Transformer-based Multi-Sensor Hybrid Fusion for Multi-Target Tracking

Xinwei Wei, Linao Zhang, Yiru Lin, Jianwei Wei, Chenyu Zhang, Wei Yi*

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

E-mail: Syinwei.Wei@gmail.com, linao819@163.com, kussoyi@gmail.com

***Abstract*—Deep learning (DL) approaches, which do not rely on models and can learn complex relationships within data, garner increasing attention in the model-free multi-target tracking (MTT) domain. However, the study of applying the DL method to multi-sensor fusion-based MTT is relatively less. In this paper, we propose a Transformer-based distributed multi-sensor MTT approach, which adopts a hybrid fusion structure with both feature-level and decision-level fusion. First, for each local sensor, the high-dimensional feature information is extracted from the measurements based on a Transformer-based tracking module, which enables continuous tracking of multiple targets and provides the predicted target states and corresponding uncertainties. Then, the outputs of local sensors are fused using the covariance interception (CI) fusion rule. Finally, to further improve the fusion performance, the decision-level information is fed into a fusion decoder with the feature-level information to obtain the predicted target state and uncertainties after deep fusion. In this way, we realize a deep utilization of different sensors' information and achieve a feature-level decision-level hybrid multi-sensor fusion, namely, Transformer-based multi-sensor hybrid fusion (TMSHF). Simulation results show that the proposed fusion method outperforms the CI algorithm in various tracking scenarios.**

***Index Terms*—Multi-sensor fusion, Transformer, Multi-target tracking, Distributed fusion**

I. Introduction

Multi-target tracking (MTT) involves continuously estimating the positions and movements of multiple dynamic targets over time, using sensor data that may contain noise. Whether in military applications like aerial reconnaissance [1] or civilian domains such as autonomous driving [2], pedestrian tracking [3], traffic management [4], MTT technology holds significant application value. The approach to solving MTT problems depends on whether they operate in a model-based or model-free environment.

Model-based Bayesian methods achieve state-of-the-art performance in cases where accurate multi-target models are known and the observations consist of low-dimensional individual target detections. There are three leading Bayesian filtering-based solutions for MTT, which are the joint probabilistic data association filter (JPDAF) [5], multiple hypothesis tracking (MHT) [6]–[8], and random finite set (RFS) approaches. In JPDAF and MHT, the core idea is to first address the data association problem between sensor data and targets, then apply traditional filtering methods for tracking [9]. In contrast, the RFS methods provide a holistic approach that simultaneously addresses both tracking and association problems. Representative examples include the PHD filter [10], [11], CPHD filter [12], [13], multi-Bernoulli (MB) filter [14]–[16], PMBM filter [17] and GLMB filter [18], [19], both of which utilize the RFS framework to address tracking problems, achieving excellent performance.

To achieve better performance, multi-sensor data fusion has gradually become the research focus. It has been widely used in target detection [20], image processing [21], [22], industrial prognosis [23], emotion recognition [24], and so on. Existing multi-sensor target tracking methods can be classified into two categories based on the type of shared information between sensors: feature-level and decision-level.

In short, the former is mainly devoted to computing accurate multi-sensor likelihood probabilities/posterior probabilities. It usually uses a centralized sensor network in which all sensors' measurements are sent to a fusion center to be federated for optimal fusion. This requires each sensor to communicate with a central node, demanding high computational, storage, and communication capabilities while exhibiting weak fault tolerance.

In contrast, decision-level fusion fuses the a posteriori estimates produced by the local filters. This method, typically implemented in a distributed manner, offers better computational scalability and reliability. The covariance intersection (CI) [25] and arithmetic average (AA) [26] are two widely adopted fusion rules for distributed fusion. For example, the CI fusion rule can fuse unknown correlation estimates generated by different but not necessarily independent sensors to avoid repeated calculation of information in fusion. In practice, fused nodes are usually unable to determine the correlation between their information and the information received by other sensor nodes. The assumption of independence [27] between different sensor nodes is not applicable in most of the real-world problems, so CI fusion rules are used to fuse the first and second order statistics provided by the local sensor nodes [28] to obtain the fused MTT results.

In recent years, the utilization of deep learning (DL) in model-free MTT has significantly risen, and several neural networks have been successfully invented, such as KalmanNet [29], OKF [30], GPSSM [31] and LGBF [32]. Notably, 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.

MT3 proposed by Pinto *et al.* [33] is a high-performance, end-to-end MTT neural network based on the Transformer architecture. The MT3v2 algorithm [34] further brings the opportunities for multi-senor fusion using deep learning method [35]. However, the MT3v2 algorithm cannot achieve continuous prediction and only realizes the fusion of single-frame target state estimation results. Later in [36], to address continuous target tracking problems, an SR-MT3 method is proposed by incorporating recursive tracking in the original MT3 based on state-autoregressive queries.

This paper proposes a Transformer-based multi-sensor hybrid fusion (TMSHF) method based on feature-level and decision-level fusion. Firstly, we extract high-dimensional feature information from measurements based on an SR-MT3 tracking module within each local sensor. The outputs of multiple local sensors are then fused at the initial decision level using the CI fusion rule to avoid double-counting of common information. Finally, to achieve better fusion performance, the decision-level and feature-level information are input into the decoder together as embedding and query, respectively, to obtain the predicted target state and uncertainty after deep fusion. This way, we realize the deep utilization of different sensor information and form a feature-level decision-level hybrid multi-sensor fusion method. In the numerical simulation, we demonstrate the efficacy of the proposed TMSHF method by comparing it with the tracking results of the Kalman filtering (KF) based CI fusion.

II. Models and Notations

In this paper, we adhere to conventional multitarget transition and observation models tailored for point traces, designate T to denote a specific frame we are tracking and t to denote a particular frame since the sensor commenced detecting measurements.

