Circuits and Systems
Mekelweg 4,
2628 CD Delft
The Netherlands
http://ens.ewi.tudelft.nl/

CAS-2019-00

M.Sc. Thesis

Particle Filter based Speaker Tracking in Distributed Pairwise Microphone Networks

Lantian Kou

Abstract

The particle filter (PF) algorithm is appropriate to solve the problem of speaker tracking in a reverberant and noisy environment using distributed pairwise microphone networks. First, complete the tracking task based on PF algorithm in centralized manner, a processing center is required to collect the signal from all microphones to carry out the PF processing. The computation complexity and time consumption of the particle filter algorithm are relatively high, mainly because of the large number of particles exploited in the filtering process since the effectiveness and accuracy of the particle filter particularly rely on the sample set size. However, almost all the existing particle filtering algorithms exploit the fixed number of particles, especially in the field of acoustic source tracking. To deal with this matter, Kullback-Leibler distance (KLD) sampling method was utilized as an adaptation technique to adjust the sample size instead of setting fixed number. Two approaches based on particle filter algorithm for tracking speaker in distributed way are proposed. Compared to the centralized scheme, each microphone pair in the distributed network executes the local PF individually and exchanges local weights or posterior parameters among neighboring nodes to efficiently achieve the global estimate of the sound source position. Finally, simulation experiments demonstrate these two methods are feasible to track the speaker in distributed microphone networks with a variable number of particles.

Particle Filter based Speaker Tracking in Distributed Pairwise Microphone Networks

Thesis

submitted in partial fulfillment of the
requirements for the degree of

Master of Science

in

Electrical Engineering

by

Lantian Kou
born in Chengdu, China

This work was performed in:

Circuits and Systems Group
Department of Microelectronics & Computer Engineering
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Delft University of Technology

Delft University of Technology
Department of
Microelectronics & Computer Engineering

The undersigned hereby certify that they have read and recommend to the Faculty of Electrical Engineering, Mathematics and Computer Science for acceptance a thesis entitled **"Particle Filter based Speaker Tracking in Distributed Pairwise Microphone Networks"** by **Lantian Kou** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: 29 November 2019

Chairman: _____
dr.ir. R. Heusdens

Advisor: _____
dr.ir. R. Heusdens

Committee Members: _____
dr.ir. R.C. Hendriks

_____
dr.ir. H. Driessen

Abstract

The particle filter (PF) algorithm is appropriate to solve the problem of speaker tracking in a reverberant and noisy environment using distributed pairwise microphone networks. First, complete the tracking task based on PF algorithm in centralized manner, a processing center is required to collect the signal from all microphones to carry out the PF processing. The computation complexity and time consumption of the particle filter algorithm are relatively high, mainly because of the large number of particles exploited in the filtering process since the effectiveness and accuracy of the particle filter particularly rely on the sample set size. However, almost all the existing particle filtering algorithms exploit the fixed number of particles, especially in the field of acoustic source tracking. To deal with this matter, Kullback-Leibler distance (KLD) sampling method was utilized as an adaptation technique to adjust the sample size instead of setting fixed number. Two approaches based on particle filter algorithm for tracking speaker in distributed way are proposed. Compared to the centralized scheme, each microphone pair in the distributed network executes the local PF individually and exchanges local weights or posterior parameters among neighboring nodes to efficiently achieve the global estimate of the sound source position. Finally, simulation experiments demonstrate these two methods are feasible to track the speaker in distributed microphone networks with a variable number of particles.

Acknowledgments

I would like to thank my supervisor dr.ir. Richard Heusdens for giving me a lot of patient guidance, numerous helpful instructions and support during the project.

I would like to thank dr.ir. Richard C. Hendriks for supporting me in this project.

I would like to thank dr.ir. Hans Driessen for the support and encouragement in this project.

I would like to thank my family for their support and trust towards me.

Lantian Kou
Delft, The Netherlands
29 November 2019

Contents

List of Figures

List of Tables

1 Introduction

A traditional microphone array consists of multiple microphones placed on fixed positions with regular geometry topology which limits their applications. Alternatively, the microphone network has been developed recently, which is composed of multiple microphone arrays randomly placed in space. Utilizing microphone arrays to collect the audio signals in indoor environment can effectively suppress noise and reverberation effects, and can also extract spatial information about the acoustic source.

Over the past years, acoustic source tracking with microphone arrays has become an important part in many fields. For example, in audio/video conference system [1, 2, 3], the location information can guide the camera to track the talker in the field of view and provide steering information for the microphone arrays to enhance the desired audio signal in the specific direction. In addition, in smart surveillance systems [4, 5], due to the influence of illumination and deformation, the video tracking has unstable performance. Considering the sound propagation is not affected by these factors, the approaches of acoustic source tracking can be used to compensate for the deficiencies to improve the accuracy of monitor system. Moreover, in the fields of robot audition [6, 7] and hearing aids [8], the algorithms of acoustic source tracking are applied for speech enhancement.

1.1 Acoustic Source Localization

Traditional approaches to localize the acoustic source in microphone arrays can be divided into direct and indirect methods, respectively.

1.1.1 Direct Method

The direct method attempts to estimate the position directly by the received signal at the current time with beamforming-based approach, mainly subdivided into two categories: delay-and-sum beamforming and filtering-and-sum beamforming. The conventional delay-and-sum beamforming considers all microphones to have an identical frequency response with equal amplitude weights, whereas the filtering-and-sum beamforming exploits different gains to each receivers [9].

For instance, steered response power-phase transform (SRP-PHAT) [10] is one of the direct methods, which searches for the source position in the whole space to maximize the output of a steered delay-and-sum beamformer, where the phase transform algorithm utilized to sharpen the peak. The drawback of SRP-PHAT algorithm is the large amount of calculation. Improving the real-time performance by changing the search strategy was proposed in [11, 12].

A microphone array for speaker localization with six microphones operated as a

steered filter-and-sum beamformer was discussed in [13], however, the performance of localization has declined when the speakers is very near or far to the microphone array.

1.1.2 Indirect Method

The indirect method is based on the time-delay estimate (TDE) approach, which usually has two steps. Firstly, estimating the time difference of arrival (TDOA) which is the time difference between the sound source arriving at the different microphones in each of the same microphone array, respectively. Secondly, solving a series of nonlinear hyperbolic equations, with the known geometry structures and the position information of microphone arrays, to obtain the estimation of position. This method has the advantages of simple structure and low calculation complexity.

Two typical indirect approaches are the well-known generalized cross-correlation (GCC) method [14] and adaptive eigenvalue decomposition (AED) method [15]. More specifically, GCC method obtains the TDOA by calculating the correlation between multiple synchronous audio signals in the same microphone array. By improving the performance in a noisy and reverberant environment, GCC and phase transform (GCC-PHAT) method was introduced in [14] which added appropriate weighting function with cross power spectrum to sharpen the peak of cross-correlation. AED method estimates the direct paths of the impulse responses in the spatial domain, which contains the information of time delay between the acoustic source and the microphones signals in each array, where the impulse responses are calculated as the eigenvector related to the minimum eigenvalue of the covariance matrix of the receiver signals.

The acoustic source localization methods merely convert the signal frame at current time into a certain form of localization function, and then searching for the peak point corresponding to the position of sound source. In practical applications, the presence of background noise and room reverberation often leads to multiple peaks in the localization function, which causes the pseudo sources in the search area. In some cases, the amplitude of the pseudo peaks could be even larger than the amplitude of the peak related to the actual sound source, resulting in low accuracy and poor performance of acoustic source localization.

1.2 Acoustic Source Tracking

In order to solve the problems mentioned above in localization method, the tracking approach can be exploited to estimate the smooth trajectory of the acoustic source.

Unlike the approaches of the acoustic source localization, the speaker tracking methods not only utilize the audio signal received at the current time but also adopt the observations obtained from the previous time. The Bayesian-based filtering algorithm is recommended in acoustic source tracking by utilizing a series of past data and current observations. Indeed, the tracking approach regards the spatial information of the speaker as a state vector updated with time, which dynamically estimates the position of the sound source since the pseudo sound sources cannot be constantly consistent with the characteristics of the state vector.

In [16], the location of the sound source was estimated preliminarily by linear motion model, then smoothing the estimation by Kalman filter with time-invariant measurement matrix in observation function. However, the accuracy of this method performs not well in the field of speaker tracking since it only regards the state space model as a linearization model. Extended Kalman filter (EKF) utilizes the first-order Taylor series expansion to linearize the non-linear observation function [17]. When EKF linearizes the system, it needs to solve Jacobian matrix of the non-linear function which leads to the complicated calculation, and the approximations result in large errors and even divergence. Unscented Kalman filter (UKF) [18] approximates the probability distribution of the system state by a set of determined sample points, and then exploits these sample points to obtain further statistical information of the system state through a non-linear model. UKF can achieve second-order approximation of state distribution by mean and covariance. In the problem of speaker tracking in reverberant and noisy environment, the performance of these algorithms is seriously degraded due to the effect of the spurious TDOA observations due to the reverberation and noise.

The sequential Monte Carlo-based particle filter (PF) algorithm can be utilized in nonlinear and non-Gaussian systems [19], and PF can recursively estimate the posterior probability density of the unknown target position conditioned on all past observations and the received data at the present moment. The PF method was firstly introduced into solving the speaker tracking problem in [3], and a general framework for tracking an acoustic source in reverberant environment was proposed in [20]. With the development in the field of machine learning, combining an acoustic map (AM) with a classification map (CM) based on spectral signatures has been utilized for tracking an acoustic source by PF algorithm in the noisy environment, especially in the environment that consists of persistent and high energy interferences [21].

To track speaker based on PF algorithm in centralized manner, a processing center is required to collect the signal from all nodes in the microphone networks to carry out the PF for tracking. This centralized processing method has a large communication burden and high requirements for data processing capacity in the central unit. In addition, once the exception occurs in the processing center, the entire network will not be able to complete the tracking task [22].

Compared to the centralized microphone arrays, each pair in the distributed microphone network runs the local PF individually and exchanges some local data among neighboring nodes to efficiently achieve the global estimate of the sound source position.

In [23], a consensus-based DPF is designed to track the speaker via the global coherent field observations. However, the distributed calculation of the global particle weights requires large data transmission and increases the communication burden as large amount of particles. In [24], a distributed EKF-based PF (DEKPF) is proposed to linearize the observation model. In the DEKPF, the Chong-Mori-Chang track-fusion theorem is utilized to fuse the local statistics of local posterior distributions. However, EKF calculation is required for each particle sampled. When the number of particles is large, the calculation amount is abundant. Additionally, in [25] another DPF is designed for the speaker tracking through approximating the posterior distribution by a Gaussian distribution and the multiple-hypothesis model is used and improved to combat the noise and reverberation. Moreover, [22] makes a literature survey about

the classification of different distributed particle filter algorithms.

As above mentioned, almost all the existing CPF and DPF algorithms exploit the fixed number of samples, especially in the field of tracking the acoustic source. The required size of particles is set according to prior experience in advance and cannot be changed during the filtering process. Usually, in order to ensure the high filtering accuracy of the system and meet the application requirements, we should set a relatively large number of particles. Nevertheless, the computation complexity of the particle filter algorithm is proportional to the number of particles.

When the size of the particle set can be adaptively changed during the filtering process under certain conditions of performance, the number of particles participating in the filtering can be effectively adjusted, thereby achieving the purpose of reducing the computation complexity and time consumption. The literature [26] introduced an approach of controlling the number of particles by setting two sets of particle filters in parallel with different numbers of particles, and then calculating the difference of some statistics results of the two sets. Another method introduced in [27] is to determine the number of particles by the Kullback-Leibler distance (KL-distance), with certain probability and error, between the sample-based estimated posterior probability density and the true posterior density in the sampling process.

1.3 Research Statement and Thesis Outline

The goals of this thesis are addressed as: To track the speaker in a noisy and reverberant environment utilizing the particle filtering algorithms in centralized scheme and distributed manner; To adaptively adjust the sample set size during the progress of particle filtering in speaker tracking problem.

The rest of the thesis is organized as follows: Chapter 2 describes some background theories; Chapter 3 introduces the method of particle filtering in centralized way and the approach to dynamically adjust the sample size of particles in detail, and provides the implementation of tracking speaker; Chapter 4 presents the adaptive distributed particle filter with variable sample size and its implementation; finally, conclusion and suggestion for future work can be found in Chapter 5.

2 Background

This chapter outlines the basis of speaker tracking based on microphone network.

2.1 General Framework

In the problem of speaker tracking, the microphones at different positions are used to pick up the voice signal and the processing unit is to extract the observation information. The dynamic state is modeled to describe the motion state of the speaker over time, which is utilized in filtering algorithms. The diagram of acoustic source tracking can be shown in Figure 2.1.

The spatial information of the moving target (such as position and velocity) is a random state vector changed over time. A system model is utilized to describe the dynamic characteristics of the sound source and an observation model is exploited to express the relationship between the sound source state and the observed data [19].

Set $\mathbf{x}_k \in \mathbb{R}^{n_x}$ as the n_x dimensional state vector of the acoustic source at time k, $\mathbf{z}_k \in \mathbb{R}^M$ is the M dimensional observation vector $\mathbf{z}_k = (z_{1,k}, \ldots, z_{M,k})^{\mathrm{T}}$ at time k, where M is the number of microphone pairs in microphone network. Then, the dynamic characteristics of the system model and observation model can be described as

$$\mathbf{x}_k = f_k(\mathbf{x}_{k-1}, \mathbf{u}_k) \tag{2.1}$$
$$\mathbf{z}_k = g_k(\mathbf{x}_k, \mathbf{v}_k) \tag{2.2}$$

The formula (2.1) is the system model, which is also called the dynamical motion model in the tracking problem. It mainly describes the law of the state transition over time, $f_k : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ is the state transition function which usually is Markov process, $\mathbf{u}_k \in \mathbb{R}^{n_u}$ is the n_u dimensional process noise which mainly represents the uncertainty of motion. Equation (2.2) is an observation model that describes the relationship between the observation $\mathbf{z}_k$ and source state $\mathbf{x}_k$, $g_k : \mathbb{R}^{n_x} \times \mathbb{R}^{n_v} \to \mathbb{R}^M$ is the observation function, $\mathbf{v}_k \in \mathbb{R}^{n_v}$ is the observation noise.

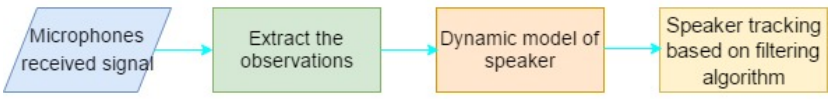


Figure 2.1: Tracking diagram

2.2 Dynamic Motion Model

2.2.1 Dynamic Model

The position of a random moving acoustic source is in three-dimensional Cartesian coordinate. Since the height of the speaker is often fixed, it is reasonable to consider the dynamic motion model that the source has no velocity in the vertical direction. The state vector of the target at any time k can be defined as $\mathbf{x}_k = [x_k, y_k, \dot{x}_k, \dot{y}_k, z_k]^T$, where the talker's position is $(x_k, y_k, z_k)^T$ and the corresponding velocity is $(\dot{x}_k, \dot{y}_k)^T$ are included. The motion model is to represent the time-varying location of a speaker moving in a room environment. One simple motion model but has been shown to work well for the movement of a speaker is Langevin model proposed in [28, 20]. Langevin equation originally describes Brownian motion, the random movement of a particle in fluid. It corresponds to the discrete process when the motion state is a Markov process. In the Langevin model the source motion is assumed to be identical and independent in each of the Cartesian coordinates. This motion is described as [24]

$$\begin{aligned} \mathbf{x}_k &= f_k(\mathbf{x}_{k-1}, \mathbf{u}_k) && (2.3)\\ &= \mathbf{A}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_k && (2.4) \end{aligned}$$

where $\mathbf{u}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_5)$ is a zero-mean Gaussian process noise with covariance $\mathbf{I}_5$, and matrix $\mathbf{A}$ and $\mathbf{B}$ are defined as

$$\mathbf{A} = \begin{bmatrix} \mathbf{I}_2 & a\Delta T \otimes \mathbf{I}_2 & \mathbf{0} \\ \mathbf{0} & a \otimes \mathbf{I}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 1 \end{bmatrix}; \mathbf{B} = \begin{bmatrix} b\Delta T \otimes \mathbf{I}_2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & b \otimes \mathbf{I}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 1 \end{bmatrix} \tag{2.5}$$

where $\mathbf{I}_m$ is the m-order identity matrix, $\otimes$ stands for the Kronecker product, $\Delta T = T_0/f_s$ is the time interval between two consecutive location estimates (T_0 and f_s denote the frame length in samples and sampling frequency, respectively), and the parameters a and b are given by

$$\begin{cases} a = e^{-\beta \Delta T} \\ b = \bar{v}\sqrt{1 - a^2} \end{cases} \tag{2.6}$$

where β and $\bar{v}$ are the rate constant and the steady-state root-mean-square(RMS) velocity, respectively.

Then the position of acoustic source can be expressed as

$$\mathbf{r}_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_k \tag{2.7}$$

2.2.2 Transition function

Since the dynamic model is a Markov process, the state transition distribution can be expressed as

$$p(\mathbf{x}_k|\mathbf{x}_{k-1}) \sim \mathcal{N}(\mathbf{x}_k; \mathbf{A}\mathbf{x}_{k-1}, \mathbf{B}\boldsymbol{\Sigma}_{\mathbf{u}_k}\mathbf{B}^{\mathrm{T}}) \tag{2.8}$$

where $\boldsymbol{\Sigma}_{\mathbf{u}_k} = \mathbf{I}_5$ and $\mathbf{A}$ and $\mathbf{B}$ are defined in (2.5).

