ADAPTING PARTICLE FILTER ON INTERVAL DATA FOR DYNAMIC STATE ESTIMATION

Fahed Abdallah, Amadou Gning and Philippe Bonnifait

Lab. HEUDIASYC, UMR CNRS 6599. Université de Technologie de Compiègne
B.P. 20529, 60205 Compiègne Cedex, FRANCE. Tel: +33.3.44.23.44.85 fax: +33.3.44.23.44.77
fahed.abdallah@hds.utc.fr, gningelh@hds.utc.fr, philippe.bonnifait@hds.utc.fr

ABSTRACT

Over the last years, Particle Filters (PF) have attracted considerable attention in the field of nonlinear state estimation due to their relaxation of the linear and Gaussian restrictions in the state space model. However, for some applications, PF are not adapted for a real-time implementation. In this paper we propose a new method, called Box Particle Filter (BPF), for dynamic nonlinear state estimation, which is based on particle filters and interval frameworks and which is well adapted for real time applications. Interval framework will allow to explain regions with high likelihood by a small number of box particles instead of a large number of particles in the case of PF. Experiments on real data for global localization of a vehicle show the usefulness and the efficiency of the proposed approach.

Keywords: State estimation, Monte Carlo methods, Mobile robots, Multisensor systems, Interval analysis.

1. INTRODUCTION

Bayesian methods provide an attractive framework for dynamic state estimation problems [2]. The Bayesian approach consist in constructing the probability density function (PDF) of the state vector based on a probabilistic formulation of all available information. A particle filter is a sequential Monte Carlo Bayesian estimator of the PDF of the state using weighted particles. The efficiency and accuracy of the filter depend mostly on the number of particles used in the estimation, and on the propagation function used to re-allocate weights to these particles at each iteration. If the imprecision, i.e. bias and noise, in the available information is high, the number of particles should be very large in order to obtain good performances. This may induce complexity problems for a real-time implementation. Several works try to combine approaches in order to overcome these shortcomings (see for example [5] and references therein). Other works use statistical approaches to increase the efficiency of particle filters by adapting the size of sample sets during the estimation process [3].

If only the maximum error on the available data is known, it will be judicious to model these errors by their maximum bound using intervals. Interval methods represent a relatively new research direction in some applications of signal processing. Though, in the closely related field of controls there has been much work that can also be applied to signal processing. In this paper, we propose an extension of the particle filter algorithm by dealing with interval data and by using interval analysis and Constraints satisfaction techniques [4]. In usual particle filtering, particles are punctual states associated with weights depending of the likelihood which is defined by a statistical model of the observation error. In the box particle filter (BPF), particles are boxes associated with weights whose likelihood is defined by a bounded model of the observation error. The algorithm of the BPF is presented in Section 4.

2. BAYESIAN FRAMEWORK AND PARTICLE FILTERS

The state space model can be described as:

$$\begin{cases} x_{k+1} = & f(x_k, u_k, v_k) \\ y_k = & g(x_k, w_k) \end{cases} \tag{1}$$

where $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_v} \longrightarrow \mathbb{R}^{n_x}$ is a possibly non-linear function defining the state at time $k+1$ from the previous state at time k, the input u_k and an independent identically distributed process noise sequence $v_k, k \in \mathbb{N}$. We note by n_x, n_u and n_v, respectively, the dimensions of the state, the input and process noise vectors. The function $g : \mathbb{R}^{n_x} \times \mathbb{R}^{n_w} \longrightarrow \mathbb{R}^{n_y}$ is a possibly non-linear function defining the relation between the state and the measurement at time k, $w_k, k \in \mathbb{N}$ is an i.i.d measurement noise sequence. n_y, n_w are dimensions of the measurement and measurement noise vectors, respectively. The states and the measurements up to time k will be represented by $X_k = \{x_i, i = 1, \cdots, k\}$ and $Y_k = \{y_i, i = 1, \cdots, k\}$, respectively.

