Multi-Frame Attention with Feature-Level Warping for Drone Crowd Tracking

Takanori Asanomi, Kazuya Nishimura, Ryoma Bise
Kyushu University, Fukuoka, Japan
bise@ait.kyushu-u.ac.jp

Abstract

Drone crowd tracking has various applications such as crowd management and video surveillance. Unlike in general multi-object tracking, the size of the objects to be tracked are small, and the ground truth is given by a point-level annotation, which has no region information. This causes the lack of discriminative features for finding the same objects from many similar objects. Thus, similarity-based tracking techniques, which are widely used for multi-object tracking with bounding-box, are difficult to use. To deal with this problem, we take into account the temporal context of the local area. To aggregate temporal context in a local area, we propose a multi-frame attention with feature-level warping. The feature-level warping can align the features of the same object in multiple frames, and then multi-frame attention can effectively aggregate the temporal context from the warped features. The experimental results show the effectiveness of our method. Our method outperformed the state-of-the-art method in DroneCrowd dataset. The code is publicly available in `https://github.com/asanomitakanori/mfa-feature-warping`.

1. Introduction

Automatic multiple-person tracking from videos taken by cameras mounted on drones (uncrewed aerial vehicles), called drone crowd tracking, has a wide range of applications, such as video surveillance and crowd management. In the video, people move their position while the background fluctuates due to the drone's motion. The task aims to find an object's location and associate the same objects in the video (Figure 1). Different from general multi-object tracking that uses a bounding box for annotation, point-level annotation is used in the public dataset [36] since the object size tends to be small.

The most significant difference between drone crowd tracking tasks and the other multi-object tracking with bounding-box (*e.g.*, tracking in a video captured by a surveillance camera) is that the object size is much small. Thus the ground truth of an object location is given by point-level annotation as shown in Figure 1. This causes the lack of discriminative features for finding the same objects from many similar objects. For example in Figure 1 (b) (Cloudy, Small, Crowded), the appearance of a person appears as a white dot, and different persons have similar appearances. Thus, similarity-based tracking techniques, widely used for multi-object tracking with bounding-box, are difficult to use. The small size of objects also causes other difficulty; the difference in the appearance of an object from the background object is not significant. For example, an appearance of a person is similar to the other objects. This causes inconsistent detection results with time, *i.e.*, an object is detected sometimes, but not at the other time.

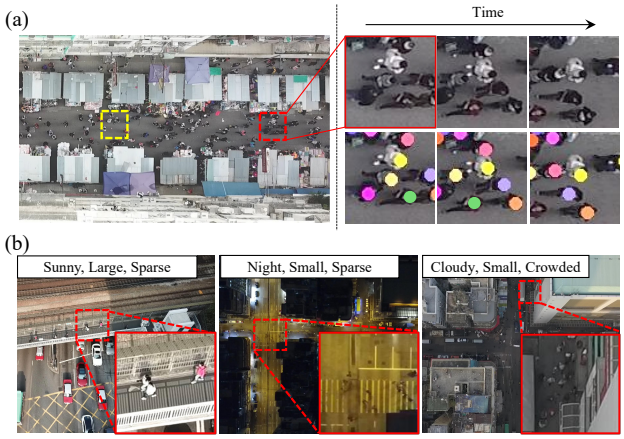

Figure 1. Example images in drone crowd tracking. (a) Example of the entire image (**Left**), enlarged sequence images at the red box (**Right-Top**), and ground-truth of tracking (**Right-Bottom:**): dots indicate target object, and the same object has the same color. (b) Example images under various conditions.

The second significant difference is that the spatial context far from objects is ineffective. For example, the yellow box in Figure 1 (a) is not related to how many people are in the red box region. In detection tasks of large objects and video analysis, the spatial context is an important clue. Thus vision transformer has been often used to aggregate the spatial context by self-attention. However, this may not be effective in drone crowd tracking tasks.

To track multiple small objects, point-based tracking

methods have been proposed. For example, STNNet [36] estimates the location and motion of objects in a multi-task manner from two continuous frames. STNNet has achieved the best performance (the state-of-the-art) in the DroneCrowd dataset [36], which contains point-level annotations as ground truth instead of bounding boxes. Point-level annotation is often used for cell tracking, which tracks small cells in microscopy images. Motion and position map [13] has been proposed to represent localization and motion given two frames simultaneously and achieved the best performance in a cell tracking dataset [15]. These methods use only two frames for motion and location estimation.

In drone crowd tracking tasks, the temporal context in the neighbor position is essential. When it is difficult to detect objects from a single frame accurately, the detection results may not be consistent in multiple frames. However, if we check such objects in several frames, we can identify persons' consistency in time.

Self-attention, which has been used in transformer, is a promising technology to aggregate the temporal context. The transformer has been widely used for many vision tasks. Almost methods use self-attention for aggregating wide-range contexts and spatial dependencies in a single input image, in which small patches in an image are inputted to the self-attention module and extracts the features using the correlation of inputted patches. Some methods use the spatial-temporal context for tracking. Zhou *et al.* [46] proposed global association with transformer. This method first detects bounding boxes in multiple frames and then associates the detected object in multiple frames using transformer, which estimates the apparent similarity of detected bounding boxes. However, the features of point-level objects are not enough to identify the same object.

In this paper, we propose multi-frame attention with feature-level warping to aggregate temporal information in multiple frames. Given feature maps extracted from multiple frames, the proposed method aggregates the temporal context using temporal self-attention. This assumes that the attention is calculated from the same position of the feature maps in multiple frames. However, the object positions change in multiple frames due to their motion. Therefore, we introduce feature map warping modules before multi-frame attention to align the features of the same object in multiple frames. Next, the warped feature maps are inputted into the multi-frame attention module to aggregate the temporal context. Then, the backward warping module warps the extracted features again to obtain the original positions for each frame. It can output consistent detection results in multiple frames using temporal context. This makes the tracking accuracy improve. The contributions of this paper are summarized as follows:

- We proposed multi-frame attention with feature-level warping. This method can align the object features in the map and aggregate the image features from multiple frames by multi-frame attention.
- Using DroneCrowd dataset [36], the proposed method achieved the best performance than comparative methods (containing the state-of-the-art method, STNNet).

