Attention-Based End-to-End Differentiable Particle Filter for Audio Speaker Tracking

Jinzheng Zhao 1, yong xu 1, xinyuan qian 1, haohe liu 1, Mark Plumbley 1, and Wenwu Wang 1

1Affiliation not available

October 31, 2023
Attention-Based End-to-End Differentiable Particle Filter for Audio Speaker Tracking

JINZHENG ZHAO1, YONG XU2,(Senior Member, IEEE), XINYUAN QIAN3, (Senior Member, IEEE), HAOHE LIU1, (Student Member, IEEE), MARK D. PLUMBLEY1, (Fellow, IEEE), AND WENWU WANG1, (Senior Member, IEEE)

1Centre for Vision, Speech and Signal Processing, University of Surrey, Guildford, GU2 7XH, UK
2Tencent AI Lab, Bellevue, WA 98004, USA
3Department of Computer Science and Technology, University of Science and Technology Beijing, Beijing 100083, China

Corresponding author: Jinzheng Zhao (email: j.zhao@surrey.ac.uk).

This research is supported by Tencent AI Lab Rhino-Bird Gift Fund and University of Surrey.

ABSTRACT Particle filter (PF) has been widely used in speaker tracking due to its capability of modeling the non-linear process or the non-Gaussian environment. However, particle filter are penalized by many issues. For example, pre-defined handcrafted measurements are often used which can limit the model performance. In addition, the transition and update models are often preset which make PF less flexible to be adapted to different scenarios. To address these issues, we propose an end-to-end differentiable particle filter framework by employing the multi-head attention to model the long-range dependencies. The proposed model employs the self-attention as the learned transition model and the cross-attention as the learned update model. To our knowledge, this is the first proposal of combining particle filter and transformer for speaker tracking, which integrates the measurement extraction, transition and update steps into an end-to-end architecture. Experimental results show that the proposed model achieves superior performance over the recurrent baseline models.

INDEX TERMS Particle Filters, Speaker Tracking.

I. INTRODUCTION

SPEAKER tracking plays an important role in speech separation [1], speech enhancement [2] and speaker diarization [3]. The task of speaker tracking is to estimate the 2D position, 3D position or Direction of Arrival (DOA) of speakers at each time step. Generally, speaker tracking consists of two steps, measurement extraction and Bayesian filtering. Speaker localization can provide measurements for speaker tracking. For speaker localization, there are two types of methods: parametric-based methods [4] and learning-based methods [5]. One of the important parametric-based methods is global coherent field (GCF), which is widely used for obtaining measurements in speaker tracking [6]. GCF map accumulates the generalized cross-correlation phase transform [4] (GCC-PHAT) generated by signals from each microphone pair. Then a grid search method is employed to find the maximum over the acoustic map. The position producing the maximum is regarded as the position of the sound source. Compared to parametric-based methods, learning-based methods are more robust against room reverberation and background noise [7] when trained on audio data recorded from different acoustic scenarios. It finds the relationships between the audio features such as GCC-PHAT [8] and the speakers’ positions through neural networks.

Tracking considers the temporal variations of the speaker trajectory. The tracking algorithms often focus on temporal variations, smoothed trajectories, removing estimation outliers and compensating for missing observations. The family of Bayesian filtering algorithms is often used to mitigate these problems, which aims to estimate target states recurrently given the previous states and the current measurements. There are two recursive steps in the Bayesian filtering, prediction, and updating. In the prediction step, the target
states are transferred from the last time step to the current
time step through a transition model. In the update step, the
states are updated from prior to posterior by the measurement
model. Several methods have been developed in this family,
including Kalman Filter (KF) and PF. KF assumes the transit-
ion process and the update process to be linear and assumes
the noise follows Gaussian distribution. It uses Gaussian
distribution to represent the target states and updates the
mean and covariance at each time step. The performance
is satisfactory in this linear Gaussian environment but the
strict requirement limits the generalization to complicated
scenarios. Particle filter is a Sequential Monte Carlo (SMC)
method and uses a group of particles instead of Gaussian
distributions to represent the target states, which can handle
the non-linear and non-Gaussian scenarios. PF contains four
steps. At first the weights of particles are initialized as the
same. The particle states are transited in the prediction step
and the particle weights are updated by the measurement
likelihood in the update step. The target states are calculated
as the weighted sum of the particle states. At last the particles
are resampled to avoid the weight degeneracy problem. The
particles with high weights are maintained and duplicated,
while the particles with low weights are discarded.