2.3 Signal Model

This section provides the signal model which simulates the audio signals received at microphones in the indoor environment.

2.3.1 Signal Model

Assume that a microphone network which consists of M pairwise microphone arrays in a reverberant and noisy environment. Let $m \in \{1, \ldots, M\}$ and $p \in \{1, 2\}$ denote the pth microphone in the microphone pair m. The signal received from a single acoustic source can be modeled as

$$y_m^p(t) = s(t) \star h_m^p(t) + n_m^p(t) \tag{2.9}$$

where $s(t)$ is the source signal, $h_m^p(t)$ is the room impulse response (RIR) from the source to the specified microphone, $n_m^p(t)$ is the additive noise, and $\star$ represents the convolution symbol.

Considering there exists walls in a rectangular room (enclosure), the microphone will not only receive the sound signal through the direct path but also collecting the gradually weakened signals from the multipath due to the reflections that occur after encountering the walls during propagation. This phenomenon calls a reverberation effect [29]. A simple illustration is depicted in Figure 2.2, the solid line represents the direct way and dashed lines perform the multipath during sound propagation.

The RIR from the source to any microphone can be rewritten in terms of two components, direct path and multipath, as [20]

$$h_m^p(t) = \frac{1}{4\pi d_m^p(t)}\delta(t - \tau_m^p(t)) + g_m^{p,rev}(t) \tag{2.10}$$

where $d_m^p(t) = \|\mathbf{r}(t) - \mathbf{r}_m^p\|$ is the Euclidean distance between the location $\mathbf{r}(t)$ of the acoustic source and the position $\mathbf{r}_m^p$ of pth microphone in the microphone pair m at time t, $\tau_m^p(t) = d_m^p(t)/c$ is the time delay of direct path, c is the speed of sound in air, and $g_m^{p,rev}(t)$ is the overall reverberant impulse response due to the multipath effect.

The RIR can be divided into three major parts as shown in Figure 2.3: direct sound dominates in the aspect of first arriving sound at microphone; early reflections have relatively strong and low-density reflections; late reverberation whose reflection density increases rapidly and holds a long tail with an exponential decline.

In practice, although the speech is non-stationary, it is assumed that the speech signal is quasi-stationary when processed by framing. Hence, in order to track the

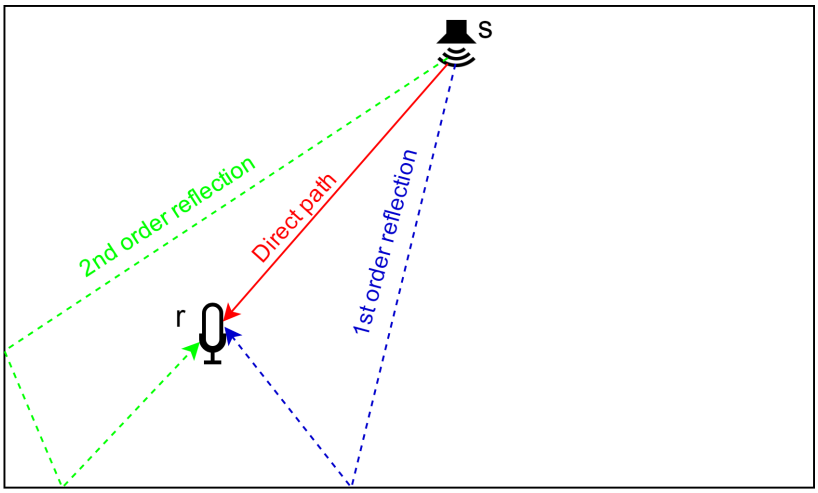


Figure 2.2: Room reverberation

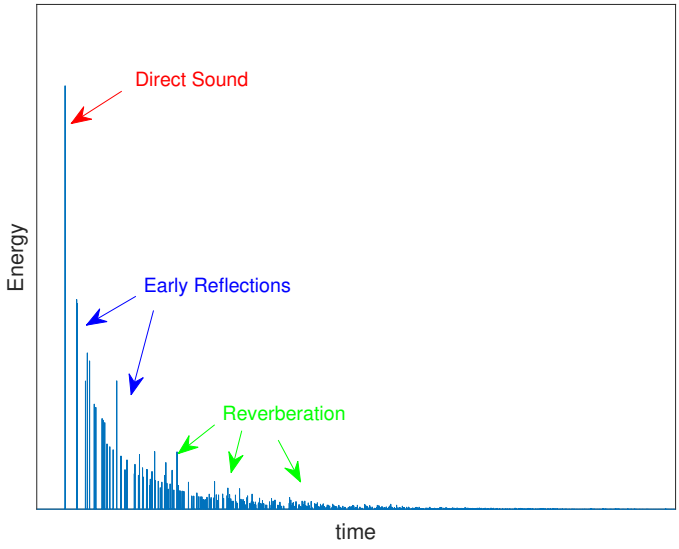


Figure 2.3: Room impulse response (RIR)

acoustic source, the signal received at each microphone is processed by frames. Let k denote the time index of frames, N_f is the length of samples in each frame, and the received signal in the pth microphone of the mth microphone pair at frame k is

$$\mathbf{y}_{m,k}^{p} = [y_m^p(kN_f), y_m^p(kN_f + 1), \ldots, y_m^p(kN_f + N_f - 1)] \tag{2.11}$$

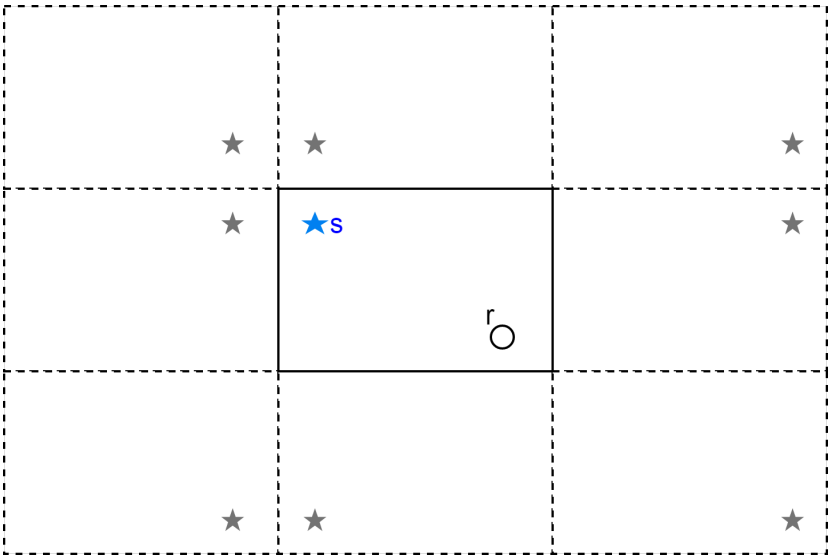


Figure 2.4: Image source pattern (the solid box is the original room)

2.3.2 RIR Based on Image Method

The direct path and multipath exist together to result in reverberation when receiving the sound signals at each microphone. The image method for simulating the reverberation effects in a room was introduced in [30]. The room is assumed to be a rectangular enclosure with an acoustic source. The basic idea of image method is when a wall exists, the reflection effect of the sound source is treated as placing an image symmetrically on the opposite side of the wall. Therefore, the reflection effect becomes the direct sound emitting from the mirror source to the microphone. Accordingly, the sound signal reaches the microphone through multiple walls and can be calculated by repeating the image process as illustrated in Fig.2.4, the blue star represents the actual sound source while the grey stars performs the images.

Due to the absorption of air and walls, when the source signal propagates in the air and reflects on the walls, the signal arriving at the microphone is an attenuated and delayed version of the source signal [31]. Moreover, the received signal is a linear superposition of the signal from direct path and all reflected paths.The RIR can be represented by

$$h_m^p(t) = \sum_{\mathbf{g}=0}^{1} \sum_{\mathbf{r}=-\infty}^{\infty} \beta_{x1}^{|n-q|} \beta_{x2}^{|n|} \beta_{y1}^{|l-j|} \beta_{y2}^{|l|} \beta_{z1}^{|m-k|} \beta_{z2}^{|m|} \times \frac{\delta[t - (\|\mathbf{R_g} + \mathbf{R_r}\|)/c]}{4\pi \|\mathbf{R_g} + \mathbf{R_r}\|} \tag{2.12}$$

where $\mathbf{R_g}$ represents eight possible transmission distance in terms of the triplet $\mathbf{g} = (q, j, k)$ which means the mirror source is the image of the acoustic source in that direction when the value is 1 in any corresponding dimension in $\mathbf{g}$, and $\mathbf{r} = (n, l, m)$ is the integer triplet as reflection order which represents the limit of image quantity.

Let $\mathbf{r}(t) = [x(t), y(t), z(t)]^{\mathrm{T}}$ and $\mathbf{r}_m^p = [x_m^p, y_m^p, z_m^p]^{\mathrm{T}}$ are the location of speaker and microphone respectively, the $\mathbf{R_g}$ and $\mathbf{R_r}$ can be expressed as

$$\mathbf{R_g} = (x(t) - x_m^p + 2qx_m^p, y(t) - y_m^p + 2jy_m^p, z(t) - z_m^p + 2kz_m^p)^T \tag{2.13}$$

$$\mathbf{R_r} = 2(nL_x, lL_y, mL_z)^T \tag{2.14}$$

where(L_x, L_y, L_z) are the room size. Thus, when the reflection order is N, there exists $8 * (2N + 1)^3$ different paths.

Image method is the most common method for simulating RIR due to the straightforward concept and efficient computation.

2.4 Time Difference of Arrival Observation

The speaker tracking based on filtering algorithm mainly consists of two steps: first, the audio signal received by each microphone pair is utilized to extract some characteristics including the spatial information of the sound source such as the time difference of arrival (TDOA) which estimates the time difference of voice arriving to the microphones at different positions; then, using TDOA candidates as the observation information combined with the geometric structure of the microphone network to estimate the position of the sound source by filtering algorithm.

At present, a variety of time delay estimation methods have been proposed, including generalized cross-correlation (GCC) function [14], least mean square (LMS) method [32] and adaptive eigenvalue decomposition (AED) algorithm [15]. Due to the effectiveness and simplicity, the GCC method is widely used to estimate time delay by constructing a generalized cross-correlation function with speech signals received by different microphones. Then the TDOA candidate is obtained by searching for the maximum point of the cross-correlation function.

2.4.1 Observation Model

The observation model describes the relationship between the state of the acoustic source and observation values. At time k, each node has a local observation $\mathbf{z}_k$ of the unknown source state $\mathbf{x}_k$. The observation model can be described as [14]

$$\mathbf{z}_k = g_k(\mathbf{x}_k, \mathbf{v}_k) \tag{2.15}$$

where g_k is the unknown observation function and $\mathbf{v}_k$ is the observation noise.

2.4.2 Generalized Cross-correlation Function

The generalized cross-correlation (GCC) is used to estimate the time delay for providing observation measurements. At time k, the GCC function at a microphone pair can be expressed as [14]

$$R_{m,k}(\tau) = \int \Phi_m(f) Y_m^1(f) Y_m^{2*}(f) e^{j2\pi f\tau} df \tag{2.16}$$

where $\Phi_m(f)$ is the weighting term in the frequency domain, $Y_m^1(f)$ and $Y_m^2(f)$ are Fourier transforms of $\mathbf{y}_{m,k}^1$ and $\mathbf{y}_{m,k}^2$, $*$ denotes the complex conjugation. The common weighting terms $\Phi_m(f)$ are smoothed coherence transform (SCOT) and phase transform (PHAT) as [14]

$$\Phi_m^{SCOT}(f) = \frac{1}{\sqrt{Y_m^1(f)Y_m^{2*}(f)}} \tag{2.17}$$

$$\Phi_m^{PHAT}(f) = \frac{1}{|Y_m^1(f)Y_m^{2*}(f)|} \tag{2.18}$$

where the PHAT coefficient aims to sharpen the peak of the GCC function by whitening the cross-power spectrum in the frequency domain, which is in order to obtain a more accurate estimation and is widely used in practice.

The TDOA measurement at node m is $\hat{\tau}_{m,k}$ that is estimated by exploring the potential TDOA candidates τ that maximizes the GCC function, which relates to the largest peak of $R_{m,k}(\tau)$

$$\hat{\tau}_{m,k} = \arg \max_{\tau \in [-\tau_m^{\max}, \tau_m^{\max}]} R_{m,k}(\tau) \tag{2.19}$$

where the maximum possibility of time delay between the microphones in the same pair is $\tau_m^{\max} = \|\mathbf{r}_m^1 - \mathbf{r}_m^2\|/c$, $\mathbf{r}_m^1$ and $\mathbf{r}_m^1$ are the positions of the two microphones in mth pair, respectively.

When indoor noise and reverberation are not serious, the time delay corresponding to the maximum peak value of the GCC function is the TDOA related to the real acoustic source. On the other hand, when the noise or reverberation is severe, the GCC function tends to present multiple spurious peaks, and these pseudo peaks can even cover the peak corresponding to the true sound source [28].In this situation, if the TDOA estimation was only obtained from the maximum peak value of the GCC function, it is prone to produce erroneous results, which leads to incorrect sound source position information. In order to suppress the adverse effects of the pseudo peaks generated by noise or reverberation, [28] and [20] proposed the approach to select multiple delays as candidate TDOAs based on several large peaks of the GCC function. Specifically, the values of peak amplitudes in the GCC function are arranged in order from large to small, and then several delays with the large peak amplitudes are picked out to constitute the observations of the node m, that is

$$\mathbf{z}_{m,k} = \{\hat{\tau}_{m,k}^1, \hat{\tau}_{m,k}^2, \ldots, \hat{\tau}_{m,k}^{N_g}\} \tag{2.20}$$

where $\hat{\tau}_{m,k}^g$ is the estimation of time delay corresponding to the gth largest peak of GCC function $R_{m,k}(\tau)$, $g = 1, 2, \ldots, N_g$ is the index of the TDOA candidates.

2.4.3 Multi-Hypothesis Likelihood Function

Consider the likelihood function $p(\mathbf{z}_{m,k}|\mathbf{x}_k)$ of the microphone pair m, it describes the probability of occurrence of observation $\mathbf{z}_{m,k}$ under given state $\mathbf{x}_k$). In the problem of speaker tracking, the multi-hypothesis likelihood function is widely utilized with

multiple TDOA candidates [28]. In fact, at most one candidate is related to the real sound source in equation (2.20), while the remaining candidates are derived from indoor noise or reverberation. In order to distinguish between these two cases, the label $\{c_{m,k}^g\}_{g=1}^{N_g}$ is introduced to indicate the association of each TDOA candidate $\mathbf{z}_{m,k}$ and its corresponding source, and establish the following hypothetical events [28]:

$$\mathcal{H}_0 = \left\{c_{m,k}^g = F; g = 1, 2, \ldots, N_g\right\} \tag{2.21}$$

$$\mathcal{H}_g = \{c_{m,k}^g = T; c_{m,k}^{g'} = F; g, g' = 1, 2, \ldots, N_g, g \neq g'\} \tag{2.22}$$

where the hypothesis $\mathcal{H}_0$ means that no candidate is associated with the real sound source, and the hypothesis $\mathcal{H}_g$ denotes that candidate g corresponding to the true target. $c_{m,k}^g = T$ indicates the $\hat{\tau}_{m,k}^g$ is related to the real sound source, whereas $c_{m,k}^g = F$ represents the $\hat{\tau}_{m,k}^g$ is derived from the pseudo sound source.

All the hypotheses in (2.21) and (2.22) are mutually exclusive and exhaustive. Thus, the likelihood function $p(\mathbf{z}_{m,k}|\mathbf{x}_k)$ can be expressed as

$$p(\mathbf{z}_{m,k}|\mathbf{x}_k) = \sum_{g=0}^{N_g} q_g p(\mathbf{z}_{m,k}|\mathbf{x}_k, \mathcal{H}_g) \tag{2.23}$$

where $q_g = p(\mathcal{H}_g)$ is the prior probability of hypothesis $\mathcal{H}_g$, and $\sum_{g=0}^{N_g} q_g = 1$, the large value of q_0 represents the position of true source is often not among the candidates due to the high level of noise and reverberation in the room.

Generally, the likelihoods of TDOA candidates related to the true or spurious source are subject to the Gaussian distribution and uniform distribution over the range $[-\tau_m^{\max}, \tau_m^{\max}]$, respectively [28]

$$p(\hat{\tau}_{m,k}^g|\mathbf{x}_k, c_{m,k}^g = T) = \mathcal{N}(\hat{\tau}_{m,k}^g; \tau_{m,k}(\mathbf{x}_k), \sigma^2) \tag{2.24}$$

$$p(\hat{\tau}_{m,k}^g|\mathbf{x}_k, c_{m,k}^g = F) = \frac{1}{2\tau_m^{\max}} \tag{2.25}$$

where $\tau_{m,k}(\mathbf{x}_k) = (\|\mathbf{r}_k - \mathbf{r}_m^1\| - \|\mathbf{r}_k - \mathbf{r}_m^2\|)/c$ is the theoretic TDOA at microphone pair m, σ^2 is the observation noise variance.

