Transformer-based Multi-Target Tracking with Bayesian Perspective

Xinwei Wei, Yiru Lin, Linao Zhang, Zhiyuan Zou, Jianwei Wei, Wei Yi*

School of Information and Communication Engineering, University of Electronic Science and Technology of China

E-mail: Syinwei.Wei@gmail.com, linyiru2021@163.com, kussoyi@gmail.com

***Abstract*—The Bayesian inference has a two-step recursion structure, i.e., prediction and updating, which can be viewed as a dynamic reasoning process. Based on this elegant structure, various multi-target tracking (MTT) algorithms have been invented and successfully applied in many areas. On the other hand, Bayesian inference MTT algorithms are model-based methods that rely on models' accuracy and first-order Markov assumption. In recent years, the MTT algorithms based on deep learning have received much attention due to their model-free property and the ability to learn from data, although they have issues such as over-fitting, generalization, etc. In this work, we propose a Transformer-based multi-target tracker whose architecture mimics the Bayesian inference, referred to as the Bayesian inference-based Transformer (BAIT) for MTT. To deal with the model mismatch issues, BAIT uses neural networks instead of the pre-assumed motion and observation models while retaining the excellent architecture of Bayesian inference. BAIT can recursively complete accurate predictions and updates via Transformer by refining the estimation of target states in a Bayesian inference-like manner. Thus, BAIT can be viewed as a combination of model-based and data-based methods. The simulation results show that, because of combining the advantages of Bayesian architecture with intelligent data association structure, BAIT is competitive in simple scenarios and achieves superior performance when the data association task becomes complicated.**

***Index Terms*—Multi-target tracking, Bayesian inference, Data association, Transformer**

I. Introduction

Multi-target tracking (MTT) endeavors to predict the trajectories of targets with unknown and varying numbers over time, utilizing a series of imprecise sensor measurements. The applications of MTT traverse a myriad of fields, including autonomous driving [1], pedestrian tracking [2], underwater operations [3], aerial reconnaissance [4], and more. Consequently, researching and implementing high-performance solutions for this task holds significant importance.

The most widely used methods in the field of MTT are currently Bayesian inference-based algorithms. In scenarios where precise multitarget models are available, and the observations consist of low-dimensional, individual object detections, state-of-the-art (SOTA) performance is obtained through model-based Bayesian methods. The Kalman filter (KF) [5] based on Bayesian filtering is a classical method that utilizes Bayes' theorem and linear system dynamics model to achieve estimation and prediction of target trajectories by recursively updating the probability distribution of the target state. The random finite set (RFS) [6] framework is also widely used to model the MTT problems in a Bayesian way. For example, the probability hypothesis density (PHD) [7], [8] filter and its extension, the cardinalized PHD (CPHD) [9], [10] filter, represent significant applications of the Bayesian inference framework, employing probabilistic hypothesis density approaches and data association techniques. Other notable examples include the Poisson multi-Bernoulli mixture (PMBM) [11] filter and the generalized labeled multi-Bernoulli (GLMB) [12], [13] filter, both of which extend and refine the Bayesian inference framework, thereby theoretically enabling Bayes-optimal estimates.

The Bayesian inference framework offers significant advantages in MTT. Through prediction and update mechanisms, it efficiently handles uncertainties and data associations inherent in complex tracking environments and can provide accurate predictions by recursively refining the estimation of target states based on Bayesian inference, enhancing tracking precision and robustness. However, the Bayesian inference algorithms mentioned above are all model-based (MB) approaches, whose state-space models (SSM) are crucial in illustrating state evolution and sensor observations. The effectiveness of conventional Bayesian trackers relies on the accuracy of prior SSMs [14]; thus, their performance notably diminishes when confronted with model mismatches. Unfortunately, obtaining a comprehensive and precise prior SSM proves challenging in intricate and non-cooperative environments.

An attractive approach for solving these problems is the utilization of deep learning (DL), which typically optimizes models with a plethora of parameters by empirical risk minimization. [15]. Indeed, DL has been increasingly applied to MTT with breakthroughs in SOTA performance [16]. On the one hand, DL is used to handle certain sub-tasks, such as KalmanNet [17] proposed by G. Revach *et al.*, which uses an RNN for training and inference on Kalman gain to assist the Kalman filter. S. Eleftheriadis *et al.* proposed a novel inference method for learning the Gaussian process state space model (GPSSM) [18], addressing the model mismatch problem to some extent. The optimized Kalman filter (OKF) [19] proposed by I. Greenberg *et al.*, which adjusts numeric optimization to the positive-definite KF parameters. And the

This work was supported in part by the National Natural Science Foundation of China under Grant 62231008 and 62301127, the China Postdoctoral Science Foundation under Grant BX20220057, 2023M730509 and GZB20230112, the "Tianfu Qingcheng" Plan of Sichuan Province under Grant 1332 and 1395.

LGBF [20], which can avoid the model mismatch and Markov restriction in Bayesian methods.

In addition, recent advancements employ DL to tackle the entire task of MTT. Pinto *et al.* [21] proposed a high-performing neural network for MTT, named Multi-Target Tracking Transformer (MT3), which leverages the Transformer architecture. Experimental results show that MT3 outperforms SOTA Bayesian trackers in some complex scenes. However, MT3 operates as a single-frame prediction algorithm in which continuous prediction cannot be realized. Our previous work, state regressive MTT using Transformers [22], builds upon the original MT3 framework, which employs state autoregressive-based queries for recursive tracking to solve the continuous tracking problem. But it's still predicting the isolated states without data association and not forming genuine trajectories that cannot perform recursive prediction in ways conforming to tracking theory like Bayesian inference. To combine the track filtering with data association, Pinto *et al.* [23] proposed a novel DL architecture specifically tailored for decouples the data association task from the smoothing task. However, it is designed for smoothing task which processes and corrects previous predictions, not for online contunuous tracking. To the best of our knowledge, currently, no research in DL integrates the data association and track filtering tasks into one algorithm trained together to solve the entire online tracking task.