A. Target Motion and Sensor Observation Model

To define the target motion model as [37], we consider the evolution of the state sequence $\{\mathbf{x}_k^t, t \in \mathbb{N}\}$ of target k, which is expressed as:

$$\mathbf{x}_{t,k} = \mathbf{f}_t\left(\mathbf{x}_{t-1,k}, \mathbf{v}_{t-1}\right), \tag{1}$$

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

Consider a group of sensors with the same field of view (FOV) detecting the same set of targets, and considering that each existing object can generate at most one measurement, the observation equation for the actual measurements captured by sensor s can be expressed as follows:

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

where $\mathbf{h}_t^s : \mathcal{R}^{d_x} \times \mathcal{R}^{d_n} \to \mathcal{R}^{d_z}$ is the observation function, $\mathbf{n}_t^s, t \in \mathbb{N}$ is an i.i.d. measurement noise sequence. Also, d_z and d_n are dimensions of the measurement and measurement noise vectors, respectively.

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

$$\mathbb{Z}_t^s = \bigcup_k \mathbf{z}_{t,k}^s \cup \mathbb{C}_t^s, \tag{3}$$

where $\mathbb{C}_t^s$ is the set of clutters in the frame t generated by sensor s.

When assuming that the posterior density at each time step is Gaussian, equation (1) and (2) can be rewritten as:

$$\mathbf{x}_t = \mathbf{F}_t \mathbf{x}_{t-1} + \mathbf{v}_{t-1}, \tag{4}$$

$$\mathbf{z}_t^s = \mathbf{H}_t^s \mathbf{x}^t + \mathbf{n}_t^s, \tag{5}$$

where $\mathbf{F}_t$ and $\mathbf{H}_t^s$ represent known matrices defining the linear functions. we consider the case when $\mathbf{v}_{t-1}$ and $\mathbf{n}_t^s$ have zero mean and are statistically independent, the covariances of them are respectively $\mathbf{Q}^{t-1}$ and $\mathbf{R}_t^s$.

B. Single Sensor Tracking Task Formulation

In single sensor tracking task, we focus on the problem of multi-target state estimation in the domain of sensor s using the estimations of last frame $\hat{\mathbb{X}}_{T-1}^s$ and a sequence of measurements $\mathbb{Z}_{T-\tau:T}^s$ from τ time-steps in the past until the current frame T. The sequence of measurements detected by sensor s is denoted as

$$\mathbb{Z}_{T-\tau:T}^s = \left\{\mathbb{Z}_{T-\tau}^s, \cdots, \mathbb{Z}_{T-1}^s, \mathbb{Z}_T^s\right\}, \tag{6}$$

where

$$\mathbb{Z}_t^s = \left\{\mathbf{Z}_{t,k}^s\right\}_{k=1}^{m_t^s} = \left\{\mathbf{Z}_{t,1}^s, \mathbf{Z}_{t,2}^s, \cdots \mathbf{Z}_{t,m_t^s}^s\right\}, \tag{7}$$

$$\mathbf{Z}_{t,k}^s = \left[x_{t,k}^s, y_{t,k}^s, t\right]', \tag{8}$$

where $'$ is the matrix transpose operation, m_t^s is the number of measurements detected by sensor s in the frame t, $x_{t,k}^s$ and $y_{t,k}^s$ are the X and Y coordinates of measurement k.

The target states estimated by sensor s in the last frame are denoted as

$$\begin{aligned}\hat{\mathbb{X}}_{T-1}^s &= \left\{\hat{\mathbf{X}}_{T-1,k}^s\right\}_{k=1}^{n_{T-1}^s} \\ &= \left\{\hat{\mathbf{X}}_{T-1,1}^s, \hat{\mathbf{X}}_{T-1,2}^s, \cdots, \hat{\mathbf{X}}_{T-1,n_{T-1}^s}^s\right\},\end{aligned} \tag{9}$$

$$\hat{\mathbf{X}}_{t,k}^s = \left[\hat{\mathbf{x}}_{t,k}^s, \mathbf{\Sigma}_{t,k}^s\right] = \left[\hat{x}_{t,k}^s, \hat{y}_{t,k}^s, \hat{\sigma}_{x_{t,k}}^s, \hat{\sigma}_{y_{t,k}}^s\right], \tag{10}$$

where n_{T-1}^s is the number of targets estimated by sensor s in the last frame, $\hat{\mathbf{x}}_{t,k}^s = \left[\hat{x}_{t,k}^s, \hat{y}_{t,k}^s\right]$ is the coordinate of target k, $\mathbf{\Sigma}_{t,k}^s = \left[\hat{\sigma}_{x_{t,k}}^s, \hat{\sigma}_{y_{t,k}}^s\right]$ is the standard deviation of covariance of target k.

The result of the single sensor tracking task from each sensor is a set of estimated states and standard deviations for targets in the current frame, formally:

$$\hat{\mathbb{X}}_T^s = \left\{\hat{\mathbf{X}}_{T,k}^s\right\}_{k=1}^{n_T^s} = \left\{\hat{\mathbf{X}}_{T,1}^s, \hat{\mathbf{X}}_{T,2}^s, \cdots, \hat{\mathbf{X}}_{T,n_T^s}^s\right\}. \tag{11}$$

