Kernel-Based Bayesian Filtering for Object Tracking

Bohyung Han[†] Ying Zhu[‡] Dorin Comaniciu[§] Larry Davis[†]

† Dept. of Computer Science
University of Maryland
College Park, MD 20742, USA
{bhhan, lsd}@cs.umd.edu

‡ Real-Time Vision and Modeling
Siemens Corporate Research
Princeton, NJ 08540, USA
Ying.Zhu@siemens.com

§ Integrated Data and Systems
Siemens Corporate Research
Princeton, NJ 08540, USA
Dorin.Comaniciu@siemens.com

Abstract

Particle filtering provides a general framework for propagating probability density functions in non-linear and non-Gaussian systems. However, the algorithm is based on a Monte Carlo approach and sampling is a problematic issue, especially for high dimensional problems. This paper presents a new kernel-based Bayesian filtering framework, which adopts an analytic approach to better approximate and propagate density functions. In this framework, the techniques of density interpolation and density approximation are introduced to represent the likelihood and the posterior densities by Gaussian mixtures, where all parameters such as the number of mixands, their weight, mean, and covariance are automatically determined. The proposed analytic approach is shown to perform sampling more efficiently in high dimensional space. We apply our algorithm to the real-time tracking problem, and demonstrate its performance on real video sequences as well as synthetic examples.

1 Introduction

Particle filtering is a Monte Carlo approach to solve the recursive Bayesian filtering problem. Although it provides tractable solutions to non-linear and non-Gaussian systems, it is faced with practical issues such as sample degeneracy and sample impoverishment [2]. Moreover, to achieve reliable filtering, the sample size can grow exponentially as the dimension of the state space increases. To overcome these issues, we explore an analytic approach to approximate density functions and introduce a new kernel-based filtering scheme. The main idea of this work is to maintain an analytic representation of relevant density functions and propagate them over time. In this paper, kernel-based density representation is adopted.

1.1 Related Work

There have been many parametric density representations proposed for various applications. In [15, 20], the authors suggest Gaussian mixture models, but their method requires knowledge of the number of components, which is difficult to know in advance. A more elaborate density representation is described in [11], where a 3-component mixture is used for the target modeling in object tracking problem, but this approach cannot overcome the drawback of parametric methods. Kernel density estimation [8] is a widely used non-parametric approach in computer vision. Its major advantage is the flexibility to represent very complicated densities effectively. But its very high memory requirements and computational complexity inhibit the use of this method.

For Bayesian filtering, Cham and Rehg [3] introduce a piecewise Gaussian function to specify the tracker state, in which the selected Gaussian components characterize the neighborhoods around the modes. This idea is applied to multiple hypothesis tracking in a high dimensional space body tracker, but the sampling and the posterior computation are not straightforward. The closest work to our paper is [13] where the posterior is represented with a Gaussian mixture in a particle filter framework. However, this solution may not provide a compact representation for the posterior, and the prediction and the update steps are oversimplified.

1.2 Our Approach

In this paper, we extend our previous work [9] which provides the main framework of Kernel-based Bayesian filtering. We introduce density approximation and density interpolation to represent density functions efficiently and effectively. In both techniques, the density function is represented by a Gaussian mixture, where the number of mixands, their weights, means and covariances are automatically determined. The density approximation is based on a mode finding algorithm [6, 7] derived from variable-bandwidth mean-shift which provides the methodology to construct a compact representation with a small number of Gaussian kernels. A density interpolation technique is introduced to obtain a continuous representation of the measurement likelihood function. Unscented transformation (UT) [12, 16] is also adopted to deal with non-linear state transition models. These techniques are integrated into the Bayesian filtering framework. In the new kernel-based Bayesian filtering algorithm, the continuous representations of density functions are propagated over time.

The advantage of maintaining an analytic representation of density functions lies in efficient sampling. This is important for solving high dimensional problems. A multi-stage sampling strategy is introduced in density interpolation for accurate approximation of the measurement likelihood function. The new

algorithm is applied to real-time object tracking, and its performance is demonstrated through various experiments.

This paper is organized as follows. Section 2 introduces the new density propagation technique in the Bayesian filtering framework. Section 3 and 4 explain the density approximation and the density interpolation method, respectively. Section 5 demonstrates its performance by various simulation results with synthetic examples. Finally, it is demonstrated in section 6 how our algorithm can be applied to object tracking in real videos.

2 Bayesian Filtering

In this section, we introduce the new Bayesian filtering framework, where the relevant density functions are approximated by kernel-based representations and propagated over time.

2.1 Overview

In a dynamic system, the process and measurement model are given by

$$\mathbf{x}_t = g(\mathbf{x}_{t-1}, \mathbf{u}_t) \tag{1}$$

$$\mathbf{z}_t = h(\mathbf{x}_t, \mathbf{v}_t) \tag{2}$$

where $\mathbf{v}_t$ and $\mathbf{u}_t$ are the process and the measurement noise, respectively. The state variable $\mathbf{x}_t$ ($t = 0, \ldots, n$) is characterized by its probability density function estimated from the sequence of measurements $\mathbf{z}_t$ ($t = 1, \ldots, n$). In the sequential Bayesian filtering framework, the conditional density of the state variable given the measurements is propagated through prediction and update stages,

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{t-1} \tag{3}$$

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \frac{1}{k}p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) \tag{4}$$

where $k = \int p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1})d\mathbf{x}_t$ is a normalization constant independent of $\mathbf{x}_t$. $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$ is the prior probability density function (pdf), $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ is the predicted pdf and $p(\mathbf{z}_t|\mathbf{x}_t)$ is the measurement likelihood function. The posterior pdf at time step t, $p(\mathbf{x}_t|\mathbf{z}_{1:t})$, is used as the prior pdf in time step $t+1$.

