Online Multi-camera People Tracking with Spatial-temporal Mechanism and Anchor-feature Hierarchical Clustering

Riu Cherdchusakulchai * Sasin Phimsiri * Visarut Trairattanapa * Suchat Tungjitnob *
Wasu Kudisthalert Pornprom Kiawjak Ek Thamwiwatthana Phawat Borisuitsawat
Teepakorn Tosawadi Pakcheera Choppradit Kasisdis Mahakijdechachai
Supawit Vatathanavaro Worawit Saetan Vasin Suttichaya *
AI and Robotics Ventures (ARV), Thailand
{riuc, sasinp, visarutt, suchatt, wasuk, pornpromk, ekt, phawatb,
teepakornt, pakcheerac, kasisdism, supawitv, worawits, vasins}@arv.co.th

Abstract

Multi-camera Multi-object tracking (MTMC) surpasses conventional single-camera tracking by enabling seamless object tracking across multiple camera views. This capability is critical for security systems and improving situational awareness in various environments. This paper proposes a novel MTMC framework designed for online operation. The framework employs a three-stage pipeline: Multi-object Tracking (MOT), Multi-target Multi-camera Tracking (MTMC), and Cross Interval Synchronization (CIS). In the MOT stage, ReID features are extracted and localized tracklets are created. MTMC links these tracklets across cameras using spatial-temporal constraints and constraint hierarchical clustering with anchor features for improved inter-camera association. Finally, CIS ensures the temporal coherence of tracklets across time intervals. The proposed framework achieves robust tracking performance, validated on the challenging 2024 AI City Challenge with a HOTA score of 51.0556%, ranking sixth. The code is available at: https://github.com/ARV-MLCORE/AIC2024_Track1_ARV

1. Introduction

Multi-camera multi-object tracking (MTMC) is a rapidly developing field within computer vision, demonstrating significant advantages over traditional object detection methodologies. Its key advantage is the ability to locate and track the same object across multiple camera views, overcoming the limitations of single-camera systems. MTMC has diverse applications, including enhancing security surveillance in both public and private spaces. Furthermore, it can address Environmental, Social, and Governance (ESG) concerns by facilitating the estimation of vehicle travel distances for carbon footprint calculations and enabling the development of forest monitoring systems. The technique can be integrated with various camera platforms to track objects beyond humans and vehicles, broadening its potential applications. The principles of MTMC can be divided into three key components:

1. **Re-identification (ReID)**: the algorithms are responsible for matching the same objects in each camera. This can be achieved by analyzing embedded vectors from MOT or other features such as an object's pose, depth, trajectory, color, or texture.
2. **Multi-object Tracking (MOT)**: MOT algorithms detect and track the positions of objects within individual image or video frames. MOT plays a crucial role in extracting the features necessary to generate the embedded vectors utilized in the ReID process.
3. **Camera Data Association**: This component determines the relationships between the spatial locations of each different camera and their interconnections. By understanding these relationships, the system can optimize ReID performance and reduce computational costs, as comparisons are focused on relevant camera views. Camera data association also allows for the accurate calculation of object trajectories across multiple cameras.

This paper presents three key contributions that advance person tracking and re-identification across non-uniform camera networks.

1. **Novel Framework for Online Multi-camera People Tracking:** We proposed a framework designed for online people tracking across heterogeneous camera setups. This framework utilizes time-window approach to capture short-term video frames. This facilitates online tracking in scenarios with challenge factors.
2. **Spatial-temporal Constraints and Refinement:** We

*These authors contributed equally to this work.

introduced the spatial-temporal constraints to ensure compatibility during tracklet concatenations. There are four constraints employed in this research: Frame Overlap Constraint for Same Camera, Frame Overlap Constraint for Non-Intersecting Camera Pairs, Position Overlap Constraint, and Neighbor Constraint.

3. **Constraint Hierarchical Clustering with Anchor Feature:** To efficiently merge anchor features across cameras, we introduced a novel clustering approach called Constraint Hierarchical Clustering with Anchor Feature. This method leverages a similarity matrix to identify potential cluster merges while enforcing hierarchical constraints to ensure the merged clusters are semantically meaningful.
4. **Evaluation and Performance:** We evaluate the effectiveness of our proposed methodology in the context of Track 1 of the 2024 AI City Challenge. On the public testing set, which consists of data from synthetic multi-camera scenes, our approach achieves a Higher Order Tracking Accuracy (HOTA) score of 51.01 [16], demonstrating its strong performance in this challenging benchmark.

According to this paper, the core sections are structured into five main parts. Section 1 provides the introduction. Section 2 describes related works. Section 3 explains the proposed method. Section 4 presents experiments and lastly, Section 5 presents conclusions.

2. Related Works

2.1. Re-identification

Re-identification (ReID) is a task that aims to establish associations between reference and target objects. It is widely applied in person and vehicle re-identification scenarios. A core challenge within ReID lies in developing models capable of extracting robust and discriminative features for each object under diverse conditions. This task has recently gained significant attention due to the integration of AI applications within smart city and security surveillance solutions, aiming to reduce manual labor and enhance overall efficiency.

