Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

MH-T2TA: a multiple-hypothesis algorithm for multi-sensor track-to-track association with an intelligent track score*#

Pingliang XU, Yaqi CUI‡, Wei XIONG

Institute of Information Fusion, Naval Aviation University, Yantai 264001, China

E-mail: xu_pingliang@163.com; cui_yaqi@126.com; xiongwei@csif.org.cn

Received Apr. 28, 2024; Revision accepted July 30, 2024; Crosschecked Nov. 25, 2025; Published online Dec. 12, 2025

Abstract: Track-to-track association (T2TA), which aims at unifying track batch numbers and reducing track redundancy, serves as a precondition and foundation for track fusion and situation awareness. The current problems of T2TA come mainly from two sources: track data and association methods. Ubiquitous problems include errors and inconsistent update periods in track data, as well as suboptimal association results and dependencies on prior information and assumed motion models for association methods. Focusing on these two aspects, we propose a multiple-hypothesis algorithm for multi-sensor T2TA with an intelligent track score (MH-T2TA). A spatial–temporal registration module is designed based on self-attention and a contrastive learning architecture to eliminate errors and unify the distributions of asynchronous tracks. A multiple-hypothesis algorithm is combined with deep learning to estimate the association score of a pair of tracks without relying on prior information or assumed motion models, and the optimal association pairs can be obtained. With three kinds of loss functions, tracks coming from the same targets become closer, tracks coming from different targets become more distant, and the estimated track scores are very similar to the real ones. Experimental results demonstrate that the proposed MH-T2TA can associate tracks in complex scenarios and outperform other T2TA methods.

Key words: Track-to-track association; Multiple-hypothesis algorithm; Track score; Neural networks
https://doi.org/10.1631/FITEE.2400340 **CLC number:** TP274

1 Introduction

In distributed multi-sensor systems, tracks reported by multiple sensors are distributed within the same scenario. This can result in batch number confusion, track redundancy, and situational ambiguity, which in turn affect the subsequent fusion of tracks from the same target (Bar-Shalom et al., 1990; He et al., 2010; Zhu et al., 2016). Track-to-track association (T2TA) is used to associate tracks of the same target reported by different sensors, so as to further unify batch numbers, reduce redundancy, and clarify the situation (Bar-Shalom and Li, 1995; Klein and Bar-Shalom, 2016). At present, with the deployment of shore-, sea-, air-, and space-based sensors, it is easy to obtain a large amount of multi-sensor track data. However, the lack of track processing algorithms, especially track association algorithms, seriously affects the effective utilization of track data. Many methods focusing on the T2TA problem and based on different technologies have been proposed, such as statistical reasoning, fuzzy, and artificial intelligence methods. However, these methods have shortcomings such as unreasonable assumptions, inappropriate models, uncertain thresholds, long association times, and suboptimal association results. Thus, new algorithms are urgently needed.

‡ Corresponding author

* Project supported by the National Natural Science Foundation of China (Nos. 62171453 and U2433216)

Electronic supplementary materials: The online version of this article (https://doi.org/10.1631/FITEE.2400340) contains supplementary materials, which are available to authorized users

ORCID: Pingliang XU, https://orcid.org/0000-0001-8357-4592; Yaqi CUI, https://orcid.org/0000-0002-4408-9962

Statistical reasoning methods associate tracks by distance, probability, and hypothesis testing. Kanyuck and Singer (1970) associated tracks using a weighted distance test. However, this method uses information only from the tracks at the current time, ignoring historical track information. Considering that the association in complex scenarios (such as dense targets, cross targets, interference, and noise) requires historical track information, based on a sequential track association algorithm, He and Zhang (2006) proposed restricted and attenuated memory track association algorithms and sequential classic assignment rules. However, this method needs hypothesis testing for the Chi-square random variable, which is imprecise when tracks are affected by random and systematic errors (Bar-Shalom, 2008). Sun et al. (2023) proposed T2TA methods based on maximum likelihood estimation (MLE) to resolve the T2TA of compact high-frequency surface wave radar (HFSWR) tracks with large measurement errors. However, this method can be applied only to tracks with synchronous sampling points.

To alleviate the dependence on hypothesis testing and the effect of random and systematic errors, some anti-bias T2TA algorithms have been proposed. Qi et al. (2017) regarded the difference between an associated track pair after coarse association as a systematic error (bias) and eliminated it by subtracting the difference between the coarsely associated tracks. Zhu and Chen (2014) derived radar bias explicitly by constructing a pseudo-measurement equation and estimated radar biases using recursive least squares (RLS). However, it is difficult to achieve accurate estimation of systematic error through track pairs after coarse association. Their estimation method relies on the modeling of the target motion and systematic error, so the association performance decreases in maneuvering scenarios. Because track association and spatial registration are presuppositions between each other, Wang et al. (2021) proposed a method that alternates between spatial registration and track association to improve the association performance. However, the combination of spatial registration and track association requires identifying a significant target to determine the systematic error. It is usually difficult to find such a target, which limits the application range of this method. Considering that the systematic error will cause the absolute position of the target to shift, but will not change the relative position between the targets, some T2TA methods based on reference topology features (REFs) have been proposed (Shi et al., 2006; Tian et al., 2014; Zhu and Han, 2014; Zhu et al., 2016; Qi et al., 2018; Sönmez and Hocaoğlu, 2022). This kind of method obtains association results by comparing the relative distances between tracks. These relative distances are calculated by traversing all track sampling points and setting them as the coordinate origin. However, too much time is consumed in traversing all track sampling points and calculating distances, especially when dealing with track sequences (Qi et al., 2018).

Some T2TA methods based on fuzzy mathematics have been proposed that focus on the random property of tracks and the fuzzy property of association decisions ignored in statistical reasoning methods. In these methods, the motion of the target is modeled by fuzzy factors, and the uncertainty is included. The association process is generally divided into the following steps: determining the composition of fuzzy element sets, selecting the membership functions, and assigning weight vectors. Finally, the degree of membership of different track pairs is determined, and the association results are obtained (Aziz, 2011). However, the selection of the membership function and weight vectors is highly subjective, which complicates the determination of the optimal values and limits the model's adaptability to various scenarios. Some improvements have been made to association performance. Du W et al. (2013) proposed a fuzzy double-threshold track association algorithm that uses an adaptive threshold and is insensitive to initial thresholds. Zhao et al. (2017) used a weight function and adopted dynamic weight sets to improve association performance.

The dependencies of the above methods on prior information (Aziz, 2011; Tokta and Hocaoglu, 2019), the hypothetical model (Kanyuck and Singer, 1970), and the threshold (Wu et al., 2021) seriously restricts their scope of application and association performance. Recently, some researchers have used data-driven and heuristic algorithms to obtain association results. This kind of algorithm uses machine learning or deep learning techniques to extract track features and learn association rules from a large number of tracks, so as to reduce the reliance on prior information, motion models, and thresholds. Cui et al. (2021) transformed

the track association problem into a classification problem and used deep convolutional neural networks to obtain association results. Xu and Fang (2021) transformed the track association problem into a binary classification problem and solved it based on AdaBoost and a decision tree. Xiong et al. (2021) processed tracks by graph neural networks (GNNs) (Wu et al., 2021), embedded tracks into a high-dimensional space, and found the nearest track pairs as the association results. Yang et al. (2022) used self-attention and cross-attention mechanisms to extract track features and constructed an association matrix by graph matching to obtain track association results between automatic identification system (AIS) and radar. Jin et al. (2023) integrated track and scene features through deep learning to associate radar and AIS tracks. Although track association methods based on machine learning or deep learning can solve the above problems, it is difficult to ensure the optimal association using the sub-optimal assignment algorithms widely used in these methods. Moreover, these methods assume that the update periods of different sensors are consistent or interpolate the tracks to ensure consistency. However, under conditions where sensor update periods are inconsistent, the direct use of these methods may lead to error accumulation.

In a multiple-hypothesis algorithm, all possible association hypotheses are generated, and impossible ones are removed by pruning to obtain the association results. Unlike a heuristic algorithm based on assignment algorithms, the process of listing all hypotheses and then pruning can obtain the optimal association results. Multiple-hypothesis tracking (MHT) is a multiple-hypothesis algorithm widely used in the field of target tracking. MHT is generally accepted as the preferred method for solving the data association problem in modern multiple-target tracking (MTT) systems (Blackman, 2004). In general, there are two categories of MHT. The first is called hypothesis-oriented MHT (HO-MHT), which maintains and updates hypotheses continually when observations are received (Reid, 1977, 1979; Chong et al., 1982; Blackman, 1986; Cox and Hingorani, 1996). The second is called track-oriented MHT (TO-MHT), which initiates, updates, and scores tracks before hypotheses are formed (Kurien, 1990; Werthmann, 1992; Blackman et al., 1993; Blostein and Richardson, 1994). TO-MHT is more efficient than the original HO-MHT (Kurien, 1990), and today most MHT applications are designed based on TO-MHT. For example, Du RZ et al. (2021) modeled measurement-to-track association as a multiple-hypothesis tree and solved the measurement-to-track association problem based on MHT. Lee et al. (2023) addressed the integrated tracking and identification problem of a maneuvering re-entry target with the help of MHT and obtained the optimal solution. The main steps of TO-MHT include measurement filtering and prediction, gate formation and measurement-to-track association, track initiation, track scoring and pruning, track clustering, hypothesis generation and scoring, and global track scoring and pruning (Werthmann, 1992). Among all the operations, branching and scoring are the most critical. The scoring operation is used to obtain scores of different hypotheses, and the branching operation is used to select the best hypothesis and prune bad hypotheses. However, existing scoring algorithms need to assume the motion model of the target, use the filtering algorithm to predict the state of the target, construct the scoring function according to the state and the covariance matrix of the target, and finally obtain the association score. Under the conditions of complex scenarios and unknown target states, it is difficult to obtain the optimal score by relying on the presumed target motion attribute and target motion model. Moreover, due to the influence of many factors such as target maneuvering, platform maneuvering, target quantity uncertainty, systematic error, and clutter interference, the scenario has significant uncertainty, resulting in a substantial reduction in scoring effectiveness. Thus, there is an urgent need to enhance the scoring algorithm within MHT to alleviate the dependencies on prior information and assumed motion models, and to diminish the impact of various uncertainties. More importantly, the aim of T2TA is also to solve the association problem. Can the multiple-hypothesis algorithm be applied to T2TA to achieve the optimal association results? To our knowledge, no researchers have yet applied a multiple-hypothesis algorithm to the T2TA problem.

Given the above analyses, we conclude that there are three difficult problems in T2TA currently. The first is that it is difficult to obtain the optimal association result using heuristic association algorithms. The second is the dependence on prior information and assumed motion models. The third is the impact of various uncertainties. A multiple-hypothesis algorithm

can solve the first problem and a heuristic algorithm can solve the second and third. Thus, in this paper, the multiple-hypothesis algorithm and deep learning (Xiong et al., 2021) are combined, and a multiple-hypothesis algorithm for T2TA with an intelligent track score (MH-T2TA) is proposed. The multiple-hypothesis algorithm is used to generate all hypothetical association relations and obtain the optimal association result. An attention-based contrastive network is used to estimate track scores under conditions of inconsistent sensor update periods and systematic errors. The intelligent track score network is data-driven and is trained with a large number of associated track pairs, making it highly robust to uncertainty without requiring prior information or assumed motion models. Compared with other T2TA methods, the proposed MH-T2TA method can solve all three problems.

Overall, the contributions of this paper are as follows:

1. We extend the widely used multiple-hypothesis algorithm in MHT to T2TA problems and combine the multiple-hypothesis algorithm with deep learning. This can alleviate the dependencies on prior information and assumed motion models, reduce the impact of various uncertainties, and obtain optimal association results.

2. We propose an attention-based network and a contrastive learning architecture to achieve the spatial–temporal registration of tracks from different sensors. A spatial–temporal mixing block is proposed to estimate track scores by sufficiently mixing of spatial–temporal features.

3. We show that layer normalization (Ba et al., 2016) is unsuitable for processing track data and will make different features of tracks gradually become indistinguishable.

4. Experimental results indicate that the proposed MH-T2TA method can achieve state-of-the-art T2TA performance.

2 Hypothesis generation and network construction

Fig. 1 shows a schematic of the proposed MH-T2TA. It consists of two parts: multiple-hypothesis generation and an intelligent track score network. We first describe the T2TA scenarios and then formulate these two parts.

This paper considers T2TA scenarios containing two sensors, A and B, in a common detection area, where the tracks reported by each sensor are unique and correspond to different targets. Thus, different tracks from the same sensor cannot correspond to the same target. Suppose that in the given scenario, the number of tracks reported by sensor S is N_S, where $S\in\{A, B\}$ and $N_A\neq N_B$ in general. The track set from sensor S is $\Phi_S=\left\{\boldsymbol{T}_1^S, \boldsymbol{T}_2^S, \cdots, \boldsymbol{T}_{N_S}^S\right\}$. However, different sensors usually have different update periods, which leads to different sampling numbers. Thus, over a period of time, tracks from different sensors have different numbers of track sampling points. One track from sensor S can be represented as follows:

$$\boldsymbol{T}_i^S=\begin{bmatrix} x_i^{1,S} & y_i^{1,S} & t_i^{1,S} \\ x_i^{2,S} & y_i^{2,S} & t_i^{2,S} \\ \vdots & \vdots & \vdots \\ x_i^{l_S,S} & y_i^{l_S,S} & t_i^{l_S,S} \end{bmatrix}, \tag{1}$$

where $x_i^{k,S}$ (k=1, 2, ⋯ , l_S) is the X coordinate of the k^{th} track sampling point of the i^{th} track from sensor S, $y_i^{k,S}$ is the Y coordinate, $t_i^{k,S}$ is the report time, and l_S is the number of track sampling points of one track from sensor S.