At each time step, the conditional distribution of the state variable $\mathbf{x}$ given a sequence of measurements $\mathbf{z}$ is represented by a Gaussian mixture. Our goal is to retain such a representation through the stages of prediction and update, and to represent the posterior probability in the following step with the same mixture form.

The proposed filtering framework is described as follows. First, unscented transformation (UT) [12, 16] is used to derive a mixture representation of the predicted pdf $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$. Second, the density interpolation technique with a multi-stage sampling is introduced to approximate the likelihood function with an mixture form. By multiplying two mixture functions, the posterior pdf is obtained through equation (4). To prevent the number of mixands from growing too large, an algorithm of density approximation based on mode finding is applied to derive a compact representation for the posterior pdf.

2.2 Prediction by Unscented Transform

Denote by $\mathbf{x}_t^i$ ($i = 1, \ldots, n_t$) a set of means in R^d and by $\mathbf{P}_t^i$ the corresponding covariance matrices at time step t. Let each Gaussian have a weight κ_t^i with $\sum_{i=1}^{n_t} \kappa_t^i = 1$, and let the prior density function be given by

$$p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1}) = \frac{1}{(2\pi)^{d/2}} \sum_{i=1}^{n_{t-1}} \frac{\kappa_{t-1}^i}{|\mathbf{P}_{t-1}^i|^{1/2}} \exp\left(-\frac{1}{2}D^2\left(\mathbf{x}_{t-1}, \mathbf{x}_{t-1}^i, \mathbf{P}_{t-1}^i\right)\right) \tag{5}$$

The unscented transformation [12, 16] is a method for calculating the statistics of a random variable which undergoes a non-linear transformation.

$$\begin{aligned}
\mathcal{X}_{t-1}^{(i,0)} &= \mathbf{x}_{t-1}^i \\
\mathcal{X}_{t-1}^{(i,j)} &= \mathbf{x}_{t-1}^i - (\sqrt{(d+\lambda)\mathbf{P}_{t-1}^i})_j \quad j = 1, \ldots, d \\
\mathcal{X}_{t-1}^{(i,j)} &= \mathbf{x}_{t-1}^i + (\sqrt{(d+\lambda)\mathbf{P}_{t-1}^i})_{j-d} \quad j = d+1, \ldots, 2d \\
W_{(i,0)} &= \lambda/(d+\lambda) \\
W_{(i,j)} &= 1/2(d+\lambda) \qquad j = 1, \ldots, 2d
\end{aligned} \tag{6}$$

where λ is a scaling parameter and $(\sqrt{(d+\lambda)\mathbf{P}_{t-1}^i})_j$ is the ith row or column of the matrix square root of $(d+\lambda)\mathbf{P}_{t-1}^i$. $W_{(i,j)}$ is the weight associated with the j-th sigma point where $\sum_{j=0}^{2d} W_{(i,j)} = 1$. These sigma vectors are propagated through the non-linear function,

$$\mathcal{X}_t^{(i,j)} = g(\mathcal{X}_{t-1}^{(i,j)}) \quad i = 0, \ldots, 2d \tag{7}$$

and the mean and covariance for $\bar{\mathbf{x}}_t^i$ are approximated using a weighted sample mean and covariance of the posterior sigma points,

$$\begin{aligned}
\bar{\mathbf{x}}_t^i &= \sum_{i=0}^{2d} W_{(i,j)}\mathcal{X}_t^{(i,j)} \\
\bar{\mathbf{P}}_t^i &= \sum_{i=0}^{2d} W_{(i,j)}(\mathcal{X}_t^{(i,j)} - \bar{\mathbf{x}}_t^i)(\mathcal{X}_t^{(i,j)} - \bar{\mathbf{x}}_t^i)^\top + \mathbf{Q}
\end{aligned} \tag{8}$$

where $\mathbf{Q}$ is the covariance matrix for the process noise.

For each mode in the prior, UT is applied independently and the density after prediction is as follows.

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) = \frac{1}{(2\pi)^{d/2}} \sum_{i=1}^{n_{t-1}} \frac{\bar{\kappa}_t^i}{|\bar{\mathbf{P}}_t^i|^{1/2}} \exp\left(-\frac{1}{2}D^2\left(\mathbf{x}_t, \bar{\mathbf{x}}_t^i, \bar{\mathbf{P}}_t^i\right)\right) \tag{9}$$

where $\bar{\kappa}_t^i = \kappa_{t-1}^i$. This non-linear transformation is guaranteed to be accurate up to the second order of the Taylor expansion.

2.3 Multi-stage Sampling and Interpolation of Measurement Likelihood

In contrast to various particle filters, we represent the measurement likelihood function in an analytic form. A continuous approximation of the likelihood function is interpolated from discrete samples. A multi-stage sampling scheme is introduced to improve the approximation progressively. The advantage of the analytic representation is that it provides a global view of the landscape of the likelihood function and thus enables efficient sample placement.

