A Spatial Domain Multiresolutional Particle Filter

Lang Hong* and Kefu Xue*

*Wright State University, Department of Electrical Engineering, Dayton, OH 45435, USA

Abstract—Particle filters have been proven to be very effective for nonlinear/non-Gaussian filtering. However, the most notorious disadvantage of a particle filter is its formidable computational complexity, since hundreds (even thousands) of particles are usually needed to achieve a required approximation accuracy. It has also been proven that one of the techniques of truly solving a computational problem is multiresolutional processing, both in temporal and spatial domains. Therefore, in this paper we propose a multiresolutional particle filter in the spatial domain using thresholded wavelets to reduce significantly the number of particles, meanwhile maintaining the full strength of a particle filter.

Keywords: Particle filters, nonlinear/non-Gaussian estimation, computational efficiency, multiresolutional processing techniques, wavelets

I. INTRODUCTION

Particle filters, falling into the category of sequential Monte Carlo methods, are effective in nonlinear and non-Gaussian filtering. A particle filter estimates a posterior density function \( p(x_k | Z^k) \) by an empirical approach

\[
\hat{p}(x_k | Z^k) = \frac{1}{N} \sum_{i=1}^{N} \delta_k(x_k^{(i)})
\]

where \( x_k^{(i)} \) are iid (independently and identically distributed) random samples of \( p(x_k | Z^k) \) and \( Z^k \) represents the accumulative measurements. The associated integration can also be achieved empirically as

\[
I(f_k) = \int f_k(x_k)p(x_k | Z^k)x_k \approx \frac{1}{N} \sum_{i=1}^{N} f_k(x_k^{(i)}). \tag{2}
\]

Although operations in Eqs. (1) and (2) are straightforward and can be applied to any nonlinear and non-Gaussian density, there is a problem: sampling directly from a posterior function is not always possible. Researchers have found a solution: instead of sampling the posterior function, samples can be drawn from a proposal function whose support includes that of the posterior function. This method is called importance sampling (IS) or sequential importance sampling (SIS) when the weight updates are performed in a recursive manner. However, IS/SIS is not problem free. As time elapses, the distribution of importance weights becomes more and more skewed. After a few time steps, only a few particles have non-zero weights. Another fix has also been proposed: a selection step called resampling is introduced to eliminate the particles with low weights and multiply the particles having high weights. More recently, many improvements have been made and the technology of particle filters is becoming mature, which includes bootstrap particle filters, extended-Kalman particle filters, unscented particle filters, etc. [7], [12]. Reference [4] presents an excellent collection of articles on particle filters. When applied to target tracking, both in temporal and spatial domains. Therefore, in this paper we propose a multiresolutional particle filter in the spatial domain using thresholded wavelets to reduce significantly the number of particles, meanwhile maintaining the full strength of a particle filter.

In general, multiresolutional processing can be applied to a particle filter in either temporal or spatial domains. A multiresolutional particle filter is called a multires particle filter if it is derived by applying multiresolutional processing in a temporal domain. On the other hand, applying multiresolutional processing in a spatial domain in designing a particle filter results in a multires particle filter. In our previous work, we have demonstrated significant computational savings by using a multirate multiple model particle filter [11]. This paper is focused on computational savings with a multires particle filter. A particle filter approximates a density function by a set of particles with weights and a multires particle filter further decomposes the weights into lowpassed and highpassed components. By thresholding the highpassed particle weights, a much smaller number of particles can represent the density function with a comparable approximation accuracy. A density can be approximated by a set of densities with different resolutions (smoothness), where a smoother density requires a smaller number of particles. As a result, the total number of particles could be much smaller than a regular particle filter (we call this a unires particle filter), while maintaining a comparable approximation accuracy.

The rest of the paper is organized as follows. Section 2 formulates a multires particle filter and Section 3 presents detailed implementations of a multires particle filter with simulation examples. Section 4 concludes the paper.