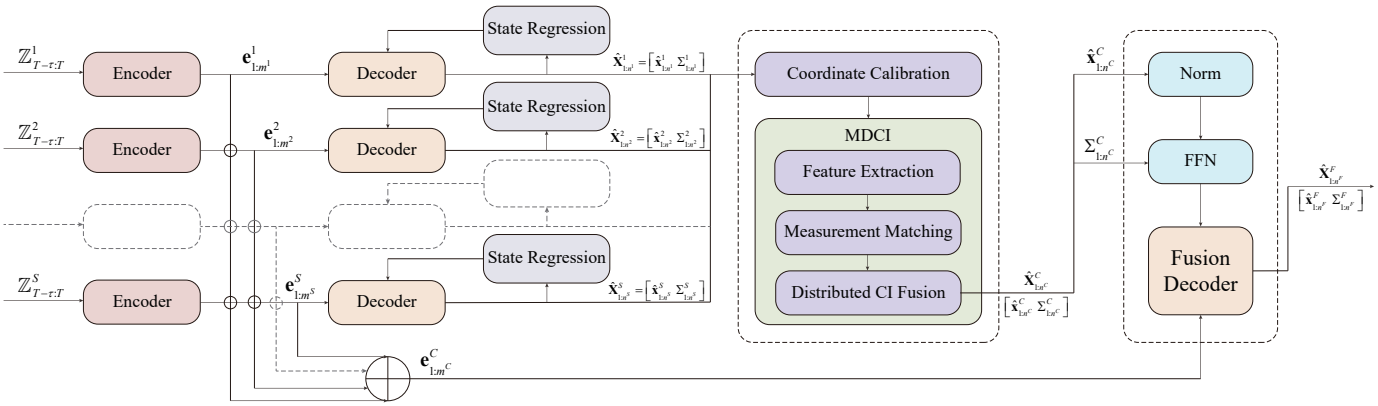

Fig. 1. High level architechture of TMSHF. The measurements $\mathbb{Z}^s_{T-\tau:T}$ in a period τ detected by each sensor s from 1 to S are first fed into S encoders respectively, generating their own embeddings $\mathbf{e}^s_{1:m^s}$. The decoders corresponding to the encoders then transform these embeddings and the estimations from the last frame into the predictions $\hat{\mathbf{X}}^s_{1:n^s}$ in the current frame T for sensor s. The estimated results are returned to the decoder after recursive processing by the state regression mechanism for the single sensor tracking task above. The coordinate calibration aligns a copy of these estimations and inputs them to the MDCI fusion algorithm to form the preliminary fusion result $\hat{\mathbf{X}}^C_{1:n^C}$. Ultimately, the fusion decoder generates the final fusion result $\hat{\mathbf{X}}^F_{1:n^F}$ using the queries generated by Norm and FFN from $\hat{\mathbf{X}}^C_{1:n^C}$ and a concatenated embeddings $\mathbf{e}^C_{1:m^C}$ from the embeddings of each encoder.

C. Multi Sensor Fusion Task Formulation

We adopt a hybrid fusion method combining feature and decision fusion for better performance in multi-sensor fusion tasks.

We use the measurements from all the sensors for the feature fusion, formally:

$$\mathbb{Z}^{1:S}_{T-\tau:T} = \left\{ \{\mathbb{Z}^s_t\}_{t=T-\tau}^{T} \right\}_{s=1}^{S}, \tag{12}$$

where S is the number of the sensors involved in the feature fusion.

As for the decision fusion, all the estimate states from the single sensor tracking are used as input, formally:

$$\hat{\mathbb{X}}^{1:S}_T = \left\{ \hat{\mathbb{X}}^s_T \right\}_{s=1}^{S} = \left\{ \left\{ \hat{\mathbf{X}}^s_{T,k} \right\}_{k=1}^{n^s_T} \right\}_{s=1}^{S}. \tag{13}$$

A set of fused estimate states and standard deviations is the result of the multi-sensor fusion task, formally:

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

where the $\hat{\mathbf{X}}^F_{T,k}$ and the $\hat{\mathbb{X}}^s_T$ in (13) is defined as equation (10).

III. Transformer-based Multi-Sensor Hybrid Fusion

We present TMSHF, a multi-target multi-sensor fusion approach based on an sequence-to-sequence Transformer [38] architecture, which is the abbreviation of Transformer based Multi-Sensor Hybrid Fusion.

The network we introduce builds upon SR-MT3: State-Regressive Multi-Target Tracking with Transformers, a high-performing deep learning method for online continuous tracking based on the Transformer architecture [36].

A. TMSHF Architecture

TMSHF uses a parallel encoder-decoder architecture for each sensor to perform MTT independently and a single decoder for fusion to integrate the results from all sensors, illustrated in Fig. 1.

Notably, each measurements in a period τ are collected by sensor s in random order to a sequence $\mathbb{Z}^s_{T-\tau:T}$ according to equation (6). And the states estimates by sensor s in the last frame are added to the sequence $\hat{\mathbb{X}}^s_{T-1}$ in equation (9).

In the prediction process of each sensor, $\mathbb{Z}^s_{T-\tau:T}$ is transformed by the Transformer encoder into the embeddings $\mathbf{e}^s_{1:m^s}$. The $\mathbf{e}^s_{1:m^s}$ is then fed to a Transformer decoder, together with the queries $\hat{\mathbb{X}}^s_{T-1}$, to produce the estimate states $\hat{\mathbb{X}}^s_T = \hat{\mathbf{X}}^s_{1:n^s}$. A copy of the estimate states is processed by the state regression mechanism and returned to the decoder as queries recursively in the next frame. The other copy is collected into the set $\hat{\mathbb{X}}^{1:S}_T$ together with the output of the other sensors. The multi-target distributed covariance intersection (MDCI) fusion algorithm then uses the estimation states set to perform preliminary fusion, generates the queries $\hat{\mathbb{X}}^C_T = \hat{\mathbf{X}}^C_{1:n^C}$ for the fusion decoder. These queries and the concatenation of the embeddings $\{\mathbf{e}^s_{1:m^s}\}_{s=1}^{S}$ are finally transformed by the fusion decoder to the final fusion estimation $\hat{\mathbb{X}}^F_T = \hat{\mathbf{X}}^F_{1:n^F}$.