Within the field of MTMC, CNN-based approaches still remain common. Their popularity originates from the suitability of object detection outputs for generating embedded vectors, which are essential for similarity matching via distance calculations. Architectures like ResNet[11, 19, 21, 24], renowned for their feature extraction efficiency, continue to be in widespread use in ReID. Many papers[35, 36] proposed an Omni-Scale Network (OSNet) taking benefits of omni-scale feature extraction. However, Transformer-based methods[5] are gaining attention, particularly as they become increasingly employed in object detection tasks[2]. Transformers have been applied in novel ways for MTMC ReID, including encoding image embeddings as sequences of patches[6, 26] or hybridizing them with ResNet[4].

Furthermore, to enhance ReID performance, techniques beyond traditional appearance-based features are being explored. These include incorporating pose estimation[10, 19] to derive pose-based embeddings, leveraging multi-body level detection for matching individual body parts[29], and employing face refinement to extract features specifically from human faces[15]. ReID model performance is primarily contingent upon the training datasets utilized. In the field[34], four benchmark datasets are commonly referenced to evaluate and benchmark the performance of ReID models: Market1501[33], DUKEREID[20], MSMT17[27], and CUHK03[14]. These datasets provide diverse environments and scenarios to facilitate comprehensive testing and validation of ReID algorithms.

2.2. Object detection and object tracking

Object detection is a fundamental task within computer vision, aiming to localize and classify objects appearing within a scene. MTMC systems often leverage the features extracted from detected objects for the ReID process. Recently CNN-based architectures remain a prevalent choice within the field.[7, 8, 19, 21, 35, 36] Some well-known examples include YOLOv5[3, 15, 18, 21, 24], YOLOv7[10], YOLOv8[11, 23], and YOLOX[10]. These methods offer the capability of performing both real-time and batch inference while exhibiting greater computational efficiency compared to transformer-based architectures. For object tracking, the objective is to track and distinguish objects with consistent attributes or features across sequential image or video frames. By predicting object trajectories[8, 10], object tracking algorithms significantly enhance ReID performance. Additionally, the features extracted during tracking are employed in both ReID and Camera data association processes. Currently, notable object tracking techniques include BoT-SORT[1, 9], ByteTrack[10, 32], and DeepSORT[28].

2.3. Camera Data Association

Camera data association refers to the process of matching and tracking objects across multiple camera views or frames. The challenge lies in dealing with occlusions, varying perspectives, and appearance changes over time. This section reviews significant advancements in the field, highlighting processing methodologies. The Online methods [17, 21, 22, 28, 30, 31] utilize real-time or near real-time tracking for instant decision-making to manage object trajectories but low accuracy. Conversely, Offline methods [9, 10, 13, 18, 24] prioritize accuracy and completeness of object trajectories by processing the entire video data post-capture. This approach allows for complex computations and optimizations that are not feasible in real time, lead-

ing to more precise tracking outcomes. However, the trade-off is the inability to support instant decision-making as the analysis is conducted after the event, making it unsuitable for applications requiring immediate response or intervention.

3. Proposed Method

3.1. Notations

Let V represent the number of cameras involved in the system. Let $\Omega = [\omega_1, \omega_2, \ldots]$ denote the list of time windows, where each ω_ι represents a time interval indexed by ι. Let $\mathcal{T}\omega\iota = \{\tau_1, \tau_2, \ldots, \tau_V\}$ indicate the set of all tracklets from camera 1 to camera V within the time window ω_ι. The set of tracklets from camera i is defined as $\tau_i = \{t_1^i, t_2^i, \ldots, t_j^i\}$, where t_j^i represents the local tracklet index j in camera i.

Let N denote the number of objects being tracked. The multi-camera tracklet profile from window 1 to window T is defined by $\Pi_T = \{P_1, P_2, \ldots, P_N\}$, where each $P_k = \{t_j^i \mid t_j^i \text{ is the tracklet of object identity } k\}$.

3.2. Architecture Overview

The proposed MTMC method consists of three main modules: Multi-object Tracking (MOT), Multi-target Multi-camera Tracking (MTMC), and Cross Interval Synchronization (CIS), as depicted in Figure 1. MOT handles human object feature extraction, individual object trajectory tracking, and determination of object velocity and position. MTMC merges IDs from all involved cameras, while CIS synchronizes IDs across time intervals.

MOT employs a time-windowing approach to segment video input from multiple cameras into subintervals. Within each subinterval, a pre-trained human object detection model, such as YOLO, identifies human objects in each frame. These detections are then directed to three specialized modules: Re-identification (ReID) Feature Extraction, Object Tracking, and Spatial Information Extraction.

ReID Feature Extraction generates feature vectors from human images, while Object Tracking assigns unique IDs to initial object detections and tracks them across frames. Spatial Information Extraction analyzes movement and surroundings to determine object position, aided by user-provided camera position data.

These modules produce tracklets fed into MTMC, which employs Constraint Hierarchy Merge to enhance tracking accuracy by merging tracklets across cameras. The refined camera tracklets are then processed by CIS, which synchronizes object trajectories across time windows by comparing tracklets with previous profiles, merging or creating profiles based on distance metrics.

3.3. Multi-object Tracking

Multi-object tracking is the procedure to acquire set of all tracklets from all cameras in specific time window, $\mathcal{T}\omega\iota$. The process consists of object detection, ReID feature extraction, object tracking, spatial information extraction, and feature aggregation.

In this work, a time-windowing approach is employed for frame extraction from videos. The video sequence is segmented into fixed-size, non-overlapping windows that are subsequently shifted along the time axis. Frames are then extracted at regular intervals within each window. Notably, this research also investigates the impact of varying time-window size on the performance of the tracking process.