2.1 Multiple-hypothesis generation

The aim of multiple-hypothesis generation is to generate all possible track association matrices in one association cluster, with no association conflicts in each track association matrix. The process of multiple-hypothesis generation is as follows.

2.1.1 Gating and coarse association

We establish a set of coarse association rules for the coarse association of tracks. The motivation is that one track cannot be associated with another track that differs greatly in position, speed, heading, and time. It can be associated only with a track that is as similar as possible to itself. Thus, we define distance gate G_{D}, speed gate G_{S}, heading gate G_{H}, and time gate G_{T}. The distance gate G_{D} is the distance between the mean positions of $\boldsymbol{T}_i^A$ and $\boldsymbol{T}_j^B$. According to the maximum systematic error characteristic of the sensor, the distance gate is set to less than or equal to

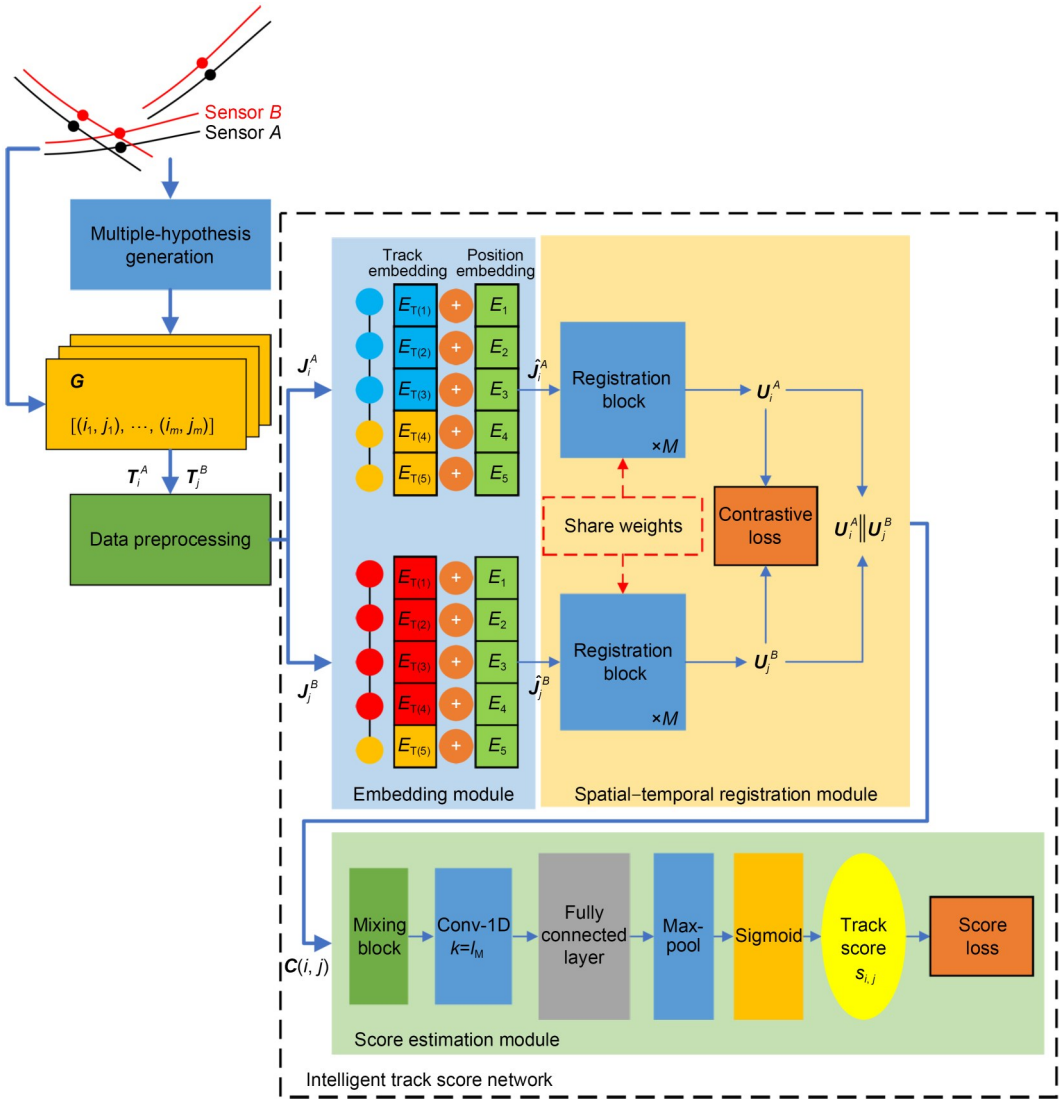

Fig. 1 A schematic of MH-T2TA, which consists of two parts: multiple-hypothesis generation and an intelligent track score network. The network consists of three modules: an embedding module, a spatial–temporal registration module, and a score estimation module. Two branches in the spatial–temporal registration module share network parameters and weights (References to color refer to the online version of this figure)

0.1. The speed gate G_S is the absolute value of the difference between the average velocities of $\boldsymbol{T}_i^A$ and $\boldsymbol{T}_j^B$, set to less than or equal to 0.1. The heading gate G_H is the absolute value of the difference between the average headings of $\boldsymbol{T}_i^A$ and $\boldsymbol{T}_j^B$, set to less than or equal to 60. The time gate G_T is the maximum of the starting times of $\boldsymbol{T}_i^A$ and $\boldsymbol{T}_j^B$ minus the minimum of their ending times. The time gate is set to less than or equal to 0, which means that the maximum of the starting times of $\boldsymbol{T}_i^A$ and $\boldsymbol{T}_j^B$ is less than or equal to their minimum ending times and thus they have a common time area. All association gates and the corresponding thresholds are shown in Eq. (2) at the top of the next page, where h_i^S is the heading of the vector from $\left(x_i^{1,S} \quad y_i^{1,S}\right)$ to $\left(x_i^{l_S,S} \quad y_i^{l_S,S}\right)$, $S \in \{A, B\}$. The heading h_i^S can be easily calculated by latitude and longitude.

The pseudo-code of the gating and coarse association is shown in Algorithm S1 in the supplementary materials. We traverse tracks $\boldsymbol{\Phi}_A$ from sensor A and $\boldsymbol{\Phi}_B$ from sensor B, and calculate the above gates. If all gates are satisfied between two tracks $\boldsymbol{T}_i^A$ and $\boldsymbol{T}_j^B$, the two tracks are coarsely associated, and we save their indexes i and j in the coarse association list $\boldsymbol{P}$.

$$\begin{cases} G_{\mathrm{D}}=\sqrt{\left(\dfrac{1}{l_A}\sum\limits_{k=1}^{l_A}x_i^{k,A}-\dfrac{1}{l_B}\sum\limits_{k=1}^{l_B}x_j^{k,B}\right)^2+\left(\dfrac{1}{l_A}\sum\limits_{k=1}^{l_A}y_i^{k,A}-\dfrac{1}{l_B}\sum\limits_{k=1}^{l_B}y_j^{k,B}\right)^2}\le 0.1, \\ G_{\mathrm{S}}=\left|\dfrac{\sqrt{\left(x_i^{l_A,A}-x_i^{1,A}\right)^2+\left(y_i^{l_A,A}-y_i^{1,A}\right)^2}}{t_i^{l_A,A}-t_i^{1,A}}-\dfrac{\sqrt{\left(x_j^{l_B,B}-x_j^{1,B}\right)^2+\left(y_j^{l_B,B}-y_j^{1,B}\right)^2}}{t_j^{l_B,B}-t_j^{1,B}}\right|\le 0.1, \\ G_{\mathrm{H}}=\left|h_i^A-h_j^B\right|\le 60, \\ G_{\mathrm{T}}=\max\left(t_i^{1,A},t_j^{1,B}\right)-\min\left(t_i^{l_A,A},t_j^{l_B,B}\right)\le 0. \end{cases} \tag{2}$$

2.1.2 Association cluster

The association hypothesis is generated according to the coarse association pairs. However, if we consider generating association hypotheses based on coarse association pairs of the entire scenario, it will result in a combination explosion and consume too much association time. For example, if there are n coarse association tracks in both sensors A and B, the time complexity of T2TA for the whole scenario is $O(n^2)$. In contrast, if the tracks in sensors A and B are clustered into k clusters, the time complexity of T2TA will decrease to $O(n^2/k)$, which is a reduction by a factor of k. Moreover, since there are multiple association hypotheses, conflicting association pairs are assigned to the same association cluster only if they conflict. This reduces the number of comparisons when generating the association hypothesis. Thus, in this step, we cluster the conflicting association pairs to reduce the amount of computation and obtain the association cluster $\boldsymbol{T}_{\mathrm{C}}$. A conflicting association pair is defined as two track association pairs that have the same track number in sensor A or B. For example, given two association pairs (i_1,j_1) and (i_2,j_2), if $i_1=i_2$ or $j_1=j_2$, they are a conflicting association pair. Moreover, the conflicting association pairs have transitivity. Let a track association pair be $\alpha_k=(i_k,j_k)$. If $\alpha_1\leftrightarrow\alpha_2$ and $\alpha_2\leftrightarrow\alpha_3$, we can obtain $\alpha_1\leftrightarrow\alpha_3$. According to the conflicting association pairs, the track pairs in the coarse association list are clustered into different groups, so that the association pairs in the same group are conflicting association pairs and those in different groups are not conflicting. The tracks in the same group are assigned to a cluster. The association cluster $\boldsymbol{T}_{\mathrm{C}}$ has a data structure of dictionary type and can be described as $\boldsymbol{T}_{\mathrm{C}}\triangleq\{\text{key:value}\}$, where "key" represents the cluster number and "value" represents the association pairs (i,j). Each key represents an association cluster and includes many association pairs. The pseudo-code of the association cluster is shown in Algorithm S2 in the supplementary materials. We traverse the coarse association list $\boldsymbol{P}$ and select the first association pair as the starting element of a new cluster. If the current association pair conflicts with any association pair in this cluster, the current association pair will be added to this cluster, and the current association pair will be deleted. The traversal is repeated until all elements in the coarse association list are deleted.

To explain the process of the association cluster more clearly, we take a case of three tracks from each sensor as an example. After gating and coarse association, the coarse association list is

$$\boldsymbol{P}=[(1,1),(1,2),(2,1),(2,2),(3,3)]. \tag{3}$$

If we do not use the association cluster, each association pair in $\boldsymbol{P}$ will be compared with other association pairs when generating the association hypothesis. However, if we assign the conflicting association pairs to the same association cluster, the number of traversal calculations will be greatly reduced. As described in Algorithm S2, $(1,1)$ is assigned to $\boldsymbol{T}_{\mathrm{C}}[1]$. Because the first index is the same for sensor A at (1, 1) and (1, 2), (1, 2) is assigned to $\boldsymbol{T}_{\mathrm{C}}[1]$. Because the second index is the same for sensor B at (1, 1) and (2, 1), (2, 1) is assigned to $\boldsymbol{T}_{\mathrm{C}}[1]$. Because the second index is the same for sensor B at (1, 2) and (2, 2), (2, 2) is assigned to $\boldsymbol{T}_{\mathrm{C}}[1]$. Finally, we have the cluster result of $\boldsymbol{T}_{\mathrm{C}}[1]$ and the corresponding elements in $\boldsymbol{P}$ are deleted, as shown in Eq. (4):

$$\boldsymbol{T}_{\mathrm{C}}[1]=\left[(1,1),(1,2),(2,1),(2,2)\right]. \tag{4}$$

Then, in the next loop, $\boldsymbol{P}=\left[(3,3)\right]$; $(3,3)$ is assigned to $\boldsymbol{T}_{\mathrm{C}}[2]$ and $\boldsymbol{P}$ is empty, which indicates that the cluster process is finished. Finally, two association clusters $\boldsymbol{T}_{\mathrm{C}}[1]$ and $\boldsymbol{T}_{\mathrm{C}}[2]$ are generated according to $\boldsymbol{P}$.

2.1.3 Hypothesis generation

After the association cluster, we generate all association hypotheses G from one association cluster $\boldsymbol{T}_{\mathrm{C}}[n]$. There is no conflict between each track in each association hypothesis. The association relations in one association cluster are transformed into an association matrix $\boldsymbol{C}$. Each element in $\boldsymbol{C}$ is a triplet of $\left(f_{i,j},i,j\right)$, as shown in Eq. (5).

$$\boldsymbol{C}=\begin{bmatrix}\left(f_{i,j},1,1\right) & \cdots & \left(f_{i,j},1,N_j\right)\\ \vdots & \vdots & \vdots\\ \left(f_{i,j},N_i,1\right) & \cdots & \left(f_{i,j},N_i,N_j\right)\end{bmatrix}, \tag{5}$$

where i is the index of the track from sensor A, j is the index of the track from sensor B, N_i is the number of all tracks from sensor A in $\boldsymbol{T}_{\mathrm{C}}[n]$, N_j is the number of all tracks from sensor B in $\boldsymbol{T}_{\mathrm{C}}[n]$, and $f_{i,j}$ is the indicator of association relation. If track i from sensor A is associated with track j from sensor B, according to the association relations, $f_{i,j}=1$; otherwise, $f_{i,j}=0$.