B. Single Sensor Tracking

We use a network similar to SR-MT3 [36] for single-sensor tracking. The set of outputs is derived from k new-birth queries and k autoregressive queries generated by the state regression mechanism shown in fig. 2, where k represents a predetermined value equivalent to the maximum number of objects. Since each sensor has its own independent state regression mechanism with the same architecture and only regresses the last frame, we choose one of them as an example

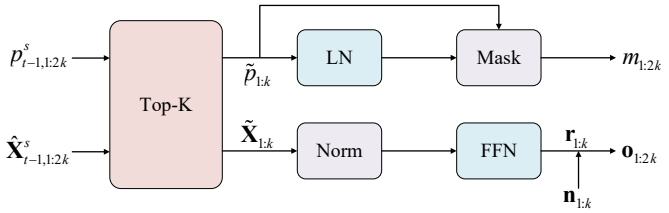

Fig. 2. Illustration of the state regression mechanism. The top-k promising states $\tilde{\mathbf{X}}_{1:k}$ is selected from the estimations $\hat{\mathbf{X}}^s_{t-1,1:2k}$ in the last frame by comparing the existence probabilities $p^s_{t-1,1:2k}$, and then fed into a Norm and an FFN layer to form the decoder queries $\mathbf{o}_{1:2k}$ with the new-birth queries $\mathbf{n}_{1:k}$. Meanwhile, a threshold g_t is calculated by the LN from the existence probabilities $\tilde{p}_{1:k}$ corresponding to $\tilde{\mathbf{X}}_{1:k}$ for creating a binary mask sequence $m_{1:2k}$.

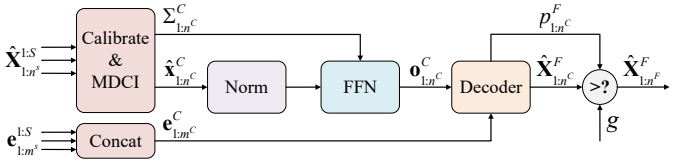

Fig. 3. Diagram of the hybrid fusion network. The output of the multi-target distributed CI fusion algorithm is composed of two components. One of them is the fused state $\hat{\mathbf{x}}^C_{1:n^C}$, which is then processed by the Norm and FFN to create a part of the queries. The other component—fused covariance corresponding to the state is input to FFN directly to form the fusion decoder queries $\mathbf{o}^C_{1:n^C}$ combined with the processed state. Meanwhile, the embeddings extracted by the encoders in each sensor's tracking process are concatenated with each other to form the embeddings for the fusion decoder. Ultimately, the fusion decoder transforms the embeddings and queries to the fusion estimates $\hat{\mathbf{X}}^F_{1:n^C}$ and corresponding existence probabilities $p^F_{1:n^C}$. The final fusion results $\hat{\mathbf{X}}^F_{1:n^F}$ is selected from $\hat{\mathbf{X}}^F_{1:n^C}$ by comparing the existence probabilities $p^F_{1:n^C}$ with a pre-specified existence threshold g.

and omit the sensor label s and the time step t inside the mechanism.

The decoder queries $\mathbf{o}_{1:2k}$ is integrated with the new-birth queries $\mathbf{n}_{1:k}$ as [39] and the autoregressive queries $\mathbf{r}_{1:k}$ tranformed from $\tilde{\mathbf{X}}_{1:k}$ through Norm and FFN. Besides, we adopt a mask based on existence probabilities to mitigate the influence of outputs that should be terminated on the current estimation, which is calculated as:

$$m_i = \begin{cases} \text{True} & i \leqslant k \text{ and } \tilde{p}_i < g_t \\ \text{False} & i > k \text{ or } \tilde{p}_i \geqslant g_t \end{cases}, i \in \mathbb{N}^{2k}. \tag{15}$$

The new-birth queries $\mathbf{n}_{1:k}$ do not possess correlating existence probabilities; thus their masks are consistently set to false.

C. Multi Sensor Fusion

The multi-sensor fusion module is combined with the multi-target distributed CI fusion algorithm (MDCI) and a hybrid fusion network (HFN). The sensors' estimations are processed by MDCI and HFN sequentially after a coordinate system calibration to form the final fusion estimates. The time step T is omitted in this section since the fusion process only involves the current frame.

1) Multi-target Distributed CI Fusion:

MDCI is used for preliminary decision fusion and fusion decoder query generation.

In the total output of the decoder $\hat{\mathbb{X}}^{1:S}_T$, there are clutter and measurements from different sensors for different targets. To group the measurements from different sensors against the same target, we need to compare the distance between the different measurements and the range threshold δ. If the distance is greater than the threshold, the two measurements are distinguished as different targets. If the distance is less than the threshold, the two measurements are classified as one target.

Following the classification of the measurements, the CI fusion rule is used to fuse the estimates of the same target. The elegance of CI fusion lies in its inherent capacity to uphold the Gaussian property of local estimates denoted by $\hat{\mathbf{x}}^s_k$, elucidated given as [37]

$$\hat{\mathbf{x}}^s_k \sim \mathcal{N}\left(\mathbf{x}_k; \boldsymbol{\mu}^s_k, \boldsymbol{\Sigma}^s_k\right), \tag{16}$$

where $\boldsymbol{\mu}^s_k$ and $\boldsymbol{\Sigma}^s_k$ are mean and covariance of local estimates, respectively. Consequently, the fused estimate seamlessly fits a Gaussian distribution [40], represented as:

$$\hat{\mathbf{x}}^C_k \sim \mathcal{N}\left(\mathbf{x}_k; \boldsymbol{\mu}^C_k, \boldsymbol{\Sigma}^C_k\right), \tag{17}$$

where the expressions of the mean $\boldsymbol{\mu}^C_k$ and the covariance $\boldsymbol{\Sigma}^C_k$ are shown as follows:

$$\boldsymbol{\Sigma}^C_k = \left[\hat{\sigma}^C_{x_k}, \hat{\sigma}^C_{y_k}\right] = \left(\sum_{s=1}^{n^s} \omega^s_k \left(\boldsymbol{\Sigma}^s_k\right)^{-1}\right)^{-1}, \tag{18}$$

$$\boldsymbol{\mu}^C_k = \left[\hat{x}^C_k, \hat{y}^C_k\right] = \boldsymbol{\Sigma}^C_k \left(\sum_{s=1}^{n^s} \omega^s_k \left(\boldsymbol{\Sigma}^s_k\right)^{-1} \boldsymbol{\mu}^s_k\right), \tag{19}$$

and ω^s_k is the mixing weight of sensor s for the k-th target's data fusion satisfying $\sum_{s=1}^{n^s} \omega^s_k = 1$. Furthermore, the fusion result can be expressed as follows:

$$\hat{\mathbf{X}}^C_k = \left[\hat{x}^C_k, \hat{y}^C_k, \hat{\sigma}^C_{x_k}, \hat{\sigma}^C_{y_k}\right]. \tag{20}$$

Ultimately, $\hat{\mathbf{X}}^C_k$ is processed by HFN to produce the final fusion result $\hat{\mathbf{X}}^F_k$.