The Bayesian filter is a one-order Markov process. The
estimation of current states depends on the state in the past
time and the measurements at the current time. Simi-
larly, transformer is an auto-regressive model when used in
temporal prediction tasks such as machine translation, text
summarization and tracking, whose output at current time
step also depends on the input and output in the previous
time steps. Based on this similarity we try to combine the
two model. One of the reason for the success of transformer
is the effectiveness of multi-head attention. It calculates
the weighted sum of values based on the similarity of
queries and keys. We use the transformer encoder as the
feature extractor to provide measurements. The self-attention
module in the decoder is used for the transition model
to predict particle states and the cross-attention module
is used for the update model to calculate measurement
likelihood, which will combines the advantages of particle
filter and transformer. 1) Particle filter is not an end-to-
end architecture. The measurement needs to be obtained
before the update step. The combination is an end-to-end
architecture and the filter and the measurement model can
be trained jointly. 2) Additionally, the transition model and
update model are often preset, which is hard to generalize
to complex scenarios. The self-attention and cross attention
modules are employed as learnable transition and update
model. 3) It is proved that the algorithm priors introduced
by particle filter will improve the model performance [9].
In addition, the prediction step and update step introduced in
transformer bring explainability to the model compared to
training in a neural network black box.

The remaining part of this paper is presented as follows:
In Section II, we summarize some related topics. In Section
IV, we present our differentiable particle filter transformer
for single-speaker tracking and discuss the extension for
multiple-speaker scenarios. In section V, we show the ex-
perimental results of the baseline methods and the proposed
methods. We also discuss the model robustness against noise
and sequence length. In Section VI, we concludes the paper
and point out the limitation and future works.

II. Related Work

A. Speaker Tracking

In the past few years, Bayesian filter based methods appear
for speaker tracking. Most methods adopt the paradigm
of filter with measurement model. In [10], an adaptive
particle filter is proposed for single speaker tracking. It uses
GCF as audio measurement and uses face detection and
color histogram as video measurement. An adaptive weight
mechanism is designed to determine the importance of audio
and visual modality dynamically. In [11], a similar algorithm
with [10] is explored under a more reverberant and noisy
environment with occlusion, speaker out of field of view
and speaker not facing the cameras. In [6], a new dataset
named CAV3D is proposed for audio-visual speaker tracking.
Compared to the mostly used AV16.3 dataset [12], CAV3D
has stronger reverberation and more complicated scenarios.

In [13], particle filter is used for multiple speaker tracking
with discriminative and generative measurement likelihood.
In [14], a two-layer particle filter is proposed. Two groups
of particles pass through the audio and visual layers separately.
The particle weights are determined by the likelihood of two
modalities.

The RFS-based method is another branch of Bayesian
filter, which can handle the varying number of speakers.
RFS stands for random finite set and contains a varying
number of elements. Both target set and measurement set can
be represented by the RFS. At each time step, the speaker
RFS is the combination of surviving speakers, spawned
speakers from last time step, and new speakers. To lower the
computational complexity, PHD filter propagates the first-
order moment of the multiple target state distribution. It has
the linear Gaussian form, representing the target in Gaussian
distribution and SMC form, representing the target with
particles. In [15], PHD filter is used for tracking unknown
and varying number of moving audio sources. In [16], the
SMC PHD filter is employed with the help of mean-shift
to move particles to the local maximum. There are several
works [17], [18] that combines particle flow with PHD filter
while particle flow can help to transfer the particles from
prior distribution to posterior distribution. Unlike PHD fil-
ter, multi-target multi-Bernoulli (MeMBer) filter propagates
posterior density function rather than the first-order moment.
The target state is represented as Bernoulli RFS, which is
empty or has a single element. In [1], generalized labeled
Bernoulli filter (GLMB) is employed to solve the problem
of multi-modal space-time permutation and deal with the
problem of varying number of speakers.
B. Differentiable Bayesian Filter