The association hypotheses are generated based on $\boldsymbol{C}$. The stack storage structure and backtracking algorithm are used to store temporary association hypotheses, ensuring that one track of one sensor can be associated with at most one track of another sensor, and that no association hypothesis is overlooked. In detail, we traverse the first row or first column of $\boldsymbol{C}$. The direction of traversal varies depending on the number of rows and columns. If the number of rows is greater than or equal to the number of columns, we recursively traverse the first column elements of each matrix. If the number of rows is smaller than the number of columns, we recursively traverse the first row elements of each matrix. If there is an associated track pair ($f_{i,j}=1$) in $\boldsymbol{C}$, the association pair (i,j) is saved and the corresponding row and column are deleted. When $\boldsymbol{C}$ can no longer be deleted, we stop processing $\boldsymbol{C}$, at which point all saved association pairs (i,j) form an association hypothesis $\boldsymbol{G}'=\left[(i_1,j_1),(i_2,j_2),\cdots,(i_m,j_m)\right]$, where m is the number of association pairs in one association hypothesis $\boldsymbol{G}'$. In one cluster, there may be more than one $\boldsymbol{G}'$ and the multiple-hypothesis generation will generate them all to form the whole association hypothesis $\boldsymbol{G}=\left[\boldsymbol{G}'(1),\boldsymbol{G}'(2),\cdots\right]$ and obtain the best association result. The pseudo-code of hypothesis generation is shown in Algorithm S3 in the supplementary materials and is presented with the traversal of the first row.

To explain the process of hypothesis generation more clearly, we take a case of three tracks from sensor A and two tracks from sensor B in one association cluster as an example. The association cluster is

$$\boldsymbol{T}_{\mathrm{C}}[n]=\left[(1,1),(1,2),(2,1),(2,2),(3,1)\right]. \tag{6}$$

Given $\boldsymbol{T}_{\mathrm{C}}[n]$, we can obtain the association matrix

$$\boldsymbol{C}=\begin{bmatrix}(1,1,1) & (1,1,2)\\ (1,2,1) & (1,2,2)\\ (1,3,1) & (0,3,2)\end{bmatrix}. \tag{7}$$

Because the number of rows is greater than the number of columns in $\boldsymbol{C}$, we recursively traverse the first column elements of each matrix. At the first traversal of $\boldsymbol{R}=\left[\boldsymbol{C},[\cdot]\right]$, we process the first element of the first column and obtain

$$\boldsymbol{R}=\left[\begin{bmatrix}(1,2,2)\\ (0,3,2)\end{bmatrix},\left[(1,1)\right]\right]. \tag{8}$$

Then, traverse the first column of $\begin{bmatrix}(1,2,2)\\ (0,3,2)\end{bmatrix}$ and obtain one association hypothesis $\boldsymbol{G}'(1)=\left[(1,1),(2,2)\right]$. At the second traversal of $\boldsymbol{R}=\left[\boldsymbol{C},[\cdot]\right]$, we process the second element of the first column and obtain

$$\boldsymbol{R}=\left[\begin{bmatrix}(1,1,2)\\ (0,3,2)\end{bmatrix},\left[(2,1)\right]\right]. \tag{9}$$

Then, traverse the first column of $\begin{bmatrix}(1,1,2)\\ (0,3,2)\end{bmatrix}$ and obtain one association hypothesis $\boldsymbol{G}'(2)=\left[(2,1),(1,2)\right]$. Thus, we can obtain the whole association hypothesis

$$\boldsymbol{G}=\begin{bmatrix}[(1,1),(2,2)]\\ [(2,1),(1,2)]\\ [(3,1),(1,2)]\\ [(3,1),(2,2)]\end{bmatrix}. \tag{10}$$

The example process of hypothesis generation is shown in Fig. 2.

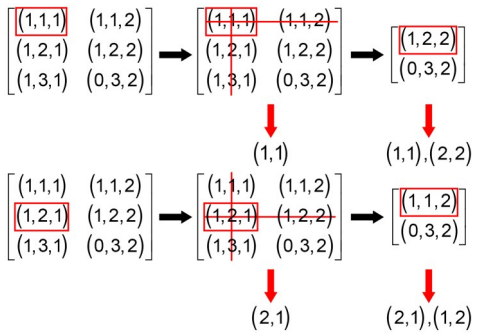

Fig. 2 An example of the hypothesis generation process

2.2 Data preprocessing

In the intelligent track score network, the time feature of the track is removed and only X and Y coordinate features are contained. It is difficult to train a neural network with unnormalized data, because neural networks are sensitive to data distribution and the distributions of these coordinate features are different. To solve this problem, we use 0 and 1 normalization to make the distribution of all features range between 0 and 1. Each feature of a track is subtracted from the minimum value of each feature in one scenario, and then the result is divided by the maximum value of each feature in this scenario minus the minimum value of each feature in this scenario. The 0 and 1 normalization is shown as follows:

$$\tilde{\boldsymbol{T}}_i^S=\left(\boldsymbol{T}_i^S-\boldsymbol{T}_{\min}\right)\odot\left(\boldsymbol{T}_{\max}-\boldsymbol{T}_{\min}\right)^{-1}, \tag{11}$$

where $\boldsymbol{T}_i^S=\begin{bmatrix}x_i^{1,S} & y_i^{1,S}\\ x_i^{2,S} & y_i^{2,S}\\ \vdots & \vdots\\ x_i^{l_S,S} & y_i^{l_S,S}\end{bmatrix}$, $\tilde{\boldsymbol{T}}_i^S=\begin{bmatrix}\tilde{x}_i^{1,S} & \tilde{y}_i^{1,S}\\ \tilde{x}_i^{2,S} & \tilde{y}_i^{2,S}\\ \vdots & \vdots\\ \tilde{x}_i^{l_S,S} & \tilde{y}_i^{l_S,S}\end{bmatrix}$ is the normalized track, $\odot$ represents element multiplication, $\boldsymbol{T}_{\max}=\begin{bmatrix}\max\limits_{i\in[1,N_S]} x_i^{k,S} & \max\limits_{i\in[1,N_S]} y_i^{k,S}\\ k\in[1,l_S] & k\in[1,l_S]\\ S\in\{A,B\} & S\in\{A,B\}\end{bmatrix}$, $\boldsymbol{T}_{\min}=\begin{bmatrix}\min\limits_{i\in[1,N_S]} x_i^{k,S} & \min\limits_{i\in[1,N_S]} y_i^{k,S}\\ k\in[1,l_S] & k\in[1,l_S]\\ S\in\{A,B\} & S\in\{A,B\}\end{bmatrix}$, i is the index of the track, k is the index of the track sampling point, and N_S is the number of all tracks from sensor S.

Then, we consider the trending position of the track. The trending position represents the overall movement trend of a track instead of a single track point. To obtain the trending position, only the starting track point $\left(\tilde{x}_i^{1,S}\quad \tilde{y}_i^{1,S}\right)$ and the ending track point $\left(\tilde{x}_i^{l_S,S}\quad \tilde{y}_i^{l_S,S}\right)$ are taken into account. The trending points are uniformly sampled from the line of the starting and ending track sampling points with the number of l_S. The position interval of coordinate X is

$$\Delta_X=\frac{\tilde{x}_i^{l_S,S}-\tilde{x}_i^{1,S}}{l_S-1}, \tag{12}$$

and the position interval of coordinate Y is

$$\Delta_Y=\frac{\tilde{y}_i^{l_S,S}-\tilde{y}_i^{1,S}}{l_S-1}. \tag{13}$$

The X coordinate trending position of the k^{th} track point of the i^{th} track from sensor S is

$$\bar{\tilde{x}}_i^{k,S}=\tilde{x}_i^{1,S}+\Delta_X(k-1), \tag{14}$$

and the Y coordinate trending position of the k^{th} track point of the i^{th} track from sensor S is

$$\bar{\tilde{y}}_i^{k,S}=\tilde{y}_i^{1,S}+\Delta_Y(k-1). \tag{15}$$

After combining the X and Y coordinate trending positions, the trending position $\bar{\tilde{\boldsymbol{T}}}_i^S$ is obtained:

$$\bar{\tilde{\boldsymbol{T}}}_i^S=\begin{bmatrix}\bar{\tilde{x}}_i^{1,S} & \bar{\tilde{y}}_i^{1,S}\\ \bar{\tilde{x}}_i^{2,S} & \bar{\tilde{y}}_i^{2,S}\\ \vdots & \vdots\\ \bar{\tilde{x}}_i^{l_S,S} & \bar{\tilde{y}}_i^{l_S,S}\end{bmatrix}. \tag{16}$$

In terms of different update periods of different sensors, according to the association period T and the minimum update period between sensors A and B, we can obtain the maximum number of sampling points l_{M}:

$$l_{\text{M}}=\frac{T}{\min\left(T_{SA},T_{SB}\right)}. \tag{17}$$

Track association is conducted in batch processing and the input track $\boldsymbol{T}_i^S$ is the track during an association period T. The track length is the maximum number of sampling points l_{M}. For each $\tilde{\boldsymbol{T}}_i^S$ and $\tilde{\tilde{\boldsymbol{T}}}_i^S$, if the number of track sampling points is smaller than l_{M}, we pad it with zero to unify its number of track sampling points to l_{M}. Thus, MH-T2TA can associate tracks with different sensor update periods and is not limited by different sensor update periods. Moreover, there is no need to use other methods to register tracks in the time domain. After normalization and trending position construction, $\tilde{\boldsymbol{T}}_i^S$ and its trending position $\tilde{\tilde{\boldsymbol{T}}}_i^S$ are concatenated along the coordinate dimension to construct the joint track $\boldsymbol{J}_i^S$.

$$\boldsymbol{J}_i^S=\left[\tilde{\boldsymbol{T}}_i^S \middle\| \tilde{\tilde{\boldsymbol{T}}}_i^S\right], \tag{18}$$

where || represents concatenation along the coordinate dimension. Next, a joint track pair, $\boldsymbol{J}_i^A$ and $\boldsymbol{J}_j^B$, is fed into the intelligent track score network to obtain the association score. To calculate the score of the combination of a track pair precisely under the conditions of errors and inconsistent update periods, the intelligent track score network needs to have the ability to register the spatial–temporal features of tracks and convert track features into scores. Furthermore, owing to the sequence property of tracks, the network needs to be able to discriminate the order of track sampling points in a track. Corresponding to these three required capabilities, the network consists of three modules: an embedding module, a spatial–temporal registration module, and a score estimation module. In the embedding module, the joint track is embedded into track embedding, and position embedding is added to discriminate the order of track sampling points. Next, the sum of track embedding and position embedding is fed into the spatial–temporal registration module. Under the constraint of contrastive loss, embeddings of the same target from different sensors are unified in space and time, reducing the impact of errors and inconsistent update periods. Finally, the unified track embedding is fed into the score estimation module, and the spatial–temporal features are mixed and enhanced by the mixing block. With rich features, the track score can be obtained precisely. The hypothesis with the highest score is selected as the optimal association hypothesis.

2.3 Embedding modules

Owing to the inconsistent update periods of different sensors, different tracks have different numbers of track sampling points. Although zero padding is used to unify the number of track sampling points, self-attention cannot discriminate the order of track sampling points, which will lead to the destruction of the sequence property of tracks. To address this problem, two kinds of embeddings (track embedding and position embedding) are used to embed tracks and positions into a unified dimension and incorporate position information.

1. Track embedding

Given an input $\boldsymbol{J}_i^S$, a fully connected (FC) layer with four input dimensions and D output dimensions is used as the track embedding layer. The four input dimensions represent the features of X, Y, trending X, and trending Y coordinates. D is a hyperparameter, which is discussed in Section 2.1 of the supplementary materials. The weights of track embeddings are learned and changed during training.