2) Hybrid Fusion Network:

The HFN is used for the actual execution of fusion tasks, shown in fig. 3.

In the single sensor tracking process, the features of the measurements detected by each sensor are extracted by the encoders into the embeddings $\{\mathbf{e}^s_{1:m^s}\}^S_{s=1}$. HFN performs the feature fusion by connecting these embeddings, formally:

$$\mathbf{e}^C_{1:m^C} = \text{concat}\left(\{\mathbf{e}^s_{1:m^s}\}^S_{s=1}\right), \; m^C = \sum_{s=1}^{S} m^s, \tag{21}$$

where m^s is the measurement number of sensor s, concat is an algorithm that connects a set of lists end to end, $\mathbf{e}^C_{1:m^C}$ is the embeddings for fusion decoder.

The decision fusion is firstly processed by MDCI mentioned in III-C1, generating a preliminary fusion result $\hat{\mathbf{X}}^C_{1:n^C}$. The state $\hat{\mathbf{x}}^C_{1:n^C}$ in $\hat{\mathbf{X}}^C_{1:n^C}$ is normalized by the FOV parameters, then input to an FFN together with the covariance $\mathbf{\Sigma}^C_{1:n^C}$ to form the fusion decoder queries $\mathbf{o}^C_{1:n^C}$ as

$$\mathbf{o}^C_i = \left[\text{FFN}\left(\text{Norm}\left(\hat{\mathbf{x}}^C_i\right)\right)\ \text{FFN}\left(\mathbf{\Sigma}^C_i\right)\right],\ 1 \leqslant i \leqslant n^C, \quad (22)$$

where n^C is the number of the existing targets after MDCI processing.

Ultimately, the feature of measurements and the results of estimates from all the sensors are stored in the embeddings $\mathbf{e}^C_{1:m^C}$ and the queries $\mathbf{o}^C_{1:n^C}$, respectively. The fusion decoder performs hybrid fusion and transforms them to the fusion estimates $\hat{\mathbf{X}}^F_{1:n^C}$. To determine whether an estimate is valid or not, a serial of corresponding existence probabilities $p^F_{1:n^C}$ is also output. The estimates in $\hat{\mathbf{X}}^F_{1:n^C}$ whose corresponding existence probability is larger than g will be selected into the final fusion estimation $\hat{\mathbb{X}}^F = \hat{\mathbf{X}}^F_{1:n^F}$, formally:

$$\mathbb{X}^F = \bigcup_{i=1}^{n^C} \tilde{\mathbf{X}}^F_i, \quad \tilde{\mathbf{X}}^F_i = \begin{cases} \hat{\mathbf{X}}^F_i & p^F_i \geqslant g \\ \emptyset & p^F_i < g \end{cases}, \quad (23)$$

$$n^F = \sum_{i=1}^{n^C} \varepsilon\left(p^F_i - g\right), \quad (24)$$

where $\varepsilon(\cdot)$ is step function.

D. TMSHF Losses

We specify the final fusion estimation $\hat{\mathbb{X}}^F_T$ as the output of a dummy sensor 0: $\hat{\mathbb{X}}^0_T$, thus the sensor label is changed to $s \in [0, S]$, and the time step T is omitted since we only calculate the losses of the current frame.

The estimate $\hat{\mathbf{X}}^s_{1:n^s}$ represent the parameters of a n^s-component multi-Bernoulli (MB) density, each $\hat{\mathbf{X}}^s_i, i \in \mathbb{N}^{n^s}$ is structured as $(\boldsymbol{\mu}^s_i, \mathbf{\Sigma}^s_i, p^s_i)$, encompassing the mean and covariance parameters for a Gaussian distribution, alongside the existence probability associated with that Bernoulli component. We supervise the predictions $\hat{\mathbb{X}}^{0:S}$ from all the sensors (including fusion result) using negative log-likelihood (NLL) losses like [34] between the ground truth $\mathbf{x}_{1:k}$:

$$\mathcal{L}\left(\mathbf{x}_{1:k}, \hat{\mathbf{X}}^s_{1:n^s}\right) = -\sum_{s=0}^{S} \log f^s\left(\mathbf{x}_{1:k}\right), \quad (25)$$

where $f^s\left(\mathbf{x}_{1:k}\right)$ represents the multi-Bernoulli density defined by $\hat{\mathbf{X}}^s_{1:n^s}$, evaluated at $\mathbf{x}_{1:k}$.

Since computing $f^s(\cdot)$ directly is computationally intractable, we extend the sequence $\mathbf{x}_{1:k}$ by appending $\emptyset$ elements, resulting in a new sequence $\tilde{\mathbf{x}}_{1:l}$ with the same number of elements as each $\hat{\mathbf{X}}^s_{1:n^s}$. Furthermore, we approximate the NLL as:

$$\begin{aligned} \log f^s\left(\mathbf{x}_{1:k}\right) &= \sum_{i=1}^{l} \log \sum_{\sigma} f^s_i\left(\tilde{\mathbf{x}}_{\sigma(i)}\right) \\ &\approx \sum_{i=1}^{l} \log f^s_i\left(\tilde{\mathbf{x}}_{\sigma(i)}\right), \end{aligned} \quad (26)$$

where σ represents a permutation function, denoted as $\sigma : \mathbb{N}^l \to \mathbb{N}^l$, with the condition $\sigma(i) = \sigma(j) \Rightarrow i = j$. This function signifies one potential association between MB components and ground-truth object states. $f^s_i\left(\tilde{\mathbf{x}}_{\sigma(i)}\right)$ represents the Bernoulli density specified by $\hat{\mathbf{X}}^s_i$ evaluated at the $\sigma(i)$-th element of $\tilde{\mathbf{x}}_{1:l}$:

$$-\log f^s_i\left(\tilde{\mathbf{x}}_j\right) = \begin{cases} \log p^s_i + \log \mathcal{N}\left(\tilde{\mathbf{x}}_j; \boldsymbol{\mu}^s_i, \mathbf{\Sigma}^s_i\right) & \tilde{\mathbf{x}}_j \neq \emptyset \\ \log\left(1 - p^s_i\right) & \text{otherwise.} \end{cases} \quad (27)$$