2. Related work

2.1. Crowd tracking

Human counting in crowds has been well studied, and many datasets for it are publicly available [7, 11]. This task aims to estimate the number of persons in a single image. Many counting methods take the density-map estimation approach, which can count people in an image but cannot localize and track individuals.

Recently, the DroneCrowd dataset [36] was published to develop a method that tracks tiny images of humans in video captured from a bird's eye view by a drone. The main differences of the task of DroneCrowd from general multi-object tracking (MOT) are three-folds; 1) because of the small size of the objects in the images, DroneCrowd uses point-level annotation, which has no region information, instead of the bounding boxes used in MOT. As well, 2) the number density of the humans in the images is high, and 3) the video shows various situations.

STNNet [36] was designed to track tiny objects for this dataset. It estimates the location and motion of objects in a multi-task manner from two successive frames. It currently has state-of-the-art performance on the DroneCrowd [36].

2.2. Multi-object tracking (MOT)

Many MOT methods have been proposed, which use active contours [18, 35, 40, 47], particle filters [25, 32] or association [41, 30, 20, 1]. A recent trend is tracking-by-detection [39, 16, 4, 9, 28]. It relies on the ability of bounding-box detectors [37, 29, 31, 20], and associates the same objects between frames based on the similarity in appearance of the detected bounding boxes. Here, the bounding box detector is unsuitable for point-level object detection, as each object lacks discriminative features due to its small size.

Point-level annotation has been used for detecting and tracking tiny objects, such as in key-point detection for pose estimation, cell tracking, and drone crowd tracking. Point-level methods [12, 13, 14, 44, 45, 39] estimates a heatmap, in which the annotated points become peaks in the heatmap, instead of bounding boxes. Then, the tracking methods associate the same object based on their position and motion estimation [12, 13, 36]. Almost all methods use the context of two successive frames for object detection and motion estimation. The temporal context in multiple frames is important information for tracking.

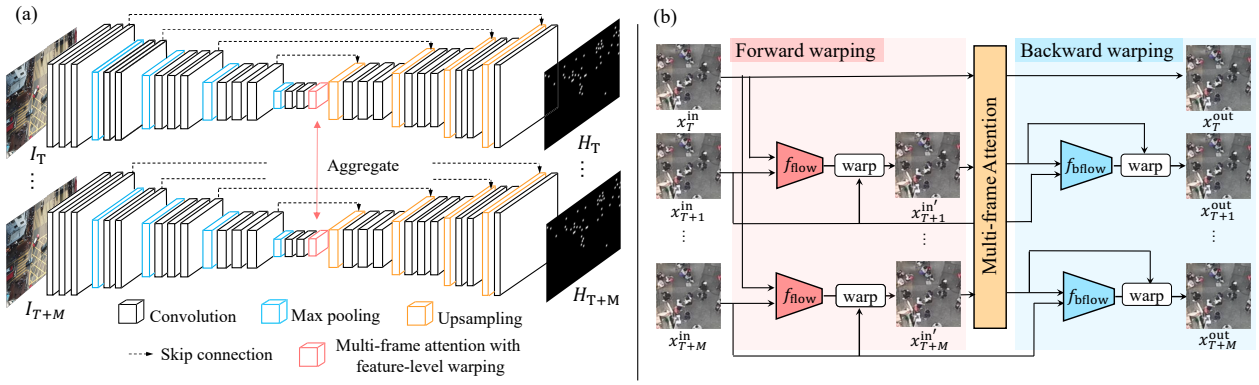

Figure 2. Overview of our method. (a) Entire network structure, (b) Multi-frame attention with feature-level warping. The module consists of three modules: forward warping, multi-frame attention, and backward warping. Forward warping aligns features according to frame T, and multi-frame attention aggregate temporal context. Since the extracted feature maps are aligned to T, the feature maps are warped into features that represent original positions by backward warping.

2.3. Transformer for video analysis

The transformer has been widely used for many tasks, including video analysis [24, 8, 5, 2, 23, 34]. For example, TimeSformer [5] and ViVit [2] have shown that temporal attention and spatial attention are effective in video classification tasks. The Video Swin transformer [23] introduces the Swin transformer [22] for video classification. The spatial-temporal attention, which computes the attention combinations of patches in multiple frames, is computationally expensive. To reduce the computational cost, the deformable video transformer [34] leverages motion cues to determine which patches to compare; it applies a handcrafted deformation to the original input images to obtain motion cues. These methods are designed for video classification tasks, which require spatial-temporal information in an entire video to be aggregated.

Some methods use transformer for multi-object tracking. TrackFormer [24] has the cascade structure of DETR [6], which is a detection method using a transformer. Trackformer extracts features at time t using the spatial context in the encoder, and these features are used as queries in the decoder at the next frame. This method is designed for tracking bounding-box-based objects. The self-attention in the encoder uses only the spatial context in each frame. The aim of using features extracted from the previous frame as queries in the decoder is for association; *i.e.*, the transformer decoder associates the objects based on object similarities of bounding-box regions. Transformer tracking [8] uses an attention-based feature fusion network, which combines template and search features by attention. These transformer-based tracking methods focus on aggregating spatial-temporal information to detect bounding boxes and extract features that represent the similarity of the same objects. Spatial information far from the object may not be useful for tracking tiny objects from a bird's eye view. Instead, our method uses temporal information at the same place by aligning feature maps in multiple frames.