In this paper, we combine the classical Bayesian architecture for prediction and update, adding data association to the previous work and expanding it into a genuine multi-target tracker. To use neural networks instead of pre-assumed motion and observation models while retaining the excellent architecture of Bayesian, we propose the BAyesian Inference using Transformers (BAIT), which combines the advantages of recursive updating in the Bayesian architecture and the ability of the Transformer architecture to deal with long sequential tasks. BAIT uses one encoder to analyze the motion states of the past in the prediction process and uses a serial dual decoder architecture to perform the data association and update process, respectively. Although BAIT is currently unable to handle track initialization and can only track the initialized targets with pre-generated labels, in comparison with both the MT3 algorithm and the conventional KF algorithm, it demonstrates competitiveness in simpler scenarios while showing more pronounced advantages in situations where the data association task becomes complicated.

II. Models and Notations for MTT

This work follows the standard models and notations for point traces. T denotes the current frame under tracking, while t signifies a specific frame from the inception of sensor detection.

A. Target Motion and Sensor Observation Model

To define the target motion model as [24], we consider the evolution of the state sequence $\{\mathbf{x}_i^t, t \in \mathbb{N}\}$ of target i given by:

$$\mathbf{x}_i^t = \mathbf{f}^t\left(\mathbf{x}_i^{t-1}, \mathbf{v}^{t-1}\right), \tag{1}$$

where $\mathbf{f}^t : \mathbb{R}^{d_x} \times \mathbb{R}^{d_v} \to \mathbb{R}^{d_x}$ represents the state transition function from the state $\mathbf{x}^{t-1}$, the sequence $\{\mathbf{v}^t, t \in \mathbb{N}\}$ denotes an i.i.d. process noise, d_x and d_v denote the dimensions of the state and process noise vectors, $\mathbb{N}$ is the set of natural numbers.

Given that each existing object may yield no more than one measurement, the observation equation for the true measurements can be represented as follows:

$$\mathbf{z}_i^t = \mathbf{h}^t\left(\mathbf{x}_i^{t-1}, \mathbf{n}^t\right), \tag{2}$$

where $\mathbf{h}^t : \mathbb{R}^{d_x} \times \mathbb{R}^{d_n} \to \mathbb{R}^{d_z}$ represents the observation function, while $\mathbf{n}^t, t \in \mathbb{N}$ stands for an i.i.d. sequence of measurement noise. The dimensions d_z and d_n correspond to those of the measurement and measurement noise vectors.

Clutters are modeled by a Poisson point process (PPP) characterized by intensity λ_c, operating independently of existing objects or true measurements. In frame t, the set of all measurements is represented by:

$$\mathbb{Z}^t = \bigcup_i \mathbf{z}_i^t \cup \mathbb{C}^t, \tag{3}$$

where $\mathbb{C}^t$ represents the set of clutters in frame t.

Thus the Bayesian inference process can be represented as (4) and (5):
Predict:

$$p\left(\mathbf{x}^t|\mathbf{z}^{1:t-1}\right) = \int p\left(\mathbf{x}^t|\mathbf{x}^{t-1}\right) p\left(\mathbf{x}^{t-1}|\mathbf{z}^{1:t-1}\right) d\mathbf{x}^{t-1}, \tag{4}$$

Update:

$$\begin{aligned} p\left(\mathbf{x}^t|\mathbf{z}^{1:t}\right) &= \frac{1}{Z^t} p\left(\mathbf{z}^t|\mathbf{x}^t\right) p\left(\mathbf{x}^t|\mathbf{z}^{1:t-1}\right), \\ Z^t &= \int p\left(\mathbf{z}^t|\mathbf{x}^t\right) p\left(\mathbf{x}^t|\mathbf{z}^{1:t-1}\right) d\mathbf{x}^t. \end{aligned} \tag{5}$$

B. Data Association Task Formulation

To explain the data association problem, we expand the measurements $\mathbb{Z}^t$ in equation (3) as

$$\mathbb{Z}^t = \left\{\mathbf{Z}_i^t\right\}_{i=1}^{m^t} = \left\{\mathbf{Z}_1^t, \mathbf{Z}_2^t, \cdots, \mathbf{Z}_{m^t}^t\right\}, \tag{6}$$

where $\mathbf{Z}_i^t$ denotes the i th measurement in the frame t, and m^t is the number of measurements in the frame t.

The state estimation set of the targets in the frame t is defined as

$$\mathbb{S}^t = \left\{\mathbf{S}_j^t\right\}_{j=1}^{n^t} = \left\{\mathbf{S}_1^t, \mathbf{S}_2^t, \cdots, \mathbf{S}_{n^t}^t\right\}, \tag{7}$$

where $\mathbf{S}_j^t$ denotes the state of j th target in the frame t, and n^t is the number of targets in the frame t. To deal with the false alarms, a dummy trajectory (similar to the dummy measurement in [25]) is introduced. Thus, the equation (7) is redefined by adding a dummy state, indexed $j = 0$, to each of the sets $\mathbb{S}^t$. Formally:

$$\mathbb{S}^t = \left\{\mathbf{S}_j^t\right\}_{j=0}^{n^t}. \tag{8}$$

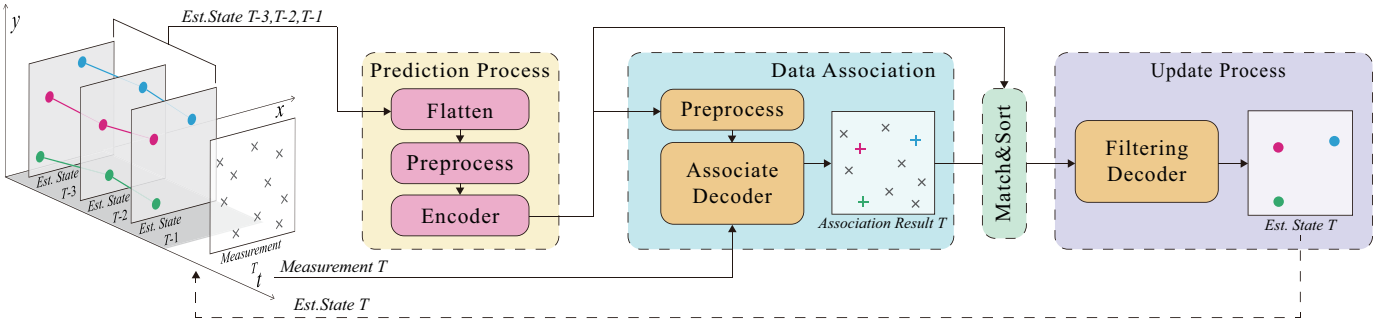

Fig. 1. High-level architecture of BAIT. The estimate states of the last τ frames (grey filled in the left stereogram) are first fed into a Transformer encoder after the flattening and preprocess, producing embeddings that represent the motion characteristics during the previous certain period, called the implicit prediction process (yellow background rounded rectangle). Then, the associate decoder performs the data association (blue background rounded rectangle) to match the measurement in the current frame (transparent background in the left stereogram) and trajectory according to these characteristics. After the measurement-trajectory pairs are transformed to arranged measurements by the match & sort mechanism (green background rounded rectangle), the filtering decoder (purple background rounded rectangle) will perform the update process and output the estimate states in the current frame. The results will return to the Transformer encoder for recursive estimation.

C. Target Tracking Task Formulation

In this investigation, we divide the target-tracking tasks into data association tasks and state filtering tasks. We focus on the problem of multi-target state estimation with identified labels using the measurements of current frame T and a sequence of estimation from τ timesteps in the past until the last frame $T-1$. The sequence of estimation is denoted as

$$\begin{aligned}\hat{\mathbb{X}}^{T-\tau:T-1} &= \left\{\hat{\mathbb{X}}^{t}\right\}_{t=T-\tau}^{T-1} \\ &= \left\{\hat{\mathbb{X}}^{T-\tau}, \cdots \hat{\mathbb{X}}^{T-2}, \hat{\mathbb{X}}^{T-1}\right\},\end{aligned} \tag{9}$$

where

$$\hat{\mathbb{X}}^{t} = \left\{\hat{\mathbf{X}}_{k}^{t}\right\}_{k=1}^{n^{t}} = \left\{\hat{\mathbf{X}}_{1}^{t}, \hat{\mathbf{X}}_{2}^{t}, \cdots \hat{\mathbf{X}}_{n^{t}}^{t}\right\}, \tag{10}$$

$$\hat{\mathbf{X}}_{k}^{t} = \left[\hat{\ell}_{k}^{t}, \hat{x}_{k}^{t}, \hat{y}_{k}^{t}, t\right]', \tag{11}$$

where $'$ stands for matrix transpose, n^t is the number of estimations in the frame t, $\hat{x}_k^t$ and $\hat{y}_k^t$ are the X and Y coordinates of estimated target k, $\hat{\ell}_k^t \in \mathbb{L} = \{\alpha_i : i \in \mathbb{N}\}$ is the unique label of estimated target k, where $\mathbb{N}$ signifies the set of positive integers, and the parameters α_i are distinct.

The measurements of the current frame are denoted as

$$\mathbb{Z}^{T} = \left\{\mathbf{Z}_{k}^{T}\right\}_{k=1}^{m^{T}} = \left\{\mathbf{Z}_{1}^{T}, \mathbf{Z}_{2}^{T}, \cdots, \mathbf{Z}_{m^{T}}^{T}\right\}, \tag{12}$$

$$\mathbf{Z}_{k}^{T} = \left[x_{k}^{T}, y_{k}^{T}\right]', \tag{13}$$

where m^T is the number of measurements in current frame T, x_k^T and y_k^T are the X and Y coordinates of measurement k.

The result of the target tracking task is a set of estimated labels and states for targets in the current frame, formally:

$$\hat{\mathbb{X}}^{T} = \left\{\hat{\mathbf{X}}_{k}^{T}\right\}_{k=1}^{n^{T}} = \left\{\hat{\mathbf{X}}_{1}^{T}, \hat{\mathbf{X}}_{2}^{T}, \cdots \hat{\mathbf{X}}_{n^{T}}^{T}\right\}, \tag{14}$$

$$\hat{\mathbf{X}}_{k}^{T} = \left[\hat{\ell}_{k}^{T}, \hat{x}_{k}^{T}, \hat{y}_{k}^{T}\right]'. \tag{15}$$

III. Bayesian Inference using Transformers

We present BAIT, an MTT approach based on a sequence-to-sequence Transformer [26] architecture, which is the abbreviation of BAyesian Inference using Transformers.

A. BAIT Architecture

BAIT employs an encoder-decoder architecture comprising a Transformer encoder and two Transformer decoders, as illustrated in Fig. 1.

