OKPS: a reactive/cooperative multi-sensors data fusion approach designed for robust vehicle localization

Adda Redouane Ahmed Bacha, Dominique Gruyer, Alain Lambert

OKPS: a reactive/cooperative multi-sensors data fusion approach designed for robust vehicle localization

A. R. Ahmed Bacha, D. Gruyer and A. Lambert

Adda Redouane AHMED BACHA and Dominique GRUYER are with the IFSTTAR/LIVIC
(The French institute of science and technology for transport, development and networks), France.
Alain LAMBERT is with University Paris Sud, France
{adda-redouane.ahmed-bacha, dominique.gruyer}@ifsttar.fr, alain.lambert@u-psud.fr

Abstract

This paper presents the Optimized Kalman Particle Swarm (OKPS) filter. This filter results of two years of research and improves the Swarm Particle Filter (SPF). The OKPS has been designed to be both cooperative and reactive. It combines the advantages of the Particle Filter (PF) and the metaheuristic Particle Swarm Optimization (PSO) for ego-vehicles localization applications. In addition to a simple fusion between the swam optimization and the particular filtering (which leads to the Swarm Particle Filter), the OKPS uses some attributes of the Extended Kalman filter (EKF). The OKPS filter innovates by fitting its particles with a capacity of self-diagnose by means of the EKF covariance uncertainty matrix. The particles can therefore evolve by exchanging information to assess the optimized position of the ego-vehicle. The OKPS fuses data coming from embedded sensors (Low cost INS, GPS and Odometer) to perform a robust ego-vehicle positioning. The OKPS is compared to the EKF filter and to filters using particles (PF and SPF) on real data from our equipped vehicle.

Keywords: Localization, mobile robotic, Extended Kalman filter, Particle Swarm Optimization, Particle filter, data fusion, vehicle positioning, GPS.

1. Introduction

Localization is a key technological component for any Advanced Driver Assistance System (ADAS). The localization information can be operated to develop new services which aim to increase drivers' autonomy and/or safety. These new services open fields for new Intelligent Transport Systems applied to Road applications (ITS-R) ie. parking valet, and automated driving... Therefore, ego-vehicle accurate and reliable localization becomes an important issue in ITS research field. The objective is: to provide a much more precise vehicle positioning than with a classic Global Navigation Satellite System (GNSS) and, a reliable and consistent accurate positioning solution with low financial costs. Thus, on-board sensors or low cost sensors are used to comply with the GPS in order to perform a data fusion. Additional sensors like inertial navigation system (INS), odometer, and steering wheel angle sensor can be used to bring new information. Fusing GPS with proprioceptive sensors get a better ego-vehicle positioning and at a higher rate than the GPS alone.
A large number of research works focuses on data fusion for vehicle localization applications [1, 2]. Estimating the position and orientation estimation is considered as a state estimation problem from a mathematical point of view [3, 4]. State estimation is generally processed using Markovian methods. The Kalman Filter (KF) is an optimal filter for state estimation in the case of linear Gaussian systems [5]. Unfortunately, accurate vehicles dynamic models are nonlinear. To deal with nonlinearity, the Extended Kalman Filter (EKF) and other Kalman variants have been proposed [6, 7, 8, 2, 9, 10, 11, 12]. The multi-hypothesis particular approach was developed to cover the EKF instability for the highly non-linear cases. The Particle Filter (PF) spread for research on autonomous vehicles localization and tracking applications [13, 14, 15]. However, the PF suffers from particles degeneracy and its accuracy depends on the particles number which have a direct impact on the computation time [16]. Resampling prevents the particle dispersion and PF divergence [17].

Aiming to create autonomous intelligent localization approaches for autonomous smart vehicles, the research community interests in hybrid approaches merging benefits from existing independent methods. Firstly intended for simulating social behavior, the Particle Swarm Optimization (PSO) [18], is a metaheuristic optimization method improved for iterative optimization issues [19, 20, 21, 22]. Particles of a swarm exchanges information and coexists in the objective of a mutual cooperation to reach the best solution. The particles move independently toward the region were the positioning probability is higher (given the sensor information). Next, particles optimize their poses by evolving toward their best neighbors (social communication) to iteratively converge to local optima or a global optimum. Localization applications, inspired from PSO, rises within these last years [14, 13, 23]. The Swarm particle Filter (SPF) is an evolved hybridization of a generic PF with PSO [13, 24, 25, 26, 27]. PSO is used to optimize the particle distribution to overcome the PF problems of sample size dependency and particles impoverishment. The SPF can be employed for tracking and localization applications [28, 13]. However, for multi-sensor data fusion based high dynamic vehicle localization, the SPF still has premature convergence and parameterization problems. A similar problem has been solved in [23] (where PSO is used for tuning noise parameters in addition to a Kalman filter) for dealing with strongly non-linear situations (aircraft abrupt dynamic changes). The resulting algorithm is a PSO aided EKF which needs anyway a PSO parameters tuning. We inspired from it and introduce an innovative localization method for vehicles. The Optimized Kalman Particle Swarm (OKPS) performs the vehicle real-time positioning considering a dynamic optimization problem. Earlier versions of our algorithm has been presented in [30] and [31]. This paper improves on our previous papers by introducing an adapted criteria for the driving tests and validation. Furthermore, our algorithm is fully detailed and a thorough comparison is realized with another algorithms. At each time step, the OKPS tries to find the best possible vehicle position according to the predicted vehicle state, the GPS measurement and the relayed information between particles (neighbors). The OKPS intends to minimize the parameterization needs while taking advantages of PSO/EKF/PF hybrid approache.

All these approaches are compared and ranked in terms of accuracy and robustness using real word experimental data of driving scenarios on the Satory-Versailles test track. Filters localization performance is evaluated in comparison with a centimetric RTK GPS used as a reference. The filters robustness is evaluated using filters uncertainty ellipses areas. The obtained results give a significant distinction between filters performances depending on the GPS quality (good, noisy, multi-path or missing signal).

Section II is dedicated to a background part; it introduces the approaches inspiring the OKPS by exposing their algorithmic and theoretical foundations. In Section III, the Optimized Kalman Particle Swarm is detailed. Section IV carries out a theoretical and experimental filters comparison with the localization results analysis. A discussion is held in Section V providing improving proposals setting out to fulfill a robust non-parameterized hybrid localization approach.

2. review of localization methods

This section describes the approaches inspiring the OKPS filter. It gives an overview of the approaches studied for the algorithm implementation. Details of the mathematic fundamentals are given for each method.

2.1. Extended Kalman Filter (EKF)

The Extended Kalman Filter (EKF) is a recursive estimator and represents the nonlinear adaptation of the linear Kalman filter (KF). For well-defined and accurate modeling of non-linear process, EKF is the most widely used filter for state estimation. The process model must be derivable to allow the linearization by Jacobians calculation. The previous estimate or state at $k-1$ and the current measurements Y_k are used to estimate the current state. The filtered current estimate is represented by $\hat{X}_{k|k}$ and $P_{k|k}$ represents the estimation uncertainty noted as a covariance matrix.

The EKF passes through two main stages: Prediction and Update. The first one produces a predicted state $\hat{X}_{k|k-1}$ using the last estimated state $\hat{X}_{k-1|k-1}$, the evolution model (Bicycle Model) and data from proprioceptive sensors. In the update step, the corrective measure Y_k is used to correct the predicted state. The EKF requires an initialization step giving initial values to the following parameters: X represents the state vector $[x\ y\ \theta]^T$, U the command vector, P the variance/covariance confidence matrix, μ the process noise, Q the variance/covariance matrix representing the process noise, v: the measurement noise, R the variance/covariance matrix representing the Measurement noise and Y the measures Vector.

2.1.1. Initialization

$$\begin{aligned} \hat{X}_0 &= E[X_0] \\ P_0 &= E[(X_0 - \hat{X}_0)(X_0 - \hat{X}_0)^T] \\ Q_0 &= E[(\mu - \overline{\mu})(\mu - \overline{\mu})^T] \\ R_0 &= E[(v - \overline{v})(v - \overline{v})^T] \end{aligned} \tag{1}$$

2.1.2. Jacobians

The Jacobian matrices are matrix of partial derivatives. A and H are derived respectively from f , the transition or evolution function and h, the observation function.

$$\begin{aligned} A_{xk} &= \nabla_x f(X, u_k, \overline{\mu})|_{X=\hat{X}_{k-1|k-1}, U_{k-1}} \\ H_k &= \nabla_x h(X, \overline{v})|_{X=\hat{X}_{k|k-1}} \end{aligned} \tag{2}$$

2.1.3. Prediction

The prediction is made using the evolution model $f(\hat{X}_{k-1|k-1}, U_{k-1})$ and its linear evolution matrix A.

$$\begin{aligned} \hat{X}_{k|k-1} &= f(\hat{X}_{k-1|k-1}, U_{k-1}) \\ P_{k|k-1} &= A_{xk} P_{k-1|k-1} A_{xk}^T + Q_k \end{aligned} \tag{3}$$

2.1.4. Update

The update is a linear correction done with the Kalman gain which is calculated using the measurement matrix H to adjust the prediction according to the available measurement. The updated state $\hat{X}_{k|k}$ is mathematically a linear weighted average between the prediction and the measurement. The weight is the Kalman gain which will tend to the prediction or the available measure according to their respective uncertainties.

$$\begin{aligned} K_k &= P_{k|k-1} H_k^T [H_k P_{k|k-1} H_k^T + R_k]^{-1} \\ \hat{X}_{k|k} &= \hat{X}_{k|k-1} + K_k [Y_k - H_k \hat{X}_{k|k-1}] \\ P_{k|k} &= (I - K_k H_k) P_{k|k-1} \end{aligned} \tag{4}$$

2.2. Particle Swarm Optimization basics (PSO)

The Particle Swarm Optimization (PSO) is a Metaheuristic Optimization method based on a set of samples called particles initially randomly and uniformly distributed in the search space according to a Gaussian distribution around an initial value. Each particle moves in the search space and represents a potential solution of the processed problem(s). Each particle has a memory capacity that allows it to know at each iteration what is its best performance to the current situation. A swarm particle also has the capacity of communicating with its neighbors (connected communicative particles), which allows it to know what is the best performance achieved by its neighboring particles. Using this information, each particle will move blending together three trends or tendencies: The tendency to keep its own way (*selfish*), the *Conservative* trend and the *Social* trend (Panurgism). For the first trend, the particle tends to use its inertia and will consequently continue keeping its own direction and evolution. By adopting the second trend, the particle tends to go back to its last best performance. For the third behavior, the particle tends to move toward the best solution found by its neighborhood. Figure 1 shows the PSO particle motion principle illustrating the tendencies collaborative mechanism of the particles.