2.3.1 Multi-stage sampling

Unlike the SIR algorithm [10] which uses the predicted pdf as the proposal distribution, we employ the multi-stage sampling strategy and progressively update the proposal function based on the observation. The predicted pdf is used as the initial proposal distribution q_0.

$$q_0(\mathbf{x}_t) = p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) \tag{10}$$

Assume that in total, N samples are to be drawn to obtain measurement data. In our multi-stage sampling scheme, N/m samples are drawn in the first stage from the initial proposal distribution (10), where m is the number of sampling stages. An initial approximation of the likelihood function $p_1(\mathbf{z}_t|\mathbf{x}_t)$ is obtained through surface interpolation with Gaussian kernels. Details of the density interpolation algorithm is provided in section 4. The proposal function is then updated by a linear combination of the initial proposal distribution and the current approximation of the likelihood function $p_1(\mathbf{z}_t|\mathbf{x}_t)$. We repeatedly approximate the likelihood function from available samples and update the proposal distribution.

$$p_j(\mathbf{z}_t|\mathbf{x}_t) = \sum_{\tau_i \neq 0} \tau_t^i \exp\left(-\frac{1}{2} D^2\left(\mathbf{x}_t, \mathbf{x}_t^i, \mathbf{R}_t^i\right)\right) \tag{11}$$

$$q_j(\mathbf{x}_t) = (1-\alpha_j) q_{j-1}(\mathbf{x}_t) + \alpha_j \frac{p_j(\mathbf{z}_t|\mathbf{x}_t)}{\int p_j(\mathbf{z}_t|\mathbf{x}_t) d\mathbf{x}_t} \tag{12}$$

where $i = 1, \ldots, \frac{j}{m}N$, $j = 1, \ldots, m$, and $\alpha_j \in [0,1]$ is *adaptation rate*.

Since the information of the observation is incorporated into the proposal distribution to guide sampling, the multi-stage sampling strategy explores the likelihood surface more efficiently than conventional particle filters. Thus, it is especially advantageous in dealing with high dimensional state space.

2.3.2 Approximation of likelihood function

As discussed previously, the measurement likelihood is estimated through the multi-stage sampling. With samples drawn from the improved proposal distributions, intermediate likelihood functions are constructed and used to update the proposal distributions. After m-step repetition of this procedure, the final measurement distribution is obtained. Algorithm 1 presents the complete procedure to compute the likelihood function, and the final measurement function with m_t Gaussians at time t is given by

Algorithm 1 Measurement Step

$S_t = \phi$
$q_0(\mathbf{x}_t) = p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$
for $i = 1$ to m **do**
  draw sample s from proposal distribution
  $S_t^i = \{s_i^{(j)} | s_i^{(j)} \sim q_{i-1}(\mathbf{x}_t), j = 1, \ldots, N/m\}$
  $S_t = S_t \cup S_t^i$
  assign mean and covariance for the element in S_t^i
  $\mathbf{m}_{(i-1)\frac{N}{m}+j} = s_i^{(j)}$
  $\mathbf{Q}_{(i-1)\frac{N}{m}+j} = c\,\mathrm{diag}(\mathrm{KNN}_1(k) \ldots \mathrm{KNN}_d(k))^2\,\mathbf{I}$ (25)
  compute likelihood of each new sample
  $l_{(i-1)\frac{N}{m}+j} = h(\mathbf{m}_{(i-1)\frac{N}{m}+j}, \mathbf{v}_t)$
  compute $\mathbf{A}$ and $\mathbf{b}$ for every element in S_t
  $\mathbf{w} = nnls(\mathbf{A}, \mathbf{b})$ (27)
  $p_i(\mathbf{z}_t|\mathbf{x}_t) = \sum_{\tau_t^j \neq 0} N(\tau_t^j, \mathbf{x}_t^j, \mathbf{R}_t^j)$
  where $\tau_t = \mathbf{w}$, $\mathbf{x}_t = \mathbf{m}$, and $\mathbf{R}_t = \mathbf{Q}$
  $q_i(\mathbf{x}_t) = (1-\alpha_i) q_{i-1}(\mathbf{x}_t) + \alpha_j \frac{p_i(\mathbf{z}_t|\mathbf{x}_t)}{\int p_i(\mathbf{z}_t|\mathbf{x}_t) d\mathbf{x}_t}$ (12)
end for
$p(\mathbf{z}_t|\mathbf{x}_t) = p_m(\mathbf{z}_t|\mathbf{x}_t)$

$$p(\mathbf{z}_t|\mathbf{x}_t) = \frac{1}{(2\pi)^{d/2}} \sum_{i=1}^{m_t} \frac{\tau_t^i}{|\,\mathbf{R}_t^i\,|^{1/2}} \exp\left(-\frac{1}{2} D^2\left(\mathbf{x}_t, \mathbf{x}_t^i, \mathbf{R}_t^i\right)\right) \tag{13}$$

where τ_t^i, $\mathbf{x}_t^i$ and $\mathbf{R}_t^i$ are the weight, mean and covariance matrix of the i-th kernel.

2.4 Update

Since both the predicted pdf and the measurement functions are represented by Gaussian mixtures, the posterior pdf, as the product of two Gaussian mixtures, can also be represented by a Gaussian mixture. Denote the Gaussian components of the predicted pdf and the likelihood function by $N(\bar{\kappa}_t^i, \bar{\mathbf{x}}_t^i, \bar{\mathbf{P}}_t^i)$ $(i = 1, \ldots, n_{t-1})$ and $N(\tau_t^j, \mathbf{x}_t^j, \mathbf{R}_t^j)$ $(j = 1, \ldots, m_t)$ respectively, the product of the two distributions is as follows.

$$\left(\sum_{i=1}^{n_{t-1}} N(\bar{\kappa}_t^i, \bar{\mathbf{x}}_t^i, \bar{\mathbf{P}}_t^i)\right)\left(\sum_{j=1}^{m_t} N(\tau_t^j, \mathbf{x}_t^j, \mathbf{R}_t^j)\right) = \sum_{i=1}^{n_{t-1}} \sum_{j=1}^{m_t} N(\bar{\kappa}_t^i \tau_t^j, \mathbf{m}_t^{ij}, \mathbf{\Sigma}_t^{ij}) \tag{14}$$

where

$$\mathbf{m}_t^{ij} = \mathbf{\Sigma}_t^{ij}((\bar{\mathbf{P}}_t^i)^{-1}\mathbf{x}_t^i + (\mathbf{R}_t^j)^{-1}\mathbf{x}_t^j) \tag{15}$$

$$\mathbf{\Sigma}_t^{ij} = ((\bar{\mathbf{P}}_t^i)^{-1} + (\mathbf{R}_t^j)^{-1})^{-1} \tag{16}$$

The resulting density function in (14) is a weighted Gaussian mixture. However, the exponential increase in the number of

components over time could make the whole procedure intractable. In order to avoid this situation, a density approximation technique is proposed to maintain a compact yet accurate density representation even after density propagation through many time steps. Details of the density approximation algorithm is given in section 3.