2. Position embedding

Position embedding is used to encode the length dimension of $\boldsymbol{J}_i^S$ and determine the order of the track sampling points of $\boldsymbol{J}_i^S$. At present, there are three widely used position embedding strategies: absolute position embedding (Vaswani et al., 2017), relative position embedding (Su et al., 2021), and learnable position embedding. For absolute position embedding, the position is represented by sine and cosine functions. The absolute position embedding for the u^{th} row and v^{th} column element in $\boldsymbol{J}_i^S$ (the u^{th} track sampling point and the v^{th} track feature) is

$$p_{u,v}=\begin{cases}\sin\left(\dfrac{u}{10\,000^{\frac{v}{D}}}\right), & v \text{ is even},\\ \cos\left(\dfrac{u}{10\,000^{\frac{v-1}{D}}}\right), & v \text{ is odd},\end{cases} \tag{19}$$

where $v\in\{0, 1, \cdots, D-1\}$ and D is the dimension of $\boldsymbol{J}_i^S$. For relative position embedding, we use rotary position embedding and the position is represented by the rotation matrix $\boldsymbol{R}_{\boldsymbol{\Theta},u}^D$, where $\boldsymbol{\Theta}$ is the set of attenuation angles and u is the index of the track sampling point in $\boldsymbol{J}_i^S$. Rotary position embedding needs to consider the multiplication between the u^{th} track sampling

point of $\boldsymbol{J}_i^S$ ($\boldsymbol{J}_i^{u,S}$) and the rotation matrix $\boldsymbol{R}_{\boldsymbol{\Theta},u}^D$. The rotary position embedding of $\boldsymbol{J}_i^{u,S}$ is

$$f\left(\boldsymbol{J}_i^{u,S}, \boldsymbol{R}_{\boldsymbol{\Theta},u}^D\right) = \boldsymbol{J}_i^{u,S}\boldsymbol{R}_{\boldsymbol{\Theta},u}^D, \tag{20}$$

where $\boldsymbol{R}_{\boldsymbol{\Theta},u}^D$ is given in Eq. (21) at the bottom of this page, and

$$\boldsymbol{\Theta}=\left\{\theta_v\middle|\theta_v=\begin{cases}10\,000^{-\frac{v}{D}}, v \text{ is even},\\ 10\,000^{-\frac{v-1}{D}}, v \text{ is odd},\end{cases} v\in\{0,1,\cdots,D-1\}\right\}. \tag{22}$$

For learnable position embedding, the position is represented by the parameters of one neural network layer. A learnable network parameter matrix with length l_{M} and dimension D is used as the position embedding. The weights of position embeddings are learned and changed during training. In Section 2.5 of the supplementary materials, we will compare the performances of these position embeddings.

For absolute position embedding and learnable position embedding, after the embedding module, track embedding and position embedding are summed up as the input $\hat{\boldsymbol{J}}_i^S$ of the spatial–temporal registration module. In particular, for relative position embedding, position embedding is added after track mapping as given in Section 2.4, and the input $\hat{\boldsymbol{J}}_i^S$ of the spatial–temporal registration module is just track embedding. The output of the embedding module is $\hat{\boldsymbol{J}}_i^S$ and will be the input of the spatial–temporal registration module. The shape of $\hat{\boldsymbol{J}}_i^S$ is $\left[l_{\mathrm{M}}, D\right]$.

2.4 Spatial–temporal registration module

The aim of the spatial–temporal registration module is to eliminate the effects of errors and inconsistent update periods and obtain the unified registration track $\boldsymbol{U}_i^S$. Motivated by bidirectional encoder representations from transformers (BERT) (Devlin et al., 2018) and considering the sequence property of tracks, we simplify the original complicated BERT and construct the spatial–temporal registration module based on self-attention. A schematic of the spatial–temporal registration module is shown in Fig. 3. The inputs of the spatial–temporal registration module are two tracks $\hat{\boldsymbol{J}}_i^A$ and $\hat{\boldsymbol{J}}_j^B$, which are the outputs of the embedding module. They are from different sensors. The critical operations of this module are contrastive learning and self-attention mechanism.

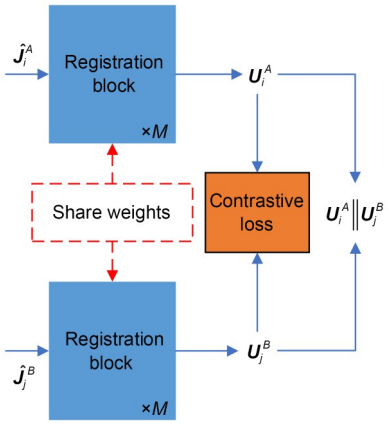

Fig. 3 Schematic of the spatial–temporal registration module

Contrastive learning is embodied in splitting two branches to deal with different tracks. The two branches have identical architecture and share weights and parameters. The aim of contrastive learning is to make full use of the associated and unassociated track pairs. With the help of contrastive loss, the features of two unified registration tracks $\boldsymbol{U}_i^A$ and $\boldsymbol{U}_j^B$

$$\boldsymbol{R}_{\boldsymbol{\Theta},u}^D=\begin{pmatrix}\cos(u\theta_0) & \sin(u\theta_0) & 0 & 0 & \cdots & 0 & 0\\ -\sin(u\theta_1) & \cos(u\theta_1) & 0 & 0 & \cdots & 0 & 0\\ 0 & 0 & \cos(u\theta_2) & \sin(u\theta_2) & \cdots & 0 & 0\\ 0 & 0 & -\sin(u\theta_3) & \cos(u\theta_3) & \cdots & 0 & 0\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & 0 & 0 & \cdots & \cos(u\theta_{D-2}) & \sin(u\theta_{D-2})\\ 0 & 0 & 0 & 0 & \cdots & -\sin(u\theta_{D-1}) & \cos(u\theta_{D-1})\end{pmatrix}. \tag{21}$$

from the same target remain unified, and the features of two unified registration tracks $\boldsymbol{U}_i^A$ and $\boldsymbol{U}_j^B$ from different targets have obvious separability. This means that the distance between $\boldsymbol{U}_i^A$ and $\boldsymbol{U}_j^B$ from the same target will be as small as possible, and the distance between $\boldsymbol{U}_i^A$ and $\boldsymbol{U}_j^B$ from different targets will be as large as possible. The contrastive loss is described in detail in Section 2.7.

The self-attention mechanism is embodied in the global fusion of all track sampling points of one track and the input is the track itself. Each branch of the spatial–temporal registration module is composed of M registration blocks and the architecture of each block is identical. A schematic of the registration block in the spatial–temporal registration module is shown in Fig. 4. The input of the first registration block is $\hat{\boldsymbol{J}}_i^S$ from sensor A or B. The query matrix $\boldsymbol{Q}_i^S$, key matrix $\boldsymbol{K}_i^S$, and value matrix $\boldsymbol{V}_i^S$ corresponding to $\hat{\boldsymbol{J}}_i^S$ are calculated first by the track mapping, and their feature dimensions are equal to that of $\hat{\boldsymbol{J}}_i^S$. For example, if the shape of $\hat{\boldsymbol{J}}_i^S$ is $\left[l_{\mathrm{M}}, D\right]$, the shapes of $\boldsymbol{Q}_i^S$, $\boldsymbol{K}_i^S$, and $\boldsymbol{V}_i^S$ are all $\left[l_{\mathrm{M}}, D\right]$.

$$\boldsymbol{Q}_i^S=\hat{\boldsymbol{J}}_i^S\boldsymbol{W}_{\mathrm{q}}, \tag{23}$$

$$\boldsymbol{K}_i^S=\hat{\boldsymbol{J}}_i^S\boldsymbol{W}_{\mathrm{k}}, \tag{24}$$

$$\boldsymbol{V}_i^S=\hat{\boldsymbol{J}}_i^S\boldsymbol{W}_{\mathrm{v}}, \tag{25}$$

where $\boldsymbol{W}_{\mathrm{q}}$, $\boldsymbol{W}_{\mathrm{k}}$, and $\boldsymbol{W}_{\mathrm{v}}$ are the learnable query mapping matrix, key mapping matrix, and value mapping matrix, respectively. Their shapes are all $\left[D, D\right]$. To fuse all track sampling points of one track globally and obtain the spatial–temporal features of different track sampling points, we use self-attention to calculate the weighted sum of $\boldsymbol{Q}_i^S$, $\boldsymbol{K}_i^S$, and $\boldsymbol{V}_i^S$. In particular, for the relative position embedding, $\boldsymbol{Q}_i^S$ and $\boldsymbol{K}_i^S$ should be added with the relative position embedding, as shown in Eqs. (26) and (27):

$$f\left(\boldsymbol{Q}_i^{u,S}, \boldsymbol{R}_{\boldsymbol{\Theta},u}^D\right)=\boldsymbol{Q}_i^{u,S}\boldsymbol{R}_{\boldsymbol{\Theta},u}^D, \tag{26}$$

$$f\left(\boldsymbol{K}_i^{u,S}, \boldsymbol{R}_{\boldsymbol{\Theta},u}^D\right)=\boldsymbol{K}_i^{u,S}\boldsymbol{R}_{\boldsymbol{\Theta},u}^D. \tag{27}$$

Then, we calculate the attention embedding $\boldsymbol{H}_i^S$ as the output of the self-attention mechanism. The process is shown in Eq. (28):

$$\boldsymbol{H}_i^S=\mathrm{SoftMax}\left(\mathrm{Mask}\left(\frac{\boldsymbol{Q}_i^S\left(\boldsymbol{K}_i^S\right)^{\mathrm{T}}}{\sqrt{D}}, \boldsymbol{M}_i^S\right)\right)\boldsymbol{V}_i^S+\hat{\boldsymbol{J}}_i^S, \tag{28}$$

where $\boldsymbol{M}_i^S$ is the mask matrix. The global fusion is embodied in the weighted calculation of all track sampling points along the time dimension. As described above, the shapes of $\boldsymbol{Q}_i^S$, $\boldsymbol{K}_i^S$, and $\boldsymbol{V}_i^S$ are all $\left[l_{\mathrm{M}}, D\right]$, and the shape of matrix multiplication $\boldsymbol{Q}_i^S\left(\boldsymbol{K}_i^S\right)^{\mathrm{T}}$ is $\left[l_{\mathrm{M}}, l_{\mathrm{M}}\right]$. The element of the p^{th} row and q^{th} column in $\boldsymbol{Q}_i^S\left(\boldsymbol{K}_i^S\right)^{\mathrm{T}}$ represents the weight between the p^{th} and q^{th} track sampling points. The attention weight of each track sampling point is included in $\boldsymbol{Q}_i^S\left(\boldsymbol{K}_i^S\right)^{\mathrm{T}}$. The denominator $\sqrt{D}$ is used to reduce the variance and enhance the stability of the calculation. Then, SoftMax is used to normalize the weights of $\boldsymbol{Q}_i^S\left(\boldsymbol{K}_i^S\right)^{\mathrm{T}}$ to between 0 and 1. Thus, the weight will have probabilistic properties. SoftMax is conducted on each row of $\boldsymbol{Q}_i^S\left(\boldsymbol{K}_i^S\right)^{\mathrm{T}}$. Assuming that $\boldsymbol{Z}$ represents one row of $\boldsymbol{Q}_i^S\left(\boldsymbol{K}_i^S\right)^{\mathrm{T}}$ and

$$\boldsymbol{Z}=\left(z_1, z_2, \cdots z_{l_{\mathrm{M}}}\right), \tag{29}$$

the SoftMax result of z_k (k=1, 2, ⋯, l_{M}) is calculated from Eq. (30):

$$\mathrm{SoftMax}\left(z_k\right)=\exp\left(z_k\right)\Big/\sum_{l=1}^{l_{\mathrm{M}}}\exp\left(z_l\right). \tag{30}$$

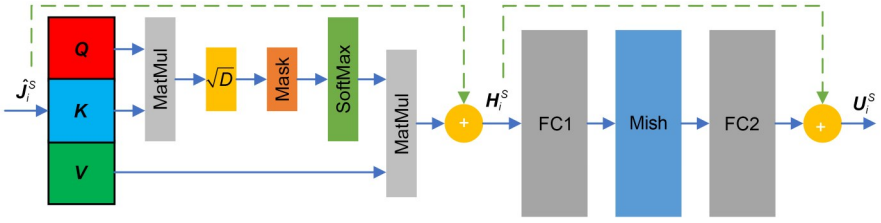

Fig. 4 Schematic of the registration block in the spatial–temporal registration module

After SoftMax, the result is multiplied by $\boldsymbol{V}_i^S$ and the shape $\left[l_{\mathrm{M}}, l_{\mathrm{M}}\right]$ is changed into $\left[l_{\mathrm{M}}, D\right]$, which is similar to the shape of $\hat{\boldsymbol{J}}_i^S$. During this process, the global weights of all track sampling points are added to the value matrix $\boldsymbol{V}_i^S$ and the result is the weighted sum of all track sampling points of $\boldsymbol{V}_i^S$. Therefore, we will pay more attention to the track sampling points that are important to T2TA, while ignoring those that are irrelevant.

In addition, Mask(·) is used to ensure that self-attention calculates only the track within the length of l_S and ignores the zero-padding parts. During zero padding, when we construct $\boldsymbol{J}_i^S$, we obtain the mask matrix $\boldsymbol{M}_i^S \in \mathbb{R}^{l_{\mathrm{M}} \times l_{\mathrm{M}}}$ simultaneously. Each row in the mask matrix $\boldsymbol{M}_i^S$ whose length exceeds l_S is 0, and the remaining elements are 1:

$$\boldsymbol{M}_i^S=\begin{bmatrix}1 & 1 & 0 & 0\\ 1 & 1 & 0 & 0\\ 1 & 1 & 0 & 0\\ 1 & 1 & 0 & 0\end{bmatrix}\in\mathbb{R}^{l_{\mathrm{M}}\times l_{\mathrm{M}}}. \tag{31}$$

After calculating $\boldsymbol{Q}_i^S\left(\boldsymbol{K}_i^S\right)^{\mathrm{T}}/\sqrt{D}$, we set the elements of $\boldsymbol{Q}_i^S\left(\boldsymbol{K}_i^S\right)^{\mathrm{T}}/\sqrt{D}$ corresponding to the 0 parts in $\boldsymbol{M}_i^S$ to $-\infty$. Then, the SoftMax function will make these $-\infty$ elements become 0, so that the multiplication with $\boldsymbol{V}_i^S$ will ignore the zero-padding elements. This can prevent the effect caused by zero paddings. Finally, the weighted result is added to the input $\hat{\boldsymbol{J}}_i^S$ to form the attention embedding $\boldsymbol{H}_i^S$ as the output of the self-attention mechanism.