A particle i at time k is characterized by a set of attributes. The first attribute is its vector in the search space $\hat{x}_k^i$. The second one is the displacement information which can be taken and is noted as a speed $\hat{v}_k^i$. Then, the third one is

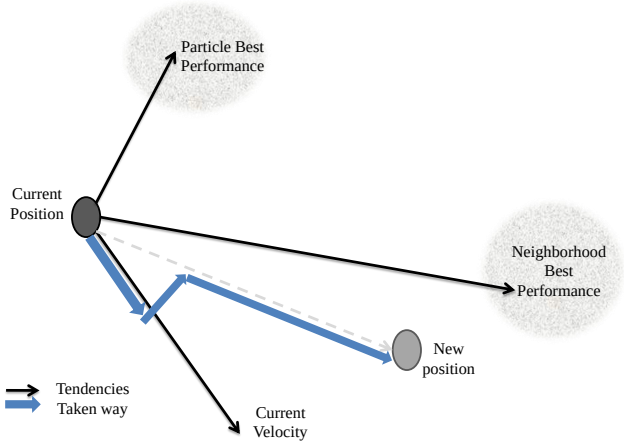

Figure 1: Particle motion in a PSO process

a set of performance information. The first performance data corresponds to the best solution found by a particle P_b^i (Personal Best). The second performance data characterizes the best solution at iteration k reached by all neighbors G_b (Global Best). At each new iteration, the position of the particle$i$ is updated using the previously mentioned information applying the PSO motion principle described in the following evolution equations:

$$\begin{aligned} \hat{x}_k^i &= \hat{x}_{k-1}^i + \hat{v}_k^i \\ \hat{v}_k^i &= \underbrace{W * \hat{v}_{k-1}^i}_{Inertia} + \underbrace{c_1 r_1 * (P_b^i - \hat{x}_{k-1}^i)}_{Personal\ Influence} + \underbrace{c_2 r_2 * (G_b - \hat{x}_{k-1}^i)}_{Social\ Influence} \end{aligned} \tag{5}$$

In Equation 5, the coefficients c_1 and c_2 are learning terms. The coefficients r_1 and r_2 are random numbers following a uniform distribution. These terms are used to produce a variation in the development of each particle providing a diversity of attitude that supports the search space exploration. The coefficient W is the inertia factor. Some variants of these evolution equations have been proposed in various studies [18, 32, 33] and [13]. Each of these variants of evolution equations is designed to balance between the tendency to explore and the tendency to converge towards the optimal solution (swarm condensation/dispersion). For more details, [34] and [35] propose comparative studies of these PSO evolution variants. For the particles interaction, it exists two principal sorts of neighborhood topologies (geographical dynamic neighborhood and static social neighborhood). The more the neighborhood is informative the more information is shared supporting the swarm convergence. The geographical neighborhood topology is based on the nearby particles and is considered as a dynamic topology because it needs to be calculated at each iteration (after particles evolution). The *social neighborhood* configurations shown in Figure 2 are set at the beginning of the process and do not require distance calculation to find the neighbors. Note that generally, in case of particles convergence, the social neighborhood tends to become geographical.

For readers interested in PSO, [34] and [36] provides an interesting analysis of recent developments in PSO evolution strategies and neighborhood topologies providing the basic algorithm and most of its variants. The details of the followed steps in the basic Particle Swarm Optimization algorithm are described in Algorithm 1.

2.3. Particle Filter (PF)

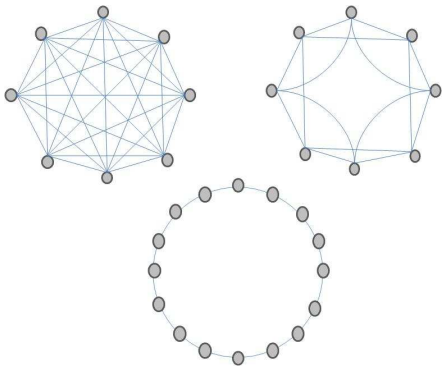

Figure 2: Different configurations of social neighborhood : Star, ring, and circle.

Algorithm 1: Basic PSO Algorithm

```
begin
Initialization()
Initialize the swarm giving initial locations and velocities stochastically assigned
while (the end criterion* is not met) do
    Fitness()
    Evaluate the particle scores using a Fitness function
    Updating()
    According to Fitness values, update the P_b and G_b
    Evolution()
    Move particles according to the evolution equations
    Estimation()
    The output value is the G_b
end
end
*: A number of iterations or an adequate fitness value.
```

The particle filtering is a method of Sequential Monte-Carlo Simulation (SMCS) [16]. Using empirical distribution of particles, the Particle Filter (PF) approximates the distribution of an estimated process. Particles explore the state space evolving independently in the search space. Each particle is a possible state of the process. Particles dispersion and concentration are managed by an Importance Sampling-Resampling (ISR) mechanism [17]. The sampling attributes a weight to each particle representing its actual consistency. The resampling updates these weights and performs a duplication-elimination mechanism according to the updated attributed weights. These two importance selection mechanisms work together to guarantee the particles distribution homogeneity and the PF stability avoiding the particles impoverishment. The PF is a good alternative to the non-linear Kalman filter variants for the ego-vehicle positioning application. With an adequate number of particles, the PF approximates the optimal probability distribution of an estimate. In comparison with the Kalman filters, the PF can be more accurate and more robust in case of strong non-linearity and noises. However, the PF performance depends directly on the available computation power (number of samples). The user must do a compromise between computational time and accuracy. The particle filter algorithmic steps are detailed in the following:

Initialization. The initial position of the ego-vehicle is represented by the state vector $X_{init} = [x_{init}\ y_{init}\ \theta_{init}]^T$. The initial cloud of particles is drawn around the initial position by assigning each particle a state vector X^i_{init} and an initial weight w^i_0. N is the number of particles (samples) which is to determined by the user. The state vectors of the particles represent a Gaussian distribution centered on the initial position according to this model:

Algorithm 2: Algorithm of the particle filter applied to the vehicle ego-localization

at time $k = 0$ **(**X_0 **et** P_0 **known)**
Initialization()
Generating the first population of particles:
N particles randomly distributed around the initial position with $W_0^i = \frac{1}{N}$.
while *(Sensor data available)* **do**
  If(Proprio data)
  Move particles in the search space according to the data and the vehicle model.
  Retain $W_0^i = \frac{1}{N}$.
  Calculate the predicted state and its uncertainty:
  $X_{vehicle} = \sum_{i=1}^{N} X_{k|k-1}^i * \frac{1}{N}$
  $P(X_{vehicle}) = \sum_{i=1}^{N} w_k^i \left(X_{k|k-1}^i - X_{vehicle}\right)\left(X_{k|k-1}^i - X_{vehicle}\right)^T$
  If (Extero data)
  Calculate / update the weight of each particle :
  $w_k^i \propto w_{k-1}^i \cdot p(Y_k|X_{k|k-1}^i) = w_{k-1}^i \cdot \frac{1}{\sqrt{(2\pi)|R_k|}} \exp\{-\frac{1}{2}(Y_k - \hat{Y}_k)^T [R_k]^{-1}(Y_k - \hat{Y}_k)\}$.
  Normalize weights $w_k^i = \frac{w_k^i}{\sum_{j=1}^{N} w_k^j}$.
  Determine the estimated state of the system and its uncertainty:
  $X_{vehicle} = \sum_{i=1}^{N} X_{k|k-1}^i * w_k^i$
  $P(X_{vehicle}) = \sum_{i=1}^{N} w_k^i \left(X_{k|k-1}^i - X_{vehicle}\right)\left(X_{k|k-1}^i - X_{vehicle}\right)^T$
  Calculate the resampling factor :
  $N_{eff} = \frac{1}{\sum_{j=1}^{N} (w_k^i)^2}$
  If ($N_{eff} \leq N_{th}$**)**
  Proceed resampling particles according to the chosen method.
  Redistribute weights to new particles.
end

Particle i has a state vector $X^i_{init} = \begin{pmatrix} x_{init} + \epsilon^i_x \\ y_{init} + \epsilon^i_y \\ \theta_{init} + \epsilon^i_\theta \end{pmatrix}$ and a weight $w^i_0 = 1/N$.