Concretely, the estimated states within a period τ are collected in a set $\hat{\mathbb{X}}^{T-\tau:T-1}$ according to equation (9). And each measurement in current frame $\mathbf{Z}_k^T$ is added in random order to the sequence $\mathbb{Z}^T$ in equation (12).

The embeddings are generated from the transformation of $\hat{\mathbb{X}}^{T-\tau:T-1}$ by the Transformer encoder in the prediction process. Then it is fed to the associate decoder, together with the queries $\mathbb{Z}^T$, to produce the probability of which trajectory each measurement belongs to, and can further be generated as the association results—measurement-trajectory pairs by the match & sort mechanism. Finally, these pairs are arranged and input to the filtering decoder to generate the estimate states $\hat{\mathbb{X}}^T$ in the update process. The result $\hat{\mathbb{X}}^T$ will return to the Transformer encoder as input recursively at the next frame $T+1$.

B. BAIT Tracking

The tracking result of BAIT is transformed following the three processes below according to Bayesian inference processes in (4) and (5):

(1) Prediction process:

$$\begin{aligned}p\left(\hat{\mathbb{X}}^{T}|\mathbb{Z}^{T-\tau:T-1}\right) &= \mathcal{F}_{P}\left(p\left(\hat{\mathbb{X}}^{T-1}|\mathbb{Z}^{T-\tau:T-1}\right)\right) \\ &= \mathcal{F}_{P}\left(p\left(\hat{\mathbb{X}}^{T-\tau:T-1}\right)\right),\end{aligned} \tag{16}$$

(2) Data association:

$$p\left(\mathbb{Z}^{T}|\hat{\mathbb{X}}^{T}\right) = \mathcal{F}_{A}\left(p\left(\hat{\mathbb{X}}^{T}|\mathbb{Z}^{T-\tau:T-1}\right), p\left(\mathbb{Z}^{T}\right)\right), \tag{17}$$

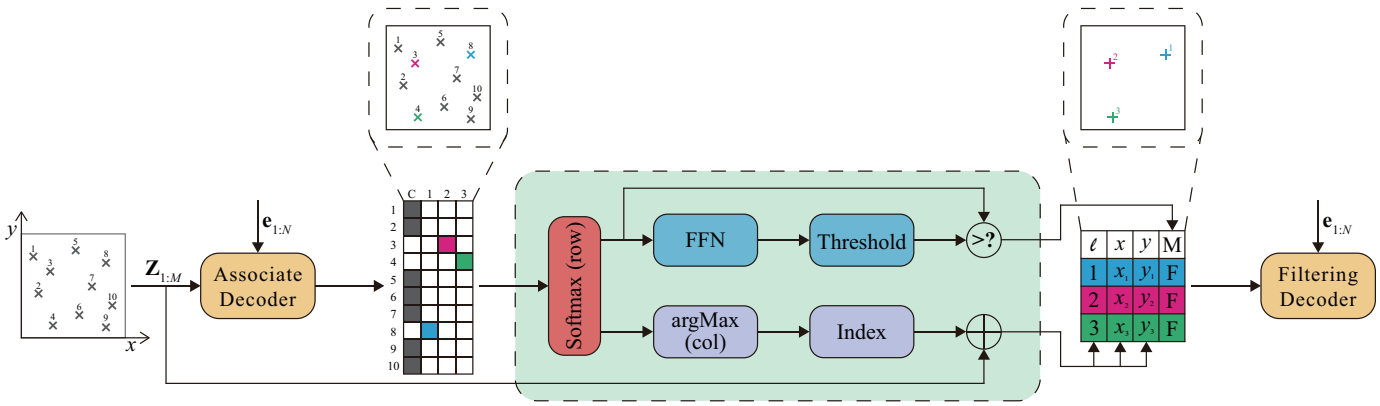

Fig. 2. Diagram of the processes between the associate decoder and filtering decoder. The associate decoder transforms the measurements $\mathbb{Z}^T$ (visualized in the leftmost graph, 10 measurements for example) and the embedding $\mathbf{e}_{1:N}$ to a match probability matrix (10×4 table on the left). Each row in the matrix represents a measurement detected in the current frame, each column represents a trajectory, and each grid denotes the probability of the corresponding measurement coming from the corresponding trajectory. The max probabilities of each row are colored, and their corresponding measurements are also colored in the visualization scene graph in the dashed square above the table. The match & sort mechanism (green filled) will then select and arrange the measurements to fill the filtering query matrix (3×4 table on the right) according to the match probability matrix. Each row in the matrix represents a matched-to-track measurement with a unique label (visualize in the dashed square above the table); clutters are excluded from the queries. These queries are finally fed to the filtering decoder to perform the update process.

(3) Update process:

$$p\left(\hat{\mathbb{X}}^T|\mathbb{Z}^{T-\tau:T}\right) = \mathcal{F}_U\left(p\left(\mathbb{Z}^T|\hat{\mathbb{X}}^T\right), p\left(\hat{\mathbb{X}}^T|\mathbb{Z}^{T-\tau:T-1}\right)\right), \tag{18}$$

where p denotes the probability density function (pdf), $\mathcal{F}$ represents a part of the deep-learning network in our algorithm to replace the Bayesian implement like the Chapman–Kolmogorov equation with a similar function.

1) Prediction Process: Transformer Encoder:

A Transformer encoder is responsible for prediction in our algorithm. After flattening and preprocessing, the encoder will extract the characteristics of the previous estimate states to an embedding. The embedding is an alternative representation of the probabilistic model for the state evolutions $\left\{p\left(\hat{\mathbb{X}}^t|\hat{\mathbb{X}}^{t-1}\right)\right\}_{t=T-\tau}^{T}$, making the encoder obtain the capability of performing implicitly prediction.