It can be assumed that, under the condition of given state $\mathbf{x}_k$ and hypothesis $\mathcal{H}_g$, each candidate TDOA in the observation $\mathbf{z}_{m,k}$ is independent. The likelihood functions under multiple hypotheses are stated as

$$p(\mathbf{z}_{m,k}|\mathbf{x}_k, \mathcal{H}_0) = \prod_{g=1}^{N_g} p(\hat{\tau}_{m,k}^g|\mathbf{x}_k, c_{m,k}^g = F) \tag{2.26}$$

$$p(\mathbf{z}_{m,k}|\mathbf{x}_k, \mathcal{H}_g) = p(\hat{\tau}_{m,k}^g|\mathbf{x}_k, c_{m,k}^g = T) \times \prod_{g'=1, g\neq g'}^{N_g} p(\hat{\tau}_{m,k}^{g'}|\mathbf{x}_k, c_{m,k}^{g'} = F) \tag{2.27}$$

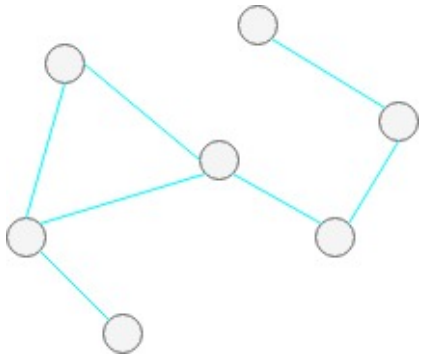


Figure 2.5: Example of a communication graph

Then substituting equations (2.24) and (2.25) into (2.26) and (2.27), the equations (2.26) and (2.27) can be rewritten as

$$p(\mathbf{z}_{m,k}|\mathbf{x}_k, \mathcal{H}_0) = \frac{1}{(2\tau_m^{\max})^{N_g}} \tag{2.28}$$

$$p(\mathbf{z}_{m,k}|\mathbf{x}_k, \mathcal{H}_g) = \frac{1}{(2\tau_m^{\max})^{N_g-1}} \mathcal{N}(\hat{\tau}_{m,k}^g; \tau_{m,k}(\mathbf{x}_k), \sigma^2) \tag{2.29}$$

Finally, substituting equations (2.28) and (2.29) into (2.23), the multiple-hypothesis likelihood function can be expressed as

$$p\left(\mathbf{z}_{m,k}\,|\mathbf{x}_k\right) = (2\tau_m^{\max})^{1-N_g} \left(\frac{q_0}{2\tau_m^{\max}} + \sum_{g=1}^{N_g} q_g \mathcal{N}\left(\hat{\tau}_{m,k}^g; \tau_{m,k}\left(\mathbf{x}_k\right), \sigma^2\right) \right) \tag{2.30}$$

2.5 Communication Graph

In distributed schematic, the topology of the microphone network is modeled as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with the vertex set $\mathcal{V} = \{1, 2, \ldots, M\}$ and edge set $\mathcal{E} \subseteq \{\mathcal{V} \times \mathcal{V}\}$. Each vertex $m \in \mathcal{V}$ denotes a unique node, and each edge $(m, m') \in \mathcal{E}$ represents that pair m and m' can communicate with each other to exchange data. The neighborhood of node m is defined as the subset of vertex vector $\mathcal{N}_m = \{m' \in \mathcal{V} \mid (m, m') \in \mathcal{E}\}$. The degree of node m in network is d_m means the number of edges that are incident to the vertex. In the example shown in Figure 2.5, the circles represent the vertices and the connecting lines perform the edges.

Adaptive Particle Filtering with Variable Sample Set Size

3

In this chapter, the fundamentals of Bayesian filtering and particle filtering are presented. Subsequently, the method using Kullback-Leibler distance (KL-distance) is introduced to adaptively adjust the number of particles during the sampling process in particle filtering. Finally, simulations of speaker tracking based on particle filtering and adapted sample set size through KL-distance approach are implemented.

3.1 Bayesian Filter

Bayesian filtering is a classic method to solve tracking problems, and it is also the theoretical basis of this thesis. Bayesian filtering recursively estimates the posterior probability density of the source state with the observation information, and then obtains the optimal estimation of the characteristics of the sound source.

3.1.1 Bayesian Theorem

As introduced in the previous chapter, the state space model of the speaker tracking system could be expressed as

$$\mathbf{x}_k = f_k(\mathbf{x}_{k-1}, \mathbf{u}_k) \tag{3.1}$$

$$\mathbf{z}_k = g_k(\mathbf{x}_k, \mathbf{v}_k) \tag{3.2}$$

From the perspective of Bayesian statistics, the system model can alternatively be described by the transition probability density function $p(\mathbf{x}_k|\mathbf{x}_{k-1})$, and the observation model can also be expressed by the measurement probability density function $p(\mathbf{z}_k|\mathbf{x}_k)$. It is depicted in Figure 3.1 that the transition function is Markov process and the observation is merely related to the current source state.

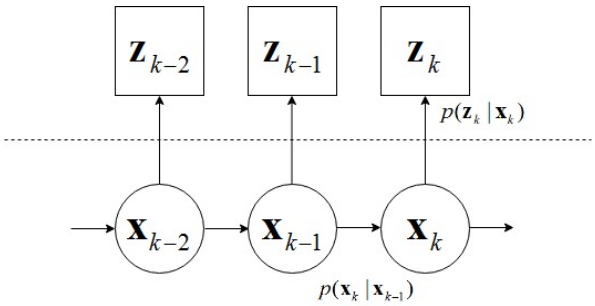


Figure 3.1: State space model

The main content of Bayesian theorem is that assuming $\mathbf{z}_k$ is a vector of independent and identical distributed measurements holds a conditional probability density $p(\mathbf{x}_k|\mathbf{z}_k)$ relative to the state vector $\mathbf{x}_k$, then the posterior probability density of the unknown parameters $\mathbf{x}_k$ is given by

$$p(\mathbf{x}_k) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1})d\mathbf{x}_{k-1} \tag{3.3}$$

$$p(\mathbf{x}_k|\mathbf{z}_k) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k)}{p(\mathbf{z}_k)} \tag{3.4}$$

where $p(\mathbf{x}_k)$ is the prior and $p(\mathbf{x}_0)$ is assumed to be known, $p(\mathbf{z}_k|\mathbf{x}_k)$ is the likelihood function given the state vector $\mathbf{x}_k$, $p(\mathbf{z}_k) = \int p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k)d\mathbf{x}_k$ is the normalized constant.

The Bayesian theorem in equation (3.4) shows the posterior is proportion to the product of the likelihood and the prior.

3.1.2 Bayesian Recursive Filtering

Bayesian filtering is to recursively estimate the posterior probability density $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ of the target state $\mathbf{x}_k$ using a prior knowledge and all available observations, where $\mathbf{z}_{1:k} = (\mathbf{z}_1, \ldots, \mathbf{z}_k) \in \mathbb{R}^{M\times k}$ denotes the cumulative measurement matrix of the system from the beginning up to the moment k. The posterior probability density $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ contains all the available information for the state $\mathbf{x}_k$, which can obtain an optimal estimate of the position under certain criteria. Assuming the initial prior probability density $p(\mathbf{x}_0)$ is known and the posterior probability density at time $k-1$ is $p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})$, then the posterior probability density $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ at time k can be obtained from the following recursions:

$$p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})d\mathbf{x}_{k-1} \tag{3.5}$$

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})} \tag{3.6}$$

where $p(\mathbf{z}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})d\mathbf{x}_k$ is the normalized constant.

The equations (3.5) and (3.6) are the prediction step and update step, respectively. The prediction step is to apply the system model function to achieve the prior probability density, and the update step is to exploit the latest measurement to correct the posterior probability density.

The detailed derivation process is as follows:

(1) Prediction step: obtain $p(\mathbf{x}_k|\mathbf{z}_{1:k-1})$ from $p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})$

$$p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{x}_k, \mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})d\mathbf{x}_{k-1} \tag{3.7}$$

$$= \int p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_{1:k-1})p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})d\mathbf{x}_{k-1} \tag{3.8}$$

Since the state $\mathbf{x}_k$ is independent with all the past observations $\mathbf{z}_{1:k-1}$ when given the $\mathbf{x}_{k-1}$, then

$$p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})d\mathbf{x}_{k-1} \tag{3.9}$$

(2) Update step: obtain $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ from $p(\mathbf{x}_k|\mathbf{z}_{1:k-1})$
First, recall the formulation in conditional probability

$$P(A|B,C) = \frac{P(A,B,C)}{P(B,C)} \tag{3.10}$$

$$= \frac{P(A,B,C)}{P(C)}\frac{P(C)}{P(B,C)} \tag{3.11}$$

$$= \frac{P(A,B|C)}{P(B|C)} \tag{3.12}$$

After obtaining the observation measurements $\mathbf{z}_k$, utilizing the Bayesian theorem to update the posterior probability density $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ from the prior probability density $p(\mathbf{x}_k|\mathbf{z}_{1:k-1})$ is

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) = p(\mathbf{x}_k|\mathbf{z}_k,\mathbf{z}_{1:k-1}) \tag{3.13}$$

$$= \frac{p(\mathbf{z}_k,\mathbf{x}_k|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})} \tag{3.14}$$

$$= \frac{p(\mathbf{z}_k|\mathbf{x}_k,\mathbf{z}_{1:k-1})p(\mathbf{x}_k|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})} \tag{3.15}$$

Since the state $\mathbf{z}_k$ is independent with all the past observations $\mathbf{z}_{1:k-1}$ when given the $\mathbf{x}_k$, then the posterior density is

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})} \tag{3.16}$$

where the $p(\mathbf{z}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})dx_k$

3.1.3 Estimation

The posterior probability density $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ contains all the available information for the state $\mathbf{x}_k$, which can achieve an optimal estimate of the position under certain criteria. The common methods of estimation are maximum a posterior (MAP) and minimum mean square error (MMSE) [33, 34],

$$\hat{\mathbf{x}}_k^{\text{MAP}} = \arg\max_{\mathbf{x}_k} p(\mathbf{x}_k|\mathbf{z}_{1:k}) \tag{3.17}$$

$$\hat{\mathbf{x}}_k^{\text{MMSE}} = E\{\mathbf{x}_k|\mathbf{z}_{1:k}\} = \int \mathbf{x}_k p(\mathbf{x}_k|\mathbf{z}_{1:k})d\mathbf{x}_k \tag{3.18}$$

Bayesian recursive filtering provides only a conceptual solution to the tracking problem in general. For dynamic systems in practice, the solution of posterior probability density can not be analytically determined [35]. When the state space model is linear and Gaussian, Kalman filtering is the optimal solution of Bayesian filtering, while for the case of the state space model is non-linear and non-Gaussian, particle filtering has better estimation performance [35]. In acoustic source tracking the performance of Kalman filtering is seriously degraded due to the effect of the spurious TDOA observations resulting from the reverberation and noise. Moreover, in the actual speaker tracking problem, the state space model can be non-linear and non-Gaussian due to the TDOA observations and reverberant environment. Thus, the particle filtering is introduced to solve the problem of speaker tracking[3].

3.2 Particle Filter

Particle filter algorithm is based on Monte Carlo simulation and recursive Bayesian filtering [19]. The results obtained by particle filtering have high accuracy and are close to the optimal estimation for the nonlinear system.

3.2.1 Importance Sampling

The core idea of the PF algorithm is to generate a set of random samples (that is, particles) in the space, and to assign them certain weights. Then approximating the posterior distribution $p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$ of the acoustic source state by a set of weighted particles $\{\mathbf{X}^i_{0:k}, w^i_k\}^{N_s}_{i=1}$ as

$$p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) \approx \sum_{i=1}^{N_s} w^i_k \delta(\mathbf{x}_{0:k} - \mathbf{X}^i_{0:k}) \tag{3.19}$$

where δ is the Dirac delta function, $\{\mathbf{X}^i_{0:k}\}^{N_s}_{i=1}$ is the set of particles with associated weights $\{w^i_k\}^{N_s}_{i=1}$ which can be described as

$$\mathbf{X}^i_{0:k} \sim q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) \tag{3.20}$$

$$w^i_k \propto \frac{p(\mathbf{X}^i_{0:k}|\mathbf{z}_{1:k})}{q(\mathbf{X}^i_{0:k}|\mathbf{z}_{1:k})} \tag{3.21}$$

where the particles are drawn from a predefined importance probability density, which is called proposal function or importance density $q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$.

The proposal function needed to be designed to ensure the weight is bounded. As a rule of thumb, choosing the $q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$ similar to $p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$ to minimize the variance of weight w^i_k [36]. Represent the posterior density by a set of randomly chosen weighted samples (particles) is shown in Figure 3.2. As the number of samples becomes very large, the characterization of samples becomes more likely to the true posterior density.

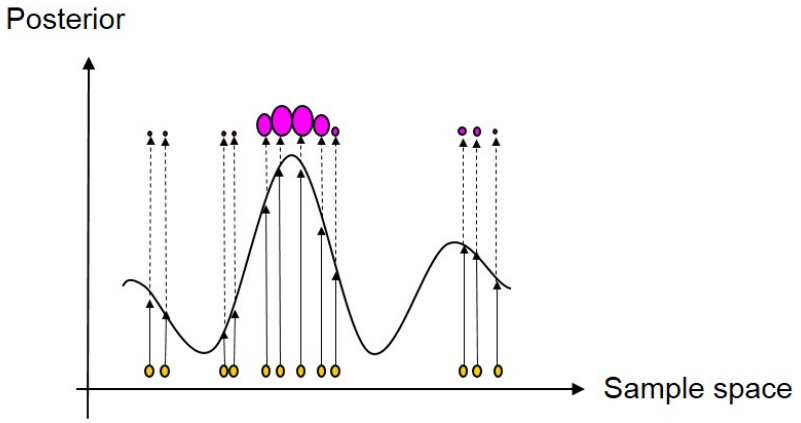


Figure 3.2: Represent the posterior density by a set of weighted particles

3.2.2 Sequential Importance Sampling Algorithm

In importance sampling, the estimation of the posterior density requires the observation data at all times, which makes the calculation of the entire algorithm complicated. In order to facilitate recursive calculations, sequential importance sampling (SIS) was proposed [19]. The SIS algorithm samples the particle set according to the importance density, and updates the weight of the particle through the current observation value. The particles and their corresponding weights are weighted and summed to approximate the acoustic source state.

Suppose that the proposal function is chosen to factorize such that

$$q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) = q(\mathbf{x}_k, \mathbf{x}_{0:k-1}|\mathbf{z}_{1:k}) \tag{3.22}$$
$$= q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k})q(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k}) \tag{3.23}$$
$$= q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k})q(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1}) \tag{3.24}$$

The equation (3.24) indicates that the particle set $\{\mathbf{X}_{0:k}^i\}_{i=1}^{N_s} \sim q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$ can be obtained by augmenting the existing set $\{\mathbf{X}_{0:k-1}^i\}_{i=1}^{N_s} \sim q(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1})$ with the current set $\{\mathbf{X}_k^i\}_{i=1}^{N_s} \sim q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k})$.

Hence, the posterior probability density can also be rewritten as

$$p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_{1:k}|\mathbf{x}_{0:k})p(\mathbf{x}_{0:k})}{p(\mathbf{z}_{1:k})} \tag{3.25}$$

$$= \frac{p(\mathbf{z}_k, \mathbf{z}_{1:k-1}|\mathbf{x}_{0:k})p(\mathbf{x}_{0:k})}{p(\mathbf{z}_{1:k})} \tag{3.26}$$

$$= \frac{p(\mathbf{z}_k|\mathbf{z}_{1:k-1}, \mathbf{x}_{0:k})p(\mathbf{z}_{1:k-1}|\mathbf{x}_{0:k})p(\mathbf{x}_{0:k})}{p(\mathbf{z}_{1:k})} \tag{3.27}$$

$$= \frac{p(\mathbf{z}_k|\mathbf{z}_{1:k-1}, \mathbf{x}_{0:k})p(\mathbf{z}_{1:k-1}|\mathbf{x}_{0:k})p(\mathbf{x}_k|\mathbf{x}_{0:k-1})p(\mathbf{x}_{0:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})p(\mathbf{z}_{1:k-1})} \tag{3.28}$$

$$= \frac{p(\mathbf{z}_k|\mathbf{z}_{1:k-1}, \mathbf{x}_{0:k})p(\mathbf{x}_k|\mathbf{x}_{0:k-1})p(\mathbf{z}_{1:k-1}|\mathbf{x}_{0:k-1})p(\mathbf{x}_{0:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})p(\mathbf{z}_{1:k-1})} \tag{3.29}$$

$$= \frac{p(\mathbf{z}_k|\mathbf{z}_{1:k-1}, \mathbf{x}_{0:k})p(\mathbf{x}_k|\mathbf{x}_{0:k-1})p(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})} \tag{3.30}$$

$$= \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})} \tag{3.31}$$

$$= p(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1})\frac{p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{z}_k|\mathbf{x}_k)}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})} \tag{3.32}$$

$$\propto p(\mathbf{x}_{0:k-1}|\mathbf{z}_{1:k-1})p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{z}_k|\mathbf{x}_k) \tag{3.33}$$

where $p(\mathbf{z}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})d\mathbf{x}_k$ is the normalized constant.