3.3.1 Object Detection

This work employed a YOLOv8 model to extract bounding boxex for indentified person. These bounding boxes are typically represented in (x, y, w, h) format, where (x, y) denotes the top-left corner coordinates of the box, and w and h represent its width and height, respectively.

3.3.2 Re-identification Feature Extraction

Re-identification (ReID) feature extraction is the subsequent step following the acquisition of individuals from the detection process. The detected bounding boxes are used to crop the image regions containing individuals. These cropped images are then fed into a ReID model, which aims to extract discriminative features that enable robust identification of the same person across different camera views.

This work adopted the Omni-Scale Network (OSNet) for ReID feature extraction due to its lightweight, multi-scale architecture. Additionally, OSNet's CNN design reduces the risk of overfitting, making it well-suited for this work.

3.3.3 Object Tracking

After completing the individual object detection stage, a multi-object tracking task (MOT) aims to track multiple objects across a video sequence by maintaining a unique identity of each object throughout video frames. The algorithm manages existing tracks by updating them with new detections. This association process utilizes a combination of motion and appearance information to determine if a new detection corresponds to an existing tracked object.

In this research, we employ ByteTrack, a tracker algorithm based on the Kalman Filter and Hungarian algorithm techniques. ByteTrack generates individual tracklets that represent local trajectories associated with specific person attributes.

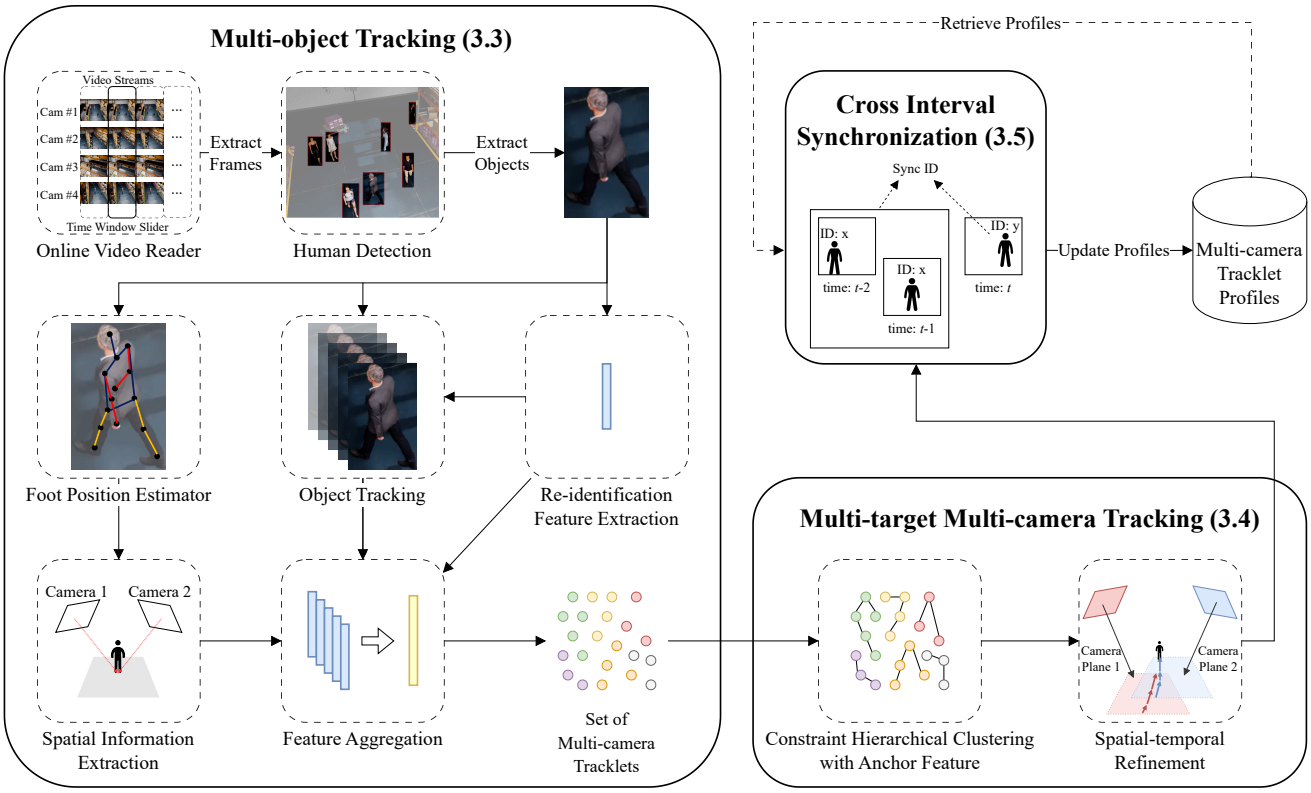

Figure 1. Proposed Framework

3.3.4 Spatial Information Extraction

This section elaborates on the Spatial Information Extraction process, a component for retrieving the world coordinate from the object's bounding box image. The process is summarized in Algorithm 1.