There have been some works that combine Bayesian filter and deep learning models for temporal prediction tasks. In [19], a backprop Kalman filter is proposed, which takes the raw image as the input and output tracking position. In [20], a dynamic weight mechanism is jointly trained with backprop Kalman filter so that the importance of different modality can be determined by the quality of the measurements. There are also some works that combine neural networks with particle filter. In [21], a differentiable particle filter is designed with a semi-supervised learning strategy to reduce the requirement of labeled data. In [22], particle filter is combined with SLAM for visual navigation. In [23], particle filter network is proposed for visual localization, which encodes the measurement model and particle filter in a single neural network. In [9], a similar architecture has been implemented. The training strategy of [9] is three steps. The transition model and measurement model are trained first. At last an end-to-end learning objective is adopted. In addition to the conventional neural network, the recurrent neural network can also be combined with particle filter. In [24], PF-LSTM and PF-GRU are proposed, which replace the deterministic update with stochastic Bayesian update. In [25], particle transformer is proposed, which leverages weighted multi-head attention for differentiable resampling. Compared to [9], [23], [25], our proposed model extends the differentiable particle filter into multiple object tracking.

III. Problem Formulation

The whole algorithm design is based on particle filter framework. Particle filter uses some particles \( s_t = \{ w_{t,i}, s_{t,i} \}_{i=1}^N \) to represent target states, which has four steps: initialization, prediction, update and resampling. In the first step, all particle weights are initialized to be the same:

\[
\{ w_{1,i} \}_{i=1}^N = 1/N
\]  

(1)

In the prediction step, the particle states \( s_t = \{ w_{t,i}, s_{t,i} \}_{i=1}^N \) are transmited from the last time step to the current time step \( s_{t+1} = \{ w_{t,i}, s_{t+1,i} \}_{i=1}^N \) using the transition model:

\[
s_{t+1} = T \cdot s_t + q
\]  

(2)

where \( T \) is the transition model and is assumed to be a velocity-constant model. \( q \) is a zero-mean Gaussian noise.

In the update step, the measurement likelihood \( I \) is first calculated:

\[
I = Z(o_{t+1} | s_{t+1})
\]  

(3)

where \( o_{t+1} \) is the observation set. \( Z \) is the measurement model. The particle weights are updated by the measurement likelihood:

\[
w_{t+1,i} = 1 \cdot w_{t,i}
\]  

(4)

The target state \( x \) is derived by the weighted summation of particle states:

\[
x_{t+1} = \sum_{i=1}^k w_{t,i} \cdot s_{t,i}
\]  

(5)

After some iterations particle filter may suffer from weight degeneracy problem, where the target state is determined by a few high-weight particles. So the last step is resampling, which will duplicate the high-weight particles and discard low-weight particles. After resampling the weights of the resampled particles are set to the same.

In this paper, we explore to use audio signal for speaker tracking. Given the binaural audio \( a = \{a_1, a_2\} \), the task of speaker localization aims to predict the DOA at each time step.

GCC-PHAT is employed as the audio features, which is commonly used in speaker tracking and calculated in the microphone pair:

\[
g_{i,j}(t, \tau) = \int_{-\infty}^{\infty} STFT_{a_i}(t, f) \cdot STFT_{a_j}^{*}(t, f) \cdot e^{j2\pi \tau df}
\]  

(6)

\( g \in R^{T \times C} \), where \( T \) is the number of windows in one time span and \( C \) is the number of coefficients of delay lags. \( \tau \) is the time delay lag. \((i, j)\) denotes one microphone pair. \( STFT \) is Short Term Fourier Transform and * denotes complex conjugate.