From a Bayesian point of view to recursive filtering, given the measurements Y_k, everything worth knowing about the state at time k is given by the conditional probability density $p(X_k|Y_k)$, which is called the posterior density. The posterior constitutes the complete solution to the sequential estimation problem. In many real applications, one has to estimate only the filtering density $p(x_k|Y_k)$ which is a marginal of the posterior density function $p(X_k|Y_k)$. The filtering density gives the belief in the state x_k at time k, taking different values, given the measurements $Y_k = \{y_i, i = 1, \cdots, k\}$. Knowing the filtering density $p(x_k|Y_k)$, one can easily calculate various estimates of the system's state like means, modes, confidence intervals.

Particle filter methods, are the class of simulation filters which recursively approximate the filtering density $p(x_k|Y_k)$ by the cloud of N discrete *particles* with probability mass, or weight, assigned to each of them. Hence, a possibly continuous probability density function is approximated with a discrete one. The sketch of a particle filter algorithm is as follows. Initially, all particles have equivalent weight attached to them. To progress to the next time instance, two steps are performed in sequence. First, at the prediction step, the state of every particle is updated according to the motion model. An accurate dynamical model is essential for robust properties of the algorithm. Next, during the measurement step, new information that became available about the system is used to adjust the particle weights. The weight is set to be the likelihood of each particle state describing the true current state of the system. This can be computed, via bayesian inference, to be proportional to the probability of the observed measurements given the particle state (assuming all object states are equiprobable). The sample states are then redistributed to obtain uniform weighting for the next algorithm iteration by resampling them from the computed posterior probability distribution. At any time, some characteristics (position, speed etc.) can be directly computed, if desired, by using the particle set and weights as an approximation of the true probability density function. For more details on the particle filter algorithm, the reader can refer to [5] and references therein.

3. INTERVAL ANALYSIS BACKGROUND

In this section we briefly present interval analysis and we introduce some useful definitions very famous in the field of *constraints propagation techniques* (or *consistency techniques*) [6].

A real interval, denoted $[x]$, is defined as a closed and connected subset of $\mathbb{R}$, and a box $[\mathbf{x}]$ of $\mathbb{R}^{n_x}$ as a cartesian product of n_x intervals: $[\mathbf{x}] = [x_1] \times [x_2] \cdots \times [x_n] = \times_{i=1}^{n_x} [x_i]$. Usually, interval analysis is used to model quantities which vary around a central value within certain bounds. When working with intervals, one should introduce the inclusion function $[f]$ of a function f, defined such that the image by $[f]$ of an interval $[x]$ is an interval $[f]([x])$ [6]. This function is calculated such that the interval enclosing the image set is optimal (the smallest interval englobing the image). One should also extend all elementary arithmetic operations like +, -, , etc to the bounded error context and extend usual operations between sets of $\mathbb{R}^n$, e.g., $\subset, \supset, \cap, \ldots$.

Different algorithms, called contractors, exist in order to reduce the size of boxes enclosing the solutions. For the fusion problem considered, we have chosen to use constraints propagation techniques [6], because of the great redundancy of data and equations. To define the *Constraints Satisfaction Problem*, we consider a system of m relations f_m linking variables x_i of a vector x of $\mathbb{R}^{n_x}$ by a set of equations of the forms: $f_j(x_1, \ldots, x_{n_x}) = 0$, $j = 1 \ldots m$, which can be written in a compact way as $f(\mathbf{x}) = \mathbf{0}$, where $\mathbf{f}$ is the cartesian product of the f_j's.

Definition 1 (Constraints Satisfaction Problem)**.** *A Constraints Satisfaction Problem $\mathcal{H}$ is the problem which gathers a vector of variables $\mathbf{x}$ from an initial domain $\mathbf{D}$ and a set of constraints $\mathbf{f}$ linking the variables x_i of $\mathbf{x}$.*

Note that under the interval framework, $\mathbf{D} = [\mathbf{x}] = \times_{i=1}^{n_x} [x_i]$. The CSP consists on finding the values of $\mathbf{x}$ which satisfy $f(\mathbf{x}) = \mathbf{0}$. The solution set of the CSP will be defined as $\mathbf{S} = \{\mathbf{x} \in [\mathbf{x}] \mid \mathbf{f}\{\mathbf{x}\} = \mathbf{0}\}$. Note that $\mathbf{S}$ is not necessary a box. Under the interval framework, solve the CSP is interpreted as *finding the minimal box* $[\mathbf{x}^{'}] \subset [\mathbf{x}]$ such that $\mathbf{S} \subset [\mathbf{x}^{'}]$.