After the update step, the final posterior distribution is given by

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \frac{1}{(2\pi)^{d/2}} \sum_{i=1}^{n_t} \frac{\kappa_t^i}{\mid \mathbf{P}_t^i \mid^{1/2}} \exp\left(-\frac{1}{2} D^2\left(\mathbf{x}_t, \mathbf{x}_t^i, \mathbf{P}_t^i\right)\right) \tag{17}$$

where n_t is the number of components at time step t.

3 Density Approximation

In this section, we review an iterative procedure of mode detection derived from variable-bandwidth mean-shift [7], and density approximation using the mode detection technique [9].

3.1 Mode Detection and Density Approximation

Suppose that $N(\kappa_i, \mathbf{x}_i, \mathbf{P}_i)$ $(i = 1 \dots n)$ is a Gaussian kernel with weight κ_i, mean $\mathbf{x}_i$, and covariance $\mathbf{P}_i$ in the d-dimensional state space, where $\sum_{i=1}^n \kappa_i = 1$. Then, we define the sample point density estimator computed at point $\mathbf{x}$ by

$$\hat{f}(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}} \sum_{i=1}^{n} \frac{\kappa_i}{\mid \mathbf{P}_i \mid^{1/2}} \exp\left(-\frac{1}{2} D^2\left(\mathbf{x}, \mathbf{x}_i, \mathbf{P}_i\right)\right) \tag{18}$$

where

$$D^2\left(\mathbf{x}, \mathbf{x}_i, \mathbf{P}_i\right) \equiv (\mathbf{x} - \mathbf{x}_i)^\top \mathbf{P}_i^{-1} (\mathbf{x} - \mathbf{x}_i). \tag{19}$$

Our purpose is to obtain the compact representation of the density function which is a Gaussian mixture. The mode location and its weight are found by mean-shift algorithm, and the covariance matrix associated with each mode is computed by using Hessian matrix.

To find the gradient ascent direction at $\mathbf{x}$, the variable-bandwidth mean-shift vector at $\mathbf{x}$ is given by

$$\mathbf{m}(\mathbf{x}) = \left(\sum_{i=1}^{n} \omega_i(\mathbf{x}) \mathbf{P}_i^{-1}\right)^{-1} \left(\sum_{i=1}^{n} \omega_i(\mathbf{x}) \mathbf{P}_i^{-1} \mathbf{x}_i\right) - \mathbf{x} \tag{20}$$

where the weights

$$\omega_i(\mathbf{x}) = \frac{\kappa_i \mid \mathbf{P}_i \mid^{-1/2} \exp\left(-\frac{1}{2} D^2\left(\mathbf{x}, \mathbf{x}_i, \mathbf{P}_i\right)\right)}{\sum_{i=1}^n \kappa_i \mid \mathbf{P}_i \mid^{-1/2} \exp\left(-\frac{1}{2} D^2\left(\mathbf{x}, \mathbf{x}_i, \mathbf{P}_i\right)\right)} \tag{21}$$

satisfy $\sum_{i=1}^n \omega_i(\mathbf{x}) = 1$. By computing the mean-shift vector $\mathbf{m}(\mathbf{x})$ and translating the location $\mathbf{x}$ by $\mathbf{m}(\mathbf{x})$ iteratively, a local maximum of underlying density function is detected. A formal check for the maximum involves the computation of the Hessian matrix

$$\hat{\mathbf{H}}(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}} \sum_{i=1}^{n} \frac{\kappa_i}{\mid \mathbf{P}_i \mid^{1/2}} \exp\left(-\frac{1}{2} D^2\left(\mathbf{x}, \mathbf{x}_i, \mathbf{P}_i\right)\right) \times \mathbf{P}_i^{-1}\big((\mathbf{x}_i - \mathbf{x})(\mathbf{x}_i - \mathbf{x})^\top - \mathbf{P}_i\big)\mathbf{P}_i^{-1} \tag{22}$$

which should be negative definite. If it is not negative definite, the convergence point might be a saddle point or a local minimum. In this case, kernels associated with such modes should be restored and considered as separate modes for further processing.

The approximate density is obtained by detecting the mode location for every sample point $\mathbf{x}_i$ and assigning a single Gaussian kernel for each mode. Suppose that the approximate density has n' unique modes of $\tilde{\mathbf{x}}_j$ $(j = 1 \dots n')$ with associated weight $\tilde{\kappa}_j$ which is equal to the sum of the kernel weights converged to $\tilde{\mathbf{x}}_j$. The Hessian matrix $\hat{\mathbf{H}}_j$ of each mode is used for the computation of $\tilde{\mathbf{P}}_j$ as follows.

$$\tilde{\mathbf{P}}_j = \frac{\tilde{\kappa}_j^{\frac{2}{d+2}}}{\mid 2\pi(-\hat{\mathbf{H}}_j^{-1}) \mid^{\frac{1}{d+2}}} (-\hat{\mathbf{H}}_j^{-1}) \tag{23}$$

The basic idea of equation (23) is to fit the covariance using the curvature in the neighborhood of the mode. The final density approximation is then given by

$$\tilde{f}(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}} \sum_{i=1}^{n'} \frac{\tilde{\kappa}_i}{\mid \tilde{\mathbf{P}}_i \mid^{1/2}} \exp\left(-\frac{1}{2} D^2\left(\mathbf{x}, \tilde{\mathbf{x}}_i, \tilde{\mathbf{P}}_i\right)\right) \tag{24}$$

and $n' \ll n$ is satisfied in most cases. The approximation error $||\hat{f}(\mathbf{x}) - \tilde{f}(\mathbf{x})||$ can be evaluated straightforwardly.