II. FORMULATION OF A MULTRES PARTICLE FILTER

For a nonlinear filtering problem, the system and measurement models are given by

\[
x_k = f(x_{k-1}) + v_{k-1} \tag{3}
\]

\[
z_k = h(x_k) + w_k \tag{4}
\]

where \( f() \) and \( h() \) are system dynamic and measurement functions. Variables \( v_{k-1} \) and \( w_k \) are governed by the following known densities, \( p_v \) and \( p_w \):

\[
p(x_k | x_{k-1}) = p_v(x_k - f(x_{k-1})) \tag{5}
\]

\[
p(z_k | x_k) = p_w(z_k - h(x_k)). \tag{6}
\]

The initial state density, \( p_s(x_0) \), is also assumed known which can be decomposed into a multires structure with \( j_1 \) levels

\[
p_s(x_0) = \sum_k \alpha_{j_1,k} \phi_{j_1,k}(x_0) + \sum_{j \geq j_1} \sum_k \beta_{j,k} \psi_{j,k}(x_0) \tag{7}
\]

with scaling and wavelet coefficients given by [2]

\[
\alpha_{j_1,k} = < p_s(x_0), \phi_{j_1,k}(x_0) > \tag{8}
\]

\[
\beta_{j,k} = < p_s(x_0), \psi_{j,k}(x_0) > \tag{9}
\]
where
\[ \{ \phi_{j,k} = 2^{-j/2} \phi(2^{-j} x - k), k \in Z \}, \]
\[ \{ \psi_{j,k} = 2^{-j/2} \psi(2^{-j} x - k), k \in Z, j \geq j_1 \}, \]
and \( \phi(x) \) and \( \psi(x) \) are the scaling and mother wavelet functions, respectively. In the form of particle representation, we have
\[ p_k(x_0) = \sum_{i=1}^{N_0} \omega_{x_0}^i \delta(x_0 - x_0^i) \] (10)
where particle \( x_0^i \) is sampled from \( p_k(x_0) \), i.e., \( x_0^i \sim p_k(x_0) \) and \( \omega_{x_0}^i \) is the corresponding weight. The inner products defined by Eqs. (8) and (9) that require nonlinear integration become
\[ \sum_{i=1}^{N_0} \omega_{x_0}^i \phi_{j,1,k}(x_0) \delta(x_0 - x_0^i) \] (11)
and
\[ \sum_{i=1}^{N_0} \omega_{x_0}^i \psi_{j,1,k}(x_0) \delta(x_0 - x_0^i) \] (12)
In this paper, we will use Harw wavelets as a vehicle for demonstration.

**Haar Wavelets**

The scaling function is a simple rectangular function
\[ \phi(t) = \begin{cases} 1 & \text{if } 0 \leq t \leq 1 \\ 0 & \text{otherwise} \end{cases} \] (13)
with only two nonzero coefficients \( h(0) = h(1) = 1/\sqrt{2} \) and the wavelet function is
\[ \psi(t) = \begin{cases} 1 & \text{if } 0 \leq t < 0.5 \\ -1 & \text{if } 0.5 \leq t \leq 1 \\ 0 & \text{otherwise} \end{cases} \] (14)
with two nonzero coefficients \( g(0) = 1/\sqrt{2} \) and \( g(1) = -1/\sqrt{2} \). A filterbank is used for the discrete wavelet transform (DWT).