In the end, σ^s represents the most probable association between objects and Bernoulli components predicted for sensor s, approximated as

$$\sigma^s = \arg\min_{\sigma} \sum_{i=1}^{l} \mathcal{L}_{\text{match}}\left(\hat{\mathbf{X}}^s_i, \tilde{\mathbf{x}}_{\sigma(i)}\right), \quad (28)$$

where the efficient calculation of $\mathcal{L}_{\text{match}}$ can be achieved through the Hungarian algorithm [41] as

$$\mathcal{L}_{\text{match}}\left(\hat{\mathbf{X}}^s_i, \tilde{\mathbf{x}}_{\sigma(i)}\right) = \begin{cases} 0 & \tilde{\mathbf{x}}_{\sigma(i)} = \emptyset \\ \left\|\mu^s_i - \tilde{\mathbf{x}}_{\sigma(i)}\right\| - \log p^s_i & \text{otherwise.} \end{cases} \quad (29)$$

IV. Results

We take the scenario that contains two radars as an example to compare the tracking results of TMSHF and KF [42] with JPDA [5] across various sensor measurements and evaluate the fusion effects of CI fusion and TMSHF.

A. Parameters in Tasks

In task 1, we compare the tracking accuracy of TMSHF and KF with CI fusion. In task 2, we focus more on comparing the fusion accuracy and uncertainty of TMSHF and CI fusion.

In all the tasks, the velocity $v \sim \pm\mathbf{U}(5, 15)\,\text{m/s}$, the field of view is a 2D square$[-50\text{m}, 50\text{m}] \times [-50\text{m}, 50\text{m}]$, $\tau = 3$, $\Delta t = 0.1\text{s}$ is the sampling period, and we use Poisson models with parameter $\lambda_0 = 4$ for the initial number of targets.

There are two radars in both task 1 and task 2; radar 1 has $R = 0.09\text{m}^2$, detection probability $P_d = 0.98$, clutter intensity $\lambda_c = 3.0$ and radar 2 has $R = 0.16\text{m}^2$, detection probability $P_d = 0.95$, clutter intensity $\lambda_c = 3.0$. Consistency in tasks 1 and tasks 2, $q_s = 0.25\text{m}^2/\text{s}^2$, and the motion duration $T = 3\text{s}$.

B. Parameters for TMSHF

We use 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. The fusion decoder has the same arguments as the decoders in the single sensor

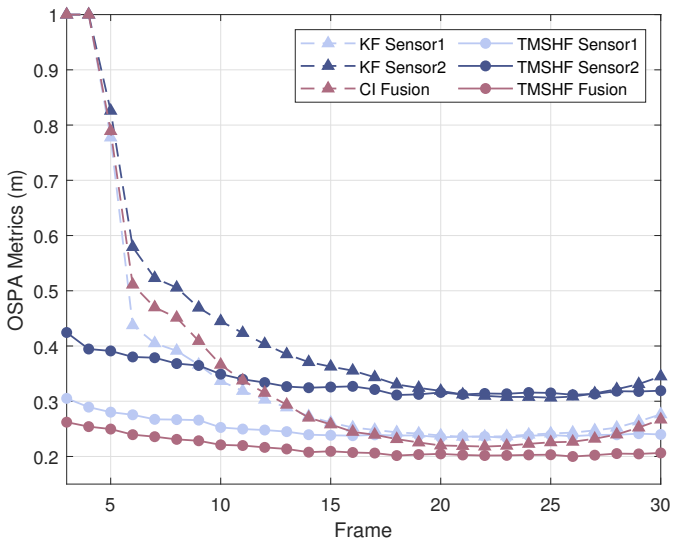

Fig. 4. Curves of OSPA scores for task 1. The light blue and dark blue triangles indicate the tracking effect of KF using sensor 1 and sensor 2 measurements, respectively. The light blue and dark blue circles indicate the tracking effect of TMSHF using sensor 1 and sensor 2 measurements, respectively. The purple triangles and circles represent the CI fusion effect with the KF predictions and the deep fusion effect of TMSHF, respectively.

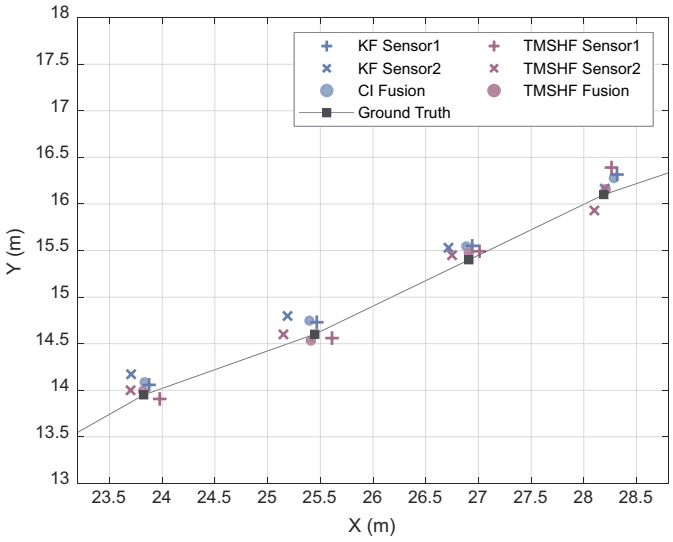

Fig. 5. Evaluation sample for task 1. This sample has three trajectories, and this figure is a zoomed-in view of one of the trajectories, showing the tracking effect of four frames under different algorithms. The blue addition and multiplication symbols represent the tracking results of KF using sensor 1 measurements and sensor 2 measurements, respectively. The purple addition and multiplication symbols represent the tracking results of TMSHF using sensor 1 measurements and sensor 2 measurements, respectively. The blue circles represent the CI fusion results with KF predictions, and the purple circles represent the deep fusion results of TMSHF. The line with the black square marker represents the true trajectory of the target.

tracking module. TMSHF was trained for each task starting with random weights, for 800k gradient descent steps, using Adam optimizer [43] with a batch size of 16.