3.2 Performance of Approximation

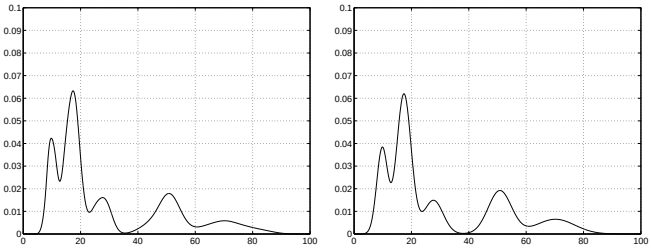
The accuracy of the density approximation is demonstrated in Figure 1. From a one-dimensional distribution composed of five weighted Gaussians, 200 samples are drawn, and the scale parameter is assigned as discussed in section 4.1. Mean Integrated Squared Error (MISE) between the original and the approximated densities is calculated for the error estimation.

The result in figure 1 shows that the mode finding based on mean-shift and the covariance estimation using the Hessian is very accurate.

Figure 2 shows the performance of the density approximation which is accurate enough to replace kernel density estimation in the multi-dimensional case.

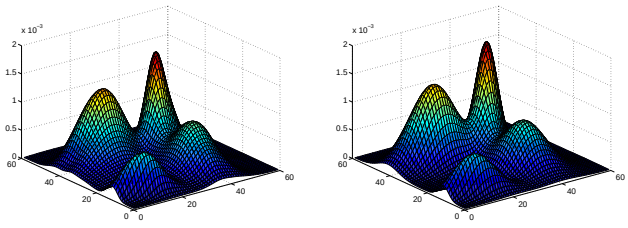
4 Density Interpolation

The density approximation presented in section 3 is an algorithm to find a compact representation when the mean, the covariance and the weight for each kernel are given. In the measurement step of Bayesian filtering, the likelihood values are



(a) (b)

Figure 1: Comparisons between kernel density estimation and density approximation (1D). For the approximation, 200 samples are drawn from the original distribution – $N(0.2, 10, 2^2)$, $N(0.35, 17, 4^2)$, $N(0.15, 27, 8^2)$, $N(0.2, 50, 16^2)$, and $N(0.1, 71, 32^2)$. (a) kernel density estimation (b) density approximation (MISE = 5.3234×10^{-5})



(a) (b)

Figure 2: Comparison between kernel density estimation and density approximations (2D). (a) kernel density estimation (b) density approximation (400 samples, MISE = 1.5237×10^{-8})

known for a set of samples. In this case, the likelihood surface can be interpolated from sample likelihood. In this section, we describe the density interpolate algorithm.

4.1 Initial Scale Selection

One of the limitations of kernel-based algorithms is that they involve the specification of a scale parameter. Various research has been performed for the scale selection problem [1, 17, 19], but it is very difficult to find the optimal scale in general. Below, we explain a strategy to determine the scale parameter for the density estimation based on *nearest neighbors*.

The basic idea of this method is very simple, and similar approaches are discussed in [4, 5]. Each sample is intended to cover the local region around itself in the d-dimensional state space with its scale. For this purpose, k-nearest neighbors (KNN) is used, and the kernel bandwidth (scale) is determined by the distance to the k-th nearest neighbor of a sample. Define $\mathrm{KNN}_j^i(k)$ $(1 \leq j \leq d)$ to be the distance to k-th nearest neighbor from sample i in the j-th dimension, and then the covariance matrix $\mathbf{P}_i$ for i-th sample is given by

$$\mathbf{P}_i = c\,\mathrm{diag}(\mathrm{KNN}_1^i(k)\ \mathrm{KNN}_2^i(k) \ldots \mathrm{KNN}_d^i(k))^2\,\mathbf{I} \qquad (25)$$

where c is a constant dependent upon the number of samples and the dimensionality, and $\mathbf{I}$ is a d-dimensional identity matrix.

By this method, samples in dense areas have small scales and the density will be represented accurately, but sparse areas convey only relatively rough information about the density function.

4.2 Interpolation

A Gaussian kernel is assigned to each sample for which mean and covariance corresponds to the sample location and the scale is initialized by the method in section 4.1, respectively. When the likelihood value on each sample is given, the weight for each kernel can be computed by the Non-Negative Least Square (NNLS) method [14].

Denote $\mathbf{x}_i$ as the mean location and $\mathbf{P}_i$ as the covariance matrix for the i-th sample ($i = 1, \ldots, n$). Also, suppose that l_i is the likelihood value on the i-th sample. The likelihood at $\mathbf{x}_j$ induced by the i-th kernel is given by

$$p_i(\mathbf{x}_j) = \frac{1}{(2\pi)^{d/2} \mid \mathbf{P}_i \mid^{1/2}} \exp\left(-\frac{1}{2} D^2(\mathbf{x}_j, \mathbf{x}_i, \mathbf{P}_i)\right). \quad (26)$$