3. Proposed Method

3.1. Overview

Figure 2 is an overview of the proposed method. Given a set of M images $\{\boldsymbol{I}_T, ..., \boldsymbol{I}_{T+M}\}(\boldsymbol{I}_{T+i} \in \mathbb{R}^{w \times h})$, where w and h indicate the width and height of the image, the network simultaneously estimates a set of the heatmaps $\{\boldsymbol{H}_T, ..., \boldsymbol{H}_{T+M}\}(\boldsymbol{H}_{T+i} \in \mathbb{R}^{w \times h})$ [12], which represents the object positions.

The entire network consists of Siamese networks, which have an encoder and a decoder, such as U-net, and produces the position heatmaps as shown in Figure 2 (a). We introduce each network to three modules, forward warping, multi-frame attention, and backward warping modules (Figure 2 (b)), in order to aggregate the temporal information in multi-frames. Siamese U-nets can interchange image features in multiple frames via these modules. Using estimated positions, objects can be tracked in a video.

3.2. Forward warping

The encoders of each network extract the image features for the object positions in each frame. To effectively use image features in different frames, we introduce a multi-frame attention module that allows the Siamese networks to interchange the extracted features in multiple frames of the same position. This assumes that the features of the same object are at the same position in each feature map. However, object positions usually change in multiple frames due to their motion, and thus the attention may not work appropriately. To deal with this problem, we introduce a forward warping module before the multi-frame attention module to align the image features in multiple frames.

Figure 2 (b) shows the forward warping module. It incorporates forward warping modules in $M+1$ Siamese networks; each module is inserted after the convolutional layers of the encoder, as shown in Figure 2. Each network f_{flow} estimates the feature-level flow in the same manner as VoxelMorph [3], for feature map alignment.

Let us denote the i-th output feature map for the j-th layer as $\boldsymbol{x}^{\text{in}}_{T+i} = f^{(j-1)}(\boldsymbol{I}_{T+i}), (i = 0, ..., M)$, these features are aligned into to one of them $\boldsymbol{x}^{\text{in}}_{T}$. Precisely, given two maps $(\boldsymbol{x}^{\text{in}}_{T}, \boldsymbol{x}^{\text{in}}_{T+i})$, f_{flow} estimates the displacement vector map $\tau_{T+i\to T}$ which warps the input feature map $\boldsymbol{x}^{\text{in}}_{T+i}$ into

$$\boldsymbol{x}^{\text{in}'}_{T+i} = \boldsymbol{x}^{\text{in}}_{T+i} \circ \tau_{T+i\to T} \quad (i = 1, \ldots, M), \tag{1}$$

where $\circ$ indicates a warping operation by $\tau_{T+i\to T}$, which specifies the vector offset from $\boldsymbol{x}^{\text{in}}_{T+i}$ to $\boldsymbol{x}^{\text{in}}_{T}$ for each pixel. The warping operation is similar to the one used in image registration methods, such as VoxelMorph [3]. The difference is that our warping is performed on every channel of a feature map, whereas the standard warping operation is performed on an image. It is expected that warping will align the features of an object to the same position.

This forward warping performs for all pairs of features in the multiple frames. The output of the module is a set of warped features $\{\boldsymbol{x}^{\text{in}'}_{T}, \boldsymbol{x}^{\text{in}'}_{T+1}, \ldots, \boldsymbol{x}^{\text{in}'}_{T+M}\}$, where $\boldsymbol{x}^{\text{in}'}_{T} = \boldsymbol{x}^{\text{in}}_{T}$. The warped feature maps are inputted to the multi-frame attention module.

3.3. Multi-frame attention

Fig. 3 is an overview of the multi-frame attention module. The warped features $\{\boldsymbol{x}^{\text{in}'}_{T}, \boldsymbol{x}^{\text{in}'}_{T+1}, \ldots, \boldsymbol{x}^{\text{in}'}_{T+M}\}$ are inputted into the multi-frame attention module to aggregate the temporal features among multiple frames. The attention is calculated from the same position of the feature maps in multiple frames. The same position of multiple frames is expected to have the same object features since the feature maps were warped by forward warping.

Let us consider a case in which an object can sometimes be clearly observed, but its appearance is unclear in some of the other frames due to the shadows of other objects. In such a case, it can be expected that the multi-frame attention module can extract the object features using other frames. This module contributes to reducing undetected objects and consistent tracking.

In contrast to standard vision transformers [10], in which cropped patches of an entire image are inputted to the transformer network (*i.e.*, self-information of a single image is used), our multi-frame attention aggregates a set of inputted images among multiple frames. This module allows the Siamese U-nets to interchange the extracted features with each other. In addition, to reduce the number of parameters and effectively extract the local features of each batch, we

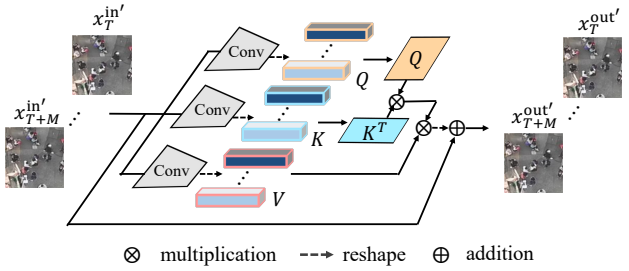

Figure 3. Multi-frame-attention. This module aggregates multiple frames information by calculating from the same position of the feature maps in multiple frames.

embed tokens in a key, query, and value by using convolution [38] instead of a fully connected layer.