C. Performance Metrics

We use the optimal sub-pattern assignment (OSPA) metric [44] to evaluate all the trackers at the difference between estimations $\hat{\mathbb{X}}$ and truth $\mathbb{X}$. 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 depicted in Fig. 4. The TMSHF algorithm undergoes initialization processes in the first two frames; hence, the average OSPA score curve is compared from the third frame onwards.

The graph indicates that the accuracy of TMSHF fusion is significantly higher than the tracking results achieved by utilizing measurements from a single sensor. The fusion performed by the CI algorithm with KF predictions does not yield outstanding results, with a slight advantage over sensor 1 only after roughly 15 frames. Overall, the TMSHF algorithm has higher tracking accuracy, both for single-sensor tracking and for the fusion with tracking results of different sensors.

Fig. 5 is a visual sample of task 1. For most of the points in the figure, the estimated target positions of TMSHF fusion are closer to the truth than those of CI fusion, consistent with the results presented in Fig. 4, showing the excellent fusion performance of TMSHF. From another perspective, the estimated positions of CI fusion tend to get closer to the position predicted by the sensor with higher accuracy (sensor 1). However, such a strategy may not always be correct, leading to the limitations of CI fusion. But, the fully trained TMSHF can efficiently overcome this problem and predict more accurate target positions.

E. Task 2 Results

In this task, the CI algorithm and TMSHF both perform the fusion process utilizing the local predictions of TMSHF to eliminate the influence of different single-sensor tracking results (KF and SR-MT3) and present the fusion effect more clearly. The average OSPA score curve for task 2 is depicted in Fig. 6.

As can be seen from the figure, the accuracy of TMSHF Fusion is significantly higher than that of single-sensor prediction. However, the fusion result of the CI algorithm is not as good as that of TMSHF sensor 1 itself (sensor 1 is the more accurate one). This suggests that only performing direct CI fusion using deep-learning results is unreliable.

Fig. 7 is a visual sample of task 2. The results on the left side indicate that when the target position truth is within the uncertainty range of sensor 1 and sensor 2, both CI and TMRHF can correctly predict the target position, but the uncertainty of TMRHF is lower. The results on the right side show that when the target position truth is outside the uncertainty range of one or even two sensors, the CI fusion still biases towards the local prediction with lower uncertainty, resulting in an incorrect prediction, while TMSHF can perform correct fusion despite this erroneous interference.

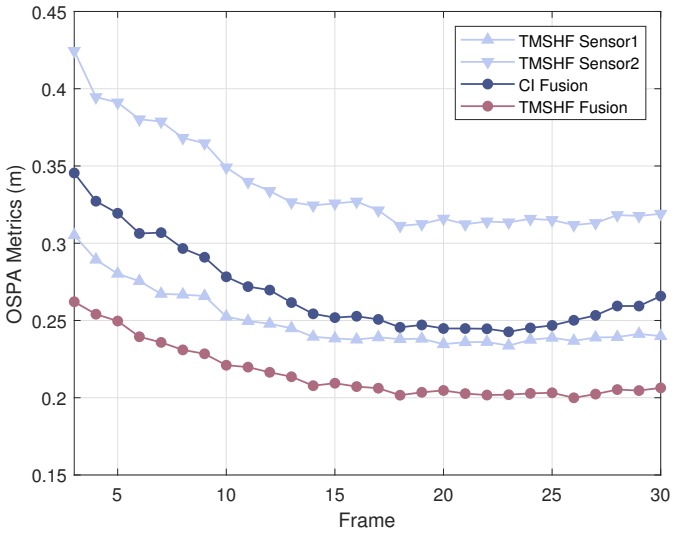

Fig. 6. Curves of OSPA scores for task 2. For clarity, the upward and downward triangles indicate the tracking performance of TMSHF using sensor 1 and sensor 2 measurements, respectively. The dark blue and purple circles indicate the CI fusion and TMSHF fusion effect using all local predictions of TMSHF, respectively.

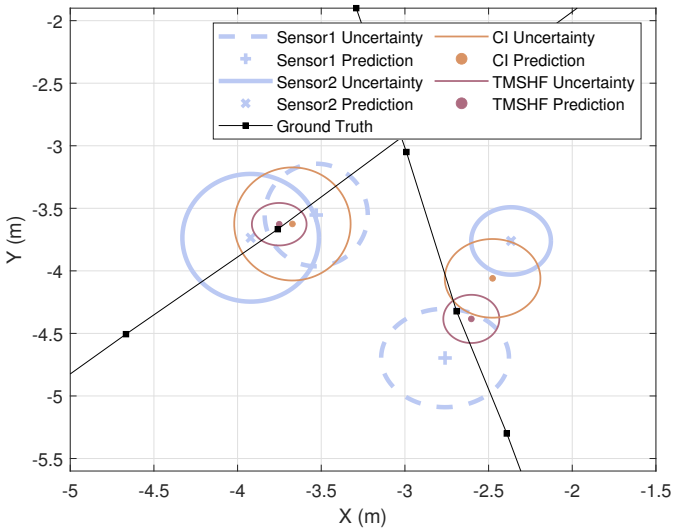

Fig. 7. Evaluation sample for task 2. This sample has four trajectories, and this figure is a partial zoom-in showing the predictions of different algorithms at frame 12. We represent the estimated uncertainties of sensor 1 and sensor 2 with blue dashed and solid circles, and indicate their estimated positions with blue additive and multiplicative symbols. Orange dots and circles depict the positions and uncertainties of the CI fusion, while purple ones depict those of TMSHF fusion.