$\epsilon^i_x, \epsilon^i_y, \epsilon^i_\theta$ are random variables representing the uncertainty of the initial positioning state. The uncertainty ellipse will be determined by the particles scattering across the search space around the initial position according to a normal distribution. The centered normal laws of probability used in this case are $N(0, \sigma_x)$, $N(0, \sigma_y)$, $N(0, \sigma_\theta)$. When, $(\sigma_x, \sigma_y, \sigma_\theta)$ are the initial standard deviations noise values given by the sensors characteristics and/or the standard deviation of an initialization data set.

Prediction. This stage will give an *a priori* ego-vehicle positioning estimate for each particle $\hat{X}^i_{k|k-1}$. Using the proprioceptive sensors data and the last localization estimate $\hat{X}^i_{k-1|k-1}$, the particles will predict the vehicle state $\hat{X}^i_{k|k-1}$. This predicted position is calculated using the bicycle vehicle model integrating the proprioceptive sensors information and noises. The prediction $\hat{X}^i_{k|k-1}$ represents the vehicle possible actual state according to the last known state and the evolutionary information until this moment. Uncertainties of the evolution model and those of proprioceptive sensors data are very important for the proper functioning of the PF filter. These uncertainties are incorporated in order to promote the diversity in the particles evolution. Giving a different prediction evolution for each particle, this integration of the measurements and modeling uncertainties is the main reason which allows the particles to better explore the search space. When these noises are too low, the filter will not explore the search space and noise conditions will not be well represented by the particles distribution. When the noises are too high, the filter suffers from particles impoverishment due to excessive scattering of particles on several consecutive steps.

Update. In this stage, the prediction of each particle is adjusted by the reassessment of particles weights according to a new updated exteroceptive sensor data provided for instance by a low cost GPS. The update can be taken as compromising between the predictive and the corrective estimates. The conciliation between these two estimates is done by taking into account their respective uncertainties. The calculation of particles weight is carried out by Equation (6).

$$P^i_k \propto P^i_{k-1} P(y_k|x^i_k) \quad \equiv \quad w^i_k \propto w^i_{k-1} * P(Y_k \mid X^i_k)$$
$$w^i_k \propto w^i_{k-1} * \frac{1}{\sqrt{(2\pi)|R|}} \exp\{-\frac{1}{2}(Y_k - \hat{Y}^i_k)^T [R_k]^{-1} (Y_k - \hat{Y}^i_k)\} \tag{6}$$

Standardization and Estimation. The weights standardization of the particles is done after updating to normalize the probabilities sum. The standardized weights are calculated according to the Equation (7).

$$w^i_k = \frac{w^i_k}{\sum_{j=1}^N w^j_k} \tag{7}$$

Then, the particles estimates are merged to give a global ego-vehicle state estimation $\hat{X}_{k|k}$. The vehicle position can be calculated with the Equation (8):

$$\hat{X}_{k|k} = \sum_{i=1}^{N} \hat{X}^i_{k|k-1} * w^i_k \tag{8}$$

Resampling. Resampling is a critical step for the stability of the particle filter. It is a selective mechanism that eliminates low weight particles and duplicates particles with high weights. The aim of applying the resampling mechanism is to control the particles dispersion without affecting the probabilistic distribution of the particles (minimum impact on the ellipse of uncertainty).

It exists a wide range of methods for resampling as well as criterions for enabling/disabling resampling. For more details, see here [17].

In this work, the used algorithm is the *systematic resampling* with the *Kong criterion* noted in the following.

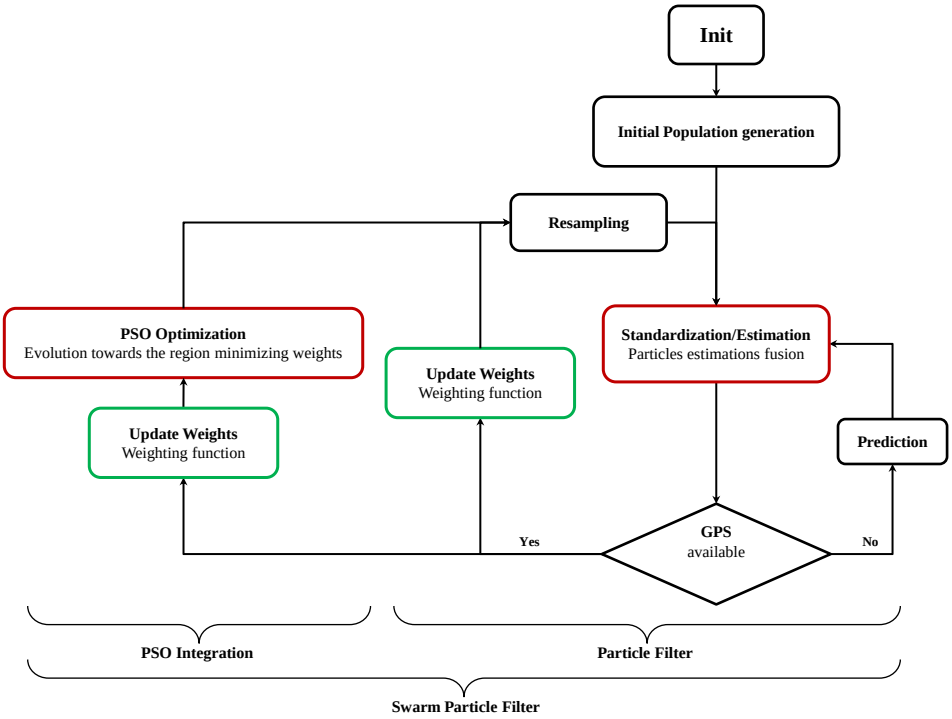

Figure 3: SPF algorithm based on PF/PSO hybridization

$$N_{eff} = \frac{1}{\sum_{j=1}^{N}(w_k^i)^2} \tag{9}$$

N_{eff} is the *Kong* criterion which indicates the number of effective particles according to their standardized weights w_k^i. N_{eff} tends to N when the distribution of the particles weights is efficient and N_{eff} tends to 1 when the distribution of the particles is inefficient.

If N_{eff} is under the defined threshold N_{th}, the resampling is then performed. The threshold N_{th} represents the minimum of required effectiveness, it is generally a fixed value within this range $0 < N_{th} \leqslant N$. It is fixed using the term α. $N_{th} = \alpha * N$ with N the number of particles and α represents the minimum desired percentage of consistent particles, for example $\alpha = 0.5$ for 50% of minimum effective particles. Note that depending on the applied resampling approach, particles weights are set to $w_k^i = \frac{1}{N}$ or recalculated depending on particles duplication number after the resampling process.

2.4. The Swarm Particle Filter (SPF)

The particles of the PF do not perform any individual correction when the GPS is available. The estimation in the PF is done only using the weighted predictions $\hat{X}_{k|k-1}^i$, see Equation (8). In order to propose an evolutionary approach which performs a real corrective step in the updating stage, the Swarm Particle Filter was developed.

Inspired by the PSO, the SPF is a hybridization of the PF with an integration of the social influence of the PSO. The SPF is expected to bring a particles interaction providing the capability for each particle to evolve in function of the neighborhood (correct the prediction). Particles go through all the normal PF steps. In addition to the classic PF stages, after the update stage, particles communicate and evolve in order to optimize their estimate and the swarm distribution. The particles move toward the region maximizing the positioning probabilities (particles weights).

The SPF performs in addition to the PF steps an evolution step just after the *update* and before the ego-vehicle positioning calculation and *resampling*. The SPF evolution step is the same concept of the evolution in PSO. However,

the evolution equations are adapted to the application, giving the Equation (10). The scores are calculated as PF weights and G_b is obtained by comparing the particles weights. The Figure 3 describes the PF hybridization giving the SPF algorithm.

$$\begin{aligned} \hat{v}_k^i &= W * \hat{v}_{k-1}^i + |randn|(G_b - \hat{X}_{k|k-1}^i) \\ \hat{X}_{k|k}^i &= \hat{X}_{k|k-1}^i + \hat{v}_k^i \end{aligned} \tag{10}$$

In the literature, approaches addressing dynamic optimization problems [XIA04] eliminate the P_b^i part from the basic PSO evolution Equation (5) (P_b^i is set equal to the particle current position), this operation is called memory erasing. By considering the optimization in a dynamic environment, the choice of erasing memories of the particles is adopted. The PSO Equation (5) gives then the SPF evolution Equation (10). This evolution variant also integrates Gaussian random numbers instead of learning factor with uniform random numbers. It was proposed and validated by [33] with the advantage of minimizing the number of parameters to be fixed in comparison with the basic PSO evolution.

The SPF filter applied to the ego-vehicle localization showed problems of premature convergence. The main reason is that the particles exchange only weight information. This weight calculated using Equation (6) considers the particle performance exclusively according to the GPS data. Thus, when the GPS data is erroneous and its measurement error matrix R_k is not sufficiently representative of the measure noise, the particles are attracted by the GPS data. By applying this principle, the SPF develops after few localization steps too much dependency on GPS data and the particles distribution is concentrated steeply around the current GPS data. This behavior penalizes the filter in the next prediction steps until a new GPS data arrives. Complying with the PF filtering and PSO evolution concept, The SPF suffers generally from premature convergence or swarm explosion problems. The SPF needs also PSO parameters tuning by the user.