After obtaining the attention embedding $\boldsymbol{H}_i^S$, two FCs are used to obtain the unified registration track $\boldsymbol{U}_i^S$ of each branch. The first FC enlarges the feature dimension by a factor of two, and the second reduces the feature dimension to the original dimension. The Mish (Misra, 2019) non-linear activation function is added between the two FCs:

$$\boldsymbol{U}_i^S=\sigma\left(\boldsymbol{H}_i^S\boldsymbol{W}_1+\boldsymbol{b}_1\right)\boldsymbol{W}_2+\boldsymbol{b}_2+\boldsymbol{H}_i^S, \tag{32}$$

where $\boldsymbol{W}_1$ and $\boldsymbol{b}_1$ are the learnable parameters of FC1, $\boldsymbol{W}_2$ and $\boldsymbol{b}_2$ are the learnable parameters of FC2, and σ is the Mish non-linear activation function. The shape of $\boldsymbol{W}_1$ is $\left[D, 2D\right]$ and that of $\boldsymbol{W}_2$ is $\left[2D, D\right]$. That is, the input dimension of FC1 is D and the output dimension is $2D$, while the input dimension of FC2 is $2D$ and the output dimension is D. Finally, we concatenate $\boldsymbol{U}_i^A$ and $\boldsymbol{U}_j^B$ from different sensors along the coordinate dimensions and feed it into the score estimation module:

$$\boldsymbol{C}\left(i,j\right)=\left[\boldsymbol{U}_i^A \middle\| \boldsymbol{U}_j^B\right], \tag{33}$$

where $\boldsymbol{C}\left(i,j\right)$, the concatenated track, is the input of the score estimation module. The shape of $\boldsymbol{C}\left(i,j\right)$ is $\left[l_{\mathrm{M}}, 2D\right]$.

2.5 Score estimation module

After the spatial–temporal registration module, the rich and unified features $\boldsymbol{U}_i^A$ and $\boldsymbol{U}_j^B$ are concatenated and $\boldsymbol{C}\left(i,j\right)$ is fed into the score estimation module. A schematic of the score estimation module is shown in Fig. 5. Because the shape of $\boldsymbol{C}\left(i,j\right)$ is $\left[l_{\mathrm{M}}, 2D\right]$ and $\boldsymbol{C}\left(i,j\right)$ still retains the temporal structure, the mixing block is designed to mix and enhance the spatial–temporal features (Xiong et al., 2024). The mixing block includes two main branches: long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) extracts temporal features and Conv-1D extracts spatial features. In addition, there are two mixing branches, temporal mixing and spatial mixing, which calculate the temporal and spatial attention factors to enhance the temporal and spatial features, respectively. After the mixing block, a Conv-1D with kernel size l_{M} is used to compress the time dimension to 1 and the spatial dimension is the same. Then, an FC is used to compress the spatial dimension to half its original size. The input dimension of this FC is $2D$ and the output dimension is D. Next, the max-pool layer selects the maximum value in the spatial dimension and reduces the spatial dimension to 1. The max-pool layer with more than one dimension gives the network strong robustness to various scenarios. Finally, sigmoid is added after the max-pool layer to ensure that the output value range is $\left[0, 1\right]$ and has the probabilistic property. The output of the score estimation module $s_{i,j}$ is the association score of $\boldsymbol{T}_i^A$ and $\boldsymbol{T}_j^B$. If track i from sensor A is associated with track j from sensor B, $s_{i,j} \to 1$; otherwise, $s_{i,j} \to 0$.

After each association track pair is scored by the intelligent track score network, the score of a given

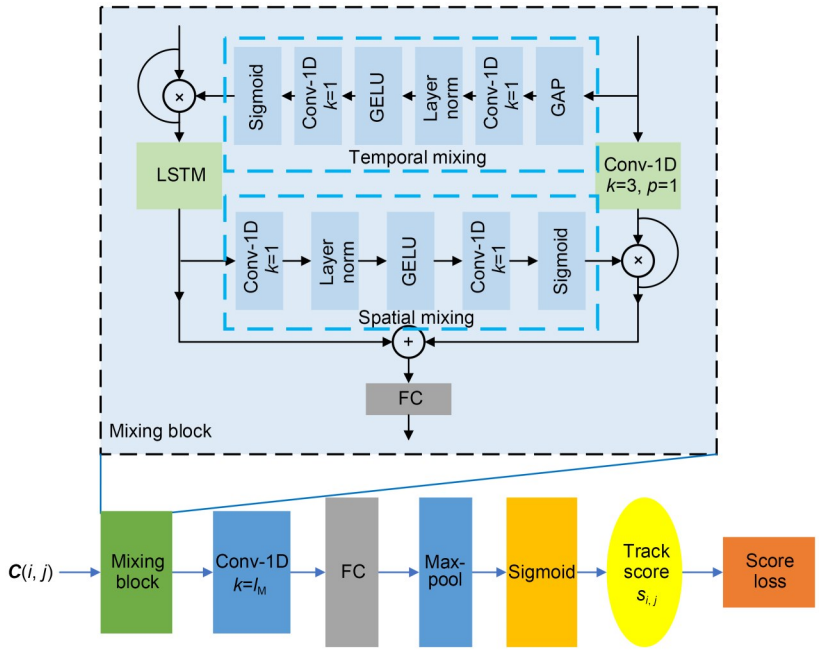

Fig. 5 Schematic of the score estimation module

hypothesis is determined by the mean score of all association track pairs in this hypothesis. The hypothesis with the highest score is selected as the association result of one cluster. Assuming that the number of hypotheses in one cluster is N_h and that the number of association track pairs in one hypothesis is n_p, where $p=\{1, 2, \cdots, N_\mathrm{h}\}$, according to the score $s_{i,j}$, we can obtain the score of hypothesis G_p:

$$S_p=\frac{1}{n_p}\sum_{(i,j)\in G_p} s_{i,j}, \tag{34}$$

where S_p is the score of hypothesis G_p. The hypothesis with the highest score is selected as the best hypothesis, G_b.

$$G_\mathrm{b}=G_p\Big| p=\arg\max_p\left(S_p\right), p=1, 2, \cdots, N_\mathrm{h}. \tag{35}$$

2.6 Inappropriate layer normalization

In the original block of BERT, layer normalization (LN) (Ba et al., 2016) is used to increase the speed of convergence and enhance the stability of the network, and LN is added after each addition operation in the registration block. LN normalizes all features in one timestep, which gradually makes each feature indistinguishable, especially when the distributions of different features are highly distinct. In natural language processing, the semantic information expressed by all features is of great importance, while the differentiation between features matters little. However, in the processing of track data, the differentiation between features is of vital importance for discriminating different tracks. For example, when LN is applied to $\boldsymbol{J}_i^S$ at timestep k, the processing can be expressed as follows:

$$\mathrm{LN}\left(\boldsymbol{J}_i^S(k)\right)=\frac{\left\{\tilde{x}_i^{k,S}, \tilde{y}_i^{k,S}, \bar{\tilde{x}}_i^{k,S}, \bar{\tilde{y}}_i^{k,S}\right\}-E\left(\boldsymbol{J}_i^S(k)\right)}{\sqrt{\triangle\left(\boldsymbol{J}_i^S(k)\right)+\varepsilon}}\cdot\gamma+\beta, \tag{36}$$

$$E\left(\boldsymbol{J}_i^S(k)\right)=E\left(\tilde{x}_i^{k,S}, \tilde{y}_i^{k,S}, \bar{\tilde{x}}_i^{k,S}, \bar{\tilde{y}}_i^{k,S}\right)=\frac{1}{4}\left(\tilde{x}_i^{k,S}+\tilde{y}_i^{k,S}+\bar{\tilde{x}}_i^{k,S}+\bar{\tilde{y}}_i^{k,S}\right), \tag{37}$$

$$V\left(\boldsymbol{J}_i^S(k)\right)=V\left(\tilde{x}_i^{k,S}, \tilde{y}_i^{k,S}, \bar{\tilde{x}}_i^{k,S}, \bar{\tilde{y}}_i^{k,S}\right)=\frac{1}{4}\left(\left(\tilde{x}_i^{k,S}-E\left(\boldsymbol{J}_i^S(k)\right)\right)^2+\left(\tilde{y}_i^{k,S}-E\left(\boldsymbol{J}_i^S(k)\right)\right)^2+\left(\bar{\tilde{x}}_i^{k,S}-E\left(\boldsymbol{J}_i^S(k)\right)\right)^2+\left(\bar{\tilde{y}}_i^{k,S}-E\left(\boldsymbol{J}_i^S(k)\right)\right)^2\right), \tag{38}$$

where $\mathrm{LN}(\cdot)$ is the layer normalization, $E\left(\boldsymbol{J}_i^S(k)\right)$ is the mean of $\boldsymbol{J}_i^S$, $V\left(\boldsymbol{J}_i^S(k)\right)$ is the variance of $\boldsymbol{J}_i^S$, γ and

β are learnable parameters, and $\varepsilon=10^{-9}$ is a small value that prevents the denominator from being zero. The critical parts that make LN unsuitable for track data are described in Eqs. (37) and (38). The mean $E\left(\boldsymbol{J}_i^S(k)\right)$ and variance $V\left(\boldsymbol{J}_i^S(k)\right)$ are calculated among all features in one timestep, and this operation will make these features over-smoothed and indistinguishable.

According to the analysis, to maintain the differentiation between the features of tracks, LN should be dropped. The validation of LN and dropping normalization will be conducted later.

2.7 Loss function

To ensure the spatial–temporal registration of the track and the accurate estimation of the track score, the network needs to satisfy three objectives:

1. For different sensors, after the spatial–temporal registration module, the tracks from the same target are close to each other, and the tracks from different targets move away from each other. However, this may make the tracks from different targets in the same sensor indistinguishable.

2. For the same sensor, after the spatial–temporal registration module, the tracks from different targets move away from each other.

3. The track score estimated by the intelligent track score network is as similar as possible to the real track score.

The spatial–temporal registration module realizes the first and second objectives and the score estimation module realizes the third one. Corresponding to the three objectives, there are three kinds of loss functions: the different sensor contrastive loss L_{dc}, the same sensor contrastive loss L_{sc}, and the score loss L_{s}.

The different sensor contrastive loss L_{dc} is used for tracks from different sensors. After the spatial–temporal registration module, it is used to make the tracks from the same target close to each other, and the tracks from different targets move away from each other. It is a process of learning by contrast, so the contrastive loss (Hadsell et al., 2006) is used as L_{dc}:

$$L_{\mathrm{dc}}=\frac{1}{2}s_{i,j}D_{\mathrm{d}}^2+\frac{1}{2}\left(1-s_{i,j}\right)\left[\max\left(0,m-D_{\mathrm{d}}\right)\right]^2, \quad (39)$$

$$D_{\mathrm{d}}=\left\|\boldsymbol{U}_i^A-\boldsymbol{U}_j^B\right\|_{\mathrm{F}}, \quad (40)$$

where $s_{i,j}$ is the label track score. As described before, if track i from sensor A is associated with track j from sensor B, $s_{i,j}=1$; otherwise, $s_{i,j}=0$. D_{d} is the Euclidean distance between $\boldsymbol{U}_i^A$ and $\boldsymbol{U}_j^B$. $\|\cdot\|_{\mathrm{F}}$ is the Frobenius norm. Parameter m is the margin distance and determines the shortest distance between $\boldsymbol{U}_i^A$ and $\boldsymbol{U}_j^B$ if they come from different targets.

The same sensor contrastive loss L_{sc} is used for tracks from the same sensor. After the spatial–temporal registration module, it is used to make the tracks from different targets move away from each other. Because L_{sc} needs to consider only the distance between $\boldsymbol{U}_i^S$ and $\boldsymbol{U}_j^S$ which come from the same sensor and represent different targets, only the negative part of the contrastive loss is adopted in L_{sc}:

$$L_{\mathrm{sc}}=\left[\max\left(0,m-D_{\mathrm{s}}\right)\right]^2, \quad (41)$$

$$D_{\mathrm{s}}=\left\|\boldsymbol{U}_i^S-\boldsymbol{U}_j^S\right\|_{\mathrm{F}}, \quad (42)$$

where D_{s} is the Euclidean distance between $\boldsymbol{U}_i^S$ and $\boldsymbol{U}_j^S$ when $i\neq j$.