Definition 2 (Contractor)**.** *A contractor is defined as an operator used to contract the initial domain of the CSP, and thus to provide* a new box $[\mathbf{x}^{'}] \subset [\mathbf{x}]$ *such that* $\mathbf{S} \subset [\mathbf{x}^{'}]$.

There are different kinds of methods to develop contractors. Each of these methods may be adapted to some specific CSPs and not to others. The method used in this paper is the Waltz algorithm [7] which is based on the constraints decomposition on primitive ones and on the use of forward-backward propagation (FBP) technique [6] for each of primitive constraints. A primitive constraint involves only an arithmetic operator or a usual function ($\cos, \exp$, etc.). The principle of the Waltz contractor is to use FBP for each constraint, without any a priori order, until the contractor *becomes inefficient*. The use of this contractor appear to be specially efficient when one has a redundancy of data and equations. In fact, this is the case for the data used in section (5). Note that Waltz algorithm is independent of the non-linearities and provide a locally consistent contractors [6]. For more details on interval analysis, readers are invited to see [6] and references therein.

4. BOX PARTICLE STRATEGY

In this section, we present in details the BPF. The key idea is to use Interval Analysis, constraint propagation techniques and to model noises by bounded errors. This is reinforced by two possible understandings of an interval in one dimension:

1. An interval represents infinity of particles continuously distributed on the interval.
2. An interval represents a particle imprecisely located in the interval.

In order to explore the state space, one can split the state space region under consideration in N boxes $\{[x^{(i)}]\}_{i=1}^{N}$ with empty intersection and associate equivalent weight for each of them. A first advantage expected with this initialization using boxes is the possibility to explore the space with a reduced number of box particles.

After the initialization step, the state of every box particle is updated according to the evolution model thanks to interval analysis tools. Knowing the box particles $\{[x^{(i)}]\}_{i=1}^{N}$ and the input $\{[u_{(k)}]\}$ at step k, the boxes at step $k+1$ are built using the following propagation equation: $[x_{k+1}^{i}] = [f]([x_k^i], [u_k])$, where $[f]$ is an inclusion function for f. The interesting propriety one can notice here is that, in order to propagate the box particles, the bounded error method is used without introducing noise.

The new measurement should be used in order to adjust the particle weights and contract the boxes. The innovation for box particle is a quantity which should indicate the proximity between the real and the predicted measure boxes. In the bounded error framework, this quantity can be evaluated as the intersection between these two boxes. Thus, for all box particles, $i = 1 \cdots N$, we have to predict box measurements using $[z_{k+1}^{i}] = [g]([x_{k+1}^{i}])$, where $[g]$ is an inclusion function for g. The innovation here consists on the intersection with the box real measure $[y_{k+1}]$. This intersection is calculated as $[r_{k+1}^{i}] = [z_{k+1}^{i}] \cap [y_{k+1}]$. With the bounded error approach, it's obvious to conclude that, a box particle for which the predicted measure box hasn't an intersection with the real measure box should be penalized and a box particle for which the predicted measure is included in the real measure box should be favorite. This lead us to construct a measure of the *box likelihood* as: $A^i = \prod_1^p A^i(j)$ where $A^i(j) = \frac{|[r_{k+1}^{i}(j)]|}{|[z_{k+1}^{i}(j)]|}$, p is the dimension of the measure and $|[X]|$ is the width of $[X]$.

In the particle filter algorithm, each particle is propagated without any information about the variance of it's position. Note that the weight of the particle gives us only an information about certainty when using this particle. In an opposite manner, after been propagated, the width of each box particle is assumed to take into account the imprecision caused by the model errors and inputs imprecisions. In order to conserve a judicious width of each box, one should use contraction algorithms which eliminate the non consistent part of the box particle with respect to the box measure [4]. This is in fact similar to the correction step of Kalman filtering when the variance-covariance matrix is corrected using the measure [5]. Note that this step, used only for box particles, doesn't appear in the particle filter algorithm.