By substituting (3.24) and (3.33) into (3.21), the weight update equation can be rewritten by

$$w_k^i \propto \frac{p(\mathbf{X}_{0:k}^i|\mathbf{z}_{1:k})}{q(\mathbf{X}_{0:k}^i|\mathbf{z}_{1:k})} \tag{3.34}$$

$$= \frac{p(\mathbf{z}_k|\mathbf{X}_k^i)p(\mathbf{X}_k^i|\mathbf{X}_{k-1}^i)p(\mathbf{X}_{0:k-1}^i|\mathbf{z}_{1:k-1})}{q(\mathbf{X}_k^i|\mathbf{X}_{0:k-1}^i, \mathbf{z}_{1:k})q(\mathbf{X}_{0:k-1}^i|\mathbf{z}_{1:k-1})} \tag{3.35}$$

$$= w_{k-1}^i \frac{p(\mathbf{z}_k|\mathbf{X}_k^i)p(\mathbf{X}_k^i|\mathbf{X}_{k-1}^i)}{q(\mathbf{X}_k^i|\mathbf{X}_{0:k-1}^i, \mathbf{z}_{1:k})} \tag{3.36}$$

Moreover, when the proposal function is under Markov assumption which means the importance density depends only on the $\mathbf{x}_{k-1}$ and $\mathbf{z}_k$ as

$$q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{1:k}) = q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_k) \tag{3.37}$$

The most common and sub-optimal choice of proposal function is the state transition function $q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_k) = p(\mathbf{x}_k|\mathbf{x}_{k-1})$ as the bootstrap filter [19].

Finalllly, the particles and weights can be updated by

$$\mathbf{X}_k^i \sim q(\mathbf{x}_k|\mathbf{X}_{k-1}^i, \mathbf{z}_k) \tag{3.38}$$

$$w_k^i \propto w_{k-1}^i \frac{p(\mathbf{z}_k|\mathbf{X}_k^i)p(\mathbf{X}_k^i|\mathbf{X}_{k-1}^i)}{q(\mathbf{X}_k^i|\mathbf{X}_{k-1}^i, \mathbf{z}_k)} \tag{3.39}$$

The particle filter based on SIS algorithm is shown in Algorithm 1.

Algorithm 1 Particle filter based on sequential importance sampling algorithm

Input: $\mathbf{X}_{k-1}^i$, w_{k-1}^i, $\mathbf{z}_k$
Output: $\mathbf{X}_k^i$, w_k^i, $\hat{\mathbf{x}}_k$
Initial: $\mathbf{X}_0^i \sim p(\mathbf{x}_0)$, $w_0^i = 1/N_s$, $i = 1, 2, \ldots, N_s$
Iteration process: $k = 1, 2, \ldots$

1: Predict step, importance sampling (draw particles): $\mathbf{X}_k^i \sim q(\mathbf{x}_k|\mathbf{X}_{k-1}^i, \mathbf{z}_k)$

2: Update step, update important weight: $w_k^i \propto w_{k-1}^i \frac{p(\mathbf{z}_k|\mathbf{X}_k^i)p(\mathbf{X}_k^i|\mathbf{X}_{k-1}^i)}{q(\mathbf{X}_k^i|\mathbf{X}_{k-1}^i, \mathbf{z}_k)}$

3: Normalize particle weight: $\tilde{w}_k^i = w_k^i / \sum_{j=1}^{N_s} w_k^j$

4: Calculate the state estimation: $\hat{\mathbf{x}}_k = \sum_{i=1}^{N_s} \tilde{w}_k^i \mathbf{X}_k^i$

Table 3.1: Particle filter based on sequential importance sampling algorithm

3.2.3 Sequential Importance Resampling Algorithm

The main problem of sequential importance sampling-based particle filter (SIS-PF) algorithm is the phenomenon of particle degradation [35]. The particle degradation occurs after multiple iterations. Most of the particles would have small weights which have negligible influence while a few particles have large weights. The result of particle degradation is that the number of effective particles in the state space becomes smaller, which causes a large amount of computation to be wasted on the update of the particles with small weights and results in performance degradation of the algorithm. The effective sample size is introduced as the measure of degeneracy described by [36]

$$N_{eff} = \frac{N}{1 + \mathrm{var}(w_k^{i\,*})} \tag{3.40}$$

where $w_k^{i\,*} = p(\mathbf{X}_k^i|\mathbf{z}_{1:k})/q(\mathbf{X}_k^i|\mathbf{X}_{k-1}^i, \mathbf{z}_k)$ represents the true weights. A small value of N_{eff} indicates severe degeneracy due to the large variance of weights. Because the exact value of N_{eff} cannot be evaluated, a common method is to approximate this value using normalized weights as [35]

$$\hat{N}_{eff} = \frac{1}{\sum_{i=1}^{N_s} (w_k^i)^2} \tag{3.41}$$

In order to reduce the influence of particle degradation on the algorithm performance, one method is to increase the number of particles, which is effective but causes a sharp increase in the amount of calculation, rarely used in practice [35]. Another way is to choose the appropriate proposal function. The most convenient approach is to use the state transition density as the importance probability density [19]. In addition, resampling technique is utilized. The basic idea of resampling is to eliminate the particles with small importance weights and duplicate the particles with large weights

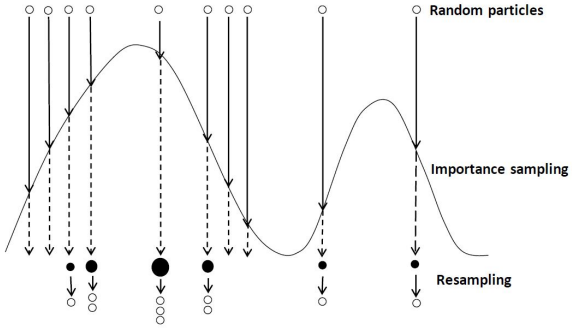


Figure 3.3: Resampling process

when the $\hat{N}_{eff}$ is below a certain threshold, which is given by

$$\{\mathbf{X}_k^i, w_k^i\}_{i=1}^{N_s} \rightarrow \{\mathbf{X}_k^{i^*}, \frac{1}{N_s}\}_{i=1}^{N_s} \tag{3.42}$$

where $\{\mathbf{X}_k^{i^*}\}_{i=1}^{N_s}$ are the particles after resampling.

The process of resampling is shown in Figure 3.3, we can see that the particles with small importance weights are abandoned and the particles with large importance weights are duplicated at the same time after the importance sampling.

The particle filter based on sequential importance resampling (SIR) algorithm is shown in Algorithm 2.

Algorithm 2 Particle filter based on sequential importance resampling algorithm

Input: $\mathbf{X}_{k-1}^i$, w_{k-1}^i, $\mathbf{z}_k$
Output: $\mathbf{X}_k^i$, w_k^i, $\hat{\mathbf{x}}_k$
Initial: $\mathbf{X}_0^i \sim p(\mathbf{x}_0)$, $w_0^i = 1/N_s$, $i = 1, 2, \ldots, N_s$
Iteration process: $k = 1, 2, \ldots$

1: Predict step, importance sampling (draw particles): $\mathbf{X}_k^i \sim q(\mathbf{x}_k|\mathbf{X}_{k-1}^i, \mathbf{z}_k)$

2: Update step, update important weight: $w_k^i \propto w_{k-1}^i \frac{p(\mathbf{z}_k|\mathbf{X}_k^i)p(\mathbf{X}_k^i|\mathbf{X}_{k-1}^i)}{q(\mathbf{X}_k^i|\mathbf{X}_{k-1}^i,\mathbf{z}_k)}$

3: Normalize particle weight: $\tilde{w}_k^i = w_k^i / \sum_{j=1}^{N_s} w_k^j$

4: Calculate the state estimation: $\hat{\mathbf{x}}_k = \sum_{i=1}^{N_s} \tilde{w}_k^i \mathbf{X}_k^i$

5: Resampling: Compute the $\hat{N}_{eff} = \frac{1}{\sum_{i=1}^{N_s} (\tilde{w}_k^i)^2}$, if $\hat{N}_{eff} < N_{th}$, resampling the particles as $\{\mathbf{X}_k^i, \tilde{w}_k^i\}_{i=1}^{N_s} \rightarrow \{\mathbf{X}_k^{i^*}, \frac{1}{N_s}\}_{i=1}^{N_s}$

Table 3.2: Particle filter based on sequential importance resampling algorithm

Resampling reduces degeneracy phenomenon, however, new problems arise. For example, the complex computation limits the performance of parallelization. In addition,

the particles with large importance weights are repeatedly selected many times leading to loss of diversity.

3.3 Adaptive Particle Filtering with Variable Sample Set Size through KLD-sampling

The computation complexity and time consumption of the particle filter algorithm are relatively high, mainly because of the large number of particles exploited in the filtering process since the computation complexity of the particle filter approach is proportional to the number of particles. Traditional particle filtering algorithm exploits a fixed number of particles for filtering. The required amount of particles is set according to prior experience in advance and cannot be changed during the filtering process. In order to ensure the high filtering accuracy of the system and meet the application requirements, we should set a relatively large number of particles.

When the number of particles can be dynamically changed during the filtering process according to the conditions of the system at different times and under the premise of ensuring certain filtering performance, the number of particles participating in the filtering can be effectively reduced, thereby achieving the purpose of reducing the filtering complexity and improving the filtering time performance. According to this idea, the literature [26] proposed a method of controlling the number of particles by setting two sets of particle filters in parallel with different numbers of particles, and then calculating the difference of some statistics results of the two sets. If the difference is greater than a certain threshold, the filtering accuracy is considered to be low, and the number of particles is increased according to a preset ratio; if the difference is less than a certain threshold, the filtering accuracy is considered to exceed the accuracy requirement, and the number of particles is reduced according to a preset ratio. This method requires simultaneous use of two sets of particle parallel filtering, one of which does not actually contribute to the filtering result. Therefore, this approach of controlling the number of particles is too expensive in terms of computation.

Another method introduced in [27] is to determine the number of samples by the Kullback-Leibler distance (KL-distance), with certain probability and error, between the sample-based estimated posterior probability density and the true posterior density in the sampling process. The key idea of the Kullback-Leibler distance sampling (KLD-sampling) approach is to use more particles when the posterior distribution area spreads out over a wide range of values, that is to say, when the uncertainty is high; on the contrary, to use fewer particles when the posterior distribution area is concentrated, in other words, when the uncertainty is low.

The goal of adaptive particle filtering with variable sample set size is to determine the number of samples such that, with a certain probability, the error between the true posterior and the sample-based representation is bounded.

3.3.1 The KL-distance

The KL-distance can also be called relative entropy introduced in [37]. The entropy of a random variable is a measure of the uncertainty of the random variable, which

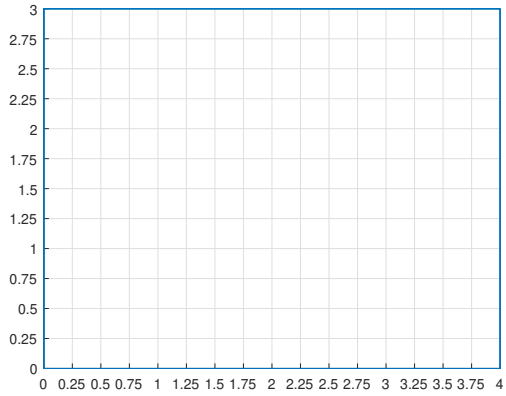


Figure 3.4: Example of bins in the two dimension room

does not depend on the actual values taken by the random variable, but only on the probabilities. The KL-distance is a measure of the difference between two probability distributions:

$$D_{KL}(\hat{p}, p) = \sum_{x} \hat{p}(x) \log \frac{\hat{p}(x)}{p(x)} \tag{3.43}$$

In the above equation (3.43), KL-distance is always non-negative and it is zero if and only if the two distributions are identical. It is not a metric of true distance since it is not symmetric and does not satisfy the triangle property. Nonetheless, it is accepted as a standard measure for the difference between two probability distributions [37]. However, the problem is that we can not obtain the true posterior distribution, which means the KL-distance can not be calculated directly. The mathematical knowledge about likelihood ratio test provides a method to connect the KL-distance and the number of samples [38]. [27] represented a sample-based approximation of the predictive belief as an estimation for the posterior.

3.3.2 Likelihood Ratio Test

In order to discretize the posterior probability distribution, we divide the room space into different small areas which are called the bins. An example is illustrated in Figure 3.4, each grid represents a bin.

Suppose that N_s samples are drawn from a discrete distribution with N_{BS} different bins, N_{BS} represents the number of bins with support in whose the bins contain at least one particle inside. Each bin has $N_{s,j}$ particles, where $j = \{1, 2, \ldots, N_{BS}\}$. Let the vector $\mathbf{N}_{SBS} = (N_{s,1}, N_{s,2}, \ldots, N_{s,N_{BS}})$ denotes the number of particles in each bin with support, which is according to multinomial distribution $\mathbf{N}_{SBS} \sim Multinomial_{N_{BS}}(N_s, \mathbf{p})$, where $\mathbf{p} = \{p_1, p_2, \ldots, p_{N_{BS}}\}$ performs the true probability of each bin and p is short for $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ in this section. The maximum likelihood estimate (MLE) of $\mathbf{p}$ is given by $\hat{\mathbf{p}} = \mathbf{N}_{SBS}/N_s$. The total number of samples can be

expressed as

$$N_s = \sum_{j=1}^{N_{BS}} N_{s,j} \tag{3.44}$$

From mathematical knowledge [38], the likelihood ratio statistic λ_{N_s} for testing p is

$$\lambda_{N_s} = \prod_{j=1}^{N_{BS}} \left(\frac{p_j}{\hat{p}_j}\right)^{N_{s,j}} \tag{3.45}$$

To inverse the ratio λ_{N_s} and then to take logarithm as

$$\log \frac{1}{\lambda_{N_s}} = \log \left(\prod_{j=1}^{N_{BS}} \left(\frac{\hat{p}_j}{p_j}\right)^{N_{s,j}}\right) \tag{3.46}$$

Since the $N_{s,j}$ is identical to $N_s\hat{p}_j$, the equation (3.46) can be rewritten as

$$-\log \lambda_{N_s} = \sum_{j=1}^{N_{BS}} N_{s,j} \log \left(\frac{\hat{p}_j}{p_j}\right) \tag{3.47}$$

$$= N_s \sum_{j=1}^{N_{BS}} \hat{p}_j \log \left(\frac{\hat{p}_j}{p_j}\right) \tag{3.48}$$

From the (3.43) and (3.48) we can find the relationship between the likelihood ratio statistic and KL-distance in terms of the MLE and true distribution:

$$-\log \lambda_{N_s} = N_s D_{KL}(\hat{\mathbf{p}}, \mathbf{p}) \tag{3.49}$$

From the mathematical knowledge of likelihood ratio test in the book [38], the distribution of $-2\log \lambda_{N_s}$ tends to a chi-square distribution with degrees of freedom $N_{BS} - 1$ as $N_s \to \infty$:

$$-2\log \lambda_{N_s} \to_d \chi^2_{N_{BS}-1} \tag{3.50}$$

Let $P_{\mathbf{p}}\left(D_{KL}(\hat{\mathbf{p}}, \mathbf{p}) \leq \varepsilon\right)$ denote the probability that the KL- distance is less than or equal to certain error bound. When N_s samples come from the true posterior distribution, the relation between the number of samples and that probability can be derived from (3.49) is as follows:

$$P_{\mathbf{p}}\left(D_{KL}(\hat{\mathbf{p}}, \mathbf{p}) \leq \varepsilon\right) = P_{\mathbf{p}}\left(2N_s D_{KL}(\hat{\mathbf{p}}, \mathbf{p}) \leq 2N_s\varepsilon\right) \tag{3.51}$$

$$= P_{\mathbf{p}}\left(-2\log \lambda_{N_s} \leq 2N_s\varepsilon\right) \tag{3.52}$$

$$\approx P\left(\chi^2_{N_{BS}-1} \leq 2N_s\varepsilon\right) \tag{3.53}$$

The quantile $\chi^2_{N_{BS}-1,1-\delta}$ of the chi-square distribution with probability $1-\delta$ is given by

$$P\left(\chi^2_{N_{BS}-1} \leq \chi^2_{N_{BS}-1,1-\delta}\right) = 1 - \delta \tag{3.54}$$

We can see that if the sample size N_s is selected as $2N_s\varepsilon = \chi^2_{N_{BS}-1,1-\delta}$, we can obtain

$$P_{\mathbf{p}}\left(D_{KL}(\hat{\mathbf{p}}, \mathbf{p}) \leq \varepsilon\right) \approx 1 - \delta \tag{3.55}$$

The derivation process is summarized as follows: if we choose the number of samples N_s as

$$N_s = \frac{1}{2\varepsilon}\chi^2_{N_{BS}-1,1-\delta} \tag{3.56}$$

where the number of particles depends only on the number of bins with support (out of J total bins), we can ensure the KL-distance between the MLE and the true distribution is less than ε with probability $1 - \delta$.

The number of particles is related to the bin size Δ since the small size of bins results in the large number of bins with support, which increases the number of particles. In addition, the number of particles is inversely proportional to the KL-distance. It can be understood that when the sampled particles are very scattered, the required number of particles increases. Contrarily, when particles are used to converge together (concentration), the required number of sampled particles is reduced.