IV. Proposed Methods

In this section we show an end-to-end differentiable architecture which combines particle filter and transformer for single speaker tracking. And we discuss the extension of the proposed model to multiple speaker tracking.

A. Attention-Based Differentiable Particle Filter

The overview of the model can be seen in Fig. 1. The self-attention provides an implicit learnable transition model to transfer the particle states without motion commands to the particles. The cross-attention module helps to calculate the measurement likelihood.

The overall architecture of our model is shown in Fig. 2, which follows the paradigm of vanilla transformer. The GCC-PHAT \( g \) is first added with the position embeddings \( g_{POS} \) along the time dimension and input to the transformer encoder.

In the transformer encoder, the input goes through the self-attention layers and the fully connected layers with the residual connection. The advantage of the proposed model
The architecture of the proposed model architecture.
GCC-PHAT of the two-channel audio is calculated and split as the input to the encoder. The decoder takes the particle embedding as input and performs self-attention and cross attention with the encoder output as the transition model and update model, respectively. Initially the colors of particles are the same, indicating the same weights. After the update, the transition model and update model, respectively (Initially the colors of particles are represented as the embedding vectors $s_i \in \mathbb{R}^{N \times D}$ where $N$ is the number of particles and $D$ is the hidden dimension. Each particle embedding implies the particle position. $L$ is the number of repeated modules in the encoder.

The particle embedding $s_t$ is first added with the position embedding $s^{POS}$ and then goes through the self-attention layer. The multi-head self-attention layer (MSA) is applied on the dimension of $N$, which is regarded as the transition model in particle filter. The transition of one particle state depends on the self-attention with other particle embedding.

$$s_t = s_t + s^{POS}$$  \hspace{1cm} (9)

$$s_{t+1} = LN (MSA (s_t)) + s_t,$$  \hspace{1cm} (10)

After the self-attention transition, particle states are transferred from $s_t$ to $s_{t+1}$. Then cross attention is used between the predicted particle states with the output of the encoder. The disturbed information for DOA estimation is also hidden in the encoder output, such as clutter, outliers and noise. The multi-head cross attention layer (MCA) is regarded as the measurement model and the output of the encoder is the measurement.

$$\hat{s}_{t+1} = LN (MCA (s_{t+1}, z^d)) + s_{t+1},$$  \hspace{1cm} (11)

A fully connected layer is used to calculate the measurement likelihood of particle embedding.

$$Z(o_{t+1} | s_{t+1}) = MLP (\hat{s}_{t+1})$$  \hspace{1cm} (12)

The particle weights are updated according to the likelihood:

$$w_{t+1,i} = Z(o_{t+1} | s_{t+1}) \cdot w_{t,i}.$$  \hspace{1cm} (13)

The corresponding DOA posterior to the updated particle states $d_{t+1,i} \in \mathbb{R}^{360}$ are derived by an MLP layer:

$$d_{t+1,i} = MLP (\hat{s}_{t+1,i})$$  \hspace{1cm} (14)

where $LN$ is the layer normalization and $MSA$ is the multi-head self attention. The final DOA posterior $d_{t+1} \in \mathbb{R}^{360}$ is the weighted summation of the DOA posterior corresponding to each particle:

$$d_{t+1} = \sum_{i=1}^{N} w_{t+1,i} d_{t+1,i}$$  \hspace{1cm} (15)

Finally, the DOA is obtained as the peak index of the posterior:

$$\hat{d}_{t+1} = \text{arg max} (d_{t+1})$$  \hspace{1cm} (16)

B. Resampling

The resampling step selects and duplicates the important particles and discards the unimportant ones. However, the resampling step is not differentiable. To integrate the resampling step into the transformer, similar to [24] [23], we employ the soft-resampling. In soft-resampling, we resample particles from a new distribution $q$ rather than resample from the original distribution $p$, where $q$ is the combination of $q$ and uniform distribution $u = 1/K$ with $0 < \alpha < 1$ being the hyperparameter to balance the two distributions.