The score loss L_{s} is used to make the track score estimated by the intelligent track score network as similar as possible to the real track score. This requirement is a typical regression problem. Thus, mean square error (MSE) loss is used as L_{s}:

$$L_{\mathrm{s}}=\mathrm{MSE}\left(\hat{s}_{i,j},s_{i,j}\right)=\frac{1}{2}\left(\hat{s}_{i,j}-s_{i,j}\right)^2, \quad (43)$$

where $\hat{s}_{i,j}$ is the estimated track score and $s_{i,j}$ is the real track score. Above all, loss function L is the sum of L_{dc}, L_{sc}, and L_{s}:

$$L=L_{\mathrm{dc}}+L_{\mathrm{sc}}+L_{\mathrm{s}}. \quad (44)$$

3 Network training, parameter selection, and ablation experiments

To choose the best embedding dimension D, select the best margin distance m, find the best number of registration blocks M in the spatial–temporal registration module, decide the proper number of multi-head attention H, analyze the performance of different position embedding strategies, validate the effectiveness of dropping normalization, demonstrate the rationality

of the loss function, and analyze the performance of different association periods, in this section, we conduct embedding dimension analysis, margin distance analysis, registration block number analysis, multi-head attention number analysis, position embedding strategy analysis, normalization validation, loss function ablation experiments, and association period analysis. These analysis experiments can be seen in Section 2 of the supplementary materials. The average association precision rate (AP), the average association recall rate (REC), and the average association F1-score (F1) are used to assess the association performance. The values of all indicators are between 0 and 1 and larger is better. Each assessment indicator is defined as the following.

1. The average association precision rate (AP):

$$\mathrm{AP}=\frac{N_{\mathrm{AP}}}{N_{\mathrm{t}}}. \tag{45}$$

The initial value of N_{AP} is 0. If a track pair is associated according to MH-T2TA and the label, N_{AP} is plus 1. N_{t} is the number of track pairs that are associated according to MH-T2TA.

2. The average association recall rate (REC):

$$\mathrm{REC}=\frac{N_{\mathrm{REC}}}{N_{\mathrm{a}}}. \tag{46}$$

The initial value of N_{REC} is 0. If a track pair is associated according to MH-T2TA and the label, N_{REC} is plus 1. N_{a} represents the number of associated pairs according to the label.

3. The average association F1-score (F1):

$$\mathrm{F1}=\frac{2\times\mathrm{AP}\times\mathrm{REC}}{\mathrm{AP}+\mathrm{REC}}. \tag{47}$$

F1 is used as a comprehensive measurement of AP and REC.

During the training period, we train all data for 50 epochs with a batch size of 256. We use the AdamW (Loshchilov and Hutter, 2017) optimizer with a learning rate of 1×10^{-4} and a weight decay of 0.01. The CosineAnnealingLR learning strategy (Loshchilov and Hutter, 2016) is used to let the learning rate decrease from 1×10^{-4} to 1×10^{-5} as the epoch increases during the 50 epochs. All experiments are conducted in a 64-bit workstation with the PyTorch deep learning framework (Paszke et al., 2019). The detailed configurations for these experiments are as follows: Ubuntu 18.04.5 LTS, 32 GB RAM, Intel Core i7-8700K CPU @3.70 GHz, NVIDIA GeForce RTX 2080Ti GPU.

For the training dataset, the association period is T=60 s and it is selected randomly for each track pair. For the test dataset, the association period is also T=60 s. To fully verify the association results at every moment in one scenario, the interval between two association periods is 60 s for each scenario. Therefore, almost every period of time in one scenario can be tested.

The dataset is constructed by the multisource track association dataset (MTAD) based on the global AIS (Cui et al., 2023). MTAD is based on the AIS track data with track cleaning, grid division, and error adding. All tracks are real ones reported by the AIS and reflect the movement law of global sea surface targets. For the original AIS tracks, we conduct the following cleaning steps:

1. Split a track that has not been updated for a long time. When the track update time is greater than 600 s, the track is truncated once until the end of the track.

2. Delete a track that is stationary or has a too low speed. If the average speed is less than or equal to 1 knot, the maximum longitude minus the minimum longitude is less than or equal to 0.5°, and the maximum latitude minus the minimum latitude is less than or equal to 0.5°, the track is dropped.

3. Delete a track that jumps from sampling points. Traversing each sampling point in one track, if the absolute value of the longitude difference between two points is greater than 0.5°, or the absolute value of the latitude difference is greater than 0.5°, the track is dropped.

4. Delete a track that is too short. Only tracks with more than 30 sampling points and a duration greater than 300 s are saved.

After the four cleaning steps, the saved tracks can be seen as the true values of tracks, which are called the original tracks. All track datasets are constructed based on the original tracks.

Then we divide the global latitude and longitude into grids by precision α=0.5°, and set the center of the scenario W_0 to translate tracks. Finally, we add random errors and systematic errors to tracks in each scenario to generate tracks from different sensors. The

training dataset contains 10 000 scenarios and each scenario consists of several to hundreds of tracks, covering various movement patterns and target types. Some scenario samples are shown in Fig. 6.

The parameters include the update period, scenario center, target detection probability, random error, and systematic error. The detailed parameters of the dataset used in this study are shown in Table 1.

Table 1 Parameters of the MTAD dataset

Parameter	Value/Description
T_{SA}(s)	10
T_{SB}(s)	20
W_0(°)	$(20, 131)$
P_{d}	0.8
E_{rA}(°)	$\mathcal{N}(0, 0.0015^2)$
E_{rB}(°)	$\mathcal{N}(0, 0.0015^2)$
E_{sA}(°)	0
E_{sB}(°)	$\mathcal{U}(-0.03, -0.01)$ or $\mathcal{U}(0.01, 0.03)$, 50%

$\mathcal{N}$ and $\mathcal{U}$ represent the Gaussian distribution and the uniform distribution, respectively. The systematic error of sensor B follows a uniform distribution $\mathcal{U}(-0.03, -0.01)$ or $\mathcal{U}(0.01, 0.03)$ with a 50% probability

The update period of sensor A is T_{SA}=10 s, the update period of sensor B is T_{SB}=20 s, the scenario center W_0=(20°, 131°), and the target detection probability P_{d}=0.8. $E_{\mathrm{r}A}$ is the random error of sensor A, $E_{\mathrm{r}B}$ is the random error of sensor B, $E_{\mathrm{s}A}$ is the systematic error of sensor A, and $E_{\mathrm{s}B}$ is the systematic error of sensor B. The random error and systematic error are added to latitude and longitude independently.

The random error affects each track sampling point, representing the instantaneous fluctuation of the track's position due to various uncertain factors. Each track sampling point is assigned a random error and the random errors between different track sampling points are independent. The systematic error affects the entire track, indicating the degree of deviation of the track from the real position. Each track is assigned a systematic error and the systematic errors between different tracks are independent.

However, if each scenario is traversed without using batch processing during training, the training time will be greatly increased. To implement batch training, the training strategy needs to be changed. First, we traverse all track scenarios, and the associated track pairs are saved separately according to sensors A and B. The two associated tracks are indexed identically when saved. Second, we choose the positive index $\mathrm{ID_P}$ and the negative index $\mathrm{ID_N}$ from the tracks of a single source and it should be guaranteed that $\mathrm{ID_P} \neq \mathrm{ID_N}$. According to $\mathrm{ID_P}$, we select the positive track A $T^A_{\mathrm{ID_P}}$ from sensor A and the positive track B $T^B_{\mathrm{ID_P}}$ from sensor B. According to $\mathrm{ID_N}$, we select the negative track A $T^A_{\mathrm{ID_N}}$ from sensor A and the negative track B $T^B_{\mathrm{ID_N}}$ from sensor B. Finally, after implementing the data preprocessing described above, the combinations of $\left[T^A_{\mathrm{ID_P}}, T^B_{\mathrm{ID_P}}\right]$ and $\left[T^A_{\mathrm{ID_N}}, T^B_{\mathrm{ID_N}}\right]$ are regarded as associated track pairs and their real track scores are 1. The combinations of $\left[T^A_{\mathrm{ID_P}}, T^B_{\mathrm{ID_N}}\right]$ and $\left[T^A_{\mathrm{ID_N}}, T^B_{\mathrm{ID_P}}\right]$ are regarded as unassociated track pairs and their real track scores are 0. The number of tracks in each combination is B', where B' is the batch size.

In addition, to carry out the quantitative analysis of the association performance, we construct five fixed testing scenarios and the number of targets in these scenarios is 3, 15, 30, 60, and 90, separately. The fixed testing scenarios are shown in Fig. 7.

4 Contrastive experiment

After the training process is finished, the parameters of the network have been determined. To verify the effectiveness of the proposed MH-T2TA, we conduct visualization of the unified registration track, real scenario testing, extension testing of more sensors, robustness testing for data variation, and contrastive experiments. Only the contrastive experiments are presented here. Other verification experiments can be seen in Section 3 of the supplementary materials. The parameters are shown in Table 2.

In this section, our proposed MH-T2TA method is compared with the following methods: the statistical reasoning methods, weighted distance (WD) (Kanyuck and Singer, 1970), and maximum likelihood estimation (MLE) (Sun et al., 2023); the anti-bias method reference topology feature (REF) (Qi et al., 2018); the fuzzy mathematical method fuzzy double-threshold (FDT) (Du W et al., 2013); the machine learning method AdaBoost (ADB) (Xu and Fang, 2021); the deep

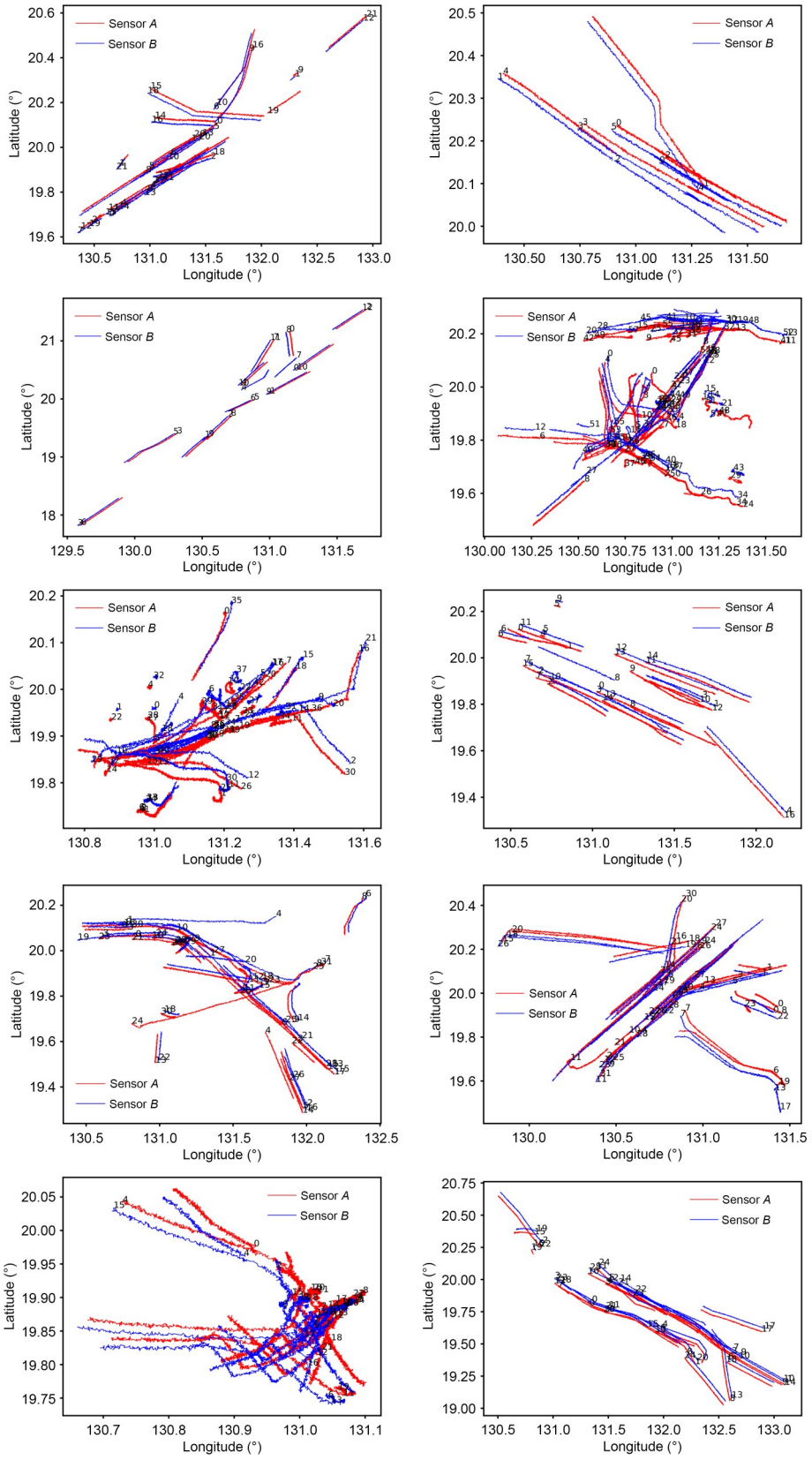

Fig. 6 Scenario samples of MTAD. The batch number is marked at the beginning of each track (References to color refer to the online version of this figure)

learning method; the integrated track and scene features (IF) method (Jin et al., 2023). In addition, some classical track distance measurement algorithms like WD can be applied to construct the association matrix, and the association results can be obtained by the Hungarian algorithm (Crouse, 2016). We also compare the association performance of the proposed MH-T2TA with those of classical track distance measurement algorithms: Hausdorff distance (HD) (Hausdorff, 1914), Fréchet distance (FD) (Alt and Godau, 1995), longest common subsequence (LCSS) (Vlachos et al., 2002), edit distance with real plenty (ERP) (Chen and Ng, 2004), and edit distance on real sequence (EDR) (Chen et al., 2005). The contrastive experiments are based on five fixed testing scenarios, and 50 Monte Carlo simulations are conducted to analyze the association results of these T2TA methods. The association results of contrastive experiments are shown in Table 3, and the association time is shown in Table 4 and Fig. 8. For clarity, only the association times of T2TA methods are plotted in Fig. 8.