Algorithm 1 Spatial Information Extraction's Algorithm

Input: Bounding box coordinate (x, y, w, h);
Homography matrix $\mathbf{H}$;
Output: World coordinate;

$I \leftarrow$ GETCROPPEDIMAGE(x, y, w, h);
$(x_l, y_l), (x_r, y_r) \leftarrow$GETFEETPOSITION$(I)$;
if (x_l, y_l) **and** (x_r, y_r) are not NULL **then**
  $\mathbf{g} \leftarrow \begin{bmatrix} \frac{x_l+x_r}{2} & \frac{y_l+y_r}{2} & 1 \end{bmatrix}^T$;
else
  $\mathbf{g} \leftarrow \begin{bmatrix} x+\frac{w}{2} & y+\frac{h}{2} & 1 \end{bmatrix}^T$;
end if
$\begin{bmatrix} X & Y & z \end{bmatrix}^T \leftarrow \mathbf{H} \times \mathbf{g}$;
return X, Y;

The process is initiated with a YOLOv8 pose estimator. It takes an input image (I) and identifies the individual foot positions within the detected bounding boxes. The identified foot positions are returned as a data structure (x_l, y_l) and (x_r, y_r), representing the pixel coordinate of the left and right feet, respectively.

Subsequently, the ground-plane coordinate $(\mathbf{g})$ is required for further analysis. When both feet are visible, $\mathbf{g}$ is determined as the midpoint between the left and right foot positions. In cases where one foot is not visible, the midpoint of the lower bounding box is utilized to approximate the position.

To obtain the actual world coordinate, the ground-plain coordinate needs to be transformed from the image pixel space to the real-world coordinate system. This transformation leverages a pre-defined Homography matrix $(\mathbf{H})$. This matrix encodes the intrinsic parameters of the camera that captured the image and establishes the crucial relationship between image pixel coordinates and real-world coordinates. The Homography matrix is applied to the ground-plain coordinate to compute the corresponding world coordinate.

3.3.5 Feature Aggregation

Feature Aggregation aims to refine the representation of a tracked person by consolidating multiple feature vectors, obtained through the ReID process, into a single anchor feature. This anchor feature serves as a more robust rep-

resentation for identification across different camera views. This work proposes an area-weighted averaging approach for computing the anchor feature representation.

Let $\mathbf{F} \in \mathbb{R}^{n \times d}$ denote the feature matrix of the target person, where n represents the number of feature vectors and d represents the feature dimensionality. Each row F_i of $\mathbf{F}$ corresponds to a feature vector. Similarly, let $\mathbf{b} = [b_1\ b_2\ \ldots\ b_n]$ be a vector representing the areas of the bounding boxes associated with each feature in $\mathbf{F}$. The anchor feature, denoted as $A(\mathbf{F}, \mathbf{b})$, is calculated using the following formula:

$$A(\mathbf{F}, \mathbf{b}) = \frac{\sum_{i=1}^{n} b_i \cdot F_i}{n \cdot \sum_{i=1}^{n} b_i}. \tag{1}$$

Equation (1) represents the area-weighted average calculation for the anchor feature, ensuring a weighted aggregation based on the respective bounding box areas. This approach enhances the accuracy and reliability of the anchor feature, contributing significantly to the overall efficacy of person tracking and identification systems. The anchor is then appended to the tracklet obtained from the Object Tracking process, which helps in maintaining continuity and accuracy in person tracking across cameras and time frames.

The format of the local tracklet is described here. Each local tracklet, t_j^i represents the path of object j within camera i across a time interval. The local tracklet is structured as follows:

- OBJECT_ID: A unique identifier assigning to object j.
- FRAME: A list containing the frame numbers within the video.
- BOUNDING_BOX: A list containing the bounding boxes for each frame, represented as $[(x, y, w, h)_1, \ldots, (x, y, w, h)_N]$.
- CONFIDENCES: A list containing confidences scores for each frame, represented as $[c_1, \ldots, c_N]$, where c_p denotes the confidence associated with the detection in frame p.
- ANCHOR_FEATURE: A list containing values from the anchor feature vector in feature space $\mathbf{F}$.
- WORLD_POSITION: A list containing the world coordinates of the object for each frame, represented as $[(X, Y)_1, \ldots, (X, Y)_N]$, where $(X, Y)_i$ denotes the coordinates for frame i. The specific calculation is detailed in Algorithm 1.

In scenarios involving multiple objects within camera i, a camera tracklet is represented as $\tau_i = \{t_1^i, t_2^i, \ldots, t_j^i\}$.

3.4. Multi-target Multi-camera Tracking

In the context of multi-camera tracking within a time interval, the subsequent procedure involves assigning a consistent ID to the same individual across various camera views. This task is performed through a series of steps, namely the Spatial-temporal Constraint, Constraint Hierarchical Clustering with Anchor feature, and Spatial-temporal Refinement.

3.4.1 Spatial-temporal Constraints

To assess the feasibility of merging two tracklets into the same trajectory, several constraints are employed:

1. **Frame Overlap Constraint for Same Camera**: This constraint verifies if two tracklets originate from the same camera and have intersecting frames. The underlying principle is that any object captured by the same camera but appearing in overlapping frames must be distinct entities.
2. **Frame Overlap Constraint for Non-intersecting Camera Pairs**: This extends the previous concept to tracklets from different cameras that do not share a visual field. The intersecting area of all camera pairs is calculated using homography matrices $\mathbf{H}$, which transform the ground plane of each camera into 3D world coordinates. This allows for the computation of intersection matrices, aiding in determining spatial overlap.
3. **Position Overlap Constraint**: For frames where tracklets exhibit overlap, this constraint evaluates whether the mean Euclidean distance between their corresponding world positions. If this distance between tracklets t_i and t_j, denoted as D_{ij}, falls below a predefined threshold, the tracklets are considered for merging. The mean distance is calculated based on the set of overlapping frames (O_{ij}):

$$D_{ij} = \frac{1}{|O_{ij}|} \sum_{k \in O_{ij}} \sqrt{(X_{i,k} - X_{j,k})^2 + (Y_{i,k} - Y_{j,k})^2}$$

   where $(X_{i,k}, Y_{i,k})$ and $(X_{j,k}, Y_{j,k})$ represent the world coordinates of tracklets t_i and t_j in frame k, respectively.
4. **Neighbor Constraint**: This constraint analyzes the potential movement of an object from the field of view of one camera to another without being captured by intermediate cameras. It leverages a pre-defined neighbor table specific to the scene being analyzed. This table encodes the adjacency information between camera views, indicating which cameras are considered directly connected.

These constraints ensure that only compatible tracklets, representing the same person's trajectory across different spaces and times, are merged, thereby enhancing the accuracy of object tracking across multiple cameras.

3.4.2 Constraint Hierarchical Clustering with Anchor Feature

This process involves comparing and grouping the anchor features of all tracklets across various camera perspectives

within specific time intervals. The methodology is outlined in Algorithm 2.

Algorithm 2 Constraint Hierarchical Clustering with Anchor Feature's Algorithm

Input: Set of all tracklets, denoted by $\mathcal{T}_{\omega_\iota} = \{\tau_1, \tau_2, \ldots, \tau_V\}$;
Distance threshold, denoted by T_{dist};
Function for checking constraint, denoted as CONSTRAINT

Output: Clusters of tracklets

1. **Initial Clustering:** This step creates the new clusters such that each cluster containing only t_j^i.
   $k \leftarrow 0$;
   for each τ_i in $\mathcal{T}_{\omega_\iota}$ **do**
   for each t_j^i in τ_i **do**
   $C_k \leftarrow \text{MAKECLUSTER}(t_j^i)$;
   $k \leftarrow k + 1$;
   end for
   end for
2. **Similarity Matrix Calculation:** Compute a similarity matrix, $\mathbf{M}$, where each element M_{ij} represents the similarity between tracklets t_i and t_j using Equation (2).
3. **Merging with Similarity and Constraint:**
   while there are still unmerged clusters **do**
   Find a pair of clusters C_a and C_b with the minimum distance $d(C_a, C_b)$ **and** satisfy CONSTRAINT(C_a, C_b);
   if $d(C_a, C_b) > T_{dist}$ **or** cannot find (C_a, C_b) that satisfies CONSTRAINT(C_a, C_b) **then**
   terminate the while loop;
   else
   Merge C_a and C_b into a new cluster $C_{a\cup b}$;
   Remove rows and columns corresponding to C_a and C_b from $\mathbf{M}$;
   Update all entries corresponding to the merged cluster $C_{a\cup b}$ to $\mathbf{M}$;
   end if
   end while
4. **return** The final set of clusters of tracklets;

This algorithm employs hierarchical clustering to iteratively merge tracklets. This approach starts by considering each tracklet as a separate cluster. The distance metric between the clusters is determined using Equation (2):

$$d(C_a, C_b) = \frac{1}{|\mathcal{A}|\,|\mathcal{B}|} \sum_{\mathbf{a}\in\mathcal{A}} \sum_{\mathbf{b}\in\mathcal{A}} \frac{\mathbf{a}\cdot\mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|}. \tag{2}$$

In this equation, C_a and C_b represent clusters. The sets $\mathcal{A}$ and $\mathcal{B}$ represent sets of anchor features associated with tracklets in clusters C_a and C_b, respectively. The vectors $\mathbf{a}$ and $\mathbf{b}$ refer to the anchor features. This distance computation measures the similarity between clusters based on the dot product of their anchor features normalized by their magnitudes.

In each iteration, the algorithm identifies the pair of clusters with the highest similarity based on a pre-computed similarity matrix, denoted by $\mathbf{M}$. This matrix stores the similarity score between every pair of clusters. It is dynamically updated as clusters merge during the process.

However, minimizing the cluster's distance alone is not sufficient to ensure meaningful clusters. To achieve this, the algorithm enforces two merging criteria:

1. **Distance Threshold:** The distance between merging clusters must be less than a predefined threshold, denoted by T_{dist}. This threshold establishes a baseline level of the cluster's distance required for cluster formation.
2. **Spatial-temporal Constraints Check:** Each potential cluster merge must also satisfy a set of Spatial-Temporal Constraints, as explained in Section 3.4.1. It prioritizes merging same-camera tracklets first (Constraint 1) before considering those from different cameras (Constraints 2, 3, and 4). This ensures a more meaningful clustering. These constraints leverage additional knowledge about the spatial-temporal data, such as camera view or motion patterns, to guide the clustering process and ensure merged clusters represent semantically similar objects.

When both similarity and constraint criteria are satisfied, the identified clusters, C_a and C_b, are merged into a new cluster, denoted as $C_{a\cup b}$. This merging process incorporates all anchor features between cluster C_a and C_b. Consequently, $C_{a\cup b}$ possesses a larger set of anchor features, enhancing its representational power. Thus, the set of anchor features of $C_{a\cup b}$ is $\mathcal{A}\cup\mathcal{B}$. Subsequently, the similarity matrix, $\mathbf{M}$, is then updated to reflect the newly formed cluster. This process continues iteratively until no further mergers satisfy both the similarity threshold and the constraints. In essence, the algorithm seeks to balance maximizing similarity with enforcing domain-specific knowledge through constraint checks.