In the multi-frame attention module, $\boldsymbol{x}^{\text{in}'}_{T+i} (i = 0, ..., M)$ is first converted into a query ($\boldsymbol{q}_{T+i}$), key ($\boldsymbol{k}_{T+i}$), and value ($\boldsymbol{v}_{T+i}$) by convolution, and the matrices are then flattened into vectors $\text{vec}(\boldsymbol{q}_{T+i})$, $\text{vec}(\boldsymbol{k}_{T+i})$, and $\text{vec}(\boldsymbol{v}_{T+i})$. Let us denote the matrices consisting of a set of queries, keys, and values of the input images as $Q = [\text{vec}(\boldsymbol{q}_T)|...|\text{vec}(\boldsymbol{q}_{T+M})]^T \in \mathbb{R}^{(M+1)\times C}$, $K = [\text{vec}(\boldsymbol{k}_T)|...|\text{vec}(\boldsymbol{k}_{T+M})]^T \in \mathbb{R}^{(M+1)\times C}$, and $V = [\text{vec}(\boldsymbol{v}_T)|...|\text{vec}(\boldsymbol{v}_{T+M})]^T \in \mathbb{R}^{(M+1)\times C}$, respectively, where $\text{vec}(\cdot)$ is the flattening operator that flattens a matrix into a vector, $C = w^{\text{in}}h^{\text{in}}ch^{\text{in}}$ is the dimension of the flattened vector. w^{in}, h^{in}, and ch^{in} are the width, height, and channel of $\boldsymbol{x}^{\text{in}'}_{T+i}$. The multi-input attention output is defined as:

$$\boldsymbol{x}^{\text{out}'}_{T+i} = \boldsymbol{x}^{\text{in}'}_{T+i} + \text{reshape}(\text{Attention}(Q, K, V)_{T+i}), \tag{2}$$

$$\text{Attention}(Q, K, V) = \text{softmax}(QK^{\top})V, \tag{3}$$

where $\text{softmax}(\cdot)$ performs the softmax operator on each row vector individually, and $\text{reshape}(\cdot)$ reshapes the input vector back into a matrix whose shape is the same as $\boldsymbol{x}^{\text{in}'}_{T+i}$. $\text{Attention}(Q, K, V)_{T+i}$ is the $(T+i)$-th vector of $\text{Attention}(Q, K, V)$. Here, in the multi-frame attention module, the information of all the input images extracted by the Siamese networks is interchanged and aggregated. To train this network, we use the sum of the loss functions for the backbone network (refer to the section of the backbone network). The extracted features $\{\boldsymbol{x}^{\text{out}'}_{T}, \boldsymbol{x}^{\text{out}'}_{T+1}, \ldots, \boldsymbol{x}^{\text{out}'}_{T+M}\}$ are then inputted to the next layer of each Siamese U-net.

Standard vision transformers, such as ViT [10], have quadratic computation complexity relative to the input image size $N = w \times h$, *i.e.*, $O(N^2 \times M)$, where M indicates the number of time frames. In contrast, multi-frame attention has linear computational complexity to the input image size and only quadratic computation to the frame number, $O(N \times M^2)$, which is much smaller than the standard one because $M << N$.

3.4. Backward warping

The decoder in a Siamese U-net decodes the extracted features to the heatmap, which has the same size as the original image. However, the forward warping module deforms the features (*i.e.*, the object positions in the warped feature map $\boldsymbol{x}^{\text{out}'}_{T+i}$ is misaligned to those of the original feature map $\boldsymbol{x}^{\text{in}}_{T+i}$). We thus introduce the backward warping module that warps the features back to their original position. The network structure and warping process are similar to the forward warping module. Precisely, given two maps $(\boldsymbol{x}^{\text{in}}_{T+i}, \boldsymbol{x}^{\text{out}'}_{T+i})$, f_{bflow} estimates the $\tau_{T\rightarrow T+i}$ which warps the input feature map $\boldsymbol{x}^{\text{out}'}_{T+i}$ into

$$\boldsymbol{x}^{\text{out}}_{T+i} = \boldsymbol{x}^{\text{out}'}_{T+i} \circ \tau_{T\rightarrow T+i} \quad (i = 1, \ldots, M), \tag{4}$$

where $\tau_{T\rightarrow T+i}$ specifies the vector offset from $\boldsymbol{x}^{\text{out}'}_{T+i}$ to $\boldsymbol{x}^{\text{out}'}_{T}$ for each pixel. The warped feature maps $\{\boldsymbol{x}^{\text{out}}_{T}, \boldsymbol{x}^{\text{out}}_{T+1}, \ldots, \boldsymbol{x}^{\text{out}}_{T+M}\}$, where $\boldsymbol{x}^{\text{out}}_{T} = \boldsymbol{x}^{\text{out}'}_{T}$, are upsampled by the decoder and the outputs are heatmaps, in which the local peaks indicates object positions. It is expected that this feature map will be warped to its original position.

3.5. Backbone models

We incorporated the proposed method into two backbone models: Heatmap [12], widely used in point-level object detection for keypoint detection in pose estimation and cell detection, and MPM [13], which is one of the state-of-art methods for cell tracking. To make this paper self-contained, we briefly describe these two methods below.

Heatmap [12]: This method estimates the heatmap of object positions by a U-net; each peak in the heatmap indicates the position of an object. From a set of annotated pixel-level object positions for an image, the ground truth of the heatmap is generated so that an object position becomes a peak with a Gaussian distribution in the map, as shown in Fig. 2 (a). The detection network (U-net) is trained using the mean square loss (MSE) $L_{det} = MSE(\boldsymbol{H}_i, \hat{\boldsymbol{H}}_i)$ between the estimation results $\hat{\boldsymbol{H}}_i$ and the ground truth of the heatmap $\boldsymbol{H}_i$.