2) Data Association: Associate Decoder:

The associate decoder connects the prediction and the update, matching the corresponding trajectories from which each measurement is generated, and provides the filtering decoder with these matched pairs, shown in Fig. 2.

The output of the associate decoder is a match probability matrix (MPM), which is used to represent the probability of each measurement coming from each trajectory. Formally:

$$\left[\hat{\mathbf{p}}_1, \hat{\mathbf{p}}_2, \cdots, \hat{\mathbf{p}}_m\right]', \tag{19}$$

where

$$\hat{\mathbf{p}}_i = \left[\hat{p}_{i,1}, \hat{p}_{i,2}, \cdots, \hat{p}_{i,s}\right], \tag{20}$$

where $\hat{p}_{i,j}$ denotes the probability of measurement i coming from trajectory j, m is the number of the measurements in the current frame, s is the max number of trajectory, the dummy trajectory $s = 0$ (Label 'C' of MPM in Fig. 2) denotes clutters.

The match & sort mechanism firstly performs softmax along the row of MPM for probability normalization. Then, it will select the most possibly generated measurements for each existing track by performing argmax along every column of MPM, producing the filtering query $\mathbf{q}_{1:s}$.

On the other branch, a linear layer processes the match probabilities to calculate an existence threshold. Tracks with a probability lower than that are considered terminated.

3) Update Process: Filtering Decoder:

After prediction and data association, the filtering decoder is in charge of the update process by transforming the filtering query and the embedding to the estimate states $\mathbf{o}_{1:k}$, where k represents the number of existing tracks in current frame.

These outputs will regress to the prediction process as $\hat{\mathbb{X}}^T$ for the next frame recursion after appending the current time stamp T.

C. BAIT Losses

All tracking predictions are supervised using both the association loss and the filtering loss.

Given MPM in (19) and ground-truth probabilities for all measurements at frame T:

$$\begin{aligned} \mathbf{a}_{1:m} &= \left(\left(p_{1,1}, \cdots, p_{1,s}\right), \left(p_{2,1}, \cdots\right), \cdots, \left(p_{m,1}, \cdots\right)\right), \\ \mathbf{l}_{1:m} &= \left(\ell_1, \ell_2, \cdots \ell_m\right), \quad \ell_i \in \mathbb{N}^s, \quad i \in \mathbb{N}^m. \end{aligned} \tag{21}$$

The association loss consists of cross-entropy (CE) loss and dice loss like the MCD loss in [27] and can be futher simplified

as:

$$\begin{aligned}\text{Loss}_{\text{Association}} &= \text{Loss}_{\text{CE}} + \text{Loss}_{\text{Dice}} \\ &= -\frac{1}{m}\sum_{i=1}^{m}\log\left(p_{i,\ell_i}\right) \\ &+ 1 - \sum_{i=1}^{m}\frac{2p_{i,\ell_i}+\gamma}{p_{i,\ell_i}^2+\gamma}.\end{aligned} \tag{22}$$

where γ is a smoothing term to prevent loss overflow.

For the filtering loss, given predictions and ground-truth states for all estimate states at frame T:

$$\begin{aligned}\mathbf{o}_{1:k} &= \left(\left(\ell_1, \hat{\mathbf{x}}_1\right), \left(\ell_2, \hat{\mathbf{x}}_2\right), \cdots, \left(\ell_k, \hat{\mathbf{x}}_k\right)\right), \\ \mathbf{g}_{1:|\mathbb{X}^T|} &= \left(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{|\mathbb{X}^T|}\right),\end{aligned} \tag{23}$$

where $\mathbb{X}^T$ is the number of true targets in frame T. Thus, the filtering loss is represented as:

$$\text{Loss}_{\text{Filtering}} = \sum_{i=1}^{k}\text{Loss}_{\text{SL1}}\left(\mathbf{o}_i, \mathbf{g}_{\sigma_*(i)}\right), \tag{24}$$

where Loss_{SL1} is the smooth L1 loss, σ_* is a match that pairs the $\mathbf{o}_i$ and $\mathbf{g}_{\ell_i}$. Since BAIT does not have the capability of track initialization, the labels of the tracks are generated in external track initiation algorithm before BAIT tracking. So that the match σ_* is also synchronously determined at the beginning. $\mathbf{g}_{\sigma_*(i)}$ will be $\emptyset$ if $\mathbf{o}_i$ has no corresponding $\mathbf{g}_{\ell_i}$ (ℓ_i is not included in $\mathbb{N}^{[\mathbb{X}^T]}$).

The total loss for backpropagation is the sum of association loss and filtering loss with respective factors.

IV. Results

The performance of BAIT is assessed in two tasks which are compared against the original MT3 [28] and KF [5] with cheap joint probabilistic data association (CJPDA) [29] for evaluation.

A. Parameters for Tasks

The performance of three trackers is compared in task 1, and the tracking details of BAIT and KF are compared in task 2:

- Task 1: Tracking in a common scenario with a moderate amount of clutters.
- Task 2: Tracking in a scenario with more clutters means a complex challenge for data association.

In all the tasks, the velocity $v \sim \pm\mathbf{U}\left(10, 20\right)\text{m/s}$, the field of view is a 2D square$[-30\text{m}, 30\text{m}] \times [-30\text{m}, 30\text{m}]$, the detection probability $P_d = 0.95$, $\Delta t = 0.1\text{s}$ is the sampling period, and we use Poisson models with parameter $\lambda_0 = 8$ for the initial number of targets.