3.4.3 Spatial-temporal Refinement

This work introduces the Spatial-temporal Refinement (STR) methodology to address the limitations of appearance-based features for person identification in multi-camera surveillance systems. These limitations arise from variations in lighting conditions and camera perspectives. This refinement step ensures that merged tracklets demonstrate spatial proximity within specific time intervals.

The core principle of STR revolves around evaluating the spatial overlap between tracklets. For each cluster of track-

lets, a positional table is constructed. This table records the world spatial coordinates of each tracklet within the cluster across every time frame. In the case where multiple spatial coordinates are registered for a single time frame, an aggregated position is computed by averaging these coordinates. This averaging process mitigates the influence of any potential positional discrepancies that might arise due to factors like occlusion or tracking inaccuracies.

Following the construction of positional tables, STR focuses on pairwise cluster comparisons. The tracklets from both clusters that share the same time frame and possess world coordinates located within a predefined intersection zone are extracted. This intersection zone is typically demarcated by the overlapping fields of view of the cameras capturing the tracklets. If this condition is satisfied, it signifies a high likelihood that the tracklets represent the same object across different camera perspectives. Consequently, STR merges these two clusters, resulting in a more robust and spatially-aware clustering outcome.

3.5. Cross Interval Synchronization

The Cross Interval Synchronization (CIS) module addresses the challenge of integrating object trajectories across distinct time windows captured by multiple cameras. This process aims to construct consistent trajectories for objects throughout the entire observation period. The process is summarized in Algorithm 3.

Algorithm 3 Cross Interval Synchronization's Algorithm

Input: Refining tracklet clusters, denoted by $\mathcal{C} = \{C_1, C_2, \ldots, C_M\}$;
Tracklet profile, denoted by $\Pi_T = \{P_1, P_2, \ldots, P_N\}$
Distance threshold, denoted by T_{dist};
Output: Updated tracklet profiles

$k \leftarrow N$;
for $i = 1$ to M **do**
  Find a pair of C_i and P_j with the minimum distance $d(C_i, P_j)$.
  if $d(C_i, P_j) < T_{dist}$ **then**
    Merge cluster C_i to profile P_j;
  else
    $k \leftarrow N + 1$;
    Create new profile P_k for cluster C_i;
  end if
end for
return $\Pi_T = \{P_1, P_2, \ldots, P_k\}$

The algorithm takes as input a set of refining tracklet clusters, denoted by $\mathcal{C} = \{C_1, C_2, \ldots, C_M\}$, a multi-camera tracklet profiles, denoted as $\Pi_T = \{P_1, P_2, \ldots, P_N\}$, and a distance threshold, denoted as T_{dist}. Each element P_k within the profile represents the trajectory of unique object identity k established through the synchronization of tracklets across all examined time intervals. It is important to note that the initial tracklet profile is derived from the first time window.

The CIS module computes a distance metric (refer to Equation (2)) between each tracklet cluster and the existing tracklet profile. If this distance between cluster C_i and profile P_j falls below the predefined threshold T_{dist}, the cluster is merged with the corresponding profile, effectively associating the detections within the cluster with the existing object track. Conversely, if no profile satisfies the distance criterion, a new tracklet profile is created to represent a new object instance.

4. Experiments

4.1. Datasets

The 2024 AI City Challenge's Track 1 dataset [25], is aimed at Multi-Camera People Tracking. The number of cameras has surged from 129 to around 1,300, and the number of individuals tracked has increased from 156 to about 3,400. Additionally, this enriched dataset includes 3D annotations and camera matrices, offering deeper insights. All video data is recorded in 1080p resolution at 30 frames per second, marking a notable advancement in the dataset's utility for precise multi-camera person tracking. We fine-tune the detection and re-identification model exclusively using the training set. The validation and testing sets, as divided in the AI City Challenge's Track 1 dataset, are utilized for experimentation, as demonstrated in Table 1 and Table 2, respectively.

4.2. Evaluation Metrics

In our research, we have chosen the Higher Order Tracking Accuracy (HOTA) metric as the primary measure to evaluate the performance of our proposed method. HOTA offers a well-balanced assessment, explicitly considering both detection accuracy and association performance within a single metric [16].

Furthermore, we conducted a self-assessment of our method using three supplementary metrics: Detection Accuracy (DetA), Association Accuracy (AssA), and Localisation Accuracy (LocA) [16].

4.3. Parameter Settings

Our proposed system leverages three pre-trained models, each fine-tuned for a specific task within the MTMC pipeline. This subsection details the parameter settings employed during the training and fine-tuning processes for these models, namely YOLOv8, OSNet, and ByteTrack.

YOLOv8 [12] serves as the person detection model. We fine-tuned the backbone of the pre-trained YOLOv8l model without freezing any layers, including the detection head.

Time Window	HOTA(%)	DetA(%)	AssA(%)	LocA(%)
30	39.99	65.76	26.66	89.46
60	55.27	68.65	46.67	89.61
120	65.69	70.58	61.65	89.70
300	**69.10**	**71.35**	**67.16**	89.75
600	68.01	70.09	66.23	**89.80**
900	67.03	68.60	65.69	89.71
1800	65.58	66.43	64.89	89.73
3600	64.24	64.81	63.83	89.75

Table 1. Proposed framework performance against time window sizes in terms of the number of frames

Rank	Team ID	Score
1	221	71.6520
2	79	67.2175
3	40	60.9261
4	142	60.8792
5	8	57.1445
6	**50 (Our)**	**51.0556**
7	5	45.1575
8	124	40.6202
9	162	40.3361
10	21	33.4879
...	...	...