In order to prevent the premature convergence problem, some particles are deprived of the neighborhood information G_b which creates troublemaking particles preventing premature swarm concentration. The troublemaking particles (blind/non evolutive particles) are selected by fixing their number at the beginning or chose randomly at each step. These kind of particles do not apply the PSO optimization encouraging the swarm expansion and the search space exploration.

Based on the SPF concept and trying to overcome the problems of the GPS attraction and premature convergence, the OKPS filter was developed to perform a reactive-cooperative ego-vehicle localization. The idea of the OKPS conception is to allow each particle to judge its uncertainty not only relatively to GPS data, but by taking also into account the swarm dynamic (theoretically the vehicle dynamic). It will make each particle acting as a sensor fitted with self-diagnostic capacity. Then, each particle will provide an estimate and its associated estimation error.

3. The Optimized Kalman Particle Swarm (OKPS)

The OKPS is an evolution of the SPF. This proposed filter integrates in addition to the social concept of the SPF a cognitive one. The cognitive concept consists in giving an intelligence self-diagnostic capacity to the swarm particles. An adaptive weighting function, called *fitness* function is also developed to allow adaptive particles weights calculation considering this new capacity. The fitness function takes the EKF gain concept in order to be as representative as possible of the particle current probability. The role of the fitness function is to give a weight for each particle considering its efficiency relatively to the GPS measure and relatively to the particle confidence on its prediction at the same time.

3.1. OKPS Implementation

To perform an optimization-filtering approach allowing to be both reactive and cooperative, the OKPS combines the advantages of the already presented techniques. The cooperative aspect consists in the information exchanging and interaction between particles. While, the reactive aspect comes out in the capacity of detecting changes in the vehicle's dynamic. This capacity is the direct consequence of fitting particles with a simple self-diagnose mechanism.

Algorithm 3: Algorithm of the SPF applied to the vehicle ego-localization

At time $k = 0$ **(**X_0 **et** P_0 **known)**

Initialization()

Generating the first population of particles:

N particles randomly distributed around the initial position with $W_0^i = \frac{1}{N}$ et $v_0^i = 0$.

while *(Sensors data available)* **do**

If(Proprio data)

Move particles in the search space according to the data and the vehicle model.

Retain $W_0^i = \frac{1}{N}$.

Calculate the predicted state and its uncertainty:

$X_{vehicle} = \sum_{i=1}^{N} X_{k|k-1}^i . \frac{1}{N}$

$P(X_{vehicle}) = \sum_{i=1}^{N} w_k^i \left(X_{k|k-1}^i - X_{vehicle}\right)\left(X_{k|k-1}^i - X_{vehicle}\right)^T$

If (Extero data)

Calculate / update the weight of each particle :

$w_k^i \propto w_{k-1}^i \cdot p(Y_k|X_{k|k-1}^i) = w_{k-1}^i \cdot \frac{1}{\sqrt{(2\pi)|R_k|}} \exp\{-\frac{1}{2}(Y_k - \hat{Y}_k)^T [R_k]^{-1}(Y_k - \hat{Y}_k)\}$.

Normalize weights $w_k^i = \frac{w_k^i}{\sum_{j=1}^{N} w_k^j}$.

Determine G_b : the best normalized weight

PSO evolution :

$\hat{v}_k^i = W.\hat{v}_{k-1}^i + |randn|(G_b - \hat{x}_{k|k-1}^i)$

$\hat{x}_{k|k}^i = \hat{x}_{k|k-1}^i + \hat{v}_k^i$

Calculate the estimated state of the system and its uncertainty:

$X_{vehicle} = \sum_{i=1}^{N} X_{k|k}^i . w_k^i$

$P(X_{vehicle}) = \sum_{i=1}^{N} w_k^i \left(X_{k|k}^i - X_{vehicle}\right)\left(X_{k|k}^i - X_{vehicle}\right)^T$

Calculate the resampling factor:

$N_{eff} = \frac{1}{\sum_{j=1}^{N} (w_k^i)^2}$

If ($N_{eff} \leq N_{th}$**)**

Proceed resampling particles according to the chosen method.

Redistribute weights to new particles.

end

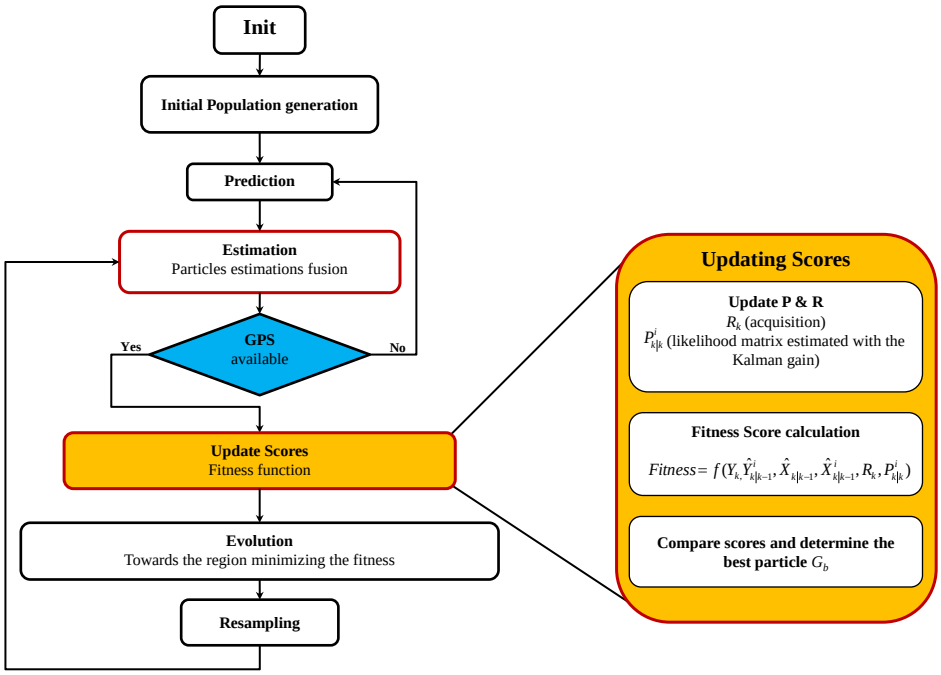

Figure 4: OKPS algorithm

The idea is performed by enhancing particles with a probability matrix P^i (Inspired from the uncertainty matrix P of the EKF method) allowing each particle to evaluate its likelihood. The P^i matrix has to be as representative as possible of the particle positioning uncertainty. It is the reason why this matrix is managed (updated at each step) by an Extended Kalman filter gain K. After an initialization step, the OKPS filter performs the localization following the logical step of prediction and update integrating the new particles and swarm features. The OKPS algorithm can be considered as an evolved SPF hybrid approach aided by a Kalman gain to reassess the particles likelihood used in an adaptive fitness function.

The OKPS algorithm is described in the flowchart presented in Figure 4 and the implementation details are given in the following:

3.1.1. Initialization

OKPS particles, in addition to all SPF particles attributes, are fitted with a probability matrix P^i_k. This matrix is initialized as an EKF variance confidence matrix.

The parameters to be fixed are the number of particles N, the inertia weight W and the resampling factor α. The initial collected data are used to define the initial position, calculated with an initial sensors data set giving $\hat{X}_0$, P_0, Q_0 and R_0 values (calculation described in Equation 1). The swarm is initialized following the initialization formulation noted in 2.3.

The initial OKPS particles attributes are: $\overrightarrow{x_i}(0)$ which represents a positioning vector $X_{init} = [x_{init}\ y_{init}\ \theta_{init}]^T$, $\overrightarrow{v_i}(0) = 0$ represents the initial value of the particle speed of evolution, $P^i_0 = P_0$ the particle initial uncertainty matrix. The particles initial (weights) fitness score value is noted $w^i_0 = 1/N$, it will be updated for every available GPS data using Equation (13), and $G^i_b(0)$ gives the global best among neighborhood initial solutions.

3.1.2. Prediction

Particles predict the vehicle state using proprioceptive data and the bicycle vehicle model. In addition to the state prediction, a prediction likelihood is calculated with $P^i_{k|k-1}$ according to the EKF prediction Equation (3).

A predicted vehicle state vector can be determined by the fusion of the predicted particles states vectors following the Equation (11).

$$\hat{X}_{k|k-1} = \sum_{i=1}^{N} \hat{X}^i_{k|k-1} * w^i_{k|k-1} \tag{11}$$

3.1.3. Updating Scores

Before the scores evaluation, the particle uncertainty $P^i_{k|k-1}$ and the measure uncertainty R_k matrices are updated. The first one is updated using a Kalman gain to be as representative as possible of the particle i uncertainty, see Equation (12). The second uncertainty matrix R_k is updated by the GPS quality data acquisition.

The updated $P^i_{k|k}$ is made according to Equation (12).

$$\begin{aligned} K^i_k &= P^i_{k|k-1} H^{iT}_k [H^i_k P^i_{k|k-1} H^{iT}_k + R_k]^{-1} \\ P^i_{k|k} &= (I - K^i_k H^i_k) P^i_{k|k-1} \end{aligned} \tag{12}$$

Then, the score for each particle is calculated with the Fitness function. The Fitness function is a minimization or maximization criterion representing one or a compromise of multiple goals, the selection of this function depends on the application and the desired result [22, 19, 35].