MPM [13]: This method estimates the position and motion map (MPM), which represents the position and moving direction between successive frames. The motion vector of each object is encoded on the pixels of the object's center position, and the distribution of the magnitudes of the vector represents the heat-map of the center positions, where the local maximum of the heatmap indicates the center position. Let us denote the ground truth and the estimation of a MPM as $\boldsymbol{H}$ and $\hat{\boldsymbol{H}}$, respectively. The loss function is defined as:

$$L_{MPM} = \frac{1}{n}\sum_{i=1}^{n}(||\boldsymbol{H}_i - \hat{\boldsymbol{H}}_i||_2^2 + (||\boldsymbol{H}_i||_2 - ||\hat{\boldsymbol{H}}_i||_2)^2), \tag{5}$$

where n is the number of training data, the first term is the squared error, and the second term is the squared error between the magnitudes of $\boldsymbol{H}$ and $\hat{\boldsymbol{H}}$.

In the traditional method, the network estimates the heatmaps or MPM individually in each frame. In contrast, our method consists of Siamese U-nets for multiple frames and the proposed warping and multi-frame attention modules that share features. Our method can be applied to a network that has an encoder and decoder, such as U-net, and estimates a position heatmap for each local frame. In our network, backbone networks are arranged in parallel and interchange their extracted features using multi-frame attention with feature-level warping.

3.6. Tracking by association

For a fair comparison, we applied the same algorithm with STNNet [36], which is a state of the art method. After detecting all objects in each frame, we applied the min-cost flow method[27], which optimizes the association.

4. Experiments

4.1. Dataset and experimental setup

We used the DroneCrowd dataset[36] in our experiments. The dataset contains time-lapse video sequences taken by drones. As shown in Figure 1, the images show streets crowded with people, and there are no significant differences in the appearances of people. Since a camera is attached to a flying drone, the background fluctuates as the persons below move from place to place. The ground truth of the object position is a point-level annotation $(ObjectID, frame, x, y)$.

In each image sequence, images were captured at 25 frames per second (FPS) with a resolution of 1920 $\times$ 1080 pixels. In this dataset, a video has three types of attributes, as follows: (1) *illumination*: three categories of illumination conditions are *Sunny*, *Cloudy* and *Night*; (2) *Object scale*: two categories of scales are *Large* (the diameter of objects $>$ 15 pixels) and *Small* (the diameter of objects $\leq$ 15 pixels); (3) *Density*: based on the average number of objects in each frame, there are two density levels, *e.g.*, *Crowded* (the average number of objects in each frame is larger than 150), and *Sparse* (the average number of objects in each frame is less than 150). Figure 1 (b) shows the examples images under different conditions. The average number of objects is 144.8, and more than 20 thousand head trajectories of people are annotated with more than 4.8 million head points in individual frames. The training, validation, and test sequences number 82, 30 and 30. This is the same setup as that of the state-of-the-art method [36].

Methods	L-mAP	L-AP$_{10}$	L-AP$_{15}$	L-AP$_{20}$
MCNN [43]	9.05	9.81	11.81	12.83
CAN [21]	11.12	8.94	15.22	18.27
CSRNet [19]	14.40	15.13	19.77	21.16
DM-Count [42]	18.17	17.90	25.32	27.59
STNNet [36]	40.45	42.75	50.98	**55.77**
Heatmap [12]	29.26±2.23	30.85	35.17	38.3
Heatmap + Ours	**32.19**±1.01	**34.49**	**38.6**	**41.4**
MPM [13]	41.07±0.4	44.7	48.92	51.22
MPM + Ours	**43.43**±1.98	**47.14**	**51.58**	**54.02**

Table 1. Localization performances in comparison using DroneCrowd; the average L-mAP, and L-AP at each threshold (L-AP$_{10}$, L-AP$_{15}$ and L-AP$_{20}$).

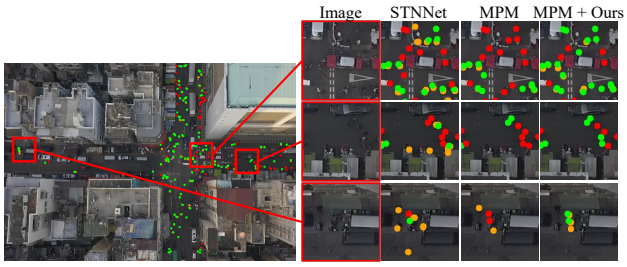

Figure 4. Examples of detection results in test images. Left: the entire image of MPM [13] + Ours detection results; Right: enlarged images of detection results from MPM [13] and MPM + Ours. Green: true positive; Red: false negative; Orange: false positive.

4.2. Experimental setup

We implemented our method by using PyTorch [26]. To train our network, we used the ADAM optimizer [17] with a learning rate of 10^{-3}, epoch $= 30$, mini-batch size $= 24$. The detection points were obtained by the thresholding of heatmap (threshold $= 0.1$). We set M (the number of aggregating frames) to 3 due to the limited memory of the GPU (NVIDIA GeForce 3090 GPU). In all experiments, for a fair comparison, we trained the networks three times with different seeds and computed the averages and standard deviation in terms of the localization and tracking performance, following [10].

We compared our method with seven methods: MCNN[43], which uses the image features from multi-scale images extracted by each expert to capture the variations in object size; CAN [21], which exploits multi-scale contextual information in density maps, and as a result, achieved the best performance for counting in both cloudy and crowded cases. DM-Count [42], which simply estimates a density map generated using 2D Gaussian; STNNet [36], which is designed for drone crowd tracking by using the neighboring context loss to guide the association, (it has achieved the state-of-the-art performance on the DroneCrowd dataset); Heamap [12], which, as described above, is widely used in object detection for point-level annotation and is one of our backbone methods; MPM [13], which simultaneously represents the position and motion of objects for tracking tiny objects, is our other backbone method. For a fair comparison, after each of the comparative methods detect objects, we used the same method for tracking, in which the detected objects were associated using min cost flow [36].