3.3.3 KLD-sampling in Particle Filter

The number of particles depends only on the number of bins with support, which means we can adjust the sample set during the sampling process by counting the bins with support. The algorithm of adaptive particle filter with variable sample set size through KLD-sampling is in Algorithm 3. In step 4 in this algorithm, we propose the method that starts to select the particle corresponding to the largest weight, and then choose the samples in descending order by weight. Note that we also need to set the maximum potential number of the particles, which guarantees to terminate when the bin size, error bound and probability are fixed in advance.

3.4 Simulation Results

In this section, to verify the validity of the proposed speaker tracking methods, some simulation experiments are executed.

3.4.1 Simulation Setup

The simulation environment is a typical room of size of $5m \times 5m \times 3m$ with $M = 8$ microphone pairs placed on the walls, and the location of microphones are known as prior information. As shown in Figure 3.5, the positions of microphones and speaker are located on the $x - y$ plane with the height of $1.5m$ (for the 2D source tracking). The spacing of each microphone pair is set to $0.6m$ and the speaker trajectory is a semi-circle moving from $(1, 2.5)m$ to $(3, 2.5)m$ with a radius of $1m$.

The source signal is a $3s$ female speech taken from the TIMIT database with a sampling frequency of 16kHz, and the frame length is $\Delta T = 32ms$. The speed of sound propagation is $340m/s$. The RIR from the speaker positions to microphones is simulated based on the image method. For the multi-hypothesis GCC-PHAT function,

Algorithm 3 SIR particle filter based on KLD-sampling algorithm

Input: $\mathbf{S}_{k-1} = \{\mathbf{X}_{k-1}^j, w_{k-1}^j\}_{j=1}^{N_{s,k-1}}$, $\mathbf{z}_k$, ε, δ, Δ, N_{min}
Output: $\mathbf{S}_k = \{\mathbf{X}_k^i, w_k^i\}_{i=1}^{N_{s,k}}$, $\hat{\mathbf{x}}_k$
Initial: $\mathbf{S}_k = \varnothing$, $N_{s,k} = 0$, $N_{KLD} = 0$, $N_{BS} = 0$, $i = 0$
Iteration process: $k = 1, 2, \ldots$

while ($N_{s,k} < N_{KLD}$ or $N_{s,k} < N_{min}$) **do**
  $N_{s,k} = N_{s,k} + 1$
  $i = N_{s,k}$
  Sample a particle with index j from the set $\mathbf{S}_{k-1}$
  Predict step, importance sampling: $\mathbf{X}_k^i \sim q(\mathbf{x}_k|\mathbf{X}_{k-1}^j, \mathbf{z}_k)$
  Update step, update important weight: $w_k^i \propto w_{k-1}^j \frac{p(\mathbf{z}_k|\mathbf{X}_k^i)p(\mathbf{X}_k^i|\mathbf{X}_{k-1}^j)}{q(\mathbf{X}_k^i|\mathbf{X}_{k-1}^j, \mathbf{z}_k)}$
  if $\mathbf{X}_k^i$ falls into empty bin b **then**
    $N_{BS} = N_{BS} + 1$
    b = non-empty
    $N_{KLD} = \frac{1}{2\varepsilon}\chi^2_{N_{BS}-1,1-\delta}$
  end if
end while
Normalize particle weight: $\tilde{w}_k^i = w_k^i / \sum_{l=1}^{N_{s,k}} w_k^l$
Calculate the state estimation: $\hat{\mathbf{x}}_k = \sum_{i=1}^{N_{s,k}} \tilde{w}_k^i \mathbf{X}_k^i$
Resampling: Compute the $\hat{N}_{eff} = \frac{1}{\sum_{i=1}^{N_{s,k}} (\tilde{w}_k^i)^2}$, if $\hat{N}_{eff} < N_{th}$, resampling the particles as $\{\mathbf{X}_k^i, \tilde{w}_k^i\}_{i=1}^{N_{s,k}} \rightarrow \{\mathbf{X}_k^{i^*}, \frac{1}{N_{s,k}}\}_{i=1}^{N_{s,k}}$ to produce $\mathbf{S}_k$

Table 3.3: SIR particle filter based on KLD-sampling algorithm

$N_g = 4$ TDOA candidates are extracted, and the initial prior probability of the hypothesis $\mathcal{H}_0$ is set as $q_0 = 0.25$. The observation noise deviation is $\sigma^2 = 5 \times 10^{-5}$. To model the dynamic motion of the speaker, parameters of the Langevin model are $\beta = 10s^{-1}$ and $\bar{v} = 1ms^{-1}$, respectively.

3.4.2 Performance Metrics

The root mean square error (RMSE) is chosen as the performance measure to indicate the deviation of the estimated positions from the truth, which is defined as [20]

$$RMSE = \sqrt{\frac{1}{K}\sum_{k}^{K} \|\mathbf{r}_k - \hat{\mathbf{r}}_k\|^2} \tag{3.57}$$

where the $\mathbf{r}_k$ and $\hat{\mathbf{r}}_k$ are the true position and the corresponding estimated position of the sound source, respectively. The lower RMSE value, the better performance of

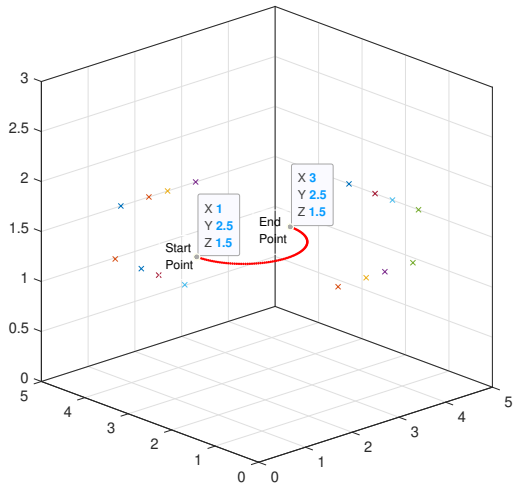


Figure 3.5: Microphone positions (crosses) and the speaker trajectory (semi-circle curve) in 3D

tracking.

3.4.3 Simulation Results

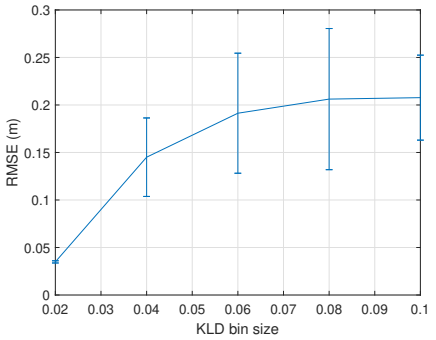
In this section, we present some experiments of speaker tracking and compare the performance between the SIR particle filtering and the SIR particle filtering based on KLD-sampling.

3.4.3.1 Parameter Settings of KLD-sampling

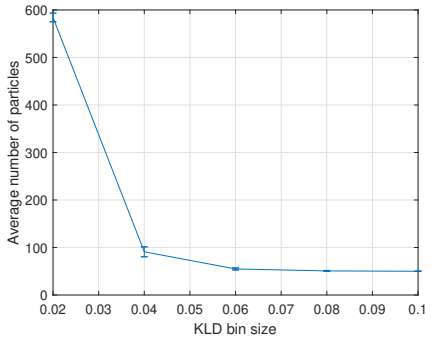
This part discusses the performance of tracking based on KLD-sampling in terms of the different choices of parameters. The critical parameters of the KLD-sampling method are the bin size Δ, the error bound ε, and the probability bound δ (the quantile is $1-\delta$). In each experiment, runs 20 Monte Carlo simulations, only one parameter is changed while the other two parameters are fixed, the noise level (signal-to-noise ratio, SNR) is SNR=20dB and reverberant time is $T_{60} = 0.2s$.

Figure 3.6(a) and Figure 3.6(b) show the influence of the different bin size Δ on the performance of KLD-sampling. The error bound $\varepsilon = 0.15$ and the quantile $1-\delta = 0.9$ are fixed. The required number of samples decreases with larger bin size since larger bin size causes smaller quantity of bins with support, thereby the corresponding tracking error rises as shown in Figure 3.6(a). The average number of particles has a dramatic decrease before the bin size increases to 0.4.

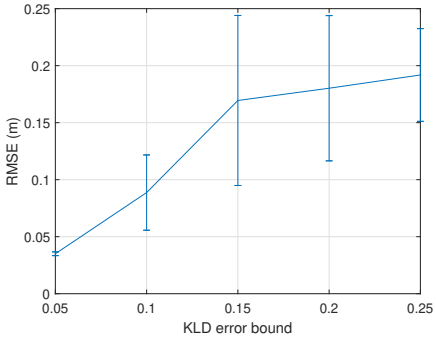
The bin size $\Delta = 0.05$ and the quantile $1-\delta = 0.9$ are fixed in Figure 3.6(c) and Figure 3.6(d). As can be seen in the formula (3.56), the required number of particles is inversely proportional to the error bound ε, which is consistent with the result in Figure 3.6(d). Similar to the result in Figure 3.6(a), the RMSE increases with the fewer



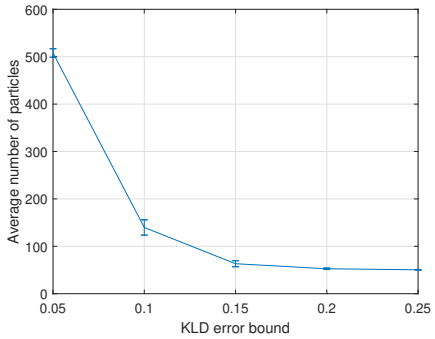
(a) RMSE versus bin size



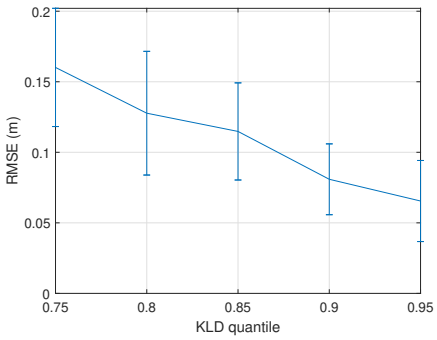
(b) Average number of particles versus bin size



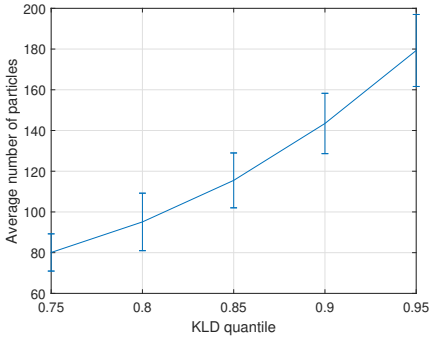
(c) RMSE versus error bound



(d) Average number of particles versus error bound



(e) RMSE versus quantile



(f) Average number of particles versus quantile

Figure 3.6: RMSE and average number of particles versus different settings of parameters in SIR particle filter based on KLD-sampling algorithm.

particles used which is shown in Figure 3.6(c). More specifically, the curve has declined drastically until the error bound reaches 0.15 from where the RMSE curve becomes a flat rising trend.

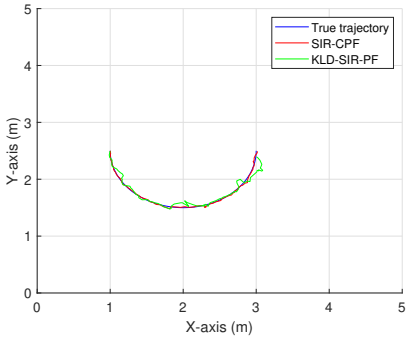
For fixed value of bin size $\Delta = 0.05$ and error bound $\varepsilon = 0.1$, the performance of KLD-sampling based on quantile $1 - \delta$ is presented in Figure 3.6(e) and Figure 3.6(f).

The required number of samples increases with larger quantile which satisfies the chi square distribution in equation (3.56). Likewise, the RMSE decreases with the larger size of samples in particle filtering. The number of particles varies less significantly with different values of probability bound δ compared with the changes of bin size Δ and error bound ε.

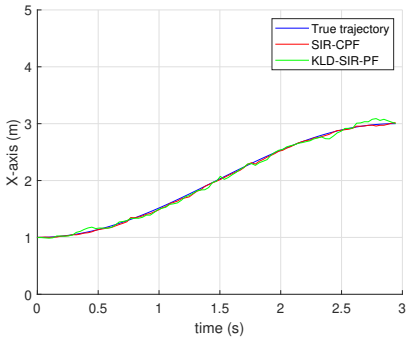
Therefore, we can set the parameters of KLD-sampling method as: the bin size $\Delta = 0.04$, the error bound $\varepsilon = 0.15$, and the quantile is $1 - \delta = 0.9$.

3.4.3.2 Speaker tracking results

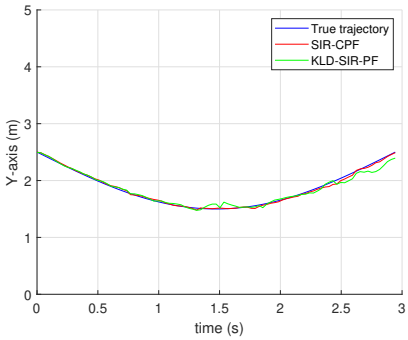
In this part, we make the comparisons of the performance between the SIR particle filtering and the SIR particle filtering based on KLD-sampling shown as in Figure 3.7. The SIR-PF has a fixed number of particles as 500, and the parameters of KLD-sampling method is set according to the conclusions of the previous section. Figure 3.7(a) illustrates both the two algorithms can successfully estimate the speaker trajectory. Figure 3.7(b) and Figure 3.7(c) are the tracking results in x-coordinate and y-coordinate, respectively. In Figure 3.7(d), we can see the adaptive adjustment of particle number by the KLD-sampling method during particle filtering progress. The RMSE results in Figure 3.7(e) shows that the SIR particle filtering has better performance of tracking accuracy since it holds 500 particles to perform the PF while the KLD-sampling approach utilizes around 100 particles in average through the progress. The results in Figure 3.7 demonstrates that the PF based on KLD-sampling algorithm is an effective approach to adjust the number of particles for the speaker tracking task within an acceptable error range.



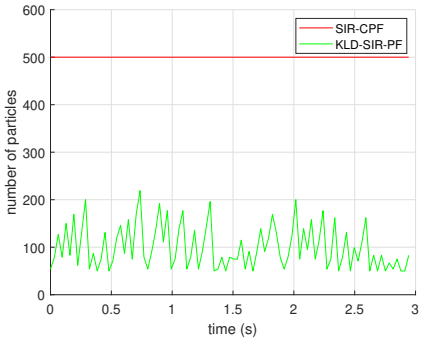
(a) x-y plane



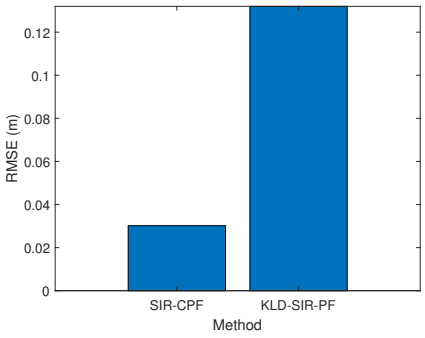
(b) x-coordinate



(c) y-coordinate



(d) number of particles



(e) RMSE results

Figure 3.7: Speaker tracking results with SNR=20dB.

Adaptive Distributed Particle Filtering with Variable Sample Set Size

4

The existing speaker tracking methods based on particle filter mostly adopt centralized processing mode, that is, all of the audio signals received by microphone arrays are transmitted to a processing center, where the task of sound source tracking is completed. The main advantage of this method is that it can exploit all the information of audio data from the entire microphone network to estimate the location of the sound source. However, this centralized processing method has a large communication burden, high requirements for data processing capacity, and the poor scalability of microphone networks. In addition, once the exception occurred in the central unit, the whole network will not be able to complete the tracking task [22]. Therefore, distributed particle filter (DPF) algorithms are appealing since all the nodes can exploit particle filtering individually to estimate a global state of target.

This chapter introduces the classification of various DPF algorithms based on different methods of data fusion, then recommends two DPF algorithms and implements the adaptation of the sample set size of particles. Finally, the simulations of speaker tracking based on DPF algorithms and comparisons with CPF algorithms are presented.

4.1 Data Fusion

Consider a microphone network composed of M microphone nodes randomly placed in space with the vertex set $\mathcal{V} = \{1, 2, \ldots, M\}$. Each microphone pair can receive and process the signal independently. Let the local observation of speaker at microphone pair m at time k is $\mathbf{z}_{m,k}$, and the relationship between local observation $\mathbf{z}_{m,k}$ and sound source state $\mathbf{x}_k$ can be described by the following local observation model:

$$\mathbf{z}_{m,k} = g_{m,k}(\mathbf{x}_k, \mathbf{v}_{m,k}), \forall m \in \mathcal{V} \tag{4.1}$$

where $g_{m,k}$ is the local observation function at node m, $\mathbf{v}_{m,k}$ is the corresponding observation noise which is independent of all past and current states.

Generally, the reliable estimation of the state of the speaker can not be obtained only by the observation of a single node [22]. Therefore, in practice, it is often necessary to analyze and synthesize the observation information of all or part of the nodes in the network, and then simultaneously acquire the global and reliable estimation of the state of the sound source. This process is called the data fusion [39].

Data fusion plays an important role in the field of distributed sensor networks and mainly includes the following three implementation architectures [39].