$$q(i) = \alpha p(i) + (1 - \alpha) u(i)$$  \hspace{1cm} (17)

Instead of being equal weights of all particles, the new particle weights are calculated as follows:

$$w^k_t = \frac{p(k)}{q(k)} = \frac{w^k_t}{\alpha w^k_t + (1 - \alpha)1/K}$$  \hspace{1cm} (18)

C. Extension to Multiple Speakers

The extension of the current model to the multiple speakers follows the conventional setting of the particle filter. In the conventional particle filter, several groups of independent particles are used for different objects. In this paper, we consider the scenario of two speakers. Two independent groups of particles are employed as input to the decoder. For the self-attention transition step, particles from two groups are processed independently with an attention mask $m = \begin{pmatrix} I & 0 \\ 0 & I \end{pmatrix}$ to make sure the self-attention transition of different groups does not interfere with each other, where the dimension of $I$ is the number of particles for one speaker.
The cross attention update is the same as that of Eq (11) \sim (14). Compared to the single speaker tracking, multiple speaker tracking has a data association step, which matches the measurements with the speaker states. The data association is hidden in the cross attention update and each particle embedding automatically finds the related measurements. When obtaining DOA, Eq (15) \sim (16) is performed for each group of particles. Since in the multiple speakers’ scenario, the number of the speakers varies. Thus we add another classifier to estimate whether the particles corresponds to an existing target.

\[
\hat{e}_{t+1} = \text{MLP}(\hat{s}_{t+1}) \tag{19}
\]

After obtaining multiple speakers’ DOA and existence probability by averaging particle states according to their weights, Hungarian algorithm is used to match the estimated DOA with the ground truth. Similar to [26], the matching strategy depends on the speaker existence probability and the DOA estimation.

\[
\sigma(x_{\sigma(i)} (y_i)) = \\
-1_{\{c_i \neq \emptyset\}} \hat{e}_{\sigma(i)} (c_i) \cdot \lambda_{cls} + 1_{\{c_i \neq \emptyset\}} \mathcal{L}_{DOA} (r_i, \hat{d}_{\sigma(i)}) \cdot \lambda_{DOA} \tag{20}
\]

where \(i\) is the index of the ground truth and \(\sigma(i)\) is the matching index. \(x_{\sigma(i)} (y_i) = (\hat{e}_{\sigma(i)}, \hat{d}_{\sigma(i)})\) and \(y_i = (c_i, r_i)\) where \(c_i\) indicates the speaker existence and \(r_i\) is the ground truth DOA. \(\mathcal{L}_{DOA}\) is the mean average error (MAE) between \(\hat{d}_{\sigma(i)}\) and \(r_i\). \(\lambda_{cls}\) and \(\lambda_{DOA}\) are the hyperparameters to balance the classification error and the DOA error.

At last, soft-resampling is performed for each group of particles, independently.

D. Learning Objective

For the DOA distance loss, we cannot use the distance between the predicted DOA and the ground truth DOA. As the argmax operation to get the DOA will block the gradients and it is not differentiable. Some works [27], [28] treat the DOA estimation task as the classification task and use the cross entropy loss. With a resolution \(r\), the DOA space are split into \(360/r\) classes. However, the cross entropy loss cannot describe the inter relationship among different classes. For instance, the error between 0° and 180° should be larger than that between 0° and 90°. But cross entropy loss will treat the same. To model the relationship between different classes, inspired by [5], first we encode the ground truth as the Gaussian distribution \(y^G\) centered on DOA.

\[
y^G \sim N (r, \sigma^2) \tag{21}
\]

We generate the Gaussian distribution centered on the ground truth DOA with resolution \(r = 1°\) and covariance \(\sigma^2 = 1°.\) The earth mover’s distance (EMD) loss [29] is employed, which was used in assessment of speech quality [30]. The EMD loss accumulating the errors in each DOA band:

\[
\mathcal{L}_{EMD} = \sum_{i=1}^{360} (q_i - y_i^G) \tag{22}
\]
corpus to 10 seconds and input to the Two Ears auditory system with a trajectory to generate binaural audio with a sampling interval of 368. We collect around 33k spatial speech corpus of more than 100 hours in total. We use 27k corpus as training set, 3k corpus as development set and 3k corpus as test set.