4.3. Crowd localization performance

Localization, which is to detect all people's locations in an image accurately, is a critical task in object tracking. We evaluated crowd localization performance using the L-mAP score following the paper that proposed DroneCrowd dataset [36]. The estimated object's points are determined by thresholding the heatmaps. L-mAP is the mean of the L-APs at various distance thresholds (1,2,. . .,25 pixels). A bigger L-mAP is the better.

Table 1 shows the L-mAP and L-AP scores with three specific distance thresholds (10, 15, and 20 pixels). Note that the performance of the comparative methods except the two backbone methods (heatmap [12] and MPM [13]) are taken from [36]. Our methods (Heatmap+Ours and MPM+Ours) improved all metrics compared with the backbone methods (heatmap [12] and MPM [13]). Moreover, MPM+Ours outperformed the state-of-the-art method (STNNet [36]). Note that when the threshold is large, *e.g.*, L-AP$_{20}$, a false positive can be associated with a ground truth (counted as a true positive). Thus, the better performance of L-AP with small distance indicates that the localization is more accurate. In particular, the improvement of the L-AP with the small distance threshold (10) from STNNet was significant (+4.39). This indicates that the localization of our method is more accurate. Fig.4 shows examples of detection results. In the results estimated by STNNet, there are many false positives (orange) and false negatives (red). MPM reduced the false positives compared to the STNNet. Furthermore, our method (MPM + Ours) significantly reduced the false negatives.

4.4. Crowd tracking performance

The goal of drone crowd tracking is to recover the trajectories of people in a video sequence. Following the paper that proposed the DroneCrowd dataset [36], we evaluated tracking performance by using the T-mAP score [36]. This metric is computed on the basis of the estimated trajectories of head points with confidence scores. Specifically, we sorted the trajectories (tracklets), formed by the locations with the same identity, based on the average confidence of their detection results. A tracklet is considered to be correct if the matched ratio between the predictions and ground-truth tracklets is more significant than a threshold. Then, the average precision with changing the confidence is com-

Methods	T-mAP	T-AP$_{0.10}$	T-AP$_{0.15}$	T-AP$_{0.20}$
MCNN [43]	9.16	11.47	9.65	6.36
CAN [21]	4.39	6.97	4.72	1.48
CSRNet [19]	12.15	17.34	12.85	6.26
DM-Count [42]	17.01	22.38	18.34	10.29
STNNet [36]	32.50	35.45	33.99	28.05
Heatmap [12]	31.44±0.20	34.5	33.02	26.80
Heatmap + Ours	**33.25±0.28**	**36.12**	**34.81**	**28.83**
MPM [13]	41.91±0.84	44.89	**43.45**	37.38
MPM + Ours	**42.08±1.18**	**44.99**	43.39	**37.67**

Table 2. Tracking performances on DroneCrowd; average T-mAP, and T-AP at each threshold (T-AP$_{0.10}$, T-AP$_{0.15}$ and T-AP$_{0.20}$)).

puted as T-AP. Following [36], we used three thresholds (0.10, 0.15, and 0.20). The T-mAP scores are the means of T-AP using different thresholds (*i.e.*, T-AP$_{0.10}$, T-AP$_{0.15}$, and T-AP$_{0.20}$). Note that the performance of the comparative methods, except the two backbone methods, are taken from [36].

Table 2 shows the crowd-tracking performance of the comparative methods. Both of our methods (Heatmap+Ours and MPM+Ours) improved T-mAP from the baseline methods. Moreover, they outperformed the state-of-the-art method (STNNet [36]). Here, we discuss why Heatmap+Ours was better than STNNet while its localization performance was worse. The detection results by STNNet tend to contain false positives than false negatives, where L-AP with the large distance threshold more penalizes the false negatives than false positives since a false positive may be associated to a ground truth. In addition, the detection results of the same object tended to be inconsistent with time. This makes the tracking performance worse. In contrast, Heatmap+Ours produces consistent detection results for the same object, *i.e.*, the same object was continuously detected or not detected over time. Therefore, even though the total detection performance of our method was worse, its tracking performance was better than that of STNNet.

In particular, MPM+Ours significantly improved the tracking performance T-mAP from the state-of-the-art method STNNet (+9.58). The features contain not only the appearance features but also the motion features of each object in MPM. Since neighbor objects may have different motions, this can be considered to have a good effect on the multi-frame attention's aggregating the temporal features of the same object, and thus it improved performance a lot.

Figure 5 shows examples of tracking results by STNNet, MPM, and MPM + Ours. STNNet includes false negative detections in every frame. The detection results are not consistent in time. For example in Figure 5 (STNNet, the 3rd row), the person in the white circles was not detected (false negatives), and the false negatives occur at different persons. Therefore, an ID switching error occurs at the third frame (white rectangle). In the results of MPM, also has

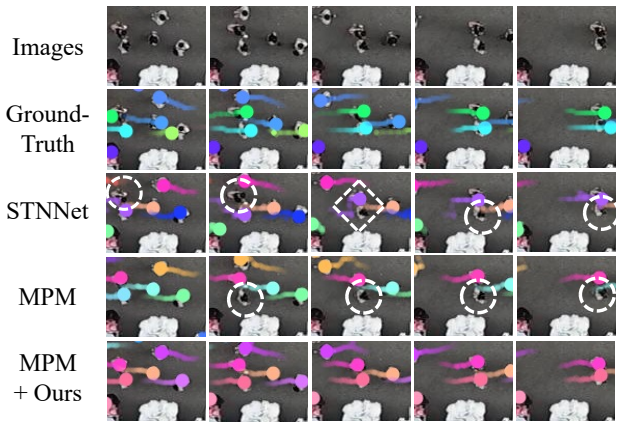

Figure 5. Example of tracking result. White dotted circles indicate false negative detection errors, and a white rectangle indicates a switching error.