Thus, if the innovation $[r^i_{k+1}]$ is not empty, then we should contract the box particle $[x^i_{k+1}]$ using the intersection box $[r^i_{k+1}]$ and Waltz algorithm to obtain a new box particle $[x^i_{k+1}]^{new}$. Else, $[x^i_{k+1}]^{new} = [x^i_{k+1}]$ and the box particle stays unchanged.

One should update the weights by multiplying the previous weight by each *box likelihood* as: $\omega^i_{k+1} = (\prod_1^p A^i(j))\omega^i_k = A^i\omega^i_k$

In order to use normalized weights, so that their sum is equal to one, we should use: $\omega^i_{k+1} \longleftarrow \frac{\omega^i_{k+1}}{\sum_{j=1}^N \omega^j_{k+1}}$.

The state can be estimated by the center means of the weighted box particles as: $\hat{x}_k = \sum_{i=1}^N \omega^i_k C^i_k$, where C^i_k is the center of the box particle i. One can use also a maximum weight estimate, i.e the state estimate will be the center of the box particle with the larger weight. A pessimist confidence in the estimation will be a very well determined area consisting in a box which contain all the possible weighted boxes. Hence one can assign for this box the noun of *enclosing box*. Note that the estimation $\hat{x}_k$ is calculated by using N vectors C^i_k. Thus, another confidence in the estimation based on the confidence of each C^i_k can be calculated with the expression: $\hat{P}_k = \sum_{i=1}^N \omega^i_k P^i_k$, where P^i_k is the partial confidence generated when using each box particle center C^i_k. In practice, P^i_k can be taken as the half width of each box particle. Thus, $\hat{P}_k = \sum_{i=1}^N \omega^i_k \frac{|[x^i_k]|}{2}$

After some iterations, only few box particles may be likely and the rest may have weights close, or even exactly equal to zero. Thus, one has to sample box particles according to their weights. Box particles that have high weights are more likely to survive, whereas those with lower weights are less likely. The resampling can be efficiently implemented using a classical algorithm for sampling N ordered independent identically distributed variables [5]. The problem of the resampling is that the resulting samples are dependent since there is a *big chance* that the samples will be drawn from a few number of ancestors. In the case of particle filter algorithm, instead of representing the smooth probability density as they should, particles are clustered into groups. Therefore, some artificial noise should be added to the resampled particles in order to lessen the dependency. This step avoid the particle filter to fall down. One can use the same strategy for box particles by adding an artificial noise to the bounds of the box. Moreover, regarding the possibilities given by boxes properties, other techniques of resampling can be considered. For example, in order to obtain independent and small boxes around regions with high likelihoods, it's easily perceived that we can divide each box by the correspondent number of realization after sampling. Nevertheless, in the bounded error area, the choice of the number of divisions that one have to do for each dimension constitutes an open problem under study [1]. After the resampling step, we have to assign the same weight for all box particles. Note that an estimation of the effective sample size N_{eff} is given by [5]: $N_{eff} = \frac{1}{\sum_{i=1}^N (\omega^i_k)^2}$. The resampling step can be performed if the effective number of samples is less than some threshold N_{th} which is determined experimentally. The BPF is summarized in Figure 1.

```
1. Initialization
   Set $k = 0$ and generate $N$ boxes $\{x^{(i)}(k)\}_{i=1}^N$ with empty
   intersection and with same width and weights equal to $\frac{1}{N}$
2. $\underline{FOR}$ $i = 1 \cdots N$
3. Propagation or prediction
   $[x^i_{k+1}] = [f]([x^i_k], [u_k])$.
4. Measurement update
     • Predicted measurement: $[z^i_{k+1}] = [g]([x^i_{k+1}])$.
     • Innovation: $[r^i_{k+1}] = [z^i_{k+1}] \cap [y_{k+1}]$.
     • likelihood: $A^i = \prod_1^p A^i(j)$, where $A^i(j) = \frac{|[r^i_{k+1}(j)]|}{|[z^i_{k+1}(j)]|}$.
     • Box particle contraction: IF $[r^i_{k+1}] \neq \emptyset$, THEN,
       contract $[x^i_{k+1}]$ using $[r^i_{k+1}]$ and Waltz algorithm to
       obtain $[x^i_{k+1}]^{new}$, ELSE, $[x^i_{k+1}]^{new} = [x^i_{k+1}]$, ENDIF.
     • Weights update: $\omega^i_{k+1} = (\prod_1^p A^i(j))\omega^i_k = A^i\omega^i_k$
   $\underline{ENDFOR}$.
5. Weights normalization
   $\underline{FOR}$ $i = 1 \cdots N$, $\omega^i_{k+1} \longleftarrow \frac{\omega^i_{k+1}}{\sum_{j=1}^N \omega^j_{k+1}}$, $\underline{ENDFOR}$
6. State estimation
   $\hat{x}_k = \sum_{i=1}^N \omega^i_k C^i_k$.   $\hat{P}_k = \sum_{i=1}^N \omega^i_k \frac{|[x^i_k]|}{2}$.
7. Resampling
   $N_{eff} = \frac{1}{\sum_{i=1}^N (\omega^i_k)^2}$. IF $N_{eff} < N_{th}$, THEN resample to
   create $N$ new particle boxes with the same weights.
8. $k = k + 1$, Goto 2 Until $k = k^{end}$
```