For the multiple speaker scenario, we randomly choose two audio clips in the dataset of single speaker and add the waveform up to simulate the two speaker scenario. In this way, we create another 100 hours of spatial speech corpus. Together with the single-speaker corpus, we use the blended corpus to train the multiple-speaker tracking model. The number of audio clips in training set, development set and test set is 60k, 6k and 6k, respectively.

B. Implementation Details

For GCC-PHAT calculation, we split one speech corpus to chunks with 368 window size to match the simulation frequency. One chunk is calculated with six previous chunks and six later chunks. The $n_{fft}$ is set to 1024 and the hop size is set to 320. The number of coefficients of delay lags is set to 96.

Both the transformer encoder and decoder use one transformer module to mimic the process of particle filter. The dimension of the query, key and value is 128. The hidden dimension of the fully-connected layer is 256. The number of heads for multi-head attention is 4.

For the particle filter, we use 30 particles with 128 hidden dimension. The alpha for soft-resampling is set to 0.2.

For model training, we adopt 5e-4 learning rate with learning rate drops of 50 epochs. The total number of training epochs is 100 with the Adam optimizer. We adopt the early stop mechanism with 30 patience. For Hungarian matching, $\lambda_{cls} = 3$ and $\lambda_{DOA} = 5/180$. For the learning objective, the hyper parameter $\lambda$ is set to 0.5. For the multiple speaker scenario, we adopt the EMD loss and cross entropy loss. We discard the ELBO loss to reduce the computational cost.

C. Evaluation Metrics

We use MAE and Accuracy to evaluate the model performance. The MAE is calculated as:

$$ MAE = \frac{1}{T \cdot M} \sum_{t=0}^{T} \sum_{i=0}^{M} (\pi - |\hat{d}_{t,i} - r_{t,i}|) $$

(25)

MAE is the average error along the time dimension within one trajectories and is smaller than 180 degree, where $T$ is the number of time step and $M$ is the number of speakers. The Accuracy is calculated as the percentage of the trajectories whose MAE is smaller than three degrees.

For the multiple speaker scenario, we also report the cardinality error, calculated as the absolute value of the estimated number of speakers and the ground truth.

D. Comparison with Other Methods

We compare the proposed model with other temporal prediction models. We compare the vanilla RNN, LSTM, and GRU. Other models which combine RNN and PF such as PF-LSTM and PF-GRU [24] are also compared. We choose the RNN-based models as comparative methods as they estimate the states in the current time step based on the current measurements and the previous states, which is similar to the Bayesian filters. When re-implementing the baseline models, we choose proper hidden dimension size such that the number of model parameters is almost the same. The GCC-PHAT is directly input to the RNN-based method to obtain DOA.

The experimental results on the simulated dataset are shown in Table 1. It is observed that the proposed particle filter transformer outperforms the baseline methods by a large margin. The transformer encoder can provide extracted features and the transformer decoder combined with particle filter can estimate the speaker state. LSTM and GRU have a better performance than that of vanilla RNN as LSTM and GRU has stronger capacity of modelling longer sequences through different gates to remember important information and discard useless information.