Define an $n \times n$ matrix $\mathbf{A}$ having an entry $p_i(\mathbf{x}_j)$ in (i, j), and an $n \times 1$ vector $\mathbf{b}$ having l_i in its i-th row. Then, the weight vector $\mathbf{w}$ can be computed by solving the following constrained least square problem,

$$\min_{\mathbf{w}} ||\mathbf{A}\mathbf{w} - \mathbf{b}||^2 \qquad (27)$$

$$\text{subject to } \mathbf{w}_i \geq 0 \text{ for } i = 1, \ldots, n,$$

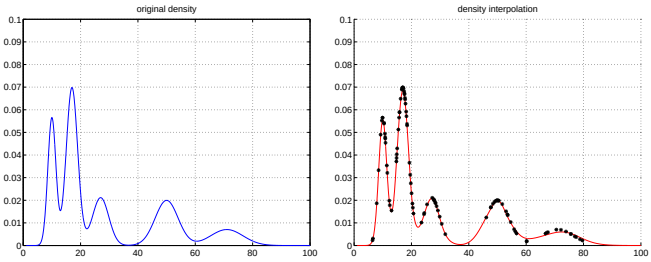
and it is denoted by $\mathbf{w} = nnls(\mathbf{A}, \mathbf{b})$. The size of matrix $\mathbf{A}$ is determined by the number of samples. When the sample size is large, sparse matrix operation methods can be used to solve $\mathbf{w}$ efficiently.

Usually, many of the weights will be zero and the final density function will be a mixture of Gaussians with a small number of components. The density interpolation simulates the heavy-tailed density function more accurately than the density approximation introduced in section 3, while the density approximation generally produces a more compact representation.

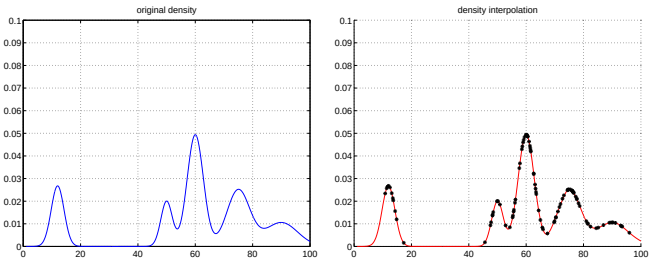
4.3 Performance of Interpolation

Figure 3 shows one-dimensional density interpolation results. For each case, 100 samples are drawn and the initial scale for each sample is given as explained in section 4.1. The estimated density function approximates the original density very accurately as seen in figure 3. Two different Gaussian mixtures – $N(0.2, 10, 2^2)$ $N(0.35, 17, 4^2)$ $N(0.15, 27, 8^2)$ $N(0.2, 50, 16^2)$ $N(0.1, 71, 32^2)$ in example 1, and $N(0.15, 12, 5^2)$ $N(0.1, 15, 4^2)$ $N(0.35, 60, 8^2)$ $N(0.25, 75, 16^2)$ $N(0.15, 90, 32^2)$ in example 2 – are tested for the interpolation.

When 50 independent realizations are performed, MISE and its variance are very small for both examples as shown in table 1.



(a) example 1



(b) example 2

Figure 3: Two examples of original density functions and their interpolations. In the interpolation graphs (right), black stars represent the sample locations (100 samples). In case (a) and (b), 22 and 24 components have non-zero weights, respectively.

Table 1: Error of density interpolation

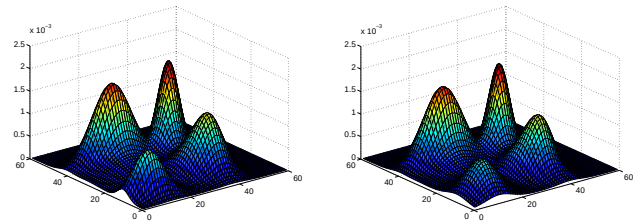
	MISE	VAR
example 1	3.9479×10^{-5}	2.5613×10^{-9}
example 2	2.6871×10^{-5}	9.5103×10^{-10}

Also, a multi-dimensional density function is interpolated in the same manner, and its performance is discussed next. In figure 4, the density interpolation produces a very accurate and stable result when 200 samples are drawn from the original density function (MISE = 4.5467×10^{-9}, VAR = 7.3182×10^{-18} on average over 50 runs).

These results show that surface interpolation is a sufficiently accurate method to approximate density function given samples and their corresponding likelihood.

5 Simulation

In this section, synthetic tracking examples are simulated, and the performance of the kernel-based Bayesian filtering is compared with the SIR algorithm [10]. Two different process models – one linear and the other non-linear – are selected, and simulations are performed for various dimensions such as 2D, 3D, 5D, 10D, 12D and 15D. The accumulated Mean Squared Error (MSE) through 50 time steps is calculated in each run, and 50 identical experiments are made based on the same data for accurate error estimation.



(a) (b)

Figure 4: Comparison between original density function and density interpolation (2D). (a) original density function (b) density interpolation with 30 non-zero weight components

The first process model is given by the following equation,

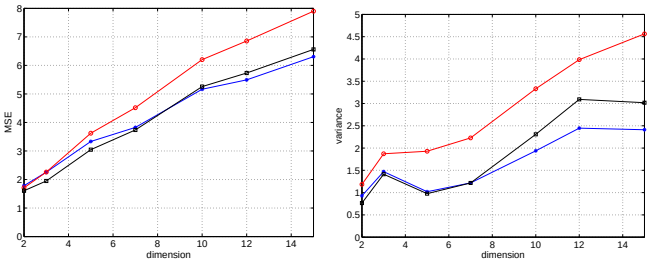
$$\mathbf{x}_t = \frac{\mathbf{x}_{t-1}}{2} + \frac{25\mathbf{x}_{t-1}}{1+\mathbf{x}_{t-1}^T\mathbf{x}_{t-1}} + 8\cos(1.2(t-1))\mathbf{1} + \mathbf{u}_t \quad (28)$$

where $\mathbf{1}$ is the vector whose elements are all ones. The process noise $\mathbf{u}_t$ is drawn from a Gaussian distribution $N(1, \mathbf{0}, (\sqrt{2}\mathbf{I})^2)$ where $\mathbf{I}$ is the identity matrix. The measurement model is given by a non-linear function

$$\mathbf{z}_t = \frac{1}{2}\mathbf{x}_t^T\mathbf{x}_t + \mathbf{v}_t \quad (29)$$

where $\mathbf{v}_t$ is drawn from a Gaussian distribution $N(1, \mathbf{0}, \mathbf{I}^2)$. For the estimation of the measurement function, fifty particles (10 particles $\times$ 5 stages) are drawn, and the posterior is estimated and propagated through the time step t ($1 \le t \le 50$).