Rewrite Eq. (10) as a signal sequence
\[ p_k(x_0) = \{ \omega_{x_0}^i \}, \quad i = 1, \ldots, N_0. \] (15)
Eqs. (11) and (12) can be expressed as a discrete convolution with a downsampling process
\[ \omega_{1}^i = \tilde{\alpha}_1^i = \sum_{k=-\infty}^{+\infty} \omega_{x_0}^i h[2i - k] \] (16)
and
\[ \omega_{h1}^i = \tilde{\beta}_1^i = \sum_{k=-\infty}^{+\infty} \omega_{x_0}^i g[2i - k] \] (17)
where only level 1 decomposition is presented. Not considering the edge effect, the process of convolution and downsampling results in \( N_0/2 \) non-zero coefficients which can be described in a summation form
\[ \omega_1 = \sum_{i=1}^{N_0/2} \omega_1^i \delta(x_0 - x_0^i) \] (18)
and
\[ \omega_{h1} = \sum_{i=1}^{N_0/2} \omega_{h1}^i \delta(x_0 - x_0^i) \] (19)
where \( x_0^i \) is the result of lowpassed and downsampled \( x_0^i \). Level 2 decomposition can be derived from \( \omega_1 \) by
\[ \omega_2^i = \sum_{k=-\infty}^{+\infty} \omega_1^i h[2i - k] \] (20)
and
\[ \omega_{h2}^i = \sum_{k=-\infty}^{+\infty} \omega_1^i g[2i - k] \] (21)
where only \( N_0/4 \) non-zero coefficients are obtained. Other levels of decomposition can be obtained similarly.

A simple thresholding can be carried out on the wavelet coefficients to reduce the number of coefficients. Let
\[ \hat{\omega}_{h1}^i = \omega_{h1}^i, \quad \text{if } \omega_{h1}^i \geq T_s; \quad \text{otherwise } \hat{\omega}_{h1}^i = 0. \] (22)
A thresholded density estimation can then be achieved as
\[ \hat{\omega}_{x0}^i = \sum_{k=-\infty}^{+\infty} \omega_1^i h[2k - i] + \hat{\omega}_{h1}^i g[2k - i]. \] (23)
\( T_s \) is an application dependent parameter which needs to be adjusted for each application depending on available computational capability. The larger \( T_s \) is, the fewer number of surviving coefficients but larger approximation errors. Thus far, the decomposition is performed for the sequence where the independent variable is the sequence index \( i \). If the independent variable is a continuous variable, such as \( x \), the decomposition is handled differently. Assume density \( p_k \) can be approximated by
\[ p_k = \sum_{i=1}^{N} \omega_i \delta(x - x^i) \] (24)
where \( x^i \) is sampled from \( p_k \), i.e., \( x^i \sim p_k \), and \( \sum_{i=1}^{N} \omega_i = 1 \). Density \( p_k \) is a continuous function of \( x \), not \( i \). To better handle Eq. (25) in multiresolution decomposition, we can rewrite it in a vector form
\[ p_k = \delta^T \omega \] (26)
where
\[ \delta = [\delta(x - x^1), \ldots, \delta(x - x^N)]^T, \quad \omega = [\omega^1, \ldots, \omega^N]^T. \] (27)
Let
\[ \omega = T^T [\omega_{h1}, \ldots, \omega_{h1+l}]^T = T^T \omega_h \] (28)
be the multiresolution decomposition of sample weights, where \( T \) is an orthogonal wavelet decomposition transform matrix, such that \( T^{-1} = T^T \). There are \( L + 1 \) levels in the decomposition. Substituting Eq. (28) into Eq. (26) results in
\[ p_k = \delta^T T^T \omega_h = (T \delta)^T \omega_h = \delta_h^T \omega_h \] (29)
where
\[ \delta_{jh} = T \delta = [\delta_{j1}, \delta_{j2}, \ldots, \delta_{j2^{L_j}+L_j}]^T. \] (30)

The multiresolution decomposition in Eq. (29) is applied to both \( \delta_1 \) and \( \delta_h \). We introduce the following variable structure
\[ \text{sgn} \{x \} = \begin{cases} 1, & x \in \text{first half of } \{x_h\} \\ -1, & x \in \text{second half of } \{x_h\} \end{cases} \] (32)
and
\[ \text{sgn}(\{x_{hj}\}) = \begin{cases} 1, & x \in \{x_{hj}\} \\ -1, & x \in \{x_{hj}\} \end{cases} \] (34)
With this implicit method, the density reconstruction can be performed as follows
\[ p_x = \sum_{i} \omega_i \delta(x - \{x_i\}) + \sum_{j \geq j_1} \sum_{i} \omega_{hj} \delta(x - \{x_{hj}\}) \text{sgn}(\{x_{hj}\}). \] (35)

