shown in Table 3.

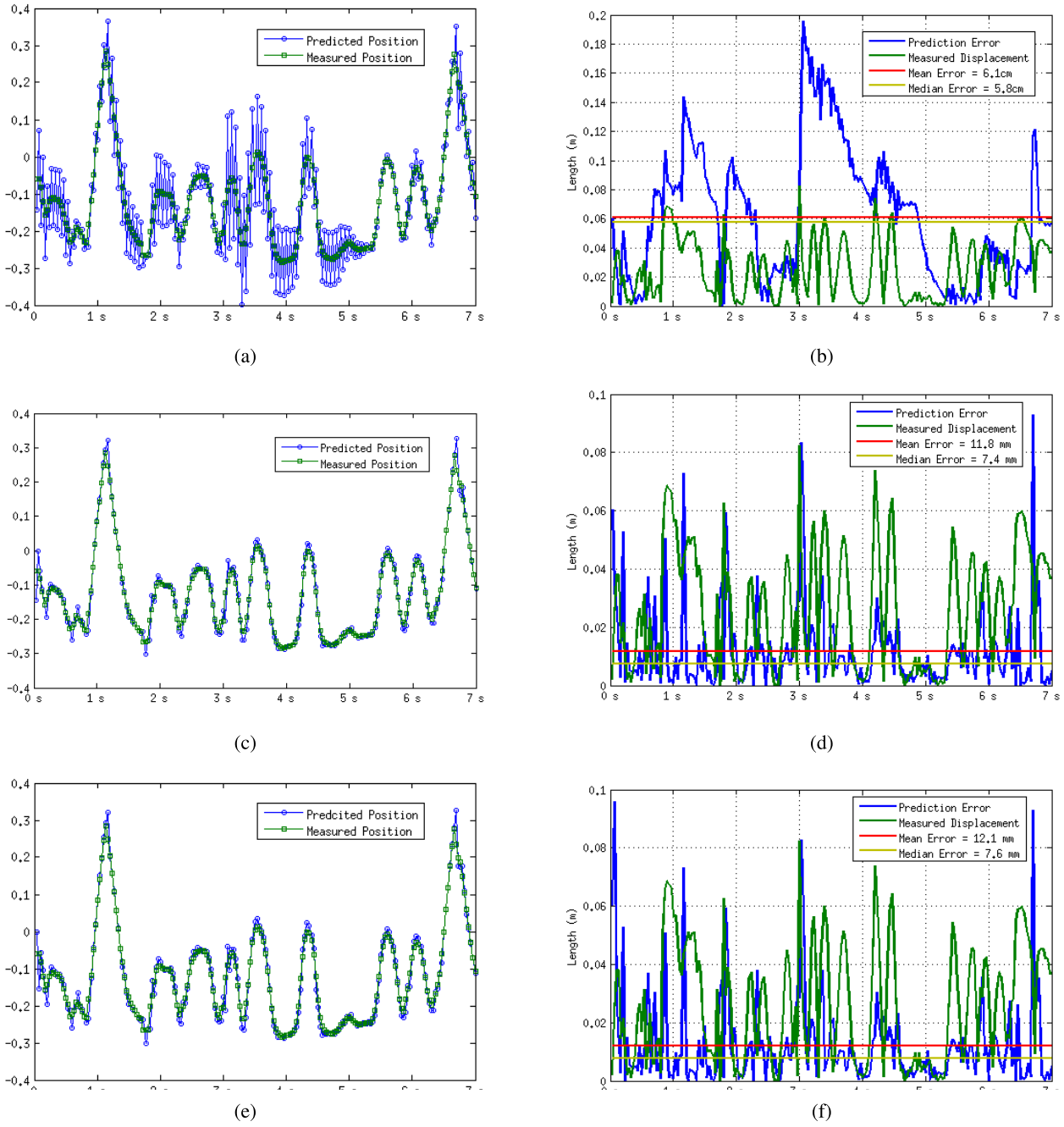

FIGURE 5. Comparative results of the CVKF and DQKF, for the real object trajectory tracking. (a) CVKF Predicted trajectory. (b) CVKF Predicted trajectory error. (c) CVKF Predicted trajectory, using the best possible σ_{ga}. (d) CVKF Predicted trajectory error, using the best possible σ_{ga}. (e) DQKF Predicted trajectory. (f) DQKF Predicted trajectory error.

TABLE 3. Comparative results from CVKF, CVKF with best possible σ_{ga} and DQKF.