Table 2 Determined parameters for MH-T2TA

Parameter	Value/Description
D	256
m	0.4
M	1
H	1
Position embedding strategy	Learnable
Normalization	Dropping
Loss functions	L_{dc}, L_{sc}, L_s

According to Table 3, overall, MH-T2TA achieves the best AF1 and the best association performance. First, we compare the association results between MH-T2TA and other T2TA methods. The AF1 of MH-T2TA gives a 0.1109 higher score than that of the REF method that ranks second, which is an obvious improvement. Looking at the F1-score of different fixed testing scenarios, the association performance of different methods varies greatly, which indicates the large performance gaps between different methods. In the scenarios containing 15 or fewer targets, except for WD that is too simple to associate tracks precisely, the other methods achieve better association performance, which indicates that these methods have the basic association ability. However, as the number of targets increases, the association effect of association methods other than MH-T2TA decreases sharply and

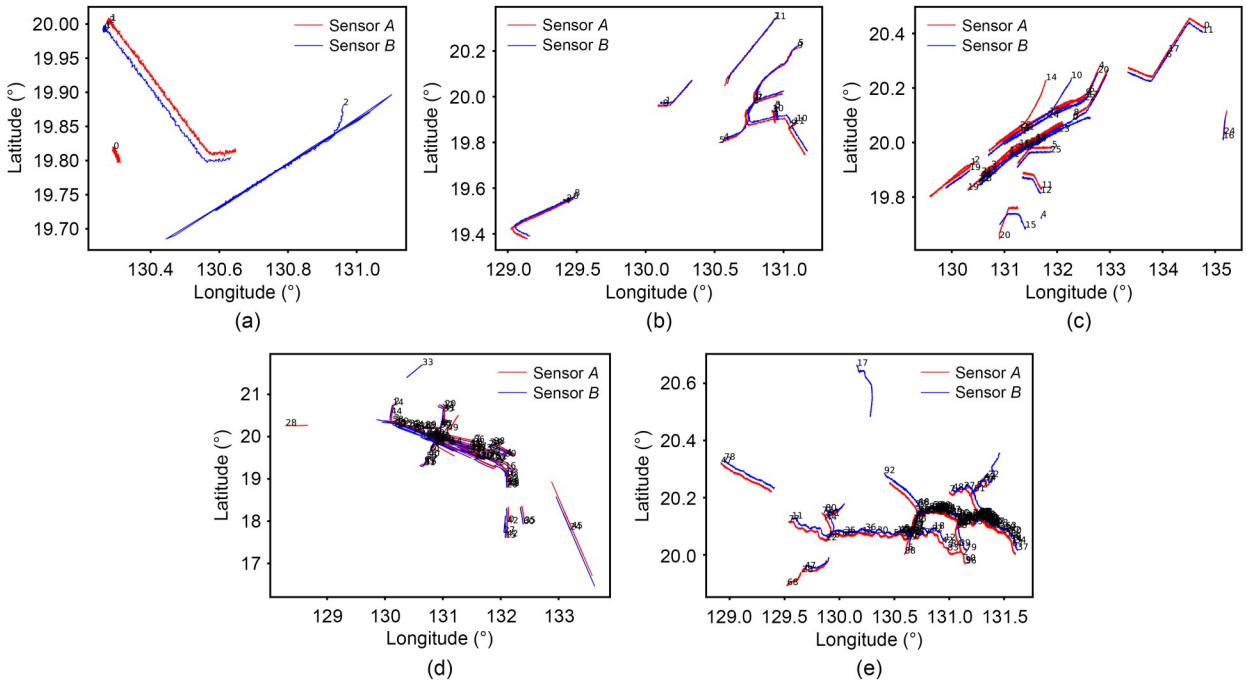

Fig. 7 Visualization of the fixed testing scenarios with 3 (a), 15 (b), 30 (c), 60 (d), and 90 (e) targets. The batch number is marked at the beginning of each track (References to color refer to the online version of this figure)

MH-T2TA still achieves the best performance. Specifically, in the very dense scenario containing 90 targets, performance gaps between MH-T2TA and other methods are very large. The F1 of MH-T2TA at 90 targets is 0.7946, which is 0.5439 higher than that of the poorest method IF and 0.0422 higher than that of the REF method that ranks second. In the IF method, the input of the network is the statistical features of tracks designed by humans instead of the original track data, and they are ambiguous in dense scenarios. Moreover, the scenarios of MTAD exhibit obvious systematic errors and asynchronous sensor update periods. Consequently, it is difficult to distinguish tracks solely based on statistical features, leading to a marked decline in the association performance of IF.

Then we analyze the results of the association between MH-T2TA and the classical track distance measurement algorithms. The AF1 of MH-T2TA achieves a 0.0631 higher score than that of the LCSS method that ranks second and the association performance of LCSS surpasses that of REF. In general, except for ERP, the methods based on the Euclidean distance are worse than those based on edit distance. Comparing the F1-score of different classical track distance measurement algorithms, we find that the association results of different methods are quite different and the best, LCSS, with an AF1 of 0.8257, surpasses ERP with an AF1 of 0.5502 by 0.2755. Using the strategies of the association threshold and the delay association, LCSS achieves optimal performance among classical track distance measurement algorithms and demonstrates high robustness to noise in tracks. EDR adopts the strategies of the association threshold and the unassociation penalty to address the

Table 3 Association results of contrastive experiments

Method	AF1*	F1				
		3	15	30	60	90
WD (Kanyuck and Singer, 1970)	0.4949	1	0.8036	0.8069	0.6452	0.4311
MLE (Sun et al., 2023)	0.6858	1	0.9463	0.9144	0.7994	0.4799
REF (Qi et al., 2018)	0.7779	1	0.8704	0.7626	0.7896	0.7524
FDT (Du W et al., 2013)	0.6155	1	0.8490	0.8518	0.6670	0.4506
ADB (Xu and Fang, 2021)	0.7254	1	0.9891	0.9740	0.7765	0.5554
IF (Jin et al., 2023)	0.4665	1	0.9454	0.8228	0.4658	0.2507
HD (Hausdorff, 1914)	0.6059	1	0.8490	0.8118	0.6575	0.4492
FD (Alt and Godau, 1995)	0.6055	1	0.8490	0.8069	0.6642	0.4456
LCSS (Vlachos et al., 2002)	0.8257	1	0.9463	0.9208	0.8508	0.7513
ERP (Chen and Ng, 2004)	0.5502	1	0.9278	0.8404	0.5151	0.3989
EDR (Chen et al., 2005)	0.7609	1	0.9546	0.9306	0.7779	0.6528
MH-T2TA (ours)	0.8888	1	0.9848	1	0.9451	0.7946

* AF1 is the weighted mean F1 value of all fix testing scenarios

Table 4 Association time of contrastive experiments for all methods

Method	Association time (s)				
	3	15	30	60	90
WD (Kanyuck and Singer, 1970)	0.0171	0.0982	0.2558	0.5603	1.1874
MLE (Sun et al., 2023)	0.0181	0.1234	0.3591	0.9249	1.8887
REF (Qi et al., 2018)	0.0323	1.3903	22.9646	402.2701	1930.4114
FDT (Du et al., 2013)	0.0199	0.1504	0.4268	1.0272	2.3011
ADB (Xu and Fang, 2021)	0.0245	0.1023	0.3214	0.8972	1.8024
IF (Jin et al., 2023)	1.0041	1.0884	1.2945	2.5919	4.9630
HD (Hausdorff, 1914)	0.0183	0.1249	0.2703	0.9036	1.9951
FD (Alt and Godau, 1995)	0.0174	0.1345	0.3249	1.0932	2.4884
LCSS (Vlachos et al., 2002)	0.0171	0.1242	0.2701	0.8823	1.9847
ERP (Chen and Ng, 2004)	0.0178	0.1502	0.3679	1.2493	2.8034
EDR (Chen et al., 2005)	0.0181	0.1527	0.3773	1.2987	2.9877
MH-T2TA (ours)	0.4418	0.9514	2.6816	7.1142	16.8338

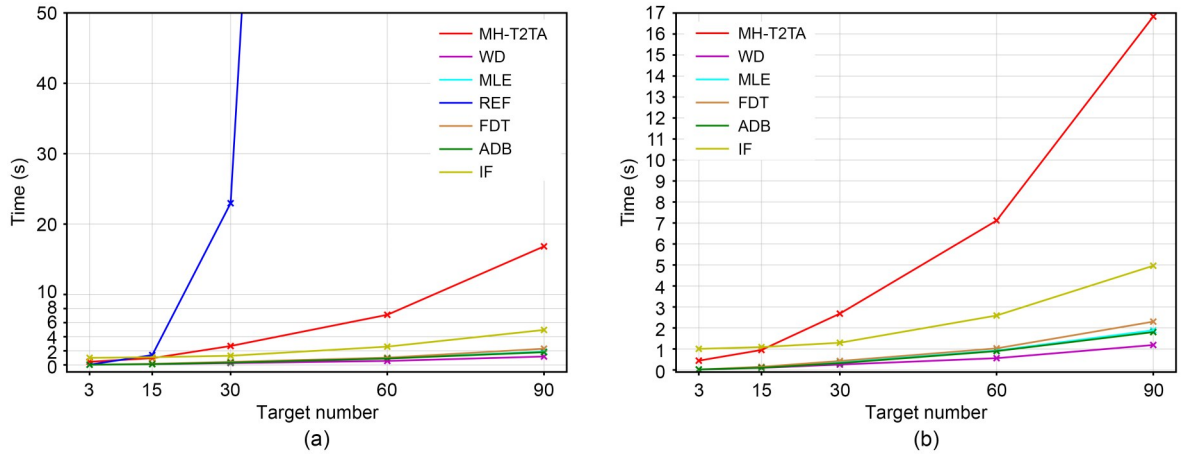

Fig. 8 Association time of contrastive experiments for T2TA methods: (a) association time of all T2TA methods; (b) association time without reference topology (References to color refer to the online version of this figure)

noise in tracks, but its association results are inferior to those of LCSS, suggesting that the delay association strategy is more effective for track association than the unassociation penalty. However, the selection of the threshold in LCSS is often difficult, and selecting an inappropriate threshold value will lead to a serious deterioration of the association results. Moreover, the crucial prerequisite for classical track distance measurement algorithms to achieve T2TA is that the update period of a pair of tracks is consistent. Therefore, it is essential for these methods to conduct time registration before distance comparison, which will result in error accumulations.

In terms of the association time in Table 4 and Fig. 8, the most obvious result is the association time of REF. As mentioned by Qi et al. (2018), the association and optimization process of REF is very complex and needs too much time. The time consumption increases sharply with the increase in the number of targets, which cannot meet the requirements of real-time association. WD, MLE, FDT, HD, FD, LCSS, ERP, and EDR take about the same amount of time and the difference does not exceed 2 s. This is because their association strategies are similar, and they all rely on the distance or probability to construct the association matrix and obtain the association result. Due to the more straightforward calculation process of machine learning compared to deep learning, ADB requires less association time than IF, with a difference of 3.1606 s. The association time of MH-T2TA is less than that of IF when the number of targets is less than or equal to 15. However, as the number of targets increases, the excessive number of tracks within a single cluster in dense scenarios results in numerous association hypotheses for each cluster, leading to a significant increase in the time required to generate these association hypotheses. When the number of targets exceeds 15, the association time required by MH-T2TA surpasses that of other methods, and the time consumption increases at an accelerating rate. Therefore, MH-T2TA needs to spend more time to obtain better association performance. How to further reduce the time required for hypothesis generation and enhance the association efficiency is a key point for further research.

5 Conclusions

In this study, based on a multiple-hypothesis algorithm, we designed a neural network to obtain an intelligent track score to solve the T2TA problem (MH-T2TA). Aiming at the ubiquitous problems of errors and inconsistent update periods in the track data, we designed a spatial–temporal registration module based on self-attention and a contrastive learning architecture to eliminate errors and unify the distributions of asynchronous tracks. Focusing on the suboptimal association results and the dependencies on prior information and the assumed motion model, we combined the multiple-hypothesis algorithm with deep learning to construct an intelligent track score network for estimating the track score of a pair of tracks. This combination achieved optimal association results

and reduced the impact of various dependencies and uncertainties.

By training the neural network, we analyzed the association performance of different parameters and selected the best one. More importantly, we demonstrated from theory and experiments that layer normalization is unsuitable for track data. The effectiveness of the spatial–temporal registration module was verified by visualizations of the unified registration track. The visual results showed that the spatial–temporal registration module has the ability to register and unify tracks to eliminate the effects of errors and inconsistent update periods. The real scenario testing showed that MH-T2TA has strong generalization ability for different sensor types. The extension testing of more sensors validated the adaptability of MH-T2TA to scenarios containing more than two sensors. Robustness testing demonstrated that MH-T2TA is robust to variations in input data characteristics and can adapt to scenarios with different error distributions. Finally, the contrastive experiments indicated that the association performance of MH-T2TA significantly exceeds those of other methods.