E. Ablation Study

We do the ablation study to show the effectiveness of the proposed end-to-end attention-based particle filter. The results of the ablation study are shown in Table 2. The first model uses transformer to obtain the measurements and uses the conventional particle filter for tracking. In each training iteration, only the transformer is optimized. For implementing the transformer for obtaining measurements, we use the transformer encoder and add an [CLS] token at the beginning of the GCC-PHAT. We use the first position of the output and pass it to a classification layer to get the DOA. We adjust the dimension of hidden layers to match

| TABLE 1. Experimental Results on the Simulated Dataset  |
|-----------------|-------------|
|                | MAE(°)      | Accuracy(%) |
| RNN            | 44.31       | 49.94       |
| LSTM           | 13.09       | 78.63       |
| GRU            | 10.27       | 80.78       |
| PF-LSTM [24]   | 23.24       | 69.42       |
| PF-GRU [24]    | 22.86       | 70.75       |
| Ours           | **4.40**    | **95.17**   |

| TABLE 2. Ablation Study          |
|-----------------|-------------|
|                | MAE(°)      | Accuracy(%) |
| w/o End2End Training        | 6.33        | 89.85       |
| w/o Temporal Prediction   | 4.44        | **95.33**   |
| Ours                      | **4.40**    | 95.17       |

ZHAO ET AL. : ATTENTION-BASED END-TO-END DIFFERENTIABLE PARTICLE FILTER FOR AUDIO SPEAKER TRACKING
<table>
<thead>
<tr>
<th>Model</th>
<th>20db MAE(°)</th>
<th>20db Accuracy(%)</th>
<th>10db MAE(°)</th>
<th>10db Accuracy(%)</th>
<th>0db MAE(°)</th>
<th>0db Accuracy(%)</th>
<th>-10db MAE(°)</th>
<th>-10db Accuracy(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN</td>
<td>62.02</td>
<td>35.96</td>
<td>69.89</td>
<td>29.17</td>
<td>78.33</td>
<td>20.88</td>
<td>83.51</td>
<td>13.63</td>
</tr>
<tr>
<td>LSTM</td>
<td>50.97</td>
<td>43.47</td>
<td>63.21</td>
<td>33.36</td>
<td>75.48</td>
<td>21.78</td>
<td>83.32</td>
<td>13.44</td>
</tr>
<tr>
<td>GRU</td>
<td>51.60</td>
<td>44.40</td>
<td>65.20</td>
<td>32.66</td>
<td>75.87</td>
<td>21.54</td>
<td>84.36</td>
<td>12.18</td>
</tr>
<tr>
<td>PF-LSTM [24]</td>
<td>57.95</td>
<td>39.68</td>
<td>67.31</td>
<td>31.27</td>
<td>77.41</td>
<td>21.49</td>
<td>83.44</td>
<td>13.38</td>
</tr>
<tr>
<td>PF-GRU [24]</td>
<td>60.89</td>
<td>38.34</td>
<td>70.37</td>
<td>29.78</td>
<td>80.00</td>
<td>20.22</td>
<td>86.44</td>
<td>12.06</td>
</tr>
<tr>
<td>Ours</td>
<td>18.39</td>
<td>78.86</td>
<td>32.03</td>
<td>63.10</td>
<td>48.53</td>
<td>43.96</td>
<td>65.50</td>
<td>25.73</td>
</tr>
<tr>
<td>Ours*</td>
<td>11.92</td>
<td>85.86</td>
<td>17.71</td>
<td>78.88</td>
<td>27.08</td>
<td>68.39</td>
<td>38.05</td>
<td>55.00</td>
</tr>
</tbody>
</table>

### G. The Impact of Noise

The simulated dataset we generate does not have noise in the binaural audio. However, in the real applications audio is always contaminated with noise. Therefore in this section we explore the model robustness against noise. So we adding noise to the development set and the test set of the simulated dataset. The noise we use is from DEMAND [35], which provides noise from different scenarios including office, park, sports fields and so on. The noise is adding on the magnitude spectrum with Signal-to-Noise Ratio (SNR) of 20db, 10db, 0db and -10db. The experimental results are demonstrated in 3. We use the pre-trained models on the clean training set and evaluate it on the noisy version of the test set without finetuning the model. It is observed that our model performs better than the baseline models on all SNR levels. The performance of RNN is the worst due to its simplest architecture. The performance of other models are in the same level. On the SNR level of 20db, 10db and 0db, our model outperforms the baseline models to a large extent (accuracy increased almost 20% on 20db and 10db, and increased almost 10% on 0db), which shows that our model is more robust to additive noise. On the SNR level of -10db, the power of the noise is bigger than that of the speaker audio and this scenario is very challenging. The performance of all models is not satisfactory. However, in the more noisy environment our model still obtain 10% increased accuracy on -10db compared to the recurrent baseline model.