Approach	Mean Error (cm)	Median Error (cm)
CVKF	6.10	5.80
CVKF - best σ_{ga}	1.18	0.74
DQKF	1.21	0.76

V. CONCLUSIONS

It was shown that an optimal Kalman Filter can be implemented, in a naive way, with adaptive parameter setting. Our proposed algorithm has shown impressive results, by closely matching the best possible CVKF in real, erratic trajectory prediction (Fig. 5) and consistently over-performing the CVKF with the standard DNCV model (Figs. 3 and 5). The mathematical reason for the improved performance is the choice of the RMS prediction error as the KF performance benchmark. This choice allowed the mapping of an optimal Q matrix to a measurable model deviation parameter, in this case the fading acceleration. Therefore, the DQKF is optimal in the sense that it will always have the best Q for the

measured model divergence, meaning it is always tunned to provide the minimum prediction error.

We mapped the best possible **Q** with respect to the CV model deviations (a_c), from $a_c = 10^{-2} m/s^2$ to $a_c = 10^2\ m/s^2$, but a wider or smaller range could also be done, depending on the intended use. Differently from previously proposed dynamical parameter setting techniques [15], [17], [18], our proposed method is unique, for it does not recalculate the parameter **Q** on every iteration of the filter, but actually calculates a trailing average of the velocity innovation and uses it to retrieve the previously mapped optimal value of Q.

With this simple setup, we achieved a near optimal prediction, with an average computational complexity increase of only approximately 7%. The hard computational problem was done beforehand, in the mapping of the optimal **Q**, so that the load of the Kalman Filter could be minimally changed.

A possible improvement to the technique could be the use of dynamical identification of measurement noise, as has been already suggested elsewhere [13], for the situation where the measurement apparatus is not well known.

As future work, we also believe the same methodology could be implemented for the Kalman Filter with near constant acceleration (CA) model, where a new *Analytic Steady-State Performance Index* would be developed, and the mapping variable would be the jerk ($\partial a/\partial t$).

ACKNOWLEDGMENT

The authors would like to thank the helpful comments of the anonymous reviewers.

REFERENCES

[1] Z. Li and H. Wu, "A survey of maneuvering target tracking using Kalman filter," in *Proc. 4th Int. Conf. Mechatronics, Mater., Chem. Comput. Eng. (ICMMCCE)*, vol. 39. 2015, pp. 542–545.

[2] M. Boutayeb and D. Aubry, "A strong tracking extended Kalman observer for nonlinear discrete-time systems," *IEEE Trans. Autom. Control*, vol. 44, no. 8, pp. 1550–1556, Aug. 1999.

[3] X. Yun and E. R. Bachmann, "Design, implementation, and experimental results of a quaternion-based Kalman filter for human body motion tracking," *IEEE Trans. Robot.*, vol. 22, no. 6, pp. 1216–1227, Dec. 2006.

[4] Y. T. Chan, A. G. C. Hu, and J. B. Plant, "Kalman filter based tracking scheme with input estimation," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-15, no. 2, pp. 237–244, Feb. 1979.

[5] C. Snyder and F. Q. Zhang, "Assimilation of simulated Doppler radar observations with an ensemble Kalman filter," *Monthly Weather Rev.*, vol. 131, no. 8, pp. 1663–1677, Aug. 2003.

[6] F. E. Daum and R. J. Fitzgerald, "Decoupled Kalman filters for phased-array radar tracking," *IEEE Trans. Autom. Control*, vol. AC-28, no. 3, pp. 269–283, Sep. 1983.

[7] R. S. Scalero and N. Tepedelenlioglu, "A fast new algorithm for training feedforward neural networks," *IEEE Trans. Signal Process.*, vol. 40, no. 1, pp. 202–210, Jan. 1992.

[8] Y. Iiguni, H. Sakai, and H. Tokumaru, "A real-time learning algorithm for a multilayered neural network based on the extended Kalman filter," *IEEE Trans. Signal Process.*, vol. 40, no. 4, pp. 959–966, Apr. 1992.

[9] Y. Barshalom and K. Birmiwal, "Variable dimension filter for maneuvering target tracking," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-18, no. 5, pp. 621–629, May 1982.

[10] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. Part V. Multiple-model methods," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 41, no. 4, pp. 1255–1321, Oct. 2005.

[11] R. K. Mehra, "On the identification of variances and adaptive Kalman filtering," *IEEE Trans. Autom. Control*, vol. AC-15, no. 2, pp. 175–184, 1970.

[12] N. K. Sinha, "Adaptive Kalman filtering using stochastic approximation," *Electron. Lett.*, vol. 9, no. 8, pp. 177–178, May 1973.