Methods	Mfa	Fw	Bw	L-mAP	T-mAP
Heatmap [12]				29.26±2.23	31.44±0.2
Heatmap + SA				21.59±12.52	17.45±11.86
Heatmap + DSTA				31.57±0.39	32.91±0.86
Heatmap + Ours	✓			30.01±1.35	31.82±2.25
Heatmap + Ours	✓	✓		31.97±0.9	32.81±0.67
Heatmap + Ours	✓	✓	✓	**32.19±1.01**	**33.25±0.28**

Table 3. L-mAP and T-mAP in ablation study. We report mean and standard deviation of all metrics over three runs with different seeds. 'SA' is self-attention. 'DSTA' is divided space-time attention. 'Mfa' is multi-frame attention. 'Fw' is forward warping. 'Bw' is backward warping.

false negatives; however, the false negatives occur in the same person. Even if the detection results are not significantly different between these two methods, such switching errors decrease the tracking performance. By introducing our method to MPM, the multi-frame context can be used for estimation. Consequently, the performance of MPM + ours is superior to other comparative methods.

4.5. Ablation study

We performed an ablation study to examine the effectiveness of each module, *i.e.*, multi-frame attention (Mfa), forward warping (Fw), and backward warping (Bw). We used the simple method (Heatmap [12]) as the backbone. In addition to the ablated methods, we evaluated two methods; 1) Heatmap+SA, which used a standard self-attention module [33] that aggregates only spatial context in each frame; 2) Heatmap+DSTA, which introduced the spatial-temporal self-attention that is used in [5] that separately applies the spatial-attention and temporal attention.

Table 3 shows the localization (L-mAP) and tracking (T-mAP) performance metrics of each ablated method. Heatmap+SA did not improve performance in either metric. We consider that the spatial context far from a tracking object is not essential for detection and tracking, and thus

Inserted block	L-mAP	T-mAP
1st (480×270)	34.92±0.78	35.38±0.14
2nd (240×135)	28.07±2.15	32.33±2.41
3rd (120×67)	31.53±1.69	31.77±1.69
4th (60×33)	32.19±1.01	33.25±0.28

Table 4. Localization and tracking performance when changing the layers inserted in the proposed modules. We report the resolutions of the feature maps in each block.

the performance was worse. Heatmap+DSTA improved the performance. Our method further improved the performance compared to these methods. In addition, each element of the proposed method improved the localization and tracking performance.

4.6. Hyper-parameter

We evaluated the localization and tracking performance by changing the layers inserting the proposed modules (forward warping, multi-frame attention, and backward warping modules). The encoder of the Siamese U-nets consists of four blocks, each consisting of three convolution layers and one pooling layer. We inserted the proposed modules after each block (the 1st, 2nd, 3rd, and 4th).

Table 4 shows the localization and tracking performance. We used the 4th block as the default setup because of low memory and high running time, which are related to the resolution of the input feature map for the module. On the other hand, the performance improvement was the best when inserting the modules after the first block, *i.e.*, the highest resolution. In all settings, the proposed method improved the tracking performance compared to the backbone (Heatmap).

4.7. Tracking performance under different conditions

As described above, DroneCrowd [36] contains videos showing many situations. We evaluated the tracking performance (T-mAP) under the following conditions: object size (large or small), density (sparse or crowd), and weather (sunny, cloudy, or night). Note that each video has several attributes (*e.g.*, sparse, small, and night).

Figure 6 shows the tracking results under different conditions: (a) objects are large, and illumination is bright; (b) objects are small, and illumination is dark due to the shadow of a building. Even though tracking people under condition (b) is difficult, our method successfully tracked the people.

Figure 7 shows a radar chart of the tracking performance, in which blue indicates Heatmap [12] and red indicates Heatmap+Ours. Besides the condition 'Night', the proposed multi-frame attention with feature-level warping improved tracking performance. In 'Night', it is too difficult to identify people due to their dark backgrounds. Therefore,

Figure 6. Examples of tracking results on each condition. (a) Cloudy, Large, Crowded. (b) Sunny, Small, Crowded.

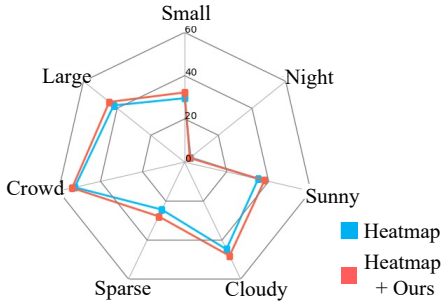

Figure 7. Radar chart of the tracking performance in each condition. Blue: Heatmap [12]; Red: Heatmap + Ours.

our multi-frame attention did not improve performance on those sequences. The comparative methods could not trace people under those conditions either. It is required to effectively represent the small differences between objects and backgrounds to deal with this problem. This is future work. Note that objects are often sparse in videos whose condition is 'Night'; thus, the performance in the 'Sparse' condition was worse than that in 'Crowd'. In all conditions except 'Night', our method improved the performance with almost the same value. It indicates that our method was effective in many situations.

5. Conclusion

This paper proposed a point-level multiple object tracking method that can track small human heads from a video captured by a drone. This method can align the object features in the map by feature-level warping and aggregate the image features from multiple frames by multi-frame attention. This makes the method able to use multi-frame context effectively. Experiments demonstrated that our method could effectively use multi-frame context and outperformed the state-of-the-art method on the DroneCrowd dataset.

Acknowledgment: This work was supported by JSPS KAKENHI Grant Number JP21K19829.

References

[1] Assaf Arbelle and Tammy Riklin Raviv. Microscopy cell segmentation via convolutional lstm networks. In *ISBI*, pages 1008–1012, 2019.