The OKPS fitness Function (13) is a maximization criterion. The calculated fitness score considers two information sources: the GPS corrective source and the particle prediction source. The compromise between these two sources will be done by a weighted average of their respective quadratic errors relatively to their respective uncertainties.

$$w^i_k = exp\{-\frac{1}{2}[(Y_k - \hat{Y}^i_k)^T [R_k]^{-1} (Y_k - \hat{Y}^i_k) + (\hat{X}_{k|k-1} - \hat{X}^i_{k|k-1})^T [P^i_{k|k}]^{-1} (\hat{X}_{k|k-1} - \hat{X}^i_{k|k-1})]\} \tag{13}$$

$\hat{Y}^i_k = H_k \hat{X}_{k|k-1} = [\ X \quad Y\ ]^T$ is the observation (predicted measure). $Y_k = [\ GPS_x \quad GPS_y\ ]^T$ is the GPS measure, R_k represents the GPS uncertainty and $P^i_{k|k}$ the updated estimation uncertainty. $\hat{X}_{k|k-1}$ is the predicted vehicle state and The G_b is determined by comparing the particles scores. The best particle G_b is the particle with highest score after scores normalization.

3.1.4. Evolution

In this step, each particle will optimize its estimation by evolving in the search space. This evolution is done by following Equation (10) and applying the Gaussian PSO motion principle [33] adapted to a dynamic environment [34]. The particle merges its inertia and its attraction to the G_b in order to move toward a region of interest. This will give a new updated position $\hat{X}^i_{k|k}$ of each particle. Thus, the swarm distribution will be optimized and concentrated on the region maximizing the fitness values. The benefit of this evolution compared to the SPF evolution is that the G_b is defined with an adaptive fitness function. Instead of SPF particles weighting which is only relative to the GPS position, this new adaptive weighting allows inertial behavior avoiding particles sticking to GPS. This adaptive evolution make the filter more robust to GPS outliers giving more smooth positioning.

3.1.5. Estimation

The particles results after evolution are fused in order to have a new ego-vehicle position estimation $\hat{X}_{k|k}$. Particles estimations $\hat{X}^i_{k|k}$ are fused using standardized weights which are the updated standardized fitness scores. The fusion for the global estimation is made with the following equation.

$$\hat{X}_{k|k} = \sum_{i=1}^{N} \hat{X}^i_{k|k} * w^i_k$$

3.1.6. Resampling

The resampling algorithm and criterion are the same used for the previously described approaches, the algorithm is the systematic one and the criterion is the Kong effective particles number.

In the same conditions of resampling threshold N_{th} and processed data, the resampling triggering for OKPS occurs less than for the SPF because of the adaptive optimization that makes the particles more reactive and consequently rises the number of effective particles. The resampling process is still a heavy computational process and avoiding useless resampling is advantageous. Moreover, the less the particle population is resampled the more the estimation dynamic characteristics are preserved.

4. Results Analysis

In this section, the filters: EKF, PF, SPF and OKPS are tested in an ego-vehicle localization application. Different scenarios with different data qualities and noises conditions are studied. The filters will be ranked in terms of accuracy and robustness. The main criteria of comparison are: the Root Mean Square Error (RMSE), the Average Euclidean Error (AEE) and the Geometric Average Error (GAE) for the category of average errors. The RMSE is the mostly used natural approximation of the mean standard deviation of the differences between predicted values and observed values (estimation error). RMSE represents a good measure of accuracy. However, as it has not a simple physical interpretation and is scale-dependent. It is generally used for probabilistic analysis. Taken from the Euclidean distance concept, the AEE represents the average of instantaneous errors $E\left[\left\|\widetilde{X}\right\|^2\right] = E\left[\sqrt{\widetilde{X}^t\widetilde{X}}\right]$while the RMSE is an approximation of the standard error $\sqrt{E(\widetilde{X}^t\widetilde{X})}$. The RMSE and AEE are analysis criteria of average errors based on the concept of the arithmetic mean. The disadvantage of this concept is that it is very sensitive to large outliers. To overcome this criteria weakness, the Geometric Average Error (GAE) is more robust to large outliers. Theoretically, the GAE is never greater than the AEE which is never even higher than the RMSE value ($GAE \leq AEE \leq RMSE$). Thus, the impact of important errors is larger on RMSE and smaller on GAE. The main interest of using GAE is to be less influenced by the large errors values and remaining close to the AEE value. $GAE = AEE$ only when the number of samples is limited to one. Moreover, the gap between the GAE, AEE and RMSE can be used to detect the presence of temporary high outliers. The more this three average criteria are close the more the mistakes were homogeneous (no error peaks). When $GAE = AEE = RMSE$, there has been no errors fluctuation and the error can be considered as a bias. Then, this bias can be counterbalanced.

In order to evaluate the instantaneous filters performance, complementary criteria to the average errors are taken into consideration, such as the instantaneous Euclidean Error (EE), axial errors, mean axial errors and mean axial standard deviations. To also evaluate filters robustness and sensitivity, the relationship between the average and instantaneous criteria is used.

To perform an even-handed experimental comparison, real word data are collected during a driving scenario on the urban area of the Satory test track (see Figure 5). Data collecting is done using a vehicle of the IFSTTAR/LIVIC laboratory equipped with embedded sensors and dedicated software. The vehicle is equipped with a steering wheel angle encoder, a gyrometer, an odometer and a low cost $AG132$ GPS running in degraded mode. Using a real data base is the best way to guarantee a real positioning tests (Data, conditions and noise similarity). The tests are then stated in the same conditions with the same data and with the same computational resources. For multi-hypothesis approaches (particular methods), the particles number is fixed to 500. The effectiveness threshold $N_{th}$for particles resampling is set to a minimum of 50% of effective particles. As said before, to avoid the SPF divergence caused by the swarm concentration around the GPS data, a percentage of communicative particles is attributed to this filter. The communicative-evolutionary SPF particles will then represent 10% of the swarm. For OKPS and SPF evolving particles, the inertia weight W is set to 0.2 to guarantee a minimum consideration of the vehicle inertia. The reference for positioning errors calculation is the centimetric differential GPS RTK . Tests are carried out in the same scenario with three different conditions. The vehicle is driven from right to left along the course delimited by flags on the Satory test track (Figure 5).

The filters performance shown by axial errors and Euclidean error graphics describe the filters instantaneous behavior at each step of the test. The tables of mean axial errors and standard deviations resume the global performance

Algorithm 4: Algorithmof the OKPS filter applied to the vehicle ego-localization

At time $k = 0$ **(**X_0 **et** P_0 **known)**

Initialization()

Generating the first population of particles:

N particles randomly distributed around the initial position with:

$W_0^i = \frac{1}{N}$

$v_0^i = 0$

$P_0^i = P_0 = E[(X_0 - \hat{X}_0)(X_0 - \hat{X}_0)^T]$

while *(Sensors data available)* **do**

If(Proprio data)

Move particles in the search space according to the data and the vehicle model.

$\hat{x}_{k|k-1}^i = f\,(\hat{x}_{k-1|k-1}^i, U_{k-1})$

$P_{k|k-1}^i = A_{xk} P_{k-1|k-1}^i A_{xk}^T + Q_k$

Retain $W_0^i = \frac{1}{N}$.

Calculate the predicted state and its uncertainty:

$\hat{X}_{k|k-1} = \sum_{i=1}^N \hat{x}_{k|k-1}^i . w_k^i$

$P(\hat{X}_{k|k-1}) = \sum_{i=1}^N w_k^i \left(X_{k|k-1}^i - \hat{X}_{k|k-1}\right)\left(X_{k|k-1}^i - \hat{X}_{k|k-1}\right)^T$

If (Extero data)

Calculate/update the weight of each particle:

R_k by acquisition

$K_k^i = P_{k|k-1}^i H_k^{iT} [H_k^i P_{k|k-1}^i H_k^{iT} + R_k]^{-1}$

$P_{k|k}^i = (I - K_k^i H_k^i) P_{k|k-1}^i$

Calculate/update the weight of each particle:

$w_k^i = exp\{-\frac{1}{2}[(Y_k - \hat{Y}_k^i)^T [R_k]^{-1} (Y_k - \hat{Y}_k^i)$
$+ (\hat{X}_{k|k-1} - \hat{X}_{k|k-1}^i)^T [P_{k|k}^i]^{-1} (\hat{X}_{k|k-1} - \hat{X}_{k|k-1}^i)]\}$

Normalize weights $w_k^i = \frac{w_k^i}{\sum_{j=1}^N w_k^j}$.

Determine G_b : the best normalized weight

PSO evolution:

$\hat{v}_k^i = W.\hat{v}_{k-1}^i + |randn|(G_b - \hat{x}_{k|k-1}^i)$

$\hat{x}_{k|k}^i = \hat{x}_{k|k-1}^i + \hat{v}_k^i$

Calculate the estimated state of the system and its uncertainty:

$\hat{X}_{k|k-1} = \sum_{i=1}^N X_{k|k}^i . w_k^i$

$P(\hat{X}_{k|k-1}) = \sum_{i=1}^N w_k^i \left(X_{k|k}^i - \hat{X}_{k|k-1}\right)\left(X_{k|k}^i - \hat{X}_{k|k-1}\right)^T$

Calculate the resampling factor:

$N_{eff} = \frac{1}{\sum_{j=1}^N (w_k^i)^2}$

If ($N_{eff} \leq N_{th}$**)**

Proceed resampling particles according to the chosen method.