Explicit Method

It can be seen in Eq. (35) that no explicit inverse transform is needed in density reconstruction, but the variable structure in Eq. (31) may pose a difficulty in implementation. Other than a complicated variable structure, we can introduce transformed variables as
\[ \mathbf{x}^h = \begin{bmatrix} x_1 \\ x_{h1} \\ \vdots \\ x_{h2^{L_j}+L_j} \end{bmatrix} = T \delta(x - \mathbf{x}^h), \] (36)
Then by using
\[ \delta_{h} = T \delta = [\delta_{1}, \ldots, \delta_{2^{L_j} + L_j}]^T, \] (37)
we have
\[ p_x = (T \delta(x - \mathbf{x}^h)) \mathbf{w}_h = (T \delta(T^T \mathbf{x}^h - \mathbf{x}_h)) \mathbf{w}_h \] (38)
which contains explicit transformation \( T \). Eq. (38) can be rewritten as
\[ p_x = \mathbf{w}^T \mathbf{p}(\delta_{h}, \mathbf{w}_h), \] (40)
where \( \mathbf{w} \) is a transformation mapping function.
If we name each level’s density component \(^3\) in Eq. (29) by \( p_{l1} = \mathbf{w}^T \mathbf{w} \)
\[ p_{l1} = \mathbf{w}^T \mathbf{w} \] (41)
\[ p_{l2} = \mathbf{w}^T \mathbf{w} \] (42)
for the lowpassed density component at level \( j \) for the implicit method or
\[ p_{l2} = \mathbf{w}^T \mathbf{w} \] (43)
for the explicit method, and
\[ p_{l2} = \mathbf{w}^T \mathbf{w} \] (44)
using the explicit method, we could rewrite Eq. (29) as
\[ p_x = p_{l1} + \sum_{j \geq j_1} p_{l2}. \] (45)

With simple thresholding, we can significantly reduce the number of particles without sacrificing the performance. Thresholding can be applied to both the implicit and explicit methods. For the implicit method, thresholding is applied to the density components
\[ \{ \mathbf{w}^T \mathbf{w}(\delta_{h}, \mathbf{w}_h) \} = \{ \mathbf{w}^T \mathbf{w}(\delta_{h}, \mathbf{w}_h) \} \] (46)
and for the explicit method, thresholding is directly applied to the wavelet coefficients
\[ \mathbf{w}_h = \{ \mathbf{w}(\delta_{h}, \mathbf{w}_h) | \mathbf{w}(\delta_{h}, \mathbf{w}_h) > T \}, \] (47)
Thus, we have
\[ p_{l2} = \sum_{j \geq j_1} \mathbf{w}^T \mathbf{w}(\delta_{h}, \mathbf{w}_h), \] (48)
for the implicit method, and
\[ p_{l2} = \sum_{j \geq j_1} \mathbf{w}^T \mathbf{w}(\delta_{h}, \mathbf{w}_h), \] (49)
for the explicit method.

Example 1.

Let the wavelet transform be
\[ \mathbf{T} = \begin{bmatrix} 1/2 & 1/2 & 1/2 & 1/2 \\ -1/2 & -1/2 & 1/2 & 1/2 \\ -\sqrt{2}/2 & \sqrt{2}/2 & 0 & 0 \\ 0 & 0 & -\sqrt{2}/2 & \sqrt{2}/2 \end{bmatrix}, \] (50)
and a 4-point density approximation be
\[ [x_1 x_2 x_3 x_4] = [2 10 4 6]^T, \quad [\omega_1 \omega_2 \omega_3 \omega_4] = \left[ \frac{1}{6} \frac{1}{4} \frac{5}{12} \frac{1}{6} \right]^T. \] (51)
and
\[ p_x = \frac{1}{6} \delta(x - 2) + \frac{5}{12} \delta(x - 4) + \frac{1}{6} \delta(x - 6) + \frac{1}{4} \delta(x - 10). \] (52)