However, because more association hypotheses need to be generated in dense scenarios, MH-T2TA takes more time than other methods. In addition, the intelligent track score network can provide only a track association score, and lacks the ability to elucidate the reason why a pair of tracks are associated or not. Therefore, how to further reduce the time required for hypothesis generation to enhance association efficiency and how to improve the interpretability of the intelligent track score network are key points for future research.

Contributors

Pingliang XU designed the research. Pingliang XU and Yaqi CUI processed the data. Pingliang XU drafted the paper. Wei XIONG helped organize the paper. All the authors revised and finalized the paper.

Conflict of interest

All the authors declare that they have no conflict of interest.

Data availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

Alt H, Godau M, 1995. Computing the Fréchet distance between two polygonal curves. *Int J Comp Geom Appl*, 5(01n02):75-91. https://doi.org/10.1142/S0218195995000064

Aziz AM, 2011. A new fuzzy clustering approach for data association and track fusion in multisensor-multitarget environment. IEEE Aerospace Conf, p.1-10. https://doi.org/10.1109/AERO.2011.5747430

Ba JL, Kiros JR, Hinton GE, 2016. Layer normalization. https://arxiv.org/abs/1607.06450

Bar-Shalom Y, 2008. On the sequential track correlation algorithm in a multisensor data fusion system. *IEEE Trans Aerosp Electron Syst*, 44(1):396. https://doi.org/10.1109/TAES.2008.4517016

Bar-Shalom Y, Li XR, 1995. Multitarget-Multisensor Tracking: Principles and Techniques. YBS Publishing, Storrs, USA.

Bar-Shalom Y, Fortmann TE, Cable PG, 1990. Tracking and data association. *J Acoust Soc Am*, 87(2):918-919. https://doi.org/10.1121/1.398863

Blackman SS, 1986. Multiple-Target Tracking with Radar Applications. Artech House, Dedham, USA.

Blackman SS, 2004. Multiple hypothesis tracking for multiple target tracking. *IEEE Aerosp Electr Syst Mag*, 19(1):5-18. https://doi.org/10.1109/MAES.2004.1263228

Blackman SS, Dempster RJ, Broida TJ, 1993. Multiple hypothesis track confirmation for infrared surveillance systems. *IEEE Trans Aerosp Electron Syst*, 29(3):810-824. https://doi.org/10.1109/7.220932

Blostein SD, Richardson HS, 1994. A sequential detection approach to target tracking. *IEEE Trans Aerosp Electron Syst*, 30(1):197-212. https://doi.org/10.1109/7.250420

Chen L, Ng R, 2004. On the marriage of Lp-norms and edit distance. Proc Int Conf on Very Large Data Bases, p.792-803. https://doi.org/10.1016/B978-012088469-8.50070-X

Chen L, Özsu MT, Oria V, 2005. Robust and fast similarity search for moving object trajectories. Proc ACM SIGMOD Int Conf on Management of Data, p.491-502. https://doi.org/10.1145/1066157.1066213

Chong C, Mori S, Tse E, et al., 1982. Distributed Hypothesis Formation in Distributed Sensor Networks. Advanced Information and Decision Systems Report, No. IR-1 I015-l.

Cox IJ, Hingorani SL, 1996. An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Trans Patt Anal Mach Intell*, 18(2):138-150. https://doi.org/10.1109/34.481539

Crouse DF, 2016. On implementing 2D rectangular assignment algorithms. *IEEE Trans Aerosp Electron Syst*, 52(4):1679-1696. https://doi.org/10.1109/TAES.2016.140952

Cui YQ, Liu Y, Tang TT, et al., 2021. A new adaptive track correlation method for multiple scenarios. *IET Radar Sonar Navig*, 15(9):1112-1124. https://doi.org/10.1049/rsn2.12101

Cui YQ, Xu PL, Gong C, et al., 2023. Multisource track association dataset based on the global AIS. *J Electr Inform Technol*, 45(2):746-756 (in Chinese).

https://doi.org/10.11999/JEIT221202

Devlin J, Chang MW, Lee K, et al., 2018. BERT: pre-training of deep bidirectional transformers for language understanding. https://arxiv.org/abs/1810.04805

Du RZ, Liu L, Bai XR, et al., 2021. A new scatterer trajectory association method for ISAR image sequence utilizing multiple hypothesis tracking algorithm. *IEEE Trans Geosci Remote Sens*, 60:1-13. https://doi.org/10.1109/TGRS.2021.3087192

Du W, Ning HS, Wei Y, et al., 2013. Fuzzy double-threshold track association algorithm using adaptive threshold in distributed multisensor-multitarget tracking systems. IEEE Int Conf on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing, p.1133-1137. https://doi.org/10.1109/GreenCom-iThings-CPSCom.2013.197

Hadsell R, Chopra S, LeCun Y, 2006. Dimensionality reduction by learning an invariant mapping. IEEE Computer Society Conf on Computer Vision and Pattern Recognition, p.1735-1742. https://doi.org/10.1109/CVPR.2006.100

Hausdorff F, 1914. Grundzüge der Mengenlehre. Veit & Comp., Leipzig, Germany.

He Y, Zhang JW, 2006. New track correlation algorithms in a multisensor data fusion system. *IEEE Trans Aerosp Electron Syst*, 42(4):1359-1371. https://doi.org/10.1109/TAES.2006.314577

He Y, Wang GH, Guan X, 2010. Information Fusion Theory with Applications. Electronic Industry Press, Beijing, China (in Chinese).

Hochreiter S, Schmidhuber J, 1997. Long short-term memory. *Neur Comput*, 9(8):1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

Jin B, Tang YF, Zhang ZK, et al., 2023. Radar and AIS track association integrated track and scene features through deep learning. *IEEE Sens J*, 23(7):8001-8009. https://doi.org/10.1109/JSEN.2023.3245647

Kanyuck AJ, Singer RA, 1970. Correlation of multiple-site track data. *IEEE Trans Aerosp Electron Syst*, AES-6(2): 180-187. https://doi.org/10.1109/TAES.1970.310100

Klein I, Bar-Shalom Y, 2016. Tracking with asynchronous passive multisensor systems. *IEEE Trans Aerosp Electron Syst*, 52(4):1769-1776. https://doi.org/10.1109/TAES.2016.150099

Kurien T, 1990. Issues in the design of practical multitarget tracking algorithms. In: Yaakov BS (Ed.), Multitarget-Multisensor Tracking: Advanced Applications. Artech House, Norwood, USA, p.43-84.

Lee CS, Whang IH, Ra WS, 2023. Knowledge-based multiple hypothesis tracking and identification of manoeuvring reentry targets. *IET Radar Sonar Navig*, 17(10):1479-1497. https://doi.org/10.1049/rsn2.12436

Loshchilov I, Hutter F, 2016. SGDR: stochastic gradient descent with warm restarts. https://arxiv.org/abs/1608.03983

Loshchilov I, Hutter F, 2017. Decoupled weight decay regularization. https://arxiv.org/abs/1711.05101

Misra D, 2019. Mish: a self regularized non-monotonic activation function. https://arxiv.org/abs/1908.08681

Paszke A, Gross S, Massa F, et al., 2019. PyTorch: an imperative style, high-performance deep learning library. Proc 33rd Conf on Neural Information Processing System, p.1-12.

Qi L, Dong K, Liu Y, et al., 2017. Anti-bias track-to-track association algorithm based on distance detection. *IET Radar Sonar Navig*, 11(2):269-276. https://doi.org/10.1049/iet-rsn.2016.0139

Qi L, He Y, Dong K, et al., 2018. Multi-radar anti-bias track association based on the reference topology feature. *IET Radar Sonar Navig*, 12(3):366-372. https://doi.org/10.1049/iet-rsn.2017.0356

Reid DB, 1977. A Multiple Hypothesis Filter for Tracking Multiple Targets in a Cluttered Environment. Lockhead Missiles & Space Company, Incorporated, USA.

Reid DB, 1979. An algorithm for tracking multiple targets. *IEEE Trans Autom Contr*, 24(6):843-854. https://doi.org/10.1109/TAC.1979.1102177

Shi Y, Wang Y, Wang SG, 2006. Fuzzy data association based on target topology of reference. *J Nat Univ Def Technol*, 28(4):105-109 (in Chinese). https://doi.org/10.3969/j.issn.1001-2486.2006.04.022

Sönmez HH, Hocaoğlu AK, 2022. Asynchronous track-to-track association algorithm based on reference topology feature. *Sign Imag Video Process*, 16(3):789-796. https://doi.org/10.1007/s11760-021-02019-9

Su JL, Lu Y, Pan SF, et al., 2021. RoFormer: enhanced transformer with rotary position embedding. https://arxiv.org/abs/2104.09864

Sun WF, Li XT, Pang ZZ, et al., 2023. Track-to-track association based on maximum likelihood estimation for T/R-R composite compact HFSWR. *IEEE Trans Geosci Remote Sens*, 61:5102012. https://doi.org/10.1109/TGRS.2023.3253784

Tian W, Wang Y, Shan XM, et al., 2014. Track-to-track association for biased data based on the reference topology feature. *IEEE Signal Process Lett*, 21(4):449-453. https://doi.org/10.1109/LSP.2014.2305305

Tokta A, Hocaoglu AK, 2019. Sensor bias estimation for track-to-track association. *IEEE Signal Process Lett*, 26(10): 1426-1430. https://doi.org/10.1109/LSP.2019.2934596

Vaswani A, Shazeer N, Parmar N, et al., 2017. Attention is all you need. https://arxiv.org/abs/1706.03762

Vlachos M, Kollios G, Gunopulos D, 2002. Discovering similar multidimensional trajectories. Proc 18th Int Conf on Data Engineering, p.673-684. https://doi.org/10.1109/ICDE.2002.994784

Wang J, Zeng YJ, Wei SM, et al., 2021. Multi-sensor track-to-track association and spatial registration algorithm under incomplete measurements. *IEEE Trans Signal Process*, 69: 3337-3350. https://doi.org/10.1109/TSP.2021.3084533

Werthmann JR, 1992. Step-by-step description of a computationally efficient version of multiple hypothesis tracking. Proc SPIE, Signal and Data Processing of Small Targets, 1698:288-300. https://doi.org/10.1109/10.1117/12.139379

Wu ZH, Pan SR, Chen FW, et al., 2021. A comprehensive survey on graph neural networks. *IEEE Trans Neur Netw Learn Syst*, 32(1):4-24. https://doi.org/10.1109/TNNLS.2020.2978386

Xiong W, Xu PL, Cui YQ, et al., 2021. Track segment association via track graph representation learning. *IET Radar Sonar Navig*, 15(11):1458-1471. https://doi.org/10.1049/rsn2.12138

Xiong W, Xu PL, Cui YQ, 2024. Unsupervised and interpretable track-to-track association based on homography estimation of radar bias. *IET Radar Sonar Navig*, 18(2):294-307. https://doi.org/10.1049/rsn2.12483

Xu Z, Fang L, 2021. An improved track association algorithm based on AdaBoost and decision tree. IEEE 4th Int Conf on Advanced Electronic Materials, Computers and Software Engineering, p.794-800. https://doi.org/10.1109/AEMCSE51986.2021.00164

Yang YP, Yang F, Sun LG, et al., 2022. Multi-target association algorithm of AIS-radar tracks using graph matching-based deep neural network. *Ocean Eng*, 266:112208. https://doi.org/10.1016/j.oceaneng.2022.112208

Zhao HC, Sha ZC, Wu J, 2017. An improved fuzzy track association algorithm based on weight function. IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conf, p.1125-1128. https://doi.org/10.1109/IAEAC.2017.8054188

Zhu HY, Chen S, 2014. Track fusion in the presence of sensor biases. *IET Signal Process*, 8(9):958-967. https://doi.org/10.1049/iet-spr.2013.0393

Zhu HY, Han SY, 2014. Track-to-track association based on structural similarity in the presence of sensor biases. *J Appl Math*, 2014:1-8. https://doi.org/10.1155/2014/294657

Zhu HY, Wang W, Wang C, 2016. Robust track-to-track association in the presence of sensor biases and missed detections. *Inform Fus*, 27:33-40. https://doi.org/10.1016/j.inffus.2015.05.002

List of supplementary materials

1 Algorithms
2 Analysis experiments
3 Verification experiments
Algorithm S1 Gating and coarse association
Algorithm S2 Association cluster
Algorithm S3 Hypothesis generation
Table S1 Association results of different embedding dimensions
Table S2 Association results of different margin distances
Table S3 Association results of different numbers of registration blocks
Table S4 Association results of different numbers of the multi-head attention
Table S5 Association results of different position embedding strategies
Table S6 Association results of different loss functions
Table S7 Association results of different association periods
Table S8 Association relations and association results between radar and ADS-B
Table S9 Parameters of the scenarios containing three sensors
Table S10 Association results from extension testing
Table S11 Error setting for each error level
Table S12 Association results for different error levels
Fig. S1 Association results of different normalizations
Fig. S2 Visualizations of the unified registration track
Fig. S3 Original track scenario of radar and ADS-B
Fig. S4 Example of fixed scenarios containing three sensors
Fig. S5 Fixed testing scenarios for each error level