Figure 5 demonstrates simulation results by comparing MSE's and variances of both algorithms. According to our experiment with the first model, the SIR filter shows better or equivalent performance in low dimensions such as 2D and 3D, but our method starts to outperform in high dimensions – more than 5D.



(a) Error (b) Variance

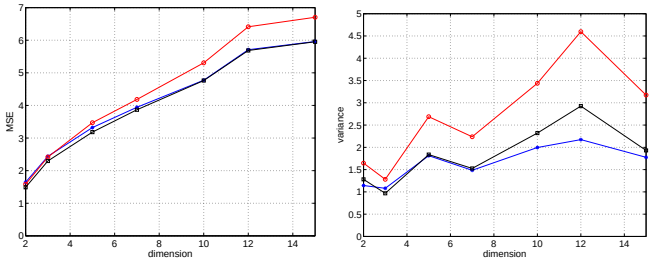
Figure 5: MSE and variance of for MSE. Kernel-based Bayesian filtering with 50 particles (blue star), SIR with 50 particles (red circle), and SIR filter with 500 particles (black square) for model 1.

The second process model is very simple linear model given by

$$\mathbf{x}_t = \frac{\mathbf{x}_{t-1}}{2} + 2\cos(2(t-1))\mathbf{1} + \mathbf{u}_t \quad (30)$$

where $\mathbf{u}_t \sim N(1, \mathbf{0}, (\sqrt{2}\mathbf{I})^2)$. The same observation model as equation (29) is employed, and fifty samples are drawn for every simulation.

Kernel-based Bayesian filtering yields smaller error in the high dimension as in previous case, and the detailed results are presented in figure 6.



(a) Error (b) Variance

Figure 6: MSE and variance of for MSE kernel-based Bayesian filtering with 50 samples (blue star), SIR filter with 50 particles (red circle), and SIR filter with 500 particles (black square) for model 2.

Two different process models produce almost similar results, and kernel-based Bayesian filtering shows better performance in high dimensional cases as expected. In order to demonstrate the benefit of kernel-based particles, we run the SIR algorithm with 500 samples, and compare the performance with our kernel-based Bayesian filtering with 50 samples. Surprisingly, the MSE's of the two cases are almost the same, and our algorithm has smaller variance of MSE than the SIR algorithm.

This result suggests that kernel-based Bayesian filtering can be applied effectively to high dimensional applications, especially, when many samples are not available and the observation process is very time-consuming.

6 Object Tracking

Particle filtering provides a convenient method for estimating and propagating the density of state variables regardless of the underlying distribution and the given system in the Bayesian framework. Additionally, our kernel-based Bayesian filtering has an advantage of managing multi-modal density functions with a relatively small number of samples. In this section, we demonstrate the performance of the kernel-based Bayesian filtering by tracking objects in real videos.

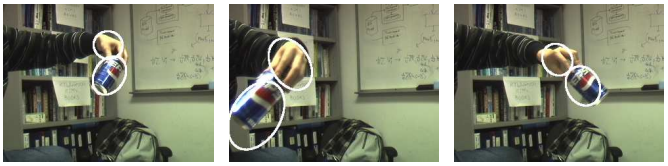
The overall tracking procedure is equivalent to what is described in section 5, and we explain the process and the measurement models briefly.

A random walk is assumed for the process model since it is very difficult to describe the accurate motion before the observation, even though our algorithm can accommodate the general non-linear function by unscented transformation described in section 2.2. 5-stage sampling is incorporated as introduced in section 2.3, and the likelihood of each particle is computed by the inverse exponentiation of the Bhattacharyya distance between the target and the candidate histograms as suggested in [18]. Based on the likelihood of each particle and the initial covariance matrix derived by the method in section 4.1, the measurement density is constructed by density interpolation.

Two sequences are tested in our experiment. In the first sequence, two objects – a hand carrying a can – are tracked with 200 samples (40 samples $\times$ 5 stages). The state space is described by a 10 dimensional vector, which is the concatenation of two 5 dimensional vectors representing two independent ellipses as follows.

$$(x_1, y_1, lx_1, ly_1, r_1, x_2, y_2, lx_1, ly_2, r_2) \tag{31}$$

where x_i and y_i $(i = 1, 2)$ are the location of ellipses, lx_i is the length of x-axis, ly_i is the length of y-axis, and r_i is the rotation variable. The tracking result is shown in figure 7, and our algorithm successfully tracks two objects for the whole sequence except the period that the side of the can is completely occluded around 470th frame.