Task 1 has $q_s = 0.09\text{m}^2/\text{s}^2$, $R = 0.01\text{m}^2$, $\lambda_c = 10$, and the motion duration $T = 2\text{s}$. In this task, we compare the continuous tracking capability of three trackers.

Task 2 has $q_s = 0.09\text{m}^2/\text{s}^2$, $R = 0.01\text{m}^2$, $\lambda_c = 20$, and the motion duration $T = 2\text{s}$. Since MT3 only predicts the target state and does not matching the estimate state to the target, we compare the tracking performance of BAIT with KF and CJPDA in the more challenging task 2 and visualize the effect of data association of BAIT.

B. Parameters for MT3 and BAIT

We use the same parameters as the original work [28] for MT3: 6 encoder and 6 decoder layers, and in all of them, the multiheaded self-attention layers have 8 heads. All FFN layers are comprised of 2048 hidden units, and the increased state dimensionality d' is set to 256. MT3 was trained for each task starting with random weights, for 800k gradient descent steps, using Adam optimizer [30] with a batch size of 16.

For BAIT, the associate decoder has 3 decoder layers, and the FFN layers inside them comprise 1024 hidden layers. The filtering decoder and the other parameters are set to be the same as the decoder and the other parameters in MT3.

C. Performance Metrics

In task1, we use the optimal sub-pattern assignment (OSPA) metric [31] to evaluate all the trackers at the difference between estimations $\hat{\mathbb{X}}$ and truth $\mathbb{X}$.

In task 2, unlike task 1, the objective is to estimate the target trajectories over time rather than the target states at each frame. We evaluate the performance of KF and BAIT using the OSPA$^{(2)}$ metric [32].

In both tasks, we choose the cutoff distance $c = 1$ and the order $p = 1$ for all the 1k Monte Carlo simulations.

D. Task 1 Results

The average OSPA score curve for task 1 is shown in fig. 3. The scores are recorded from the 5th frame onwards since MT3 and BAIT are initializing in the first 4 frames.

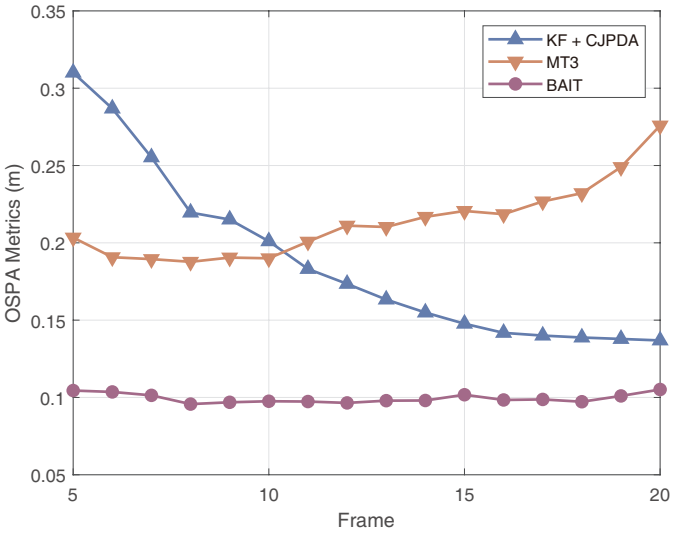

Fig. 3. Curves of OSPA scores for task 1. The blue upward triangle, the orange downward triangle, and the purple circle represent KF, MT3, and BAIT, respectively.

The OSPA score indicates that the tracking accuracy of BAIT remains stable throughout the entire tracking process and is significantly better than KF and MT3. The error of KF is a little high at first but gradually declines over time, and the decreasing trend has slowed down near frame 10. The performance of MT3 is stable and better than KF before

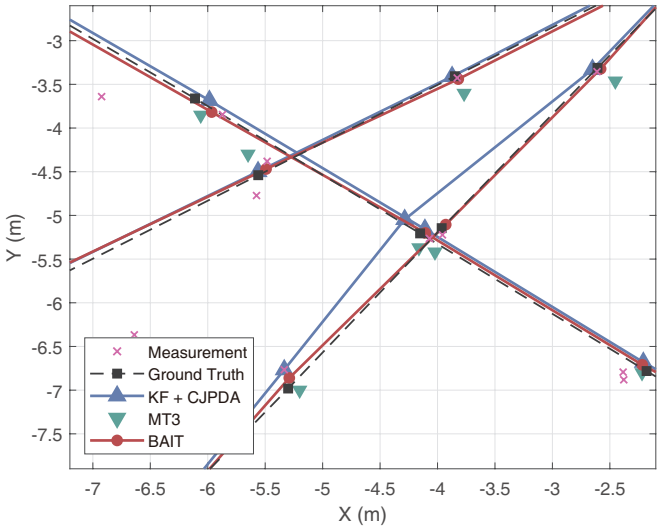

Fig. 4. Evaluation sample for task 1. This sample has 8 trajectories, and this figure is a localized zoom-in and involves three targets. The measurements are illustrated as magenta crosses, while black circles denote the ground-truth positions. Predictions for the three methods are presented: KF is represented by the blue upward triangle, MT3 by the green downward triangle, and BAIT by the red circle.