- DPF based on fusion center
  At every moment, each node in the network obtains the local estimation of the

state by local particle filter, and then transmits the local estimation to the fusion center. Finally, the fusion center merges the local estimation of each node to get the global estimation of the state. In this kind of algorithm, each node communicates directly with the fusion center. Since the nodes do not execute any particle filtering process, this method can not be regarded as real DPF, which means it also has the same problems as CPF [22].

- DPF based on leader agent
  In this kind of algorithm, information propagates and accumulates along a path composed of a series of adjacent nodes which saves the costs of energy [40]. Specifically, at each time only one active node, the leader agent (LA), performs particle filtering, which estimates the state of the target at the current time by using its own observation and the accumulated information from the previous agent; then, according to a certain node scheduling algorithm, the new LA for the next time is selected from its neighborhood nodes; finally, the posterior estimation of the state or the posterior parameters obtained by the LA is transferred to the next active node. However, when the topological structure of the network changes, an appropriate online node scheduling algorithm is needed to pick out the LA, which causes huge computation [22]. Moreover, how to restore the original information and how to choose another active node when the LA fails at a certain time also increase the complexity of the algorithm.

- DPF based on consensus
  In this algorithm, information is communicated between neighboring nodes. At each time, all microphone pairs in the network carry out particle filtering simultaneously to estimate the posterior probability density of the state. Specifically, each node can make an agreement on certain global quantities through the consensus algorithm, and then the quantities are adopted by each node to obtain the estimation of the global posterior probability density of the state by particle filter. Compared with the previous two DPF algorithms, the consensus based DPF algorithm has many advantages such as scalability and robustness. For example, scalability mainly refers to the communication bandwidth or power consumption of each node that does not depend on (or only partially depends on) the total number of network nodes since each node only communicates with its instantaneous neighbor nodes. In addition, when some nodes or communication links in the network are abnormal or damaged, the whole network can still work since each node performs the particle filtering individually [22].

4.2 Consensus Algorithm

In the distributed networks, consensus mainly refers to that each node reaches a unified agreement on some certain parameters, which often depend on the information of all nodes. The goal of the consensus algorithm is to make each node in the network exchange information with its neighbors continuously to obtain the global parameters. Consensus algorithms are iterative schemes for distributed computation and the most common types are the average consensus and max-consensus [22, 41, 42].

4.2.1 Average Consensus

Let $\mathbf{c}_{m,k}$ represent the local parameters of microphone pair $m \in \mathcal{V}$ at the time k. The goal of the average consensus algorithm is to achieve the average of the parameters of all the nodes by distributed iteration, that is, $\bar{\mathbf{c}}_k = \frac{1}{M}\sum_{m=1}^{M} \mathbf{c}_{m,k}$. In general, the iterative formula is defined as [41, 42]

$$\mathbf{c}_{m,k}(\ell+1) = \zeta_{mm}\mathbf{c}_{m,k}(\ell) + \sum_{m' \in \mathcal{N}_m} \zeta_{mm'}\mathbf{c}_{m',k}(\ell) \tag{4.2}$$

where the ℓ is the index of consensus iteration, $\zeta_{mm'}$ are the consensus weights between node m and m'. Let $\mathbf{c}_{m,k}(0) = \mathbf{c}_{m,k}$, repeat the iterative process of the above formula (4.2), the estimation $\mathbf{c}_{m,k}(\ell)$ at each node m can finally converge to the global average $\bar{\mathbf{c}}_k$ as

$$\lim_{\ell \to \infty} \mathbf{c}_{m,k}(\ell) = \bar{\mathbf{c}}_k, \forall m \in \mathcal{V} \tag{4.3}$$

where the equation (4.3) holds if and only if [41]

$$\mathbf{1}^T\boldsymbol{\zeta} = \mathbf{1}^T \tag{4.4}$$

$$\boldsymbol{\zeta}\mathbf{1} = \mathbf{1} \tag{4.5}$$

$$\rho(\boldsymbol{\zeta} - (1/M)\mathbf{1}\mathbf{1}^T) < 1 \tag{4.6}$$

where $\mathbf{1} = [1, 1, \ldots, 1]^T$, $\rho(\cdot)$ denotes the spectral radius of a matrix (the maximum absolute value of the eigenvalue), $\boldsymbol{\zeta}$ is the weight matrix of average consensus.

Under the constraint of formula (4.4), (4.5) and (4.6), the average consensus weight matrix has many choices. Different weight matrices affect the convergence speed of the average consensus algorithm. In practice, the most widely used weight matrices are max-degree weights and metropolis weights:

- Max-degree weights

$$\zeta_{mm'} = \begin{cases} \frac{1}{M} & \text{if } m' \in \mathcal{N}_m, \\ 1 - \frac{d_m}{M} & \text{if } m = m', \\ 0 & \text{otherwise.} \end{cases} \tag{4.7}$$

- Metropolis weights

$$\zeta_{mm'} = \begin{cases} \frac{1}{1+\max\{d_m, d_{m'}\}} & \text{if } m' \in \mathcal{N}_m, \\ 1 - \sum\limits_{m' \in \mathcal{N}_m} \zeta_{mm'} & \text{if } m = m', \\ 0 & \text{otherwise.} \end{cases} \tag{4.8}$$

The maximum-degree weights in the equation (4.7) require each node to know the total number of nodes in the network, while the metropolis weight matrix does not need to know the information about the entire network. The metropolis weights in (4.8) merely need to know the degree of the neighborhood of each node, thus the metropolis weights are employed to construct the weight matrix of average consensus in this thesis.

4.2.2 Max-consensus

Let $\mathbf{c}_{m,k}$ represent the local parameters of microphone pair $m \in \mathcal{V}$ at the time k. The goal of max-consensus is to obtain the maximum of the parameters $\mathbf{c}_k^{\max} = \max_{m \in \mathcal{V}} \mathbf{c}_{m,k}$ at all the pairs by distributed iteration, and the max-consensus iteration is defined as [22]

$$\mathbf{c}_{m,k}(\ell+1) = \max\left\{\mathbf{c}_{m,k}(\ell), \{\mathbf{c}_{m',k}(\ell)\}_{m' \in \mathbf{N}_m}\right\} \tag{4.9}$$

Unlike the average consensus, each $\mathbf{c}_{m,k}$ in (4.9) can converge to the global maximum $\mathbf{c}_k^{\max}$ within a finite number of iterations which does not exceed the diameter of the communication graph [22]. Therefore, in order to ensure that each node has the same result, the max-consensus is often used after the average consensus step.

4.3 Consensus-based DPF

In particle filtering, the posterior distribution of the acoustic source state is approximated by a set of weighted particles $\{\mathbf{X}_k^i, w_k^i\}_{i=1}^{N_s}$. And in consensus-based DPF, all nodes run particle filtering simultaneously and individually. Therefore, we can consider two types of quantities to be distributed computed: the particle weights and the parameters of the posterior.

4.3.1 Particle Weights Consensus-based DPF

The weights consensus-based DPF algorithm assumes that each node in the network shares the identical set of particles [22] and runs the local PF. For each particle set, its global weights are obtained by the consensus fusion algorithm. The specific steps are as follows:

(1) Prediction step: Suppose that each node holds the same particles $\{\mathbf{X}_{k-1}^i\}_{i=1}^{N_s}$ at time $k-1$. Draw the particles according to the state transition function

$$\mathbf{X}_{m,k}^i \sim p(\mathbf{x}_k|\mathbf{X}_{k-1}^i), \forall m \in \mathcal{V}, i = 1, 2, \ldots, M \tag{4.10}$$

which requires the local random number generators at each node are synchronized to ensure the entire network obtains the same pseudo-random numbers $\mathbf{X}_k^i = \mathbf{X}_{m,k}^i$ [22] such that the same particles are drawn in prediction step at each microphone pair.

(2) Update step: For each particle, the node $m \in \mathcal{V}$ calculates the local weights by the local observation $\mathbf{z}_{m,k}$ [22]

$$w_{m,k}^i = p(\mathbf{z}_{m,k}|\mathbf{X}_k^i) \tag{4.11}$$

The goal of DPF algorithm is to get the global weights of particles at each node

$$w_k^i = p(\mathbf{z}_k|\mathbf{X}_k^i) \tag{4.12}$$

(3) Fusion step: When the observations of each node in the network are independent under the condition of the given state $\mathbf{X}_k^i$, then the global likelihood can be decomposed into the product form of each local likelihood as

$$p(\mathbf{z}_k|\mathbf{X}_k^i) = \prod_{m=1}^{M} p(\mathbf{z}_{m,k}|\mathbf{X}_k^i) \tag{4.13}$$

Taking the logarithm of (4.13) yields

$$\ln w_k^i = \ln\left(\prod_{m=1}^{M} p(\mathbf{z}_{m,k}|\mathbf{X}_k^i)\right) \tag{4.14}$$

$$= \sum_{m=1}^{M} \ln p(\mathbf{z}_{m,k}|\mathbf{X}_k^i) \tag{4.15}$$

Then, taking the exponentiation of (4.15) and substituting (4.11), we can the expression of global weights

$$w_k^i = \exp\left(\ln w_k^i\right) \tag{4.16}$$

$$= \exp\left(\sum_{m=1}^{M} \ln p(\mathbf{z}_{m,k}|\mathbf{X}_k^i)\right) \tag{4.17}$$

$$= \exp\left(\sum_{m=1}^{M} \ln w_{m,k}^i\right) \tag{4.18}$$

It can be seen from the equation (4.18) that the calculation of the global weights of particles can be transformed into the exponential form of the sum of logarithm of the corresponding local weights, which can be computed by the average consensus algorithm.

Let $\varpi_{m,k}^i = \ln w_{m,k}^i$, $\bar{\varpi}_{m,k}^i = \frac{1}{M}\sum_{m=1}^{M} \varpi_{m,k}^i$, where the $\bar{\varpi}_k^i$ can be calculated by the average consensus in (4.2)

$$\varpi_{m,k}^i(\ell+1) = \zeta_{mm}\varpi_{m,k}^i(\ell) + \sum_{m'\in\mathcal{N}_m} \zeta_{mm'}\varpi_{m',k}^i(\ell) \tag{4.19}$$

Finally, the global weights can be rewritten as

$$w_k^i = \exp(M\bar{\varpi}_{m,k}^i) \tag{4.20}$$

If errors resulting from the consensus iteration are disregarded, this DPF based on weights consensus algorithm can obtain the same estimation performance as the centralized PF (Note that M has to be provided to each microphone pair in advance or

Algorithm 4 Particle weights consensus-based DPF algorithm

Input: $\mathbf{X}_{k-1}^i$, $\mathbf{z}_k$
Output: $\mathbf{X}_k^i$, w_k^i, $\hat{\mathbf{x}}_k$
Initial: $\mathbf{X}_0^i \sim p(\mathbf{x}_0)$, $w_0^i = 1/N_s$, $i = 1, 2, \ldots, N_s$
Iteration process: $k = 1, 2, \ldots$

1: Predict step, importance sampling (draw particles): $\mathbf{X}_k^i \sim p(\mathbf{x}_k|\mathbf{X}_{k-1}^i)$
2: Update step, update important weight: $w_{m,k}^i = p(\mathbf{z}_{m,k}|\mathbf{X}_k^i)$
3: Taking the logarithm of local weight: $\varpi_{m,k}^i = \ln w_{m,k}^i$
4: Calculate the $\bar{\varpi}_{m,k}^i$ by average consensus: $\varpi_{m,k}^i(\ell+1) = \zeta_{mm}\varpi_{m,k}^i(\ell) + \sum_{m' \in \mathcal{N}_m} \zeta_{mm'}\varpi_{m',k}^i(\ell)$
5: Calculate the global particle weights w_k^i by equation: $w_k^i = \exp(M\bar{\varpi}_{m,k}^i)$
6: Normalize particle weight: $\tilde{w}_k^i = w_k^i / \sum_{j=1}^{N_s} w_k^j$
7: Calculate the state estimation: $\hat{\mathbf{x}}_k = \sum_{i=1}^{N_s} \tilde{w}_k^i \mathbf{X}_k^i$
8: Resampling: Compute the $\hat{N}_{eff} = \frac{1}{\sum_{i=1}^{N_s} (\tilde{w}_k^i)^2}$, if $\hat{N}_{eff} < N_{th}$, resampling the particles as $\{\mathbf{X}_k^i, \tilde{w}_k^i\}_{i=1}^{N_s} \rightarrow \{\mathbf{X}_k^{i^*}, \frac{1}{N_s}\}_{i=1}^{N_s}$

Table 4.1: Particle weights consensus-based DPF algorithm

a distributed algorithm for counting the total number of nodes may be exploited) [22] since the calculation. On the other hand, the communication requirement is related to the number of particles since each particle weights are exchanged between neighbor nodes in consensus step.

The particle weights consensus-based DPF algorithm is shown in Algorithm 4.

4.3.2 Posterior Parameters Consensus-based DPF

Each node utilizes its local observation to run PF to achieve the Gaussian approximation of the local posterior probability density of the state [22, 43]. Then, the parameters of the local posterior approximation of each node are fused in a distributed manner utilizing the average consensus, and finally the global posterior estimation of each node is obtained.

Let each node $m \in \mathcal{V}$ have the Gaussian approximation $\mathcal{N}(\mathbf{x}_{k-1}; \hat{\mathbf{x}}_{k-1}, \mathbf{P}_{k-1})$ of the posterior $p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})$ [43]. The progress of posterior parameters consensus-based DPF is as follows:

(1) Prediction step: Once the estimated global mean and covariance at time $k-1$ are achieved, they can be propagated through the state transition function to predict the probability density function $p(\mathbf{x}_k|\mathbf{z}_{1:k-1})$ as

$$p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})d\mathbf{x}_{k-1} \tag{4.21}$$

$$\approx \int p(\mathbf{x}_k|\mathbf{x}_{k-1})\mathcal{N}\left(\mathbf{x}_k;\hat{\mathbf{x}}_{k-1},\mathbf{P}_{k-1}\right)d\mathbf{x}_{k-1} \tag{4.22}$$

where the $p(\mathbf{x}_k|\mathbf{z}_{1:k-1})$ is still approximated as the Gaussian approximation since the transition function $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ is complied to Gaussian distribution,

$$p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) \approx \mathcal{N}\left(\mathbf{x}_k;\hat{\mathbf{x}}_{k|k-1},\mathbf{P}_{k|k-1}\right) \tag{4.23}$$

where the $\hat{\mathbf{x}}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$ are estimated by scaled unscented transformed (UT) method [44].

In the UT, the n_x dimensional state $\mathbf{x}_k$ is approximated by a set of $2n_x+1$ weighted samples or sigma points firstly as [44]

$$\begin{cases} \mathcal{X}_0 = \hat{\mathbf{x}}_{k-1}, & W_0 = \frac{\lambda}{n_x+\lambda} \\ \mathcal{X}_{i_s} = \hat{\mathbf{x}}_{k-1} + (\sqrt{(n_x+\lambda)\mathbf{P}_{k-1}})_{i_s}, & W_{i_s} = \frac{1}{2(n_x+\lambda)}, \quad i_s = 1,\ldots,n_x \\ \mathcal{X}_{i_s+n_x} = \hat{\mathbf{x}}_{k-1} - (\sqrt{(n_x+\lambda)\mathbf{P}_{k-1}})_{i_s}, & W_{i_s+n_x} = \frac{1}{2(n_x+\lambda)}, \quad i_s = 1,\ldots,n_x \end{cases} \tag{4.24}$$

where the $\mathcal{X}_{i_s}$ is the i_sth sigma point associated with the weight W_{i_s}, $(\sqrt{(n_x+\lambda)\mathbf{P}_{k-1}})_{i_s}$ is the i_s column of the matrix square root (Cholesky decomposition matrix) of $\sqrt{(n_x+\lambda)\mathbf{P}_{k-1}}$, and the λ is the scaling control parameter to adjust the distance between the sigma points, which can be any number (positive or negative) providing that $(n_x+\lambda) \neq 0$.