Implicit Method

From Eqs. (28) and (30), we have
\[ \begin{bmatrix} \delta_{h1} \\ \delta_{h2} \\ \delta_{h21} \\ \delta_{h22} \end{bmatrix} = \mathbf{T} \mathbf{w} = \begin{bmatrix} 1/2 \delta(x - 2) + \delta(x - 4) + \delta(x - 6) + \delta(x - 10) \\ 1/2 \delta(x - 2) - \delta(x - 4) + \delta(x - 6) + \delta(x - 10) \\ \sqrt{2}/2 \delta(x - 2) - \delta(x - 4) + \delta(x - 6) + \delta(x - 10) \\ \sqrt{2}/2 \delta(x - 2) - \delta(x - 4) + \delta(x - 6) + \delta(x - 10) \end{bmatrix} \] (53)
and
\[ \begin{bmatrix} \omega_1 \\ \omega_{h1} \\ \omega_{h21} \\ \omega_{h22} \end{bmatrix} = \mathbf{w} \mathbf{w} = \begin{bmatrix} 1/2 \\ 1/2 \\ -\sqrt{2}/2 \\ -\sqrt{2}/2 \end{bmatrix}. \] (54)
For this example, the explicit method with and without thresholding using the implicit method.

\[ p_x = \delta(x - 2) + \delta(x - 4) + \delta(x - 6) + \delta(x - 10) \]

\[ = \frac{1}{4}\delta(x - 2) + \delta(x - 4) + \delta(x - 6) + \delta(x - 10) \]

\[ = \frac{1}{4}\delta(x - 2) + \delta(x - 4) + \delta(x - 6) + \delta(x - 10) \]

\[ = \frac{1}{8}\delta(x - 2) + \delta(x - 4) + \frac{1}{24}\delta(x - 6) + \delta(x - 10). \] (56)

For this example, \( \{x_t\} = \{2, 4, 6, 10\} \), \( \{x_{h1}\} = \{2, 4, 6, 10\} \), \( \{x_{h2}\} = \{2, 4\} \) and \( \{x_{h2}\} = \{6, 10\} \). If we set the threshold as \( \frac{T}{2} \), applying thresholding to Eq. (56) results in

\[ p_x \approx \delta(x - 1) + \delta(x - 3) + \delta(x - 5) + \delta(x - 9) \]

\[ = \frac{1}{4}\delta(x - 1) + \delta(x - 3) + \delta(x - 5) + \delta(x - 9) \]

\[ = \frac{1}{8}\delta(x - 1) + \delta(x - 3) + \frac{1}{24}\delta(x - 5) + \delta(x - 9). \] (57)

Figures 1 and 2 show the reconstruction of \( p_x \) from \( \rho_{j1} \) and \( \rho_{j2} \) with and without thresholding using the implicit method.

### Explicit Method

Using Eq. (36), we can find the following transformed variables

\[ \omega_{h} = [115\sqrt{2}2\sqrt{2}]^T. \] (59)

The reconstruction of the density can be performed separately by variable and weight reconstructions, which are

\[ x = [24610]^T = T^T \omega_{h}. \] (60)

and

\[ \omega = [16512614]^T = T^T \omega_{h}. \] (61)

The density reconstruction is then

\[ p_x = \frac{1}{6}\delta(x - 2) + \frac{5}{12}\delta(x - 4) + \frac{1}{6}\delta(x - 6) + \frac{1}{4}\delta(x - 10) \] (62)

which is illustrated in Figure 3. If the threshold is set as \( T_x = \sqrt{2}/24 \), then we have

\[ \omega_{h} = [12-12\sqrt{2}80]^T, \] (63)

which results in

\[ \omega = [16512624524]^T. \] (64)

Figures 3 and 4 demonstrate the results of density reconstruction using the explicit method, with and without thresholding.