Table 2. Top 10 rankings of AI City Challenge 2024

The batch size parameter was customized to 256, while all other parameters remained consistent with the default YOLOv8 settings.

OSNet [34–36] was fine-tuned as the ReID model. The ADAM optimizer was pre-defined with a learning rate of 3.5×10^{-4}. A single-step scheduler with a fixed step size of 20 was employed for the decaying learning rate.

ByteTrack [32] was utilized as the tracking model, operating on sequences of frames extracted from each video.

4.4. Experimental Results

This section presents the experimental evaluation of the proposed MTMC framework. We participated in Track 1 of the 2024 AI City Challenge, which focused on online MTMC system. We investigated the impact of time window size on the framework's performance using various window lengths as detailed in Table 1. Smaller window sizes represent a closer approximation to a true online approach.

Table 1 summarizes the performance of our framework across different time window sizes, measure by HOTA, DetA, AssA, and LocA. A time window size of 300 frames achieves the best overall performance for our system. This setting yields the highest score for HOTA (69.10%), DetA (71.35%), and AssA (67.16%).

From the experimental results, we select a time window size of 300 frames for evaluating the challenge's test set. As shown in Table 2, this configuration results in performance score of 51.0556 HOTA(%), ranking us 6th out of 17 participating team worldwide.

5. Discussions

This section discusses the influence of time window size on the performance of the proposed MTMC framework. We explore the relationship between windows size and evaluation metrics, as presented in Table 1.

The critical aspect of the proposed framework is determining an optimal window size for accurate tracking and ReID. The experiments show that the window size of 300 achieves the best performance. Deviations from this value lead to a decline in all metrics. This effect is attributed due to the role of anchor features, which are significantly impacted by the suitability of the time window.

A large window size introduces irrelevant noise into the system. This occur because of incorporation of features extracted under varying poses and lighting conditions across a longer time span. The large window interfere the process of feature aggregation, where the anchor feature is calculated from the sequence of ReID features.

Conversely, a window size that is too small fails to capture a comprehensive representation of the target appearance. This result in anchor features that lack the ability to effectively characterize the individual of each target.

6. Conclusions

In the presented research, we introduced a framework designed for real-time tracking of individuals across multiple camera feeds. The methodology is structured around a three-step process pipeline: Multi-object Tracking (MOT), Multi-target Multi-camera Tracking (MTMC), and Cross Interval Synchronization (CIS). The MOT phase is dedicated to extracting ReID anchor features and constructing local tracklets within a single camera's view. Following this, the MTMC step is engaged to establish connections between the set of tracklets across various cameras. This is achieved by applying spatial-temporal constraints, employing a constrained hierarchical clustering mechanism with anchor features, and conducting spatial-temporal refinement to enhance the tracking accuracy. The final phase, CIS, is implemented to maintain synchronization across different time intervals, ensuring that the multi-camera tracklet profiles are coherent and accurately represent the tracked individuals' movements. Our framework demonstrates its efficacy in tracking and identifying individuals across multiple cameras, as evidenced by the results obtained in the 2024 AI City Challenge. Here, our approach achieved a HOTA score of 51.0556% on the Track 1 dataset, securing the sixth place in the competition.

References

[1] Nir Aharon, Roy Orfaig, and Ben-Zion Bobrovsky. Bot-sort: Robust associations multi-pedestrian tracking, 2022. 2

[2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers, 2020. 2

[3] Long Chen, Haizhou Ai, Zijie Zhuang, and Chong Shang. Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In *2018 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2018. 2

[4] Ying Chen, Shixiong Xia, Jiaqi Zhao, Yong Zhou, Qiang Niu, Rui Yao, Dongjun Zhu, and Dongjingdian Liu. Restreid: Transformer block-based residual learning for person re-identification. *Pattern Recognition Letters*, 157:90–96, 2022. 2

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. 2

[6] Shuting He, Hao Luo, Pichao Wang, Fan Wang, Hao Li, and Wei Jiang. Transreid: Transformer-based object re-identification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15013–15022, 2021. 2

[7] Yunzhong Hou, Liang Zheng, Zhongdao Wang, and Shengjin Wang. Locality aware appearance metric for multi-target multi-camera tracking. *arXiv preprint arXiv:1911.12037*, 2019. 2

[8] Hung-Min Hsu, Tsung-Wei Huang, Gaoang Wang, Jiarui Cai, Zhichao Lei, and Jenq-Neng Hwang. Multi-camera tracking of vehicles based on deep features re-id and trajectory-based camera link models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2019. 2

[9] Hsiang-Wei Huang, Cheng-Yen Yang, Zhongyu Jiang, Pyong-Kun Kim, Kyoungoh Lee, Kwangju Kim, Samartha Ramkumar, Chaitanya Mullapudi, In-Su Jang, Chung-I Huang, and Jenq-Neng Hwang. Enhancing multi-camera people tracking with anchor-guided clustering and spatio-temporal consistency id re-assignment. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 5239–5249, 2023. 2