[13] M. Oussalah and J. de Schutter, "Adaptive Kalman filter for noise identification," in *Proc. Int. Seminar Modal Anal.*, vol. 3, 1998, pp. 1225–1232.

[14] I. Hashlamon and K. Erbatur, "An improved real-time adaptive Kalman filter with recursive noise covariance updating rules," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 24, no. 2, pp. 524–540, 2016.

[15] H. R. Wang, Z. H. Deng, B. Feng, H. B. Ma, and Y. Q. Xia, "An adaptive Kalman filter estimating process noise covariance," *Neurocomputing*, vol. 223, pp. 12–17, Feb. 2017.

[16] F. Sun, X. Hu, Y. Zou, and S. Li, "Adaptive unscented Kalman filtering for state of charge estimation of a lithium-ion battery for electric vehicles," *Energy*, vol. 36, no. 5, pp. 3531–3540, 2011.

[17] D. Guo and X. Wang, "Quasi-Monte Carlo filtering in nonlinear dynamic systems," *IEEE Trans. Signal Process.*, vol. 54, no. 6, pp. 2087–2098, Jun. 2006.

[18] M. Karasalo and X. Hu, "An optimization approach to adaptive Kalman filtering," *Automatica*, vol. 47, no. 8, pp. 1785–1793, 2011.

[19] X. R. Li and P. J. Vesselin, "Survey of maneuvering target tracking. Part I. Dynamic models," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 39, no. 4, pp. 1333–1364, 2003.

[20] G. Vivone, P. Braca, and J. Horstmann, "Knowledge-based multitarget ship tracking for HF surface wave radar systems," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 7, pp. 3931–3949, Jul. 2015.

[21] K. Saho and M. Masugi, "Automatic parameter setting method for an accurate Kalman filter tracker using an analytical steady-state performance index," *IEEE Access*, vol. 3, pp. 1919–1930, Sep. 2015.

[22] C.-F. Wang and K. Liu, "A novel particle swarm optimization algorithm for global optimization," *Comput. Intell. Neurosci.*, vol. 2016, Art. no. 9482073, doi: 10.1155/2016/9482073.

GABRIEL F. BASSO received the bachelor's and Ph.D. degrees in physics from the Federal University of Paraiba, João Pessoa, Paraiba, Brazil, in 2009 and 2014, respectively. He was an Anbima Certified Money Manager–CGA in 2016. He currently holds a post-doctoral position with the Embedded Systems and Robotics Laboratory, Informatics Center, Federal University of Paraiba, where he investigates non linear modeling of dynamic systems.

THULIO GUILHERME SILVA DE AMORIM is currently pursuing the bachelor's degree in computer science at the Federal University of Paraiba, João Pessoa, Paraiba, Brazil. He is involved with NavRob: navigation systems of mobile robot teams in dynamic environments with the Laboratory of Embedded Systems and Robotics.

ALISSON V. BRITO received the B.Sc. and M.Sc. degrees in computer science from the Federal University of Campina Grande (UFCG), Campina Grande - PB, Brazil, in 2001 and 2003, respectively, and the Ph.D. degree in electrical engineering in the field of microelectronics at URCG in cooperation with the Karlsruhe Institute of Technology, Karlsruhe, Germany, in 2008. He possesses experience in computer science with an emphasis on design and development of embedded systems, mainly in the following themes: design and simulation of embedded systems and applications using unmanned aerial vehicles. He is currently a Professor with the Computer Center, a Coordinator with the Laboratory of Embedded Systems and Robotics, and involved with the Graduate Program in Computer Science of the Federal University of Paraiba (UFPB), João Pessoa, Paraiba, Brazil, where he teaches courses mainly in the areas of computer architecture and embedded systems for students from computer science and computer engineering. He is a Reviewer of national and international journals and conferences.

TIAGO P. NASCIMENTO (M'04) received the B.S. degree in mechatronics engineering from the College of Technology and Science–FTC, Salvador–Bahia, Brazil, in 2007, the M.S. degree in electrical engineering from the Federal University of Bahia, Salvador, Brazil, in 2009, and the Ph.D. degree in electrical and computer engineering from Oporto University, Porto, Portugal, in 2012. In 2013, he joined the Department Computer Systems, Federal University of Paraiba, João Pessoa, Brazil, as an Assistant Professor. His main research interests are mobile robotics, multi-robots systems, process control, vehicle dynamics, and intelligent control. He has been a member of the IEEE RAS since 2004.