### III. Multires Bootstrap Particle Filter Algorithm

This section presents a major development of a spatial domain multiresolution (multires) particle filter for computationally efficient processing. Given a density in a multires decomposition form

\[ p(x_{k-1} | z_{1:k-1}) = p_{z_{1:k-1} j1} + \sum_{j2 \geq j1} p_{z_{1:k-1} j2} \] (65)

where

\[ p_{z_{1:k-1} j1} = \delta_{k-1}^{j1} \omega_{h_{j1}} \] (66)

and

\[ p_{z_{1:k-1} j2} = \sum_{j} \delta_{k-1}^{j} \omega_{h_{j1}} \omega_{h_{j2}}, \quad j = j_1, \ldots, L. \] (67)

Figures 3 and 4 demonstrate the results of density reconstruction using the explicit method, with and without thresholding.
In the following, we will derive a complete update cycle for multires densities.

A. The Implicit Method

- **Multires Propagation:**

  \[ p(x_k | x_{k-1}) = \int p(x_k | x_{k-1}) p(x_{k-1} | z_{1:k-1}) dx_{k-1} \]

  \[ = \int p(x_k | x_{k-1}) p(x_{k-1}) \sum_{j \geq j_1} p(x_{k-1}) dx_{k-1} \]

  \[ = \sum_{j \geq j_1} \int p(x_k | x_{k-1}) \omega_{k-1}^j \delta(x_{k-1} - \{x_j\}_{k-1}) dx_{k-1} + \]

  \[ \sum_{j \geq j_1} \sum_{i=1}^{N_i} \sum_{m} \int p(x_k | \{x_{h_{jn}}\}_{k-1}) \omega_{k-1}^i \times \]

  \[ \delta(x_{k-1} - \{x_{h_{jn}}\}_{k-1}) \text{sgn}(\{x_{h_{jn}}\}_{k-1}) dx_{k-1} \]

  \[ = \sum_{j \geq j_1} \sum_{i=1}^{N_i} \omega_{k-1}^j \delta(x_k - \{x_j\}_{k-1}) + \sum_{j \geq j_1} \sum_{i=1}^{N_i} \sum_{m} \omega_{k-1}^i \times \]

  \[ \delta(x_k - \{x_{h_{jn}}\}_{k-1}) \text{sgn}(\{x_{h_{jn}}\}_{k-1}) \]

  where \{x_j\}_{k-1} \sim p(x_k | \{x_j\}_{k-1}) \text{ and } \{x_{h_{jn}}\}_{k-1} \sim p(x_k | \{x_{h_{jn}}\}_{k-1}). \text{ Since both } \{x_j\}_{k-1} \text{ and } \{x_{h_{jn}}\}_{k-1} \text{ are associated with a same set of particles at the original resolution level, only on one propagated set (either } \{x_j\}_{k-1} \text{ or } \{x_{h_{jn}}\}_{k-1} \text{ is obtained from sampling and others can be derived from the sample set.}

  The propagated state estimate can be derived as

  \[ \hat{x}_{k|k-1} = \sum_{j \geq j_1} \sum_{i=1}^{N_i} \omega_{k-1}^j \{x_j\}_{k-1} + \]

  \[ \sum_{j \geq j_1} \sum_{i=1}^{N_i} \sum_{m} \omega_{k-1}^i \{x_{h_{jn}}\}_{k-1} \times \]

  \[ \text{sgn}(\{x_{h_{jn}}\}_{k-1}) \].