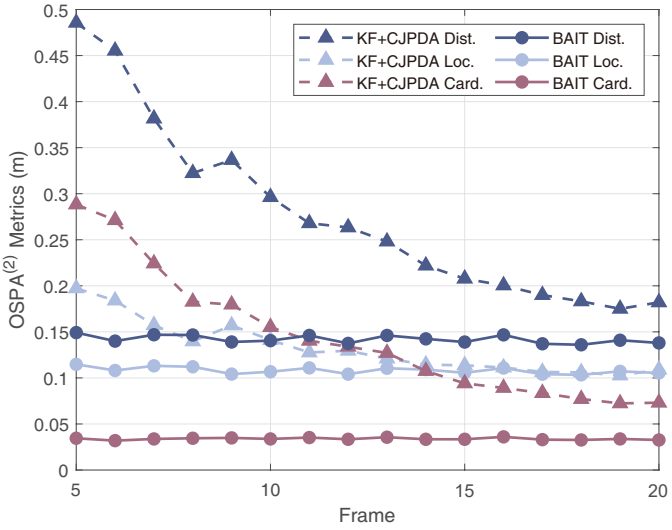

Fig. 5. Curves of OSPA$^{(2)}$ scores for task 2. Use the upward triangle for KF and the circle for BAIT for clarity. In the legend, Dist., Loc., and Card. represent distance, cardinality, and localization in the OSPA$^{(2)}$ metrics, respectively.

frame 10, but it gradually diverges and has worsened since then. These may be caused by most of the tracks crossing near frame 10. MT3 is a single-frame prediction algorithm without independent data association; thus, the prediction error during the track crossing will accumulate during the following prediction process and interfere with the tracking accuracy.

Fig. 4 is a visual sample of task 1. It can be seen that the localization accuracy of BAIT is higher than KF and MT3 for most of the points (e.g. at approximately (-4, -3.5) and (-5.3, -7)). BAIT fully utilizes the recursive advantage of Bayesian algorithms and the long sequence comprehension capability of Transformer architecture, achieving an accurate grasp of the trajectory trends. (eg. at approximately (-6, -4), the tracking accuracy of the three algorithms is very close, but the connecting line of BAIT is closer to the ground truth). And because of this accurate grasp, BAIT can maintain a stable performance without interference while KF has more significant errors due to the competition between targets for measurement, or MT3 performs worse predictions due to the track crossing (e.g. at approximately (-4, -5)).

E. Task 2 Results

The resulting average OSPA$^{(2)}$ scores for task 2 are shown in fig. 5. OSPA$^{(2)}$ distance is combined with localization and cardinality errors.

The localization error is the distance between the ground truth and the estimated states. It can be seen that the localization error of BAIT is lower than KF before the 14th frame, and their errors become close later. The cardinality error indicates the accuracy of the target number estimation. When the estimated number deviates from the truth, the cardinality error increases. The cardinality curve demonstrates that BAIT's cardinality error is stable at a lower level than KF's. In general, BAIT can obtain better tracking performance in this challenging scenario.

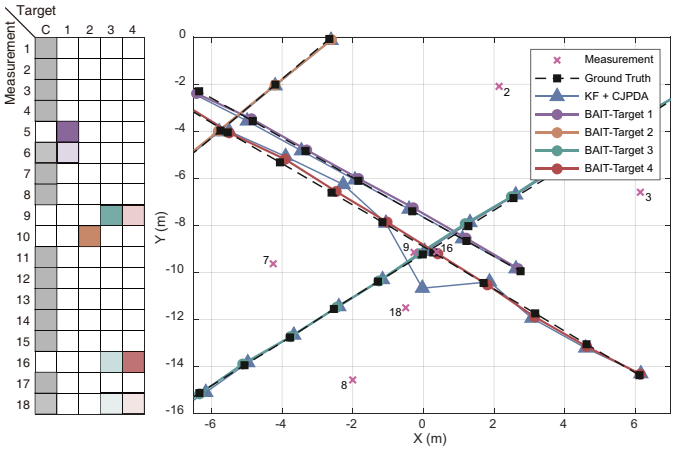

Fig. 6. Evaluation sample for task 2. This sample has 8 trajectories, and this figure is a localized zoom-in and involves four targets. In the right figure, we draw all the measurement points in the 9th frame for this example, while different colors represent the tracking trajectories for different BAIT targets. The left figure is a visualization of MPM in the 9th frame, where the higher color saturation indicates the larger association probability, and 'C' stands for clutter.

Fig. 6 is a visual sample of tracking results (right) and MPM (left) for task 2. It can be seen from the right graph that when the track of target 3 and target 4 are crossing at frame 9, KF experiences significant tracking errors because of the adjacent measurement interference and association fault, while BAIT can also perform accurate measurement-target association and maintain outstanding tracking performance. The results indicate our algorithm can choose the maximum measurement-target matching probabilities from the MPM calculated by the

associate decoder to surmount the interference and guarantee excellent estimate accuracy.

V. Conclusion

In this work, we employ neural networks instead of pre-assumed motion and observation models, presenting a Transformer-based multi-target tracker named BAIT, which combines model-free merits with the advantages of the Bayesian architecture.

We compare our algorithm with MT3 and KF+CJPDA. The results show that, in some complex scenarios, BAIT can complete the data association task with higher accuracy and achieve better performance in multi-target tracking.

In the future, we intend to implement a track management module to facilitate the initialization and allocation of track labels. This will enable the algorithm to manage the entire tracking process without the need for an additional external track initiation method or pre-generated labels.

References

[1] H.-k. Chiu, J. Li, R. Ambruş, and J. Bohg, "Probabilistic 3D multi-modal, multi-object tracking for autonomous driving," in *2021 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2021, pp. 14 227–14 233.

[2] S. Han, P. Huang, H. Wang, E. Yu, D. Liu, and X. Pan, "Mat: Motion-aware multi-object tracking," *Neurocomputing*, vol. 476, pp. 75–86, 2022.

[3] K. Panetta, L. Kezebou, V. Oludare, and S. Agaian, "Comprehensive underwater object tracking benchmark dataset and underwater image enhancement with gan," *IEEE Journal of Oceanic Engineering*, vol. 47, no. 1, pp. 59–75, 2021.