Fig. 1. BPF algorithm.

5. APPLICATION TO DYNAMIC LOCALIZATION

We present results of applying the BPF on a real experiments which consist on a localization problem of a land vehicle, and show that BPF can be efficiently implemented with a small number of box particles. At instant k, we note $\delta_{S,k}$ and $\delta_{\theta,k}$ the elementary displacement and elementary rotation given by sensors. We make specific static tests to determine interval measurements error. The position and heading angle of the vehicle which is at time k, $[X_k] = [x_k] \times [y_k] \times [\theta_k]$ are calculated in time by using linear and angular velocities via the following discrete representation:

$$\begin{cases} x_{k+1} = & x_k + \delta_{S,k}\cos(\theta_k + \frac{\delta_{\theta,k}}{2}) \\ y_{k+1} = & y_k + \delta_{S,k}\sin(\theta_k + \frac{\delta_{\theta,k}}{2}) \\ \theta_{k+1} = & \theta_k + \delta_{\theta,k} \end{cases} \tag{2}$$

The measurement of the position at time consists here in a Global Position System (GPS) which is (x_{GPS}, y_{GPS}). The "longitude, latitude" estimated point of the GPS is converted in a Cartesian local frame and the GPS bounded error measurement is obtained thanks to the GST NMEA sentence [4]. The width of the GPS measure box can be quantified using the standard deviation σ_{GPS} on x and y estimated in real time by the GPS receiver (GST frame), $[x_{GPS}] = [x_{GPS} - 3\sigma_{GPS}, x_{GPS} + 3\sigma_{GPS}]$, $[y_{GPS}] = [y_{GPS} - 3\sigma_{GPS}, y_{GPS} + 3\sigma_{GPS}]$. The GPS measurement $([x_{GPS}], [y_{GPS}])$ is used to initialize the box state position $([x_1], [y_1])$ at instant

t_1. Note that we haven't a direct measure of the heading angle, so the heading state of the vehicle should be initialized as $[\theta_1] = [-\infty, +\infty]$.

In order to be able to compute estimation errors, we have used a Thales Navigation GPS receiver used in a Post-Processed Kinematic (PPK) mode working with a local base (a Trimble 7400). We report hereafter the analysis of 4.7 Km path with a mean speed of 50 Km/h using a 3GHz Pentium 4 and a Matlab implementation. The two filters provide outputs at the frequency of the GPS (5Hz). More details about the experiments can be found in [4].

The resampling method used for the BPF consists on a subdivision strategy. The idea consists first to sample box particles according to their weights using for example a classical deterministic algorithm, and second to divide each resampled box to several boxes with number equal to the realizations of this box resulting from the resampling algorithm. This type of resampling will allows us to refine the solution around regions with high likelihood and to eliminate boxes with low weights. Nevertheless, one has to determine the number of divisions to do for each dimension. For the state considered in the case of the model (2), which is a three variables state, the box particles will be in $\mathbb{R}^3$. We suggest to give the preference to bisect boxes heading angle $[\theta]$ since we haven't a direct measure on this variable, but only an elementary rotation δ_θ of the mobile. This division is firstly performed until the width of the interval on θ of the resampled box is more than a fixed quantity (two degrees for example). For the choice between the subdivision of intervals on x or intervals on y, we give the preference to intervals with larger width.