  **Example 2.**

  Continuing on Example 2. Assume the system model is

  \[ x_{k+1} = x_k^2 + v_k, \quad v_k \sim N(0, \sigma_v^2) \]

  where \(\sigma_v\) is the propagation density.

  \[ p(x_k | x_{k-1}) = \mathcal{N}(x_k - x_{k-1}^{2/3}, \sigma_v^2) \]

  Take \(\{x_i\}_{k-1} = \{2, 4, 6, 10\}\) for propagation. By sampling \(p(x_k | \{x_i\}_{k-1})\), we have

  \[ \{x_i\}_{k-1} = \{1.5, 2.8, 3.5, 5.4\} \]

  Due to the relationship defined by \(T\) in Eq. (50), we can derive other propagated sets

  \[ \{x_{h_{11}}\}_{k-1} = \{1.5, 2.8, 3.5, 5.4\}, \quad \{x_{h_{21}}\}_{k-1} = \{1.5, 2.8\} \]

  and

  \[ \{x_{h_{22}}\}_{k-1} = \{3.5, 5.4\}. \]

  **Multires Update:**

  The update of propagated multires density is given by

  \[ p(x_k | z_{1:k}) \]

  \[ = \frac{1}{C} p(z_k | x_k) \int p(x_k | x_{k-1}) p(x_{k-1} | z_{1:k-1}) dx_{k-1} \]

  \[ = \frac{1}{C} p(z_k | x_k) p(x_{k-1} | z_{1:k-1}) \]

  \[ = \frac{1}{C} \left[ \sum_{j \geq j_1} \sum_{i=1}^{N_i} \sum_{m} p(z_k | x_k) \omega_{k-1}^j \delta(x_k - \{x_j\}_{k-1}) + \right. \]

  \[ \sum_{j \geq j_1} \sum_{i=1}^{N_i} \sum_{m} p(z_k | x_k) \omega_{k-1}^i \delta(x_k - \{x_{h_{jn}}\}_{k-1}) \times \]

  \[ \text{sgn}(\{x_{h_{jn}}\}_{k-1}) \]

  \[ = \frac{1}{C} \left[ \sum_{j \geq j_1} \sum_{i=1}^{N_i} \sum_{m} p(z_k | x_k) \omega_{k-1}^j \delta(x_k - \{x_{h_{jn}}\}_{k-1}) \delta(x_k - \{x_{h_{jn}}\}_{k-1}) \times \]

  \[ \text{sgn}(\{x_{h_{jn}}\}_{k-1}) \]
where

\[
\hat{p}^j_{h,k-1} = \mathbf{T} \left[ \begin{array}{c} f(x_{k-1}') \\ \vdots \\ f(x_{k-1}') \\ \vdots \\ f(x_{k-1}') \end{array} \right]^j + \mathbf{T} \left[ \begin{array}{c} v_{k-1} \\ \vdots \\ v_{k-1} \\ \vdots \\ v_{k-1} \end{array} \right]^j
\] (87)

\[
= \mathbf{T}^T \left( \left( \hat{p}^j_{h,k-1} \right) + w_{h,k-1} \right)
\] (88)

\[
= \mathbf{T}^T \left( \hat{p}^j_{h,k-1} \right) + w_{h,k-1}.
\] (89)

From Eq. (84) to Eq. (85), a partition of \( N \) samples into \( N/2^L \) blocks is used where \( L \) is the level number of the multires decomposition. Since it is impossible to analytically derive \( f_T \) in Eq. (89), we take another approach.

For the thresholded \( \hat{p}^j_{h,k-1} \), we can find the corresponding \( \hat{p}^j_{h,k-1} \). Therefore, we can just propagate the elements in \( \hat{p}^j_{h,k-1} = \mathbf{T}^T \hat{p}^j_{h,k-1} \) which correspond to the distinct elements in \( \hat{p}^j_{h,k-1} = \mathbf{T}^T \hat{p}^j_{h,k-1} \). For the repeated elements of \( \hat{p}^j_{h,k-1} \), we can propagate one representative element in \( \hat{p}^j_{h,k-1} \). The number of particle propagations is controlled by the thresholding operation, which can significantly reduce the computation in propagation while maintaining performance.