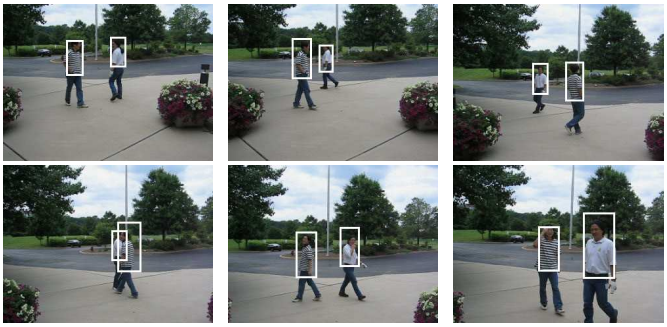
(a) $t = 43$ (b) $t = 304$ (c) $t = 387$

(d) $t = 456$ (e) $t = 570$ (c) $t = 898$

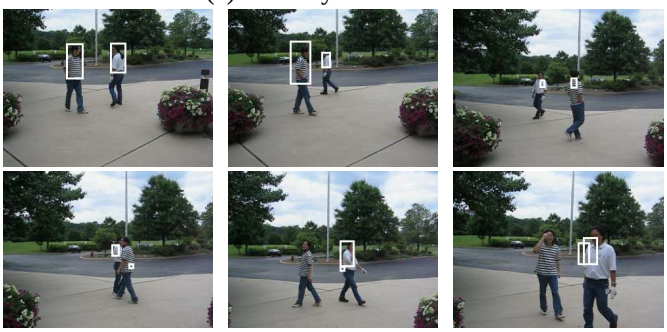
Figure 7: Object tracking result of *can* sequence.

The upper bodies of two persons are tracked in the second sequence, in which one occludes the other completely several times. The state vector is constructed by the same method as in the *can* sequence, but two rectangles are used instead of ellipses. A 6 dimensional vector – $(x, y, scale)$ for each rectangle – is used to describe the state, and 100 samples (20 samples $\times$ 5 stages) are used. Figure 8 (a) demonstrates the tracking results, and our algorithm shows good performance in spite of severe occlusions.

The tracker based on SIR algorithm is also implemented, and compared with our algorithm. As seen in figure 8 (b), the SIR algorithm shows unstable performance for the same sequence. According to experiments, one would need to run the SIR algorithm using about 400 particles to obtain a comparable result with our algorithm using 100 samples.



(a) result by our method



(a) result by the SIR algorithm

Figure 8: Object tracking result of *person* sequence at $t = 1, 95, 133, 193, 217, 300$.

7 Discussion and Conclusion

In this paper, we proposed a new Bayesian filtering framework where analytic representations are used to approximate relevant density functions. Density approximation and interpolation technique are introduced in density propagation. Various simulations and tests on object tracking in real videos show the effectiveness of our density approximation methods and the kernel-based Bayesian filtering. By maintaining analytic representations of the density functions, we can sample in the state space more effectively and more efficiently. This advantage is significant for high dimensional problems. In addition, the approximation error can be monitored and analyzed. Our future work is focused on analyzing the approximation error in the posterior distribution and its propagation over time.

Acknowledgment

The support by Siemens Corporate Research and VACE project (DOD2004H840200000) is gratefully acknowledged.

References

[1] I. Abramson, "On bandwidth variation in kernel estimates - a square root law," *The Annals of Statistics*, vol. 10, no. 4, pp. 1217–1223, 1982.

[2] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for on-line non-linear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 174–189, 2002.

[3] T. Cham and J. Rehg, "A multiple hypothesis approach to figure tracking," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Fort Collins, CO, volume II, 1999, pp. 239–219.

[4] W. Cleveland, "Robust locally weighted regression and smoothing scatterplots," *J. Amer. Statist. Assn.*, vol. 74, pp. 829–836, 1979.

[5] W. Cleveland and C. Loader, "Smoothing by local regression: Principles and methods," *Statistical Theory and Computational Aspects of Smoothing*, pp. 10–49, 1996.

[6] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, no. 5, pp. 603–619, 2002.

[7] D. Comaniciu, V. Ramesh, and P. Meer, "The variable bandwidth mean shift and data-driven scale selection," in *Proc. 8th Intl. Conf. on Computer Vision*, Vancouver, Canada, volume I, July 2001, pp. 438–445.

[8] A. Elgammal, R. Duraiswami, D. Harwood, and L. Davis, "Background and foreground modeling using nonparametric kernel density estimation for visual surveillance," *Proceedings of IEEE*, vol. 90, pp. 1151–1163, 2002.

[9] B. Han, D. Comaniciu, Y. Zhu, and L. Davis, "Incremental density approximation and kernel-based bayesian filtering for object tracking," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Washington DC, 2004, pp. 638–644.

[10] M. Isard and A. Blake, "Condensation - Conditional density propagation for visual tracking," *Intl. J. of Computer Vision*, vol. 29, no. 1, 1998.

[11] A. Jepson, D. Fleet, and T. El-Maraghi, "Robust online appearance models for visual tracking," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Hawaii, volume I, 2001, pp. 415–422.

[12] S. Julier and J. Uhlmann, "A new extension of the Kalman filter to nonlinear systems," in *Proceedings SPIE*, volume 3068, 1997, pp. 182–193.

[13] J. Kotecha and P. Djuric, "Gaussian sum particle filtering," *IEEE Trans. Signal Process.*, vol. 51, no. 10, pp. 2602–2612, 2003.

[14] C. L. Lauwon and B. J. Hanson, *Solving Least Squares Problems*. Prentice-Hall, 1974.

[15] S. McKenna, Y. Raja, and S. Gong, "Tracking colour objects using adaptive mixture models," *Image and Vision Computing Journal*, vol. 17, pp. 223–229, 1999.

[16] R. Merwe, A. Doucet, N. Freitas, and E. Wan, "The unscented particle filter," Technical Report CUED/F-INFENG/TR 380, Cambridge University Engineering Department, 2000.

[17] B. Park and J. Marron, "Comparison of data-driven bandwidth selectors," *J. of Amer. Stat. Assoc.*, vol. 85, pp. 66–72, 1990.

[18] P. Perez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in *Proc. European Conf. on Computer Vision*, Copenhagen, Denmark, volume I, 2002, pp. 661–675.

[19] S. Sheather and M. Jones, "A reliable data-based bandwidth selection method for kernel density estimation," *J. Royal Statist. Soc. B*, vol. 53, pp. 683–690, 1991.

[20] C. Stauffer and W. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, no. 8, pp. 747–757, 2000.