[4] S. Hossain and D.-j. Lee, "Deep learning-based real-time multiple-object detection and tracking from aerial imagery via a flying robot with gpu-based embedded devices," *Sensors*, vol. 19, no. 15, p. 3371, 2019.

[5] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 03 1960.

[6] R. Mahler, *Statistical multisource-multitarget information fusion*. Artech, 2007.

[7] R. P. Mahler, "Multitarget Bayes filtering via first-order multitarget moments," *IEEE Transactions on Aerospace and Electronic systems*, vol. 39, no. 4, pp. 1152–1178, 2003.

[8] B.-N. Vo and W.-K. Ma, "The Gaussian mixture probability hypothesis density filter," *IEEE Transactions on signal processing*, vol. 54, no. 11, pp. 4091–4104, 2006.

[9] R. Mahler, "PHD filters of higher order in target number," *IEEE Transactions on Aerospace and Electronic systems*, vol. 43, no. 4, pp. 1523–1543, 2007.

[10] B.-T. Vo, B.-N. Vo, and A. Cantoni, "Analytic implementations of the cardinalized probability hypothesis density filter," *IEEE transactions on signal processing*, vol. 55, no. 7, pp. 3553–3567, 2007.

[11] Á. F. García-Fernández, J. L. Williams, K. Granström, and L. Svensson, "Poisson multi-Bernoulli mixture filter: Direct derivation and implementation," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 54, no. 4, pp. 1883–1901, 2018.

[12] B.-T. Vo and B.-N. Vo, "Labeled random finite sets and multi-object conjugate priors," *IEEE Transactions on Signal Processing*, vol. 61, no. 13, pp. 3460–3475, 2013.

[13] B.-N. Vo, B.-T. Vo, and D. Phung, "Labeled random finite sets and the Bayes multi-target tracking filter," *IEEE Transactions on Signal Processing*, vol. 62, no. 24, pp. 6554–6567, 2014.

[14] M. Khodarahmi and V. Maihami, "A review on Kalman filter models," *Archives of Computational Methods in Engineering*, vol. 30, no. 1, pp. 727–747, 2023.

[15] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT press, 2016.

[16] C.-Y. Chong, "An overview of machine learning methods for multiple target tracking," in *2021 IEEE 24th International Conference on Information Fusion (FUSION)*. IEEE, 2021, pp. 1–9.

[17] G. Revach, N. Shlezinger, X. Ni, A. L. Escoriza, R. J. Van Sloun, and Y. C. Eldar, "KalmanNet: Neural network aided Kalman filtering for partially known dynamics," *IEEE Transactions on Signal Processing*, vol. 70, pp. 1532–1547, 2022.

[18] S. Eleftheriadis, T. Nicholson, M. Deisenroth, and J. Hensman, "Identification of Gaussian process state space models," *Advances in neural information processing systems*, vol. 30, 2017.

[19] I. Greenberg, N. Yannay, and S. Mannor, "Optimization or architecture: How to hack Kalman filtering," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[20] C. Zhang, J. Deng, and W. Yi, "Data-driven online tracking filter architecture: A lightgbm implementation," *Signal Processing*, vol. 221, p. 109477, 2024.

[21] J. Pinto, G. Hess, W. Ljungbergh, Y. Xia, H. Wymeersch, and L. Svensson, "Deep learning for model-based multi-object tracking," *IEEE Transactions on Aerospace and Electronic Systems*, 2023.

[22] X. Wei, L. Chen, C. Zhang, Y. Lin, L. Zhang, X. Liu, J. Jiang, and W. Yi, "Transformer based online continuous multi-target tracking with state regression," in *2023 12th International Conference on Control, Automation and Information Sciences (ICCAIS)*. IEEE, 2023, pp. 393–398.

[23] J. Pinto, G. Hess, Y. Xia, H. Wymeersch, and L. Svensson, "Transformer-based multi-object smoothing with decoupled data association and smoothing," *arXiv preprint arXiv:2312.17261*, 2023.

[24] M. 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*, vol. 50, no. 2, pp. 174–188, 2002.

[25] P. Storms and F. Spieksma, "An LP-based algorithm for the data association problem in multitarget tracking," in *Proceedings of the Third International Conference on Information Fusion*, vol. 1, 2000, pp. TUD2/10–TUD2/16 vol.1.

[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[27] L. Wenna, Z. Shunsheng, and W. Wenqin, "Multitarget-tracking method for airborne radar based on a Transformer network," *Journal of Radars*, vol. 11, no. 3, pp. 469–478, 2022.

[28] J. Pinto, G. Hess, W. Ljungbergh, Y. Xia, L. Svensson, and H. Wymeersch, "Next generation multitarget trackers: Random finite set methods vs transformer-based deep learning," in *Proc. Int. Conf. Inform. Fusion.* IEEE, 2021, pp. 1–8.

[29] R. J. Fitzgerald, "Development of practical PDA logic for multitarget tracking by microprocessor," in *1986 American Control Conference*. IEEE, 1986, pp. 889–898.

[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[31] D. Schuhmacher, B.-T. Vo, and B.-N. Vo, "A consistent metric for performance evaluation of multi-object filters," *IEEE Trans. Signal Process.*, vol. 56, no. 8, pp. 3447–3457, 2008.

[32] M. Beard, B. T. Vo, and B.-N. Vo, "OSPA (2): Using the OSPA metric to evaluate multi-target tracking performance," in *2017 International Conference on Control, Automation and Information Sciences (ICCAIS)*. IEEE, 2017, pp. 86–91.