V. Conclusion

In this work, we integrate the intelligent target tracking task and multi-sensor fusion task into a unified algorithm and realize a deep hybrid fusion for multiple sensors named TMSHF combined with feature fusion and decision fusion.

Compared to the tracking results of KF and the fusion results obtained through conventional CI methods, TMSHF exhibits remarkable advantages, particularly in intricate tasks, by delivering high precision and robustness in its fusion capabilities.

References

[1] Z. Cao, C. Fu, J. Ye, B. Li, and Y. Li, "HiFT: Hierarchical feature Transformer for aerial tracking," in *Proc. ICCV*, 2021, pp. 15 457–15 466.

[2] R. Ravindran, M. J. Santora, and M. M. Jamali, "Multi-object detection and tracking, based on DNN, for autonomous vehicles: A review," *IEEE Sensors Journal*, vol. 21, no. 5, pp. 5668–5677, 2020.

[3] P. Voigtlaender, M. Krause, A. Osep, J. Luiten, B. B. G. Sekar, A. Geiger, and B. Leibe, "Mots: Multi-object tracking and segmentation," in *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 2019, pp. 7942–7951.

[4] G. Wang, R. Gu, Z. Liu, W. Hu, M. Song, and J.-N. Hwang, "Track without appearance: Learn box and tracklet embedding with local and global motion patterns for vehicle tracking," in *Proc. ICCV*, 2021, pp. 9876–9886.

[5] T. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Sonar tracking of multiple targets using joint probabilistic data association," *IEEE journal of Oceanic Engineering*, vol. 8, no. 3, pp. 173–184, 1983.

[6] D. Reid, "An algorithm for tracking multiple targets," *IEEE transactions on Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979.

[7] R. L. Streit and T. E. Luginbuhl, "Maximum likelihood method for probabilistic multihypothesis tracking," in *Signal and data processing of small targets 1994*, vol. 2235. SPIE, 1994, pp. 394–405.

[8] S. S. Blackman, "Multiple hypothesis tracking for multiple target tracking," *IEEE Aerospace and Electronic Systems Magazine*, vol. 19, no. 1, pp. 5–18, 2004.

[9] B.-N. Vo, M. Mallick, Y. Bar-Shalom, S. Coraluppi, R. Osborne, R. Mahler, and B.-t. Vo, "Multitarget tracking," *Wiley encyclopedia of electrical and electronics engineering*, no. 2015, 2015.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

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

[15] B.-T. Vo, B.-N. Vo, and A. Cantoni, "The cardinality balanced multi-target multi-Bernoulli filter and its implementations," *IEEE Transactions on Signal Processing*, vol. 57, no. 2, pp. 409–423, 2008.

[16] B.-N. Vo, B.-T. Vo, N.-T. Pham, and D. Suter, "Joint detection and estimation of multiple objects from image observations," *IEEE Transactions on Signal Processing*, vol. 58, no. 10, pp. 5129–5141, 2010.

[17] 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.

[18] 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.

[19] 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.

[20] J. Ma, L. Tang, M. Xu, H. Zhang, and G. Xiao, "STDFusionNet: An infrared and visible image fusion network based on salient target detection," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–13, 2021.

[21] A. P. James and B. V. Dasarathy, "Medical image fusion: A survey of the state of the art," *Information fusion*, vol. 19, pp. 4–19, 2014.

[22] J. Ma, W. Yu, P. Liang, C. Li, and J. Jiang, "FusionGAN: A generative adversarial network for infrared and visible image fusion," *Information fusion*, vol. 48, pp. 11–26, 2019.

[23] A. Diez-Olivan, J. Del Ser, D. Galar, and B. Sierra, "Data fusion

and machine learning for industrial prognosis: Trends and perspectives towards industry 4.0," *Information Fusion*, vol. 50, pp. 92–111, 2019.
[24] S. Liu, P. Gao, Y. Li, W. Fu, and W. Ding, "Multi-modal fusion network with complementarity and importance for emotion recognition," *Information Sciences*, vol. 619, pp. 679–694, 2023.
[25] S. J. Julier and J. K. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," in *Proceedings of the 1997 American Control Conference (Cat. No. 97CH36041)*, vol. 4. IEEE, 1997, pp. 2369–2373.
[26] T. Li, H. Fan, J. García, and J. M. Corchado, "Second-order statistics analysis and comparison between arithmetic and geometric average fusion: Application to multi-sensor target tracking," *Information Fusion*, vol. 51, pp. 233–243, 2019.
[27] T. Bailey, S. Julier, and G. Agamennoni, "On conservative fusion of information with unknown non-Gaussian dependence," in *2012 15th International Conference on Information Fusion*. IEEE, 2012, pp. 1876–1883.
[28] S. J. Julier and J. K. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," in *Proceedings of the 1997 American Control Conference (Cat. No. 97CH36041)*, vol. 4. IEEE, 1997, pp. 2369–2373.
[29] 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.
[30] I. Greenberg, N. Yannay, and S. Mannor, "Optimization or architecture: How to hack Kalman filtering," *Advances in Neural Information Processing Systems*, vol. 36, 2024.
[31] 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.
[32] C. Zhang, J. Deng, and W. Yi, "Data-driven online tracking filter architecture: A lightgbm implementation," *Signal Processing*, vol. 221, p. 109477, 2024.
[33] 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.
[34] 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.
[35] L. Li, C. Dai, Y. Xia, and L. Svensson, "Deep fusion of multi-object densities using Transformer," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.
[36] 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.
[37] 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.
[38] 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.
[39] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *ECCV*. Springer, 2020, pp. 213–229.
[40] S. J. Julier, "An empirical study into the use of Chernoff information for robust, distributed fusion of Gaussian mixture models," in *2006 9th International Conference on Information Fusion*. IEEE, 2006, pp. 1–8.
[41] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.
[42] 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.
[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[44] 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.