Then, each sigma point is propagated through the process model to yield a set of transformed samples

$$\mathcal{Y}_{i_{st}} = f_k(\mathcal{X}_{i_{st}}, \mathbf{u}_k), i_{st} = 0, 1, \ldots, 2n_x \tag{4.25}$$

The $\hat{\mathbf{x}}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$ are computed as

$$\hat{\mathbf{x}}_{k|k-1} = \sum_{i_{st}=0}^{2n_x} W_{i_{st}}\mathcal{Y}_{i_{st}} \tag{4.26}$$

$$\mathbf{P}_{k|k-1} = \sum_{i_{st}=0}^{2n_x} W_{i_{st}}\left(\mathcal{Y}_{i_{st}} - \hat{\mathbf{x}}_{k|k-1}\right)\left(\mathcal{Y}_{i_{st}} - \hat{\mathbf{x}}_{k|k-1}\right)^T \tag{4.27}$$

According to (4.23), we can get the Gaussian approximation of $p(\mathbf{x}_k|\mathbf{z}_{1:k-1})$ and utilize it as the local proposal function to sample the particles at each node at time k as [43]

$$\mathbf{X}^i_{m,k} \sim q(\mathbf{X}^i_{m,k}|\mathbf{z}_{1:k-1},\mathbf{z}_{m,k}) \approx \mathcal{N}\left(\mathbf{x}_{m,k};\hat{\mathbf{x}}_{m,k|k-1},\mathbf{P}_{m,k|k-1}\right) \tag{4.28}$$

(2) Update step: Update the local particle weights based on the local observation as

$$w_{m,k}^{i} = \frac{p(\mathbf{X}_{m,k}^{i}|\mathbf{z}_{1:k-1}, \mathbf{z}_{m,k})}{q(\mathbf{X}_{m,k}^{i}|\mathbf{z}_{1:k-1}, \mathbf{z}_{m,k})} \tag{4.29}$$

$$= \frac{p(\mathbf{z}_{m,k}|\mathbf{X}_{m,k}^{i})p(\mathbf{X}_{m,k}^{i}|\mathbf{z}_{1:k-1})}{q(\mathbf{X}_{k}^{i}|\mathbf{z}_{1:k-1}, \mathbf{z}_{s,k})} \tag{4.30}$$

$$= \frac{p(\mathbf{z}_{m,k}|\mathbf{X}_{m,k}^{i})\mathcal{N}\left(\mathbf{x}_{m,k}; \hat{\mathbf{x}}_{m,k|k-1}, \mathbf{P}_{m,k|k-1}\right)}{\mathcal{N}\left(\mathbf{x}_{m,k}; \hat{\mathbf{x}}_{m,k|k-1}, \mathbf{P}_{m,k|k-1}\right)} \tag{4.31}$$

$$= p(\mathbf{z}_{m,k}|\mathbf{X}_{m,k}^{i}) \tag{4.32}$$

Then we can obtain the parameters $\hat{\mathbf{x}}_{m,k}$ and $\mathbf{P}_{m,k}$ of the Gaussian approximation $\mathcal{N}\left(\mathbf{x}_k; \hat{\mathbf{x}}_{m,k}, \mathbf{P}_{m,k}\right)$ of local posterior $p(\mathbf{x}_k|\mathbf{z}_{1:k-1}, \mathbf{z}_{m,k})$ by

$$\hat{\mathbf{x}}_{m,k} = \sum_{i=1}^{N_s} \tilde{w}_{m,k}^{i} \mathbf{X}_{m,k}^{i} \tag{4.33}$$

$$\mathbf{P}_{m,k} = \sum_{i=1}^{N_s} \tilde{w}_{m,k}^{i} \left(\mathbf{X}_{m,k}^{i} - \hat{\mathbf{x}}_{m,k}\right)\left(\mathbf{X}_{m,k}^{i} - \hat{\mathbf{x}}_{m,k}\right)^{\mathrm{T}} \tag{4.34}$$

where the $\tilde{w}_{m,k}^{i}$ is the normalized particle weights of $w_{m,k}^{i}$.

(3) Fusion step: The local parameters $\hat{\mathbf{x}}_{m,k}$ and $\mathbf{P}_{m,k}$ are fused into the parameters of the Gaussian approximation $\mathcal{N}\left(\mathbf{x}_k; \hat{\mathbf{x}}_k, \mathbf{P}_k\right)$ of the global posterior $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ by means of consensus-based fusion rule. The estimation of global state vector $\hat{\mathbf{x}}_k$ and its covariance matrix $\mathbf{P}_k$ of $\mathbf{x}_k$ based on merge algorithm for the linear minimum-variance unbiased estimate in [43] are expressed as

$$\mathbf{P}_{k}^{-1} = \sum_{m=1}^{M} \mathbf{P}_{m,k}^{-1} \tag{4.35}$$

$$\hat{\mathbf{x}}_k = \left(\sum_{m=1}^{M} \mathbf{P}_{m,k}^{-1}\right)^{-1} \sum_{m=1}^{M} \mathbf{P}_{m,k}^{-1} \hat{\mathbf{x}}_{m,k} \tag{4.36}$$

$$= \mathbf{P}_k \sum_{m=1}^{M} \mathbf{P}_{m,k}^{-1} \hat{\mathbf{x}}_{m,k} \tag{4.37}$$

Moreover, let $\mathbf{I}_{m,k}(0) = \mathbf{P}_{m,k}^{-1}$, $\bar{\mathbf{I}}_k = \frac{1}{M}\sum_{m=1}^{M} \mathbf{P}_{m,k}^{-1}$, $\mathbf{J}_{m,k}(0) = \mathbf{P}_{m,k}^{-1}\hat{\mathbf{x}}_{m,k}$, $\bar{\mathbf{J}}_k = \frac{1}{M}\sum_{m=1}^{M} \mathbf{P}_{m,k}^{-1}\hat{\mathbf{x}}_{m,k}$, then the $\bar{\mathbf{I}}_k$ and $\bar{\mathbf{J}}_k$ can be calculated through average consensus algorithm with finite consensus iterations

$$\mathbf{I}_{m,k}(\ell+1) = \zeta_{mm}\mathbf{I}_{m,k}(\ell) + \sum_{m' \in \mathcal{N}_m} \zeta_{mm'}\mathbf{I}_{m',k}(\ell) \tag{4.38}$$

$$\mathbf{J}_{m,k}(\ell+1) = \zeta_{mm}\mathbf{J}_{m,k}(\ell) + \sum_{m' \in \mathcal{N}_m} \zeta_{mm'}\mathbf{J}_{m',k}(\ell) \tag{4.39}$$

Finally, the estimation of global state vector $\hat{\mathbf{x}}_k$ and its covariance matrix $\mathbf{P}_k$ of $\mathbf{x}_k$ is

$$\mathbf{P}_k = \frac{1}{M}\bar{\mathbf{I}}_k^{-1} \tag{4.40}$$

$$\hat{\mathbf{x}}_k = \bar{\mathbf{I}}_k^{-1}\bar{\mathbf{J}}_k^{-1} \tag{4.41}$$

However, computing the covariance matrix $\mathbf{P}_k$ in (4.40) is still needed to be provided the total number of microphone pairs in the distributed network, which reduces the network scalability. Therefore, we can calculate the covariance matrix $\mathbf{P}_{m,k}$ at each node individually when we obtain the global estimate of state $\mathbf{x}_k$ in (4.41) after fusion step as

$$\mathbf{P}_{m,k} = \sum_{i=1}^{N_s} \tilde{w}_{m,k}^i \left(\mathbf{X}_{m,k}^i - \hat{\mathbf{x}}_k\right)\left(\mathbf{X}_{m,k}^i - \hat{\mathbf{x}}_k\right)^{\mathrm{T}} \tag{4.42}$$

The posterior parameters consensus-based DPF algorithm is shown in Algorithm 5. Compared with the common particle filtering algorithm, the posterior parameters consensus-based DPF algorithm does not need the resampling process since it adopts the Gaussian approximation to sample the particles which reduces the computational complexity.

Algorithm 5 Posterior parameters consensus-based DPF algorithm

Input: $\hat{\mathbf{x}}_{k-1}$, $\mathbf{P}_{m,k-1}$, $\mathbf{z}_k$
Output: $\hat{\mathbf{x}}_k$, $\mathbf{P}_{m,k}$
Initial: $\hat{\mathbf{x}}_0$, $\mathbf{P}_{m,0}$, $i = 1, 2, \ldots, N_s$
Iteration process: $k = 1, 2, \ldots$

1: Unscented transform: utilize the $\hat{\mathbf{x}}_{k-1}$ and $\mathbf{P}_{m,k-1}$ to calculate the $\hat{\mathbf{x}}_{k|k-1}$ and $\mathbf{P}_{m,k|k-1}$ via UT method
2: Predict step, importance sampling (draw particles): $\mathbf{X}_{m,k}^i \sim \mathcal{N}\left(\mathbf{x}_{m,k}; \hat{\mathbf{x}}_{m,k|k-1}, \mathbf{P}_{m,k|k-1}\right)$
3: Update step, update particle weights: $w_{m,k}^i = p(\mathbf{z}_{m,k}|\mathbf{X}_{m,k}^i)$
4: Normalize particle weight: $\tilde{w}_k^i = w_k^i / \sum_{j=1}^{N_s} w_k^j$
5: Obtain the local parameters $\hat{\mathbf{x}}_{m,k}$ and $\mathbf{P}_{m,k}$ via (4.33) and (4.34)
6: Calculate the average consensus iterations in (4.38) and (4.39)
7: Estimate the global state $\hat{\mathbf{x}}_k$ by equation (4.41)
8: Compute the covariance matrix $\mathbf{P}_{m,k}$ at each node by equation (4.42)

Table 4.2: Posterior parameters consensus-based DPF algorithm

4.4 Adaptive Distributed Particle Filtering with Variable Sample Set Size through KLD-sampling

As mentioned in the previous chapter, the sample set of the particle filtering can be adaptively adjusted during the sampling progress. Therefore, we can insert the KLD-sampling algorithm in Algorithm 6 into the sampling step in which $\mathbf{X}_{m,k}^{i}$ is generated in Algorithm 4 and Algorithm 5.

Algorithm 6 KLD-sampling algorithm

Input: $\{\mathbf{X}_{k-1}^{j}\}_{j=1}^{N_{s,k-1}}$, $\mathbf{z}_k$, ε, δ, Δ, N_{min}
Output: $\{\mathbf{X}_{k}^{i}\}_{i=1}^{N_{s,k}}$
Initial: $N_{s,k} = 0$, $N_{KLD} = 0$, $N_{BS} = 0$, $i = 0$

1: **while** ($N_{s,k} < N_{KLD}$ or $N_{s,k} < N_{min}$) **do**
2: $N_{s,k} = N_{s,k} + 1$
3: $i = N_{s,k}$
4: Sample a particle with weight with index j from the set $\mathbf{S}_{k-1}$
5: Predict step, importance sampling: $\mathbf{X}_{k}^{i} \sim q(\mathbf{x}_k|\mathbf{X}_{k-1}^{j}, \mathbf{z}_k)$
6: **if** $\mathbf{X}_{k}^{i}$ falls into empty bin b **then**
7: $N_{BS} = N_{BS} + 1$
8: b = non-empty
9: $N_{KLD} = \frac{1}{2\varepsilon}\chi^2_{N_{BS}-1,1-\delta}$
10: **end if**
11: **end while**

Table 4.3: KLD-sampling algorithm

4.5 Simulation Results

In this section, the performance of the speaker tracking algorithm in distributed manner is demonstrated.

4.5.1 Simulation Setup

Same as the previous chapter, the environment conditions and signal characteristics are consistent with the previous chapter. The communication graph of the distributed microphone network is shown in Figure 4.1, each microphone pair (node) can only communicate with its neighboring nodes.

For the consensus-based DPF algorithm, the initial mean and covariance are according to the Gaussian distribution near the truth, that is, the prior distribution of the speaker state satisfies the Gaussian distribution $\mathcal{N}(\mathbf{x}_k; \mathbf{x}_0, \mathbf{P}_0)$, where $\mathbf{x}_0 = [1, 2.5, 0.01, 0.01]^T$ and $\mathbf{P}_0 = diag([0.05, 0.05, 0.0025, 0.0025])$ [25].

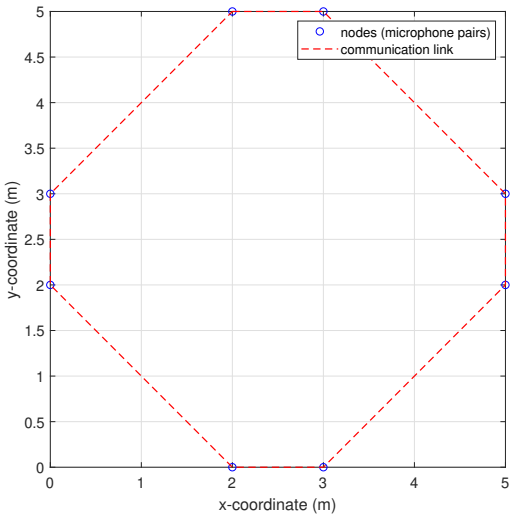
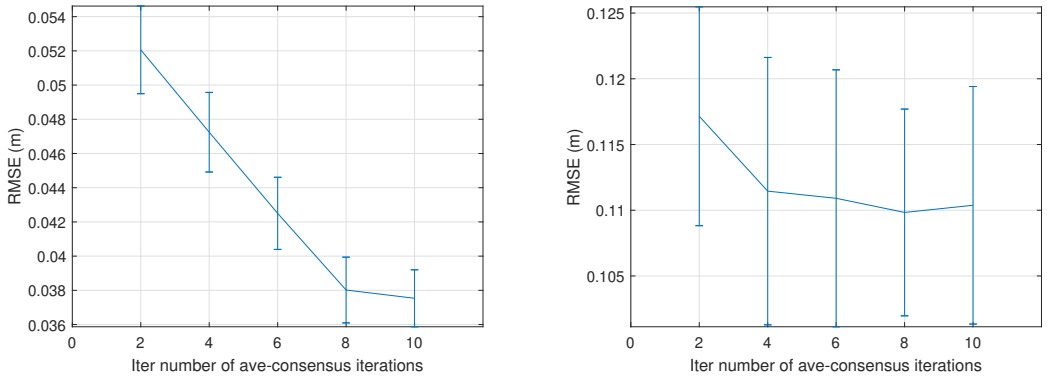


Figure 4.1: Communication graph with microphone pairs (circles) and the communication link (dotted line)



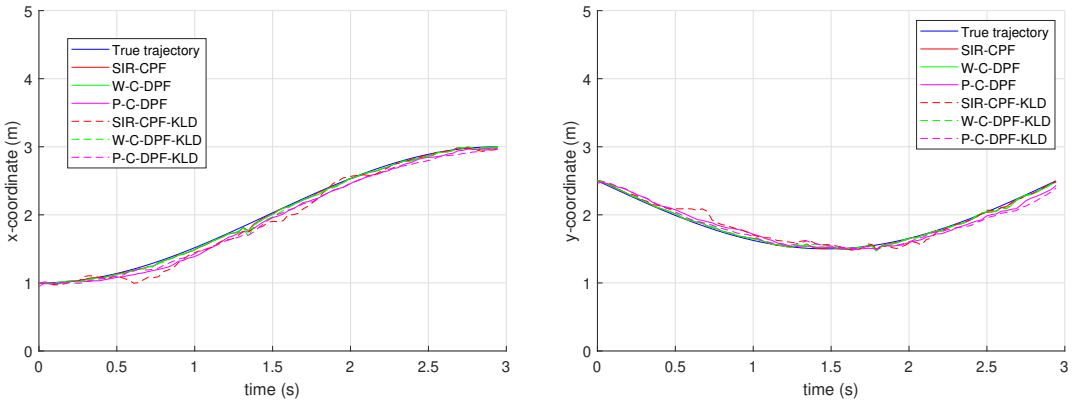
(a) weights consensus-based DPF algorithm

(b) posterior parameters consensus-based DPF algorithm

Figure 4.2: RMSE of different consensus iterations in consensus-based DPF for speaker tracking.

4.5.2 Simulation Results

In this section, we will present some experiments of speaker tracking in distributed methods and compare its performance of tracking speaker with centralized approaches including adjusting the sample size utilizing KLD-sampling algorithm.



(a) x-coordinate (b) y-coordinate

Figure 4.3: Speaker tracking results with SNR=20dB.

4.5.2.1 Convergency of DPF based on consensus

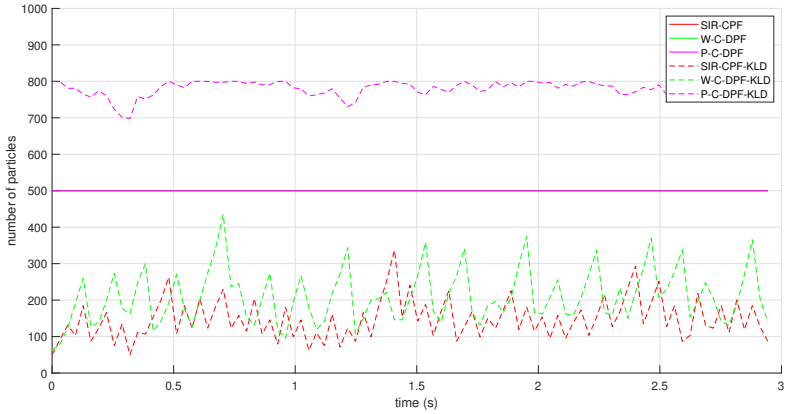
Figures 4.2(a) and 4.2(b) indicate both the two consensus-based DPFs converge with the increase number of iterations. We can see that, as expected, better performance is achieved for a larger number of consensus iterations. That is, we can adopt the average consensus algorithm to reach an agreement on some certain parameters for tracking sound source in the distributed networks. In addition, the tracking accuracy of the particle weights consensus-based DPF (W-C-DPF) outperforms the posterior parameters consensus-based DPF (P-C-DPF) since the W-C-DPF shares the identical particles at each node. The improvement is not significant for more than eight iterations for the W-C-DPF while the number of iterations is about four for the P-C-DPF.