Redistribute weights to new particles.

end

Figure 5: Satory Test Track-Urban Area

by mean values giving an idea of the average performance. The accuracy will be stated by analyzing the average errors and the robustness will be an analysis of the axial standard deviation relation with axial errors.

The first test is done using the synchronized data base. As data are synchronized, the data of slower sensors are interpolated. The localization process will perform both prediction and correction at each time step. This kind of situation are possible when the sensors are fast or when the vehicle moves slowly (parking maneuver). The problem with this data base is that the full data availability makes the filters very confident and optimist. The uncertainty ellipses and values will be smaller than normal and the challenge is to be capable of detecting outliers in spite of the confidence in the measurement. This data base is characterized by a bias and a high level of noise caused by the synchronization step. The filters have to be robust and precise at the same time. As said before, the difficulty is to filter outliers and not to be fully optimistic about data quality.

test AG132	EKF	PF	SPF	OKPS	GPS
$RMSE$	4.37	4.37	4.35	3.89	5.87
AEE	4.28	4.27	4.28	3.76	5.29
GAE	4.18	4.16	4.22	3.60	4.38

Table 1: RMSE, AEE and GAE final values for the synchronized AG132 test

The results of the AG132 synchronous test are shown in Figure 6 and tables 1 and 2. The Figure 6 describes according to the subfigures from left to right and from top to bottom the following criteria : the Euclidean Error, the RMSE, the AEE, the GAE, the longitudinal X axial error and the lateral Y axial error. The Table 2 gives the axial mean errors for each filter in comparison with the GPS one and gives also the axial standard deviations. The Table 1 compares the average errors rating the errors fluctuation using the gap between the GAE, AEE and RMSE. As noted in Table 1 the gap between the average errors criteria is significant which supports mathematically the presence of fluctuations (presence of temporary high outliers). The variations are also visible in the GPS Euclidean error results. The global results show that all the filters correct the errors and perform an acceptable ego-vehicle positioning. However, the filters remain more attracted by the biased GPS data. Here the OKPS outperforms the other filters by always keeping particles reactive and cooperative. To achieve this result, the OKPS gives an adequate likelihood value to each particle at each time step using the adaptive fitness function. Then, these values of particles likelihood (scores) are used to give a global estimation value for the ego-vehicle localization. The OKPS is finally

test AG132	EKF	PF	SPF	OKPS	GPS
X_{mean}	-1.87	1.82	-.191	-1.77	-2.86
σ_X	0.49	0.52	0.37	0.71	2.48
Y_{mean}	3.82	3.83	3.79	3.22	4.01
σ_Y	0.90	0.92	0.82	1.06	2.01

Table 2: Axis Mean Errors and standard deviations (m) for the synchronized AG132 test

test AG132	EKF	PF	SPF	OKPS	GPS
X_{mean}	-1.26	-1.29	-1.54	-1.30	-6.66
σ_X	1.43	1.50	1.38	1.28	8.96
Y_{mean}	3.76	3.88	3.89	3.66	4.82
σ_Y	0.69	0.66	0.63	0.78	6.41

Table 3: Axis Mean Errors and standard deviations (m) for AG132 Asynchronous data test

test AG132	EKF	PF	SPF	OKPS	GPS
$RMSE$	4.28	4.40	4.45	4.17	13.72
AEE	4.19	4.33	4.38	4.08	13.13
GAE	4.10	4.25	4.31	3.98	12.46

Table 4: RMSE, AEE and GAE final values for AG132 Asynchronous data test

test AG132	EKF	PF	SPF	OKPS	GPS
X_{mean}	-1.44	-1.43	-0.65	-1.12	-7.67
σ_X	1.45	1.28	2.26	1.03	8.79
Y_{mean}	4.18	4.13	4.56	2.44	5.43
σ_Y	1.51	1.49	2.33	2.08	9.11

Table 5: Axis Mean Errors and standard deviations (m) for AG132 Asynchronous data with multipaths

test AG132	EKF	PF	SPF	OKPS	GPS
$RMSE$	4.89	4.79	5.63	3.55	15.73
AEE	4.64	4.56	5.12	3.20	14.70
GAE	4.43	4.37	4.70	2.91	13.69

Table 6: RMSE, AEE and GAE final values for AG132 Asynchronous data with multipaths

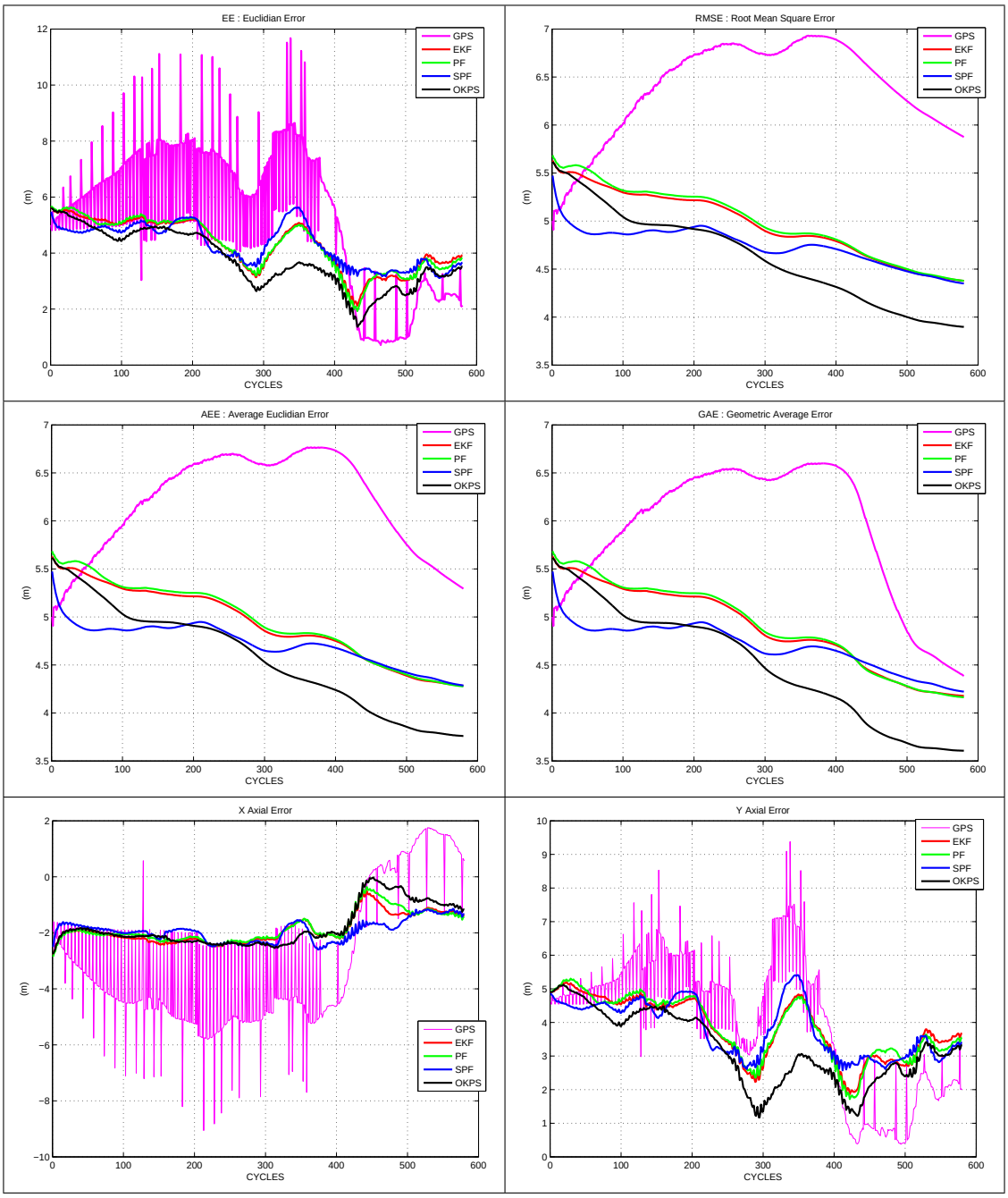

Figure 6: AG132 Synchronized data test

more accurate and more robust to strong noises and bias. The filters axial standard deviations noted in Table 2 show that the OKPS did not become more optimistic than necessary which allows it to filter more effectively GPS outliers. According to the axial mean errors and standard deviation relationship the OKPS is $11,7\%$ more accurate than the EKF and $11,4\%$ than the PF, the OKPS is also 7.8% more robust than the EKF and 6.8% less sensitive than the PF.