[10] Yuntae Jeon, Dai Quoc Tran, Minsoo Park, and Seunghee Park. Leveraging future trajectory prediction for multi-camera people tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 5398–5407, 2023. 2

[11] Hancheol Park Jeongho Kim, Wooksu Shin and Jongwon Baek. Addressing the occlusion problem in multi-camera people tracking with human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2023. 2

[12] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics YOLO, 2023. 7

[13] Jeongho Kim, Wooksu Shin, Hancheol Park, and Jongwon Baek. Addressing the occlusion problem in multi-camera people tracking with human pose estimation. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 5463–5469, 2023. 2

[14] Wei Li, Rui Zhao, Tong Xiao, and Xiaogang Wang. Deepreid: Deep filter pairing neural network for person re-identification. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 152–159, 2014. 2

[15] Zongyi Li, Runsheng Wang, He Li, Bohao Wei, Yuxuan Shi, Hefei Ling, Jiazhong Chen, Boyuan Liu, Zhongyang Li, and Hanqing Zheng. Hierarchical clustering and refinement for generalized multi-camera person tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5519–5528, 2023. 2

[16] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. Hota: A higher order metric for evaluating multi-object tracking. *International Journal of Computer Vision*, 129(2): 548–578, 2020. 2, 7

[17] Elena Luna, Juan C. SanMiguel, José M. Martínez, and Marcos Escudero-Viñolo. Online clustering-based multi-camera vehicle tracking in scenarios with overlapping fovs. *CoRR*, abs/2102.04091, 2021. 2

[18] Quang Qui-Vinh Nguyen, Huy Dinh-Anh Le, Truc Thi-Thanh Chau, Duc Trung Luu, Nhat Minh Chung, and Synh Viet-Uyen Ha. Multi-camera people tracking with mixture of realistic and synthetic knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5495–5505, 2023. 2

[19] Ergys Ristani and Carlo Tomasi. Features for multi-target multi-camera tracking and re-identification. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6036–6046, 2018. 2

[20] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *European conference on computer vision*, pages 17–35. Springer, 2016. 2

[21] Kyujin Shim, Kangwook Ko, Jubi Hwang, Hyunsung Jang, and Changick Kim. Fast online multi-target multi-camera tracking for vehicles. *Applied Intelligence*, 53(23):28994–29004, 2023. 2

[22] Francesco Solera, Simone Calderara, and Rita Cucchiara. Learning to divide and conquer for online multi-target tracking. *CoRR*, abs/1509.03956, 2015. 2

[23] Andreas Specker and Jürgen Beyerer. Reidtrack: Reid-only multi-target multi-camera tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5441–5451, 2023. 2

[24] Gábor Szűcs, Regő Borsodi, and Dávid Papp. Multi-camera trajectory matching based on hierarchical clustering and constraints. *Multimedia Tools and Applications*, pages 1–24, 2023. 2

[25] Shuo Wang, David C. Anastasiu, Zheng Tang, Ming-Ching Chang, Yue Yao, Liang Zheng, Mohammed Shaiqur Rahman, Meenakshi S. Arya, Anuj Sharma, Pranamesh

Chakraborty, Sanjita Prajapati, Quan Kong, Norimasa Kobori, Munkhjargal Gochoo, Munkh-Erdene Otgonbold, Ganzorig Batnasan, Fady Alnajjar, Ping-Yang Chen, Jun-Wei Hsieh, Xunlei Wu, Sameer Satish Pusegaonkar, Yizhou Wang, Sujit Biswas, and Rama Chellappa. The 8th AI City Challenge. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2024. 7

[26] Tao Wang, Hong Liu, Pinhao Song, Tianyu Guo, and Wei Shi. Pose-guided feature disentangling for occluded person re-identification based on transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2540–2549, 2022. 2

[27] Longhui Wei, Shiliang Zhang, Wen Gao, and Qi Tian. Person transfer gan to bridge domain gap for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[28] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric, 2017. 2

[29] Wenjie Yang, Zhenyu Xie, Yaoming Wang, Yang Zhang, Xiao Ma, and Bing Hao. Integrating appearance and spatial-temporal information for multi-camera people tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5259–5268, 2023. 2

[30] Ju Hong Yoon, Chang-Ryeol Lee, Ming-Hsuan Yang, and Kuk-Jin Yoon. Online multi-object tracking via structural constraint event aggregation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1392–1400, 2016. 2

[31] Jianming Zhang, Liliana Lo Presti, and Stan Sclaroff. Online multi-person tracking by tracker hierarchy. In *2012 IEEE Ninth International Conference on Advanced Video and Signal-Based Surveillance*, pages 379–385, 2012. 2

[32] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. Bytetrack: Multi-object tracking by associating every detection box, 2022. 2, 8

[33] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. Scalable person re-identification: A benchmark. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1116–1124, 2015. 2

[34] Kaiyang Zhou and Tao Xiang. Torchreid: A library for deep learning person re-identification in pytorch. *arXiv preprint arXiv:1910.10093*, 2019. 2, 8

[35] Kaiyang Zhou, Yongxin Yang, Andrea Cavallaro, and Tao Xiang. Omni-scale feature learning for person re-identification. In *ICCV*, 2019. 2

[36] Kaiyang Zhou, Yongxin Yang, Andrea Cavallaro, and Tao Xiang. Learning generalisable omni-scale representations for person re-identification. *TPAMI*, 2021. 2, 8