4.5.2.2 Comparisons of different speaker tracking algorithms

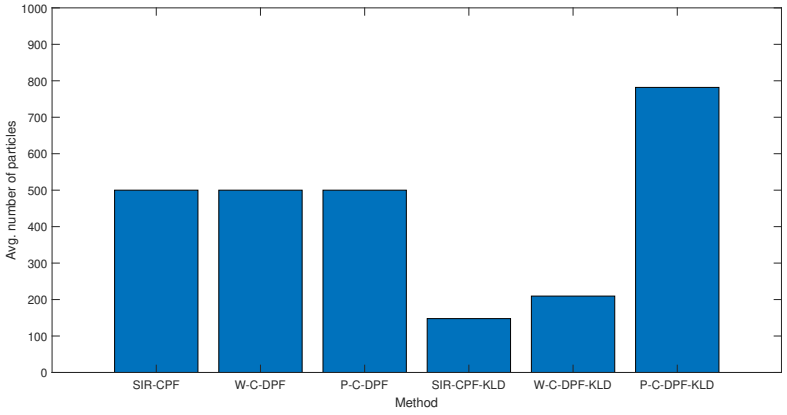
Figure 4.3 and Figure 4.4 present the speaker tracking results based on different methods when SNR is 20dB. The six approaches composed of three methods with 500 fixed particles: SIR-CPF in a centralized manner, the W-C-DPF, the P-C-DPF, and their adaptive methods which adjust the sample set size utilizing KLD-sampling algorithm named SIR-CPF-KLD, W-C-DPF-KLD, and P-C-DPF-KLD, respectively.

From Figure 4.3(a) and Figure 4.3(b) we can see that all the algorithms can successfully track the speaker trajectory in two dimensions which means all the algorithms proposed in this chapter can be worked for tracking the speaker in distributed microphone networks.

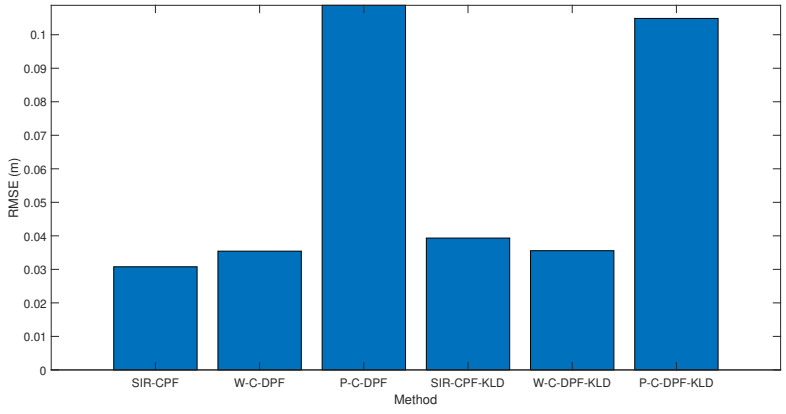
Moreover, Figure 4.4 represents the adaptation of sample size during the tracking progress based on particle filtering algorithm. The results in Figure 4.4(b) and Figure 4.4(c) demonstrate that the particle weights consensus-based DPF algorithm can exploit the KLD-sampling method to adaptively adjust the sample size. The average number of particles is significantly reduced while there is not an obvious rise in the RMSE value through the P-C-DPF-KLD algorithm. It can effectively decrease the communication burden since at each consensus step all the weights need to exchange with neighboring



(a) number of particles

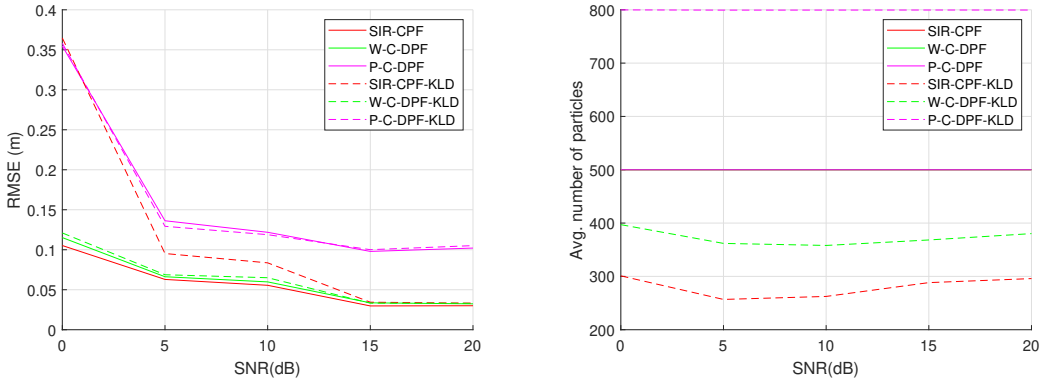


(b) average number of particles



(c) RMSE results

Figure 4.4: Speaker tracking results with SNR=20dB in terms of particle number and RMSE.



(a) RMSE results versus SNR

(b) average number of particles versus SNR

Figure 4.5: Speaker tracking results versus SNR

nodes. The results of the RMSE of W-C-DPF algorithm and SIR-CPF are quite similar, the slight difference caused by insufficiently converged consensus algorithm.

In Figure 4.4(a), we can see that the number of particles with the P-C-DPF-KLD method quickly reached the set limit and remained constant. Although the performance of adaption in the P-C-DPF-KLD is not as well as the W-C-DPF-KLD method, the KLD-sampling can also be employed as a method to adapt the sample size combined with the posterior parameters consensus-based DPF. Because the posterior parameters consensus-based DPF dose not require all the microphone pairs to be synchronized, this method is much suitable to implement in reality.

4.5.2.3 Comparisons of different SNR conditions

In this simulation, we change the parameter error bound from $\varepsilon = 0.15$ to $\varepsilon = 0.1$ since the previous value was obtained in a low noise environment. In order to satisfy the severe noise environment, we could set a tighter error bound of the KLD-sampling algorithm.

The effect of noise on the tracking performance is also evaluated for the six different tracking methods. Results from Figure 4.5 illustrate that all the approaches achieve better tracking accuracy as the SNR level increases since the observations extracted from the received signal at each microphone are more reliable with the larger the SNR (less noise interference).

From Figure 4.5(a) we can see that the RMSE results of the all algorithms are large when the noise effect is severe, and it decreases as the SNR value increases. The performance of the SIR-CPF and W-C-DPF are similar and better than the P-C-DPF. However, the SIR-CPF-KLD algorithm shows the largest tracking error when the SNR is very low whereas the W-C-DPF-KLD performs best which infers it is more robust against noise.

Figure 4.5(b) indicates the number of particles in different SNR conditions. Although the number of particles does not change significantly with different noise environments, it can still be seen that W-C-DPF-KLD is an effective tracking algorithm.

Specifically, in a high SNR environment, the SIR-CPF-KLD method could obtain a good tracking result with fewer particle numbers; however, in a low SNR environment, it cannot significantly boost the number of particles to effectively track the speaker. On the other hand, although the W-C-DPF-KLD utilizes more particles than the SIR-CPF-KLD, the W-C-DPF-KLD performs better and is more robust in severe noise environment.

5 Conclusions and Future Work

5.1 Conclusions

In this thesis, two methods based on particle filter algorithm for tracking speaker in distributed pairwise microphone networks were proposed and adjusted by KLD-sampling approach in terms of adapting the number of particles.

In Chapter 2, the general frameworks including the acoustic source motion model, signal model and observation model in speaker tracking problem were introduced.

In Chapter 3, the particle filter algorithm has been studied in a centralized manner, which is suitable to track the sound source in a reverberant and noisy environment. The drawback of the required amount of particles should be set in advance according to prior experience is not well suitable for practical application of speaker tracking problem. To handle this problem, the KLD-sampling method was presented to adjust the number of particles during the particle filtering process instead of setting fixed number. The experimental results showed that the comparisons of different settings of KLD-sampling and the validity of acoustic source tracking based on adaptive particle filtering algorithm with variable sample set size within an acceptable error range.

In Chapter 4, two distributed particle filter (DPF) algorithms were proposed and modified, the particle weights consensus-based DPF and the posterior parameters consensus-based DPF approaches. The particle weights consensus-based DPF method shares the identical set of particles to execute the local PF, which requires that the random number generators are synchronized at each microphone pair, while the posterior parameters consensus-based DPF approach exploits different sets of particles and the synchronization of random number generators is not required. We developed the posterior parameters consensus-based DPF method such that we did not need to provide the total number of microphone pairs in the distributed networks. The KLD-sampling was also applied to these two DPF algorithms to dynamically adjust the number of particles. Especially for the particle weights consensus-based DPF method, the number of particles is a critical factor which affects the complexity of computation in the data fusion process since the microphone pairs exchange all the local particle weights with their neighboring nodes. The simulations proved that these two algorithms are feasible to track the acoustic source in distributed microphone networks. The tracking accuracy of the particle weights consensus-based DPF method outperformed the posterior parameters consensus-based DPF approach. On the other hand, the posterior parameters consensus-based DPF approach held less communication burden and no synchronized generators required, which is more practicable in practice.

5.2 Future Work

Future research could be aimed in the following directions.

In this thesis, a framework for the speaker tracking using distributed pairwise microphone networks in a noisy and reverberant environment was tested. The work has been verified on some simulations, but further testing is needed to test the effectiveness and robustness of the proposed algorithms in real-life situations.

In addition, the parameters of KLD-sampling in this thesis are selected according to the fixed situation. Further study on how the parameters adaptively change with the environment is needed.

Furthermore, the speaker state utilized in this thesis only considers the spatial position and velocity information, which has a low dimension. In practical applications, higher-dimensional speaker states can be used and appropriate motion models can be selected as well, and the effect of increased state dimensions on the performance of the tracking algorithm can be researched.

Finally, this thesis focuses on the tracking problem of a single speaker. In practice, when multiple speakers appear in the room at the same time, the motion trajectory of each speaker needs to be estimated separately. The multi-speaker tracking problem could be a further research direction.

Bibliography

[1] T. Nishiura, R. Gruhn, and S. Nakamura. Automatic steering of microphone array and video camera toward multi-lingual tele-conference through speech-to-speech translation. In *IEEE International Conference on Multimedia and Expo, 2001. ICME 2001.*, pages 447–450, Aug 2001.

[2] N. Strobel, S. Spors, and R. Rabenstein. Joint audio-video object localization and tracking. *IEEE Signal Processing Magazine*, 18(1):22–31, Jan 2001.

[3] J. Vermaak, M. Gangnet, A. Blake, and P. Perez. Sequential monte carlo fusion of sound and vision for speaker tracking. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 1, pages 741–746 vol.1, July 2001.

[4] B. Chen, C. Chen, and J. Wang. Smart homecare surveillance system: Behavior identification based on state-transition support vector machines and sound directivity pattern analysis. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 43(6):1279–1289, Nov 2013.

[5] I. D. Gebru, C. Evers, P. A. Naylor, and R. Horaud. Audio-visual tracking by density approximation in a sequential bayesian filtering framework. In *2017 Hands-free Speech Communications and Microphone Arrays (HSCMA)*, pages 71–75, March 2017.

[6] K. Nakadai, H. Nakajima, M. Murase, S. Kaijiri, K. Yamada, T. Nakamura, Y. Hasegawa, H. G. Okuno, and H. Tsujino. Robust tracking of multiple sound sources by spatial integration of room and robot microphone arrays. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 4, pages IV–IV, May 2006.

[7] C. Evers, Y. Dorfan, S. Gannot, and P. A. Naylor. Source tracking using moving microphone arrays for robot audition. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6145–6149, March 2017.

[8] J. Thiemann, J. Lücke, and S. van de Par. Speaker tracking for hearing aids. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, Sep. 2016.

[9] J. C. Chen and R. E. Hudson and. Maximum-likelihood source localization and unknown sensor location estimation for wideband signals in the near-field. *IEEE Transactions on Signal Processing*, 50(8):1843–1854, Aug 2002.

[10] J. DiBiase, F. Silverman, and S. Brandstein. Robust localization in reverberant rooms. In *Microphone Arrays*, pages 157–180. Springer, Berlin, Heidelberg, 2001.

[11] H. Do, H. F. Silverman, and Y. Yu. A real-time srp-phat source location implementation using stochastic region contraction(src) on a large-aperture microphone

array. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, volume 1, pages I–121–I–124, April 2007.

[12] A. Dehghan Firoozabadi and H. R. Abutalebi. A new region search method based on doa estimation for speech source localization by srp-phat method. In *2010 18th European Signal Processing Conference*, pages 656–660, Aug 2010.

[13] N. Strobel, T. Meier, and R. Rabenstein. Speaker localization using a steered filter-and-sum beamformer. In *In Proc. Erlangen Workshop on Vision, Modeling, and Visualization*, pages 195–202, 1999.

[14] C. Knapp and G. Carter. The generalized correlation method for estimation of time delay. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 24(4):320–327, August 1976.

[15] Jacob Benesty. Adaptive eigenvalue decomposition algorithm for passive acoustic source localization. *The Journal of the Acoustical Society of America*, 107:384–391, 2000.

[16] D. Bechler, M. Grimm, and K. Kroschel. Speaker tracking with a microphone array using kalman filtering. *Advances in Radio Science - Kleinheubacher Berichte*, 1, 05 2003.

[17] T. G. Dvorkind and S. Gannot. Speaker localization exploiting spatial-temporal information. In *in Proceedings of the International Workshop on Acoustic Echo and Noise Control (IWAENC '03*, pages 295–298, 2003.

[18] J. J. LaViola. A comparison of unscented and extended kalman filtering for estimating quaternion motion. In *Proceedings of the 2003 American Control Conference, 2003.*, volume 3, pages 2435–2440 vol.3, June 2003.

[19] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings F - Radar and Signal Processing*, 140(2):107–113, April 1993.

[20] D. B. Ward, E. A. Lehmann, and R. C. Williamson. Particle filtering algorithms for tracking an acoustic source in a reverberant environment. *IEEE Transactions on Speech and Audio Processing*, 11(6):826–836, Nov 2003.

[21] M. Crocco, S. Martelli, A. Trucco, A. Zunino, and V. Murino. Audio tracking in noisy environments by acoustic map and spectral signature. *IEEE Transactions on Cybernetics*, 48(5):1619–1632, May 2018.

[22] O. Hlinka, F. Hlawatsch, and P. M. Djuric. Distributed particle filtering in agent networks: A survey, classification, and comparison. *IEEE Signal Processing Magazine*, 30(1):61–81, Jan 2013.

[23] Q. Zhang, Z. Chen, and F. Yin. Global coherence field and distributed particle filter-based speaker tracking in distributed microphone networks. *The Journal of Computational Acoustics*, 23(3), 2015.

[24] X. Zhong, A. Mohammadi, W. Wang, A. B. Premkumar, and A. Asif. Acoustic source tracking in a reverberant environment using a pairwise synchronous microphone network. In *Proceedings of the 16th International Conference on Information Fusion*, pages 953–960, July 2013.

[25] Q. Zhang, Z. Chen, and F. Yin. Speaker tracking based on distributed particle filter in distributed microphone networks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(9):2433–2443, Sep. 2017.

[26] R. Karlsson and F. Gustafsson. Monte carlo data association for multiple target tracking. In *IEE Target Tracking: Algorithms and Applications (Ref. No. 2001/174)*, volume Seminar, pages 13/1–13/5 vol.1, Oct 2001.

[27] D. Fox. Adapting the sample size in particle filters through kld-sampling. *The International Journal of Robotics Research*, 22(12):985–1003, 2003.

[28] J. Vermaak and A. Blake. Nonlinear filtering for speaker tracking in noisy and reverberant environments. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, volume 5, pages 3021–3024 vol.5, May 2001.

[29] Heinrich Kuttruff. *Room Acoustics*. CRC Press, 6th edition edition, 2016.

[30] J. B. Allen and D. A. Berkley. Image method for efficiently simulating small-room acoustics. *Journal of the Acoustical Society of America*, 65(4):943–950, 1979.

[31] E. A. Lehmann and A. M. Johansson. Prediction of energy decay in room impulse responses simulated with an image-source model. *The Journal of the Acoustical Society of America*, 124(1):269–277, 2008.

[32] D. Youn, N. Ahmed, and G. Carter. On using the lms algorithm for time delay estimation. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 30(5):798–801, October 1982.

[33] B.D.O. Anderson and J.B. Moore. *Optimal Filtering*. Prentice-Hall, Englewood Cliffs, NJ, 1979.

[34] A. Jazwinski. *Stochastic processes and filtering theory*. Mathematics in science and engineering. Acad. Press, New York, NY [u.a.], 1970.

[35] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, Feb 2002.

[36] A. Doucet, S. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10(3):197–208, July 2000.

[37] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, Inc., New Jersey, USA, second edition edition, 2006.

[38] J.A. Rice. *Mathematical Statistics and Data Analysis*. Duxbury Press, second edition edition, 1995.

[39] Hugh Durrant-Whyte and Thomas C. Henderson. *Multisensor Data Fusion*, pages 585–610. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[40] J. L. Williams, J. W. Fisher, and A. S. Willsky. Approximate dynamic programming for communication-constrained sensor network management. *IEEE Transactions on Signal Processing*, 55(8):4300–4311, Aug 2007.

[41] Lin Xiao and S. Boyd. Fast linear iterations for distributed averaging. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475)*, volume 5, pages 4997–5002 Vol.5, Dec 2003.

[42] L. Xiao, S. Boyd, and S. Lall. A scheme for robust distributed sensor fusion based on average consensus. In *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.*, pages 63–70, April 2005.

[43] A. Simonetto and T. Keviczky. *Distributed Nonlinear Estimation for Diverse Sensor Devices*, pages 147–169. Springer London, London, 2012.

[44] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte. A new method for the nonlinear transformation of means and covariances in filters and estimators. *IEEE Transactions on Automatic Control*, 45(3):477–482, March 2000.