The OKPS merges both information from prediction and correction, the adaptive weighting mechanism (fitness) and the particles self-diagnose (uncertainty matrix) allow to get the best estimation of the vehicle position. However, these results which are obtained by a GPS information penalization because of their strongly degraded and noisy

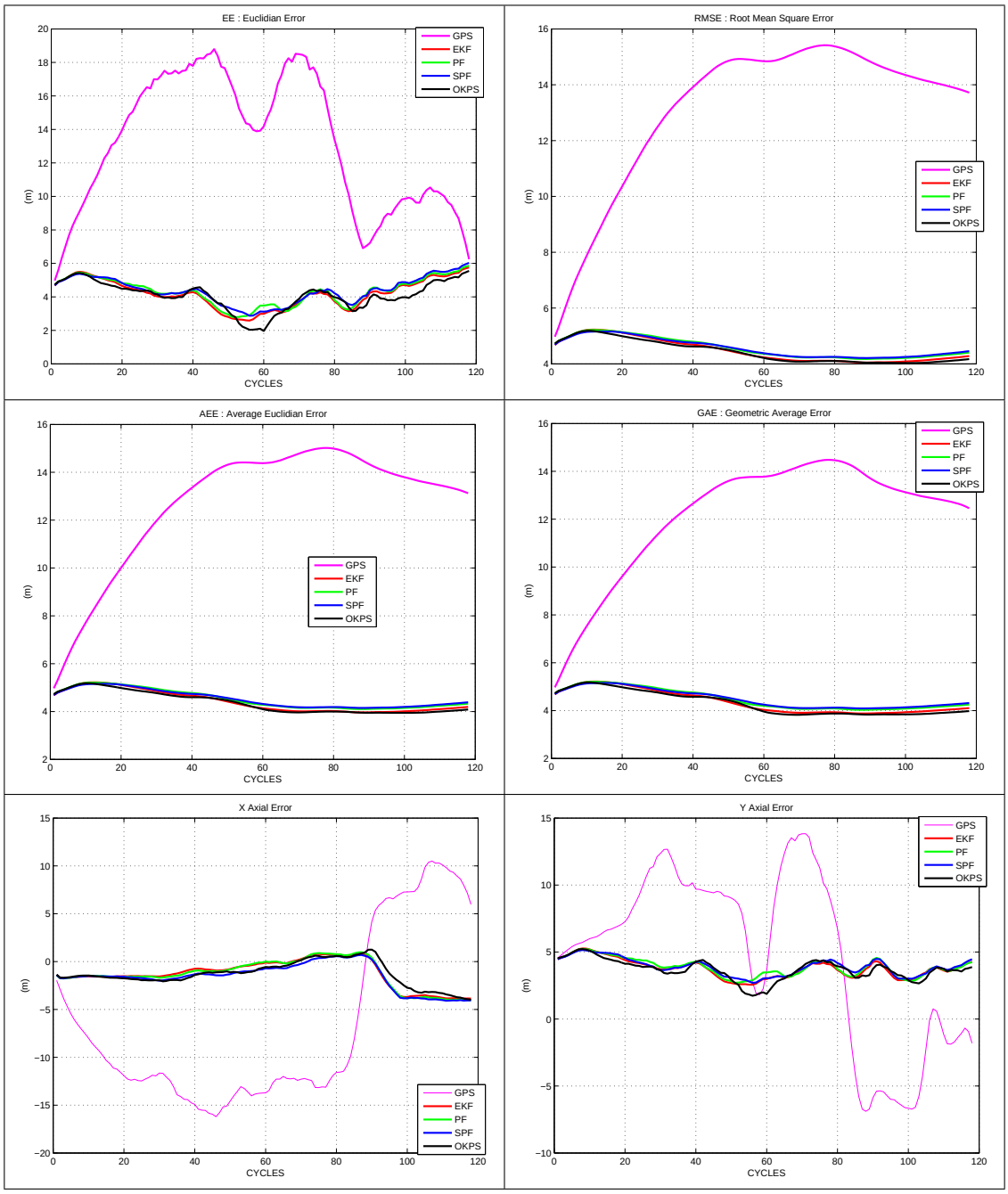

Figure 7: AG132 Asynchronous data test

character, seems to hide an OKPS divergence.

To test our filter consistency and to ensure its integrity, further tests are carried out. First, the sensors data will be taken in their raw version (no signal pre-processing and no synchronization). This test will give an idea about the approaches performance in a localization application for a standard vehicle equipped with low cost sensors. After that, a GPS disturbance is generated and included to the GPS measures. The generated GPS degradation simulates the effect of GPS multi-reflexion in urban canyons. This final test will test the ego-vehicle localization performance for an urban driving scenario.

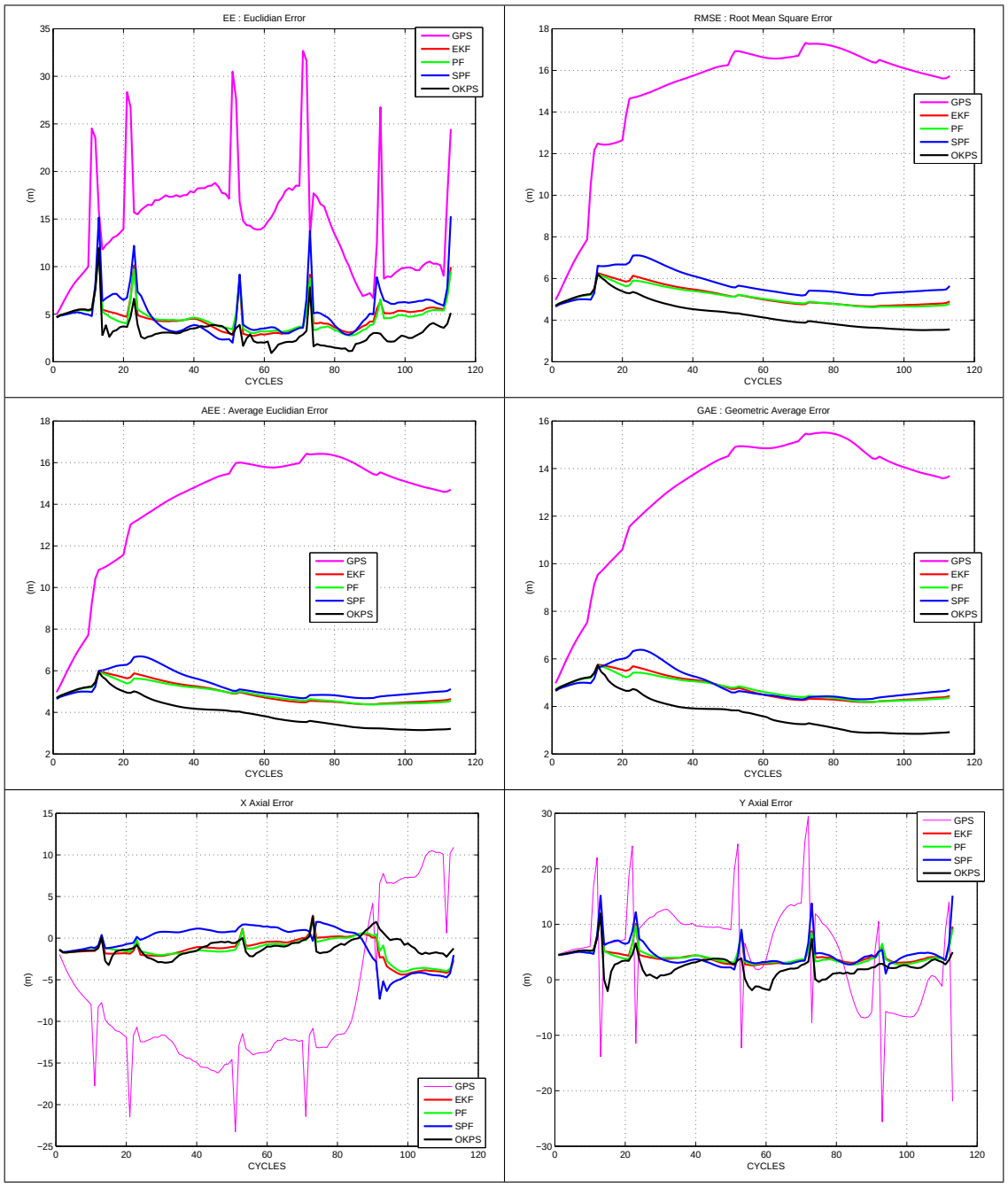

Figure 8: AG132 Asynchronous data with multipaths test

The results of the second test done with the raw asynchronous GPS data are shown in Figure 7, Table 3 and 4. The graphics shows that the OKPS is still performing good ego-vehicle localization in normal signal conditions. This result ensures the OKPS efficiency for different scenarios and supports the OKPS integrity. From axial errors analysis and standard deviations, we can say that the OKPS is 3.6% more accurate than the SPF and 1.2% more robust than the PF. For this test, the SPF an PF particles have the same behavior, the SPF outperforms the PF because of the SPF evolutionary capacity (optimization) which is not possible for the PF particles.

In order to test the filters reactivity and sensitivity in another situation, this third and last test is carried out. The

results of the urban canyon driving scenario are synthesized in the Figure 8, Table 5 and 6. The OKPS is designed to deal with this kind of situations, that is why it performs the best localization results. In this test, the GPS data are punctually disturbed by a multipath noise. These disturbances are shown by the peaks in graphics, especially in instantaneous criteria such as Euclidean Error and axial errors. The filters estimations are almost influenced by the peaks but the OKPS remains the less sensitive one. In a city with urban canyons where GPS systems generally suffers from multipath, multireflexion and outage problems, the OKPS will be the most appropriate localization approach. It is 18.4% more accurate than the EKF and 17.9% more than the PF. It is also 7.3% more robust than the SPF to multipath. The gap between RMSE, AEE, and GAE is more important than for the two previous tests which confirms the presence of important punctual disturbances.

The OKPS outcompetes the other approaches especially in signal multireflexion cases. It also remains less sensitive to the GPS positioning outliers and vehicle dynamic changes than the EKF, PF and SPF filters. The OKPS performs a better positioning with a higher accuracy in different signal and driving situations. These tests conclude that the OKPS is better overall the three scenarios and overtakes the other filters especially in case of GPS multipaths and sensors data disturbance.