To increase the model robustness against noise, we also train the model on the noisy dataset. Specifically, we add Gaussian white noise to the training set and the SNR level is 20dB. We test the model on the noisy dataset contaminated by DEMAND [35]. It is observed that the model performance (Ours*) improves obviously as the Gaussian white noise is the most general form. The model trained with Gaussian white noise can be generalized to the specific noise.

### H. Visualization

To present the tracking results more intuitively, we show the trajectories and particle states in Fig. 4. As the DOA classification resolution is 1°, the estimation DOA cannot...
FIGURE 4. Visualization of trajectories. The horizontal axis is the time and the vertical axis is DOA. The red line and green line represent the estimated trajectories and ground truth, respectively. The star represents the particles whose colors represent the particle weights. Darker colors mean higher particle weights.

TABLE 4. Experimental Results on the Two-Speaker Scenario. (Card. denotes the cardinality error.)

<table>
<thead>
<tr>
<th></th>
<th>Card.</th>
<th>MAE(°)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DETR [26] + PMBM [36]</td>
<td>0.90</td>
<td>31.48</td>
</tr>
<tr>
<td>Ours</td>
<td>0.35</td>
<td>31.91</td>
</tr>
</tbody>
</table>

be decimal points and the estimated trajectories seem to be sawtoothed. It can be seen that at the start period, the particles are scattered in different positions, which is the same as conventional particle filters. After some iterations, the particles converge to certain points. Although several outliers exist (the impulses in the third and fifth sub-figures), the estimation is close to the ground truth trajectories.

I. Results on Multi-Speaker Scenario
We report the results of two speaker tracking in Table 4. We compare the baseline method DETR [26] combined with the Poisson multi-Bernoulli mixture (PMBM) filter [36], which is a two-stage process. DETR [26] is used to extract audio measurements. The vanilla DETR is proposed for the task of object detection, which has the classification head and a localization head to determine the position of the bounding box. We modify the classification head to binary classifier to determine the speaker existence and modify the localization head to 360 classification related to DOA angles. DETR is trained on the simulated two-speaker dataset. PMBM is used for state estimation. Compared to PF, PMBM can estimate the varying number of speakers while PF needs the number of speakers as a prior. PMBM uses Poisson distribution to represent undetected targets and uses multiple Bernoulli mixture to represent detected targets. PMBM has been used for vehicle tracking [37] and multiple speaker tracking [38]. PMBM takes the audio measurements from DETR [26] and estimates the number of speakers and each speaker’s DOA.

It is observed that the proposed method performs competitively with the baseline method on the MAE and outperforms the baseline method on the cardinality estimation. It can be seen that the multiple speaker tracking is more challenging than single speaker tracking. There are two reasons. On the one hand, the GCC-PHAT feature is hard to be adapted in the multiple speaker scenario [7]. The performance of GCC-PHAT degrades significantly as the number of speakers increases [13]. On the other, data association is needed to match the measurements with the target.

VI. Conclusion and Future Work
In this paper, we combine the particle filter and transformer. Particle filter gives explainability for the transformer and transformer brings strong measurement model. This combination abandons the traditional pattern of tracking, which first extracts measurements and then feeds the measurements to the Bayesian filter. Instead, it provides an end-to-end differentiable architecture. Experiments on the simulated dataset show that the proposed model has stronger modeling capacity and is more robust to long sequence and noise. In the future, we will improve the performance of multiple speaker tracking.
REFERENCES