Table 1 shows the mean square error for GPS, PF and BPF. As a conclusion, the BPF and the PF give equivalent filtering performances. Nevertheless, for the BPF running, we use only 10 box particles comparing with 3000 particles for the PF. Thus, one can reduce significatively the particles number by using BPF instead of PF (for this application, the factor is about 300). Table 1 gives the mean of the running time of one step for each algorithm. Since the output frequency of each filter is 5 HZ, the running time for BPF satisfy real time constraints despite the use of interval arithmetic programs under Matlab and without code optimization. This is not the case of the PF. Figure 2 shows the interval error for x and y estimated for GPS (dashed black), BPF (bold black) and PF (solid blue). For PF, the interval error is calculated by using 3σ errors bounds around the point estimate. It can be seen that for this nonlinear problem, the two filters are consistent. Note that the interval error contain "0" is equivalent to say that the interval contains the PPK's point.

The maximum error on the heading estimation angles provided by the BPF and the PF are of the same magnitude (Between zero and two degrees). As shown in Table 1, the mean square errors for BPF and PF are, respectively, 0.445 and 0.446. One can conclude that the BPF is able to reconstruct a non directly measured variable. Note that the reference heading angle was built manually from the PPK measurements.

	GPS	PF	BPF
mean square error for x(m)	0.134	0.129	0.119
mean square error for y(m)	0.374	0.217	0.242
particle number	-	3000	10
one step running time (ms)	-	666	149
mean square error for θ(degrees)	-	0.446	0.445

Table 1. Comparison of PF and BPF.

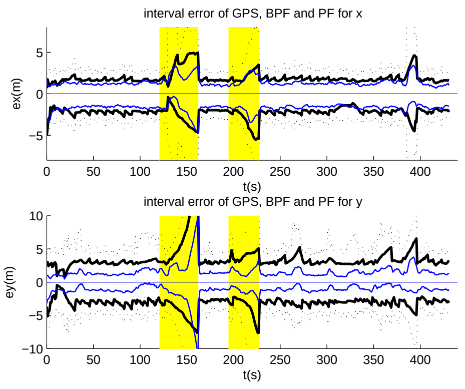

Fig. 2. Interval error for x and y estimated for GPS (dashed black), BPF (bold black) and PF (solid blue).

6. CONCLUSION AND FUTURE WORKS

A new fusion strategy that mixes Interval Analysis techniques and particle filters for data fusion and state estimation purposes was presented. The method requires a small number of box particles. This, in fact, answers one of major problems when using particle filters techniques. A comparison on real data shows that BPF outperforms the PF solution in term of running time. In our opinion, the BPF is particularly adapted when using map data with rectangular roads. Indeed, in this case the boxes are adapted to calculate an intersection with the map. Thus, future works will consist on applying the BPF algorithm for Map Matching problems.

7. REFERENCES

[1] Cendes, T. and Ratz, D. "Subdivision direction selection in interval methods for global optimization". *SIAM Journal of numerical Analysis*, 34:922-938, 1997.

[2] A. Doucet, S. Godsill, and C. Andrieu. "On sequential Monte Carlo sampling methods for Bayesian fltering". *Statistics and computing*, 10:197-208, 2000.

[3] D. Fox. "Adapting the Sample Size in Particle Filters Through KLD-Sampling". *The international Journal of robotics research*, vol. 22, 12. pp. 985-1004, Dec 2003.

[4] A. Gning, Ph. Bonnifait. "Constraints Propagation Techniques on Intervals for a Guaranteed Localization using Redundant Data". *Automatica*. Volume 42, Issue 7, pages 1167-1175, July 2006.

[5] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, P. Nord-lund. "Particle filters for positioning, navigation, and tracking". *IEEE Transactions on Signal Processing*, vol. 50, pp. 425- 435, Feb. 2002.

[6] Jaulin L., M. Kieffer, O. Didrit and E. Walter. *"Applied Interval Analysis"*. Springer-Verlag, 2001.

[7] D. Waltz. "Generating semantic descriptions from drawings of scenes with shadows", *The psychology of computer Vision*, New York, NY pp. 19-91, 1975.