5. Conclusions

This paper shows the Optimized Kalman Particle Swarm theoretical formulation and experimental performance. It highlights the cooperative reactive aspect of the OKPS which performs accurate ego-vehicle localization in degraded conditions (noises and multireflexions). Our OKPS fits the particles with an uncertainty matrix. The covariance uncertainty matrix represents the capacity of auto-diagnose of the particles which is incorporated to the adaptive weighting system (Fitness function). Thanks to the added covariance matrix, the particles of the OKPS become more reactive to abrupt dynamic changes and more robust to noises.

The advantage of the OKPS positioning is stated during a driving scenario test. The OKPS outperforms the other filters using its reactivity and cooperative particles. More explicitly, the adaptive multi-objective fitness function allows the swarm to evolve to high scores regions. Each particle merge its self-diagnose with the GPS data. The described cooperative process makes the OKPS effective in high dynamic on-road ego-vehicle localization applications. The OKPS performs the best ego-vehicle positioning especially for the urban driving scenario with GPS multipaths: it out-competes the EKF, PF and SPF for ego-vehicle localization application. Even though the OKPS is more computationally complex and more time consuming, its promising results make it one of the most suitable localization methods. The OKPS needs less tuning parameters than the metaheuristic hybrid localization approaches. An auto-attraction-repulsion mechanism insures the swarm homogeneity, diversification and effectiveness. This mechanism prevents the swarm premature convergence for a full connected neighborhood topology.

In future works, the OKPS will be tested in more diverse driving scenarios (for example stop and go, parking and strong braking and acceleration scenarios) with additional sensors which will improve the OKPS localization accuracy and integrity. This approach will then be tested in comparison with the Interacting Multi-Model Filter developed by the LIVIC laboratory for autonomous vehicle localization applications. Next, we intend to add nice properties of the IMM in the OKPS.

References

[1] E. Seignez, A. Lambert, T. Maurin, Autonomous parking carrier for intelligent vehicle, in: IEEE International Conference on Intelligent Vehicle, 2005, pp. 411–416.

[2] A. Ndjeng Ndjeng, A. Lambert, D. Gruyer, S. Glaser, Experimental comparison of kalman filters for vehicle localization, IEEE Intelligent Vehicles Symposium (2009) 441–446.

[3] P. S. Maybeck, Stochastic Models, Estimation and Control, Academic Press, New York, 1982.

[4] F. L. Lewis, Optimal Estimation : with an introduction to stochastic control theory, John Wiley & Sons, New York, 1986.

[5] R. E. Kalman, A new approach to linear filtering and prediction system, Transactions of the ASME-Journal of Basic Engineering 82 (D) (1960) 35–45.

[6] S. J. Julier, J. Uhlmann, A new extension of the kalman filter to nonlinear systems, in: International Symposium on Aerospace/Defense Sensing, Simulation and Controls, Orlando, Florida, 1997, pp. 182–193.

[7] K. Ito, K. Xiong, Gaussian filters for nonlinear filtering problems, IEEE Transaction on Automatic Control 45 (5) (2000) 910–927.

[8] S. J. Julier, J. Uhlmann, A general method for approximating nonlinear transformation of probability distribution, Tech. rep., University of Oxford (1996).

[9] T. Lefebvre, H. Bruyninckx, J. De Schutter, Kalman filters for non-linear systems: a comparison of performance, International Journal of Control 77 (7) (2004) 639–653.
[10] B. Mourllion, D. Gruyer, A. Lambert, S. Glaser, Kalman filters comparison for vehicle localization data alignment, in: IEEE/RSJ International Conference on Advanced Robotics, 2005, pp. 178 – 185.
[11] J. Ali, M. R. Ullah Baig Mirza, Performance comparison among some nonlinear filters for a low cost sins/gps integrated solution, Nonlinear Dynamics (2010) 1–12.
[12] R. Kandepu, B. Foss, L. Imsland, Applying the unscented kalman filter for nonlinear state estimation, Journal of Process Control 18 (7-8) (2008) 753–768.
[13] R. Havangi, M. A. Nekoui, M. Teshnehlab, A multi swarm particle filter for mobile robot localization, International Journal of Computer Science 7 (3) (2010) 15–22.
[14] J. Godoy, D. Gruyer, A. Lambert, J. Villagra, Development of an particle swarm algorithm for vehicle localization, in: IEEE Intelligent Vehicles Symposium, 2012, pp. 1114–1119.
[15] L. Bazzani, D. Bloisi, V. Murino, A comparison of multi hypothesis kalman filter and particle filter for multi-target tracking, in: Performance Evaluation of Tracking and Surveillance workshop at CVPR, 2009, pp. 47–54.
[16] S. Arulampalam, S. Maskell, N. Gordon, T. Clapp, A tutorial on particle filters for online non-linear/non-gaussian bayesian tracking, IEEE Transactions on Signal Processing 50 (2) (2002) 174–188.
[17] J. D. Hol, T. B. Schön, F. Gustafsson, On resampling algorithms for particle filters, in: Nonlinear Statistical Signal Processing Workshop, 2006.
[18] Y. Shi, R. Eberhart, A modified particle swarm optimizer, in: IEEE International Conference on Evolutionary Computation, 1998, pp. 69–73.
[19] M. Reyes-Sierra, C. A. C. Coello, Multi-objective particle swarm optimizers: A survey of the state-of-the-art, International Journal of Computational Intelligence Research 2 (3).
[20] C. A. Coello Coello, M. S. Lechuga, Mopso: a proposal for multiple objective particle swarm optimization, in: IEEE Congress on Evolutionary Computation, Vol. 02, 2002, pp. 1051–1056.
[21] A. Banks, J. Vincent, C. Anyakoha, A review of particle swarm optimization. part i: background and development, Springer Science, Business Media.
[22] S. C. Esquivel, C. A. C. Coello, On the use of particle swarm optimization with multimodal functions, IEEE Congress on Evolutionary Computation (2003) 1130–1136.
[23] D.-J. Jwo, S.-C. Chang, Particle swarm optimization for gps navigation kalman filter adaptation, Aircraft Engineering and Aerospace Technology 81 Iss: 4 (2009) 343 – 352.
[24] J. Zhang, T.-S. Pan, J.-S. Pan, A parallel hybrid evolutionary particle filter for nonlinear state estimation, in: Robot, Vision and Signal Processing (RVSP), 2011 First International Conference on, 2011, pp. 308–312. doi:10.1109/RVSP.2011.77.
[25] G. Tong, Z. Fang, X. Xu, A particle swarm optimized particle filter for nonlinear system state estimation, in: IEEE Congress on Evolutionary Computation, CEC 2006., IEEE, 2006, pp. 438–442.
[26] J. Zhang, T.-S. Pan, J.-S. Pan, A parallel hybrid evolutionary particle filter for nonlinear state estimation, in: Proceedings of the 2011 First International Conference on Robot, Vision and Signal Processing, RVSP '11, IEEE Computer Society, Washington, DC, USA, 2011, pp. 308–312. doi:10.1109/RVSP.2011.77.
URL http://dx.doi.org/10.1109/RVSP.2011.77
[27] Z. Fang, G.-f. Tong, X.-h. Xu, Particle swarm optimized particle filter 22 (3) (2007) 273.
[28] Z.-m. CHEN, Y.-m. BO, P.-l. WU, Q.-x. CHEN, A new hybrid algorithm for particle filtering and its application to radar target tracking, Acta Armamentarii 1 (2012) 013.
[29] A. Ahmed Bacha, D. Gruyer, S. Mammar, A new robust cooperative-reactive filter for vehicle localization: The extended kalman particle swarm, in: Intelligent Vehicles Symposium (IV), IV 2013 IEEE, 2013, pp. 195–200. doi:10.1109/IVS.2013.6629470.
[30] A. Ahmed Bacha, D. Gruyer, A. Lambert, A robust hybrid multisource data fusion approach for vehicle localization, Scientific Research Publish (SCIRP) Positioning 4 (4) (2013) 271–281. doi:10.4236/pos.2013.44027.
[31] A. Ahmed Bacha, D. Gruyer, A. Lambert, A performance test for a new reactive-cooperative filter in an ego-vehicle localization application, in: Intelligent Vehicles Symposium (IV), IV 2014 IEEE, 2014. doi:10.1109/IVS.2013.6629470.
[32] M. Clerc, J. Kennedy, The particle swarm - explosion, stability, and convergence in a multidimensional complex space, Evolutionary Computation, IEEE Transactions on 6 (1) (2002) 58–73.
[33] R. A. Krohling, Gaussian swarm: a novel particle swarm optimization algorithm, in: IEEE Conference on Cybernetics and Intelligent Systems, Vol. 1, 2004.
[34] R. E. Xiaohui Hu, Yuhui Shi, Recent advances in particle swarm, Congress on Evolutionary Computation 1.
[35] V. D. B. Frans, An analysis of particle swarm optimizers, Ph.D. thesis, University of Pretoria, South Africa (2002).
[36] X. Liang, W. Li, Y. Zhang, Y. Zhong, M. Zhou, Recent advances in particle swarm optimization via population structuring and individual behavior control, in: Networking, Sensing and Control (ICNSC), 2013 10th IEEE International Conference on, 2013, pp. 503–508. doi:10.1109/ICNSC.2013.6548790.