The propagation of the thresholded quantity is then given by

\[
\hat{p}^j_{h,k-1} = f(\mathbf{T}^T \hat{p}^j_{h,k-1}) + w_{h,k-1} \] (90)

which in turn defines the density propagation as

\[
p(x_k|z_{1:k-1}) = \sum_{j=1}^{N/2^L} \delta^T (\hat{z}^j_k - \hat{z}^j_{k-1}) \hat{p}^j_{h,k-1}.
\] (91)

\section*{Explicit Multires Update}

Given the measurement model

\[
z_k = g(x_k) + w_k
\] (92)

where \( w_k \sim N(0, R_k) \), we want to update the propagated density

\[
p(x_k|z_{1:k-1}) = \sum_{j=1}^{N/2^L} \delta^T (\hat{z}^j_k - \hat{z}^j_{k-1}) \hat{p}^j_{h,k-1} \] (93)

with the explicit multires method. The likelihood functions are

\[
t^i = N(z_k - g(\hat{x}_{k-1}^i), R_k), \quad i = 1, \ldots, \hat{N}
\] (94)

where \( \hat{N} \) is the total number of distinct particles, which could be much smaller than \( N \) depending on the threshold. The density update can be carried out by

\[
p(x_k|z_{1:k}) = \sum_{i=1}^{\hat{N}} \frac{1}{C} \hat{p}^j_{k-1} \hat{t}^i \delta(x_k - \hat{x}^i_{k-1})
\] (95)

where \( C = \sum_{i=1}^{\hat{N}} \hat{p}^j_{k-1} \hat{t}^i \).

\section*{Example 3.}

The nonlinear system and measurement models are given by

\[
x_k = \frac{x_{k-1}}{2} + \frac{5x_{k-1}}{1 + x^2_{k-1}} + v_k, \quad v_k \sim N(0,1)
\] (96)

and

\[
z_k = 10 \tan^{-1}\left(\frac{x_k}{10}\right) + w_k, \quad w_k \sim N(0,1)
\] (97)

with the initial nonlinear (bi-modal) state density being

\[
p_0(x_0) = 0.7 N(10,2^2) + 0.3 N(20,4^2).
\] (98)

In the following, we will show how \( p_0(x_0) \) is propagated from \( t = 0 \) to \( t = 1 \) using particles with and without the multiresolution approach. Then the propagated density will be updated by \( \hat{z}_1 = 15 \tan^{-1}(\frac{z_0}{10}) + w_1 \) using particles with and without the multiresolution approach, where

\[
\hat{x}_{0|0} = \sum_{i=1}^{N} x_i \omega_i
\]

with \( x_0 \sim p_0(x_0) \).

A numerical comparison of estimation accuracy and computational complexity of both unires and multires approaches is given in Table I. It can be clearly seen that the proposed multires approach performs comparably (or even slightly better sometimes) to the unires approach, but is significantly more efficient!

\section*{IV. Conclusions}

We have developed a framework for computationally efficient particle filtering using the spatial domain multiresolutional approach. Both implicit and explicit implementation methods for a multires particle filter are presented. Computational efficiency of a particle filter is achieved by drastically reducing the needed number of particles while maintaining a comparable estimation accuracy.

\section*{REFERENCES}


\section*{TABLE I}

A COMPARISON OF COMPUTATIONAL COMPLEXITY AND ESTIMATION ACCURACY (500 RUNS).

<table>
<thead>
<tr>
<th>Approaches</th>
<th>RMS Errors</th>
<th>No. of particles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unires Particle Filter</td>
<td>0.0950</td>
<td>1,000</td>
</tr>
<tr>
<td>Multires Particle Filter (( T_s = 0.0002 ))</td>
<td>0.0451</td>
<td>562</td>
</tr>
<tr>
<td>Multires Particle Filter (( T_s = 0.001 ))</td>
<td>0.0443</td>
<td>234</td>
</tr>
<tr>
<td>Multires Particle Filter (( T_s = 0.002 ))</td>
<td>0.0452</td>
<td>165</td>
</tr>
</tbody>
</table>