[2] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *ICCV*, pages 6836–6846, 2021.

[3] Guha Balakrishnan, Amy Zhao, Mert R Sabuncu, John Guttag, and Adrian V Dalca. Voxelmorph: a learning framework for deformable medical image registration. *IEEE transactions on medical imaging*, 38(8):1788–1800, 2019.

[4] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *ICCV*, pages 941–951, 2019.

[5] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, volume 2, page 4, 2021.

[6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229, 2020.

[7] Antoni B Chan, Zhang-Sheng John Liang, and Nuno Vasconcelos. Privacy preserving crowd monitoring: Counting people without people models or tracking. In *ICCV*, pages 1–7, 2008.

[8] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking. In *CVPR*, pages 8126–8135, 2021.

[9] Peng Chu and Haibin Ling. Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking. In *ICCV*, pages 6172–6181, 2019.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[11] Yanyan Fang, Biyun Zhan, Wandi Cai, Shenghua Gao, and Bo Hu. Locality-constrained spatial transformer network for video crowd counting. In *ICME*, pages 814–819, 2019.

[12] Junya Hayashida and Ryoma Bise. Cell tracking with deep learning for cell detection and motion estimation in low-frame-rate. In *MICCAI*, pages 397–405, 2019.

[13] Junya Hayashida, Kazuya Nishimura, and Ryoma Bise. Mpm: Joint representation of motion and position map for cell tracking. In *CVPR*, pages 3823–3832, 2020.

[14] Junya Hayashida, Kazuya Nishimura, and Ryoma Bise. Consistent cell tracking in multi-frames with spatio-temporal context by object-level warping loss. In *WACV*, pages 1727–1736, 2022.

[15] Dai Fei Elmer Ker, Sungeun Eom, Sho Sanami, Ryoma Bise, Corinne Pascale, Zhaozheng Yin, Seung-il Huh, Elvira Osuna-Highley, Silvina N Junkers, Casey J Helfrich, et al. Phase contrast time-lapse microscopy datasets with automated and manual cell tracking annotations. *Scientific data*, 5(1):1–12, 2018.

[16] Chanho Kim, Fuxin Li, and James M Rehg. Multi-object tracking with neural gating using bilinear lstm. In *ECCV*, pages 200–215, 2018.

[17] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[18] Kang Li, Eric D Miller, Mei Chen, Takeo Kanade, Lee E Weiss, and Phil G Campbell. Cell population tracking and lineage construction with spatiotemporal context. *Medical image analysis*, 12(5):546–566, 2008.

[19] Yuhong Li, Xiaofan Zhang, and Deming Chen. Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In *CVPR*, pages 1091–1100, 2018.

[20] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, pages 21–37, 2016.

[21] Weizhe Liu, Mathieu Salzmann, and Pascal Fua. Context-aware crowd counting. In *CVPR*, pages 5099–5108, 2019.

[22] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 10012–10022, 2021.

[23] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *CVPR*, pages 3202–3211, 2022.

[24] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. In *CVPR*, pages 8844–8854, 2022.

[25] Kenji Okuma, Ali Taleghani, Nando de Freitas, James J Little, and David G Lowe. A boosted particle filter: Multitarget detection and tracking. In *ECCV*, pages 28–39, 2004.

[26] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8026–8037, 2019.

[27] Hamed Pirsiavash, Deva Ramanan, and Charless C Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *ICCV*, pages 1201–1208, 2011.

[28] Yaadhav Raaj, Haroon Idrees, Gines Hidalgo, and Yaser Sheikh. Efficient online multi-person 2d pose tracking with recurrent spatio-temporal affinity fields. In *CVPR*, pages 4620–4628, 2019.

[29] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, pages 779–788, 2016.

[30] Markus Rempfler, Valentin Stierle, Konstantin Ditzel, Sanjeev Kumar, Philipp Paulitschke, Bjoern Andres, and Bjoern H Menze. Tracing cell lineages in videos of lens-free microscopy. *Medical image analysis*, 48:147–161, 2018.

[31] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[32] Ihor Smal, Wiro Niessen, and Erik Meijering. Bayesian tracking for fluorescence microscopic imaging. In *ISBI*, pages 550–553, 2006.

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[34] Jue Wang and Lorenzo Torresani. Deformable video transformer. In *CVPR*, pages 14053–14062, 2022.

[35] Xiaoxu Wang, Weijun He, Dimitris Metaxas, Robin Mathew, and Eileen White. Cell segmentation and tracking using texture-adaptive snakes. In *ISBI*, pages 101–104, 2007.

[36] Longyin Wen, Dawei Du, Pengfei Zhu, Qinghua Hu, Qilong Wang, Liefeng Bo, and Siwei Lyu. Detection, tracking, and counting meets drones in crowds: A benchmark. In *CVPR*, pages 7812–7821, 2021.

[37] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *ICIP*, pages 3645–3649, 2017.

[38] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *ICCV*, pages 22–31, 2021.

[39] Zhenbo Xu, Wei Zhang, Xiao Tan, Wei Yang, Huan Huang, Shilei Wen, Errui Ding, and Liusheng Huang. Segment as points for efficient online multi-object tracking and segmentation. In *ECCV*, pages 264–281, 2020.

[40] Fuxing Yang, Michael A Mackey, Fiorenza Ianzini, Greg Gallardo, and Milan Sonka. Cell segmentation, tracking, and mitosis detection using temporal context. In *MICCAI*, pages 302–309, 2005.

[41] Zhaozheng Yin, Takeo Kanade, and Mei Chen. Understanding the phase contrast optics to restore artifact-free microscopy images for segmentation. *Medical image analysis*, 16(5):1047–1062, 2012.

[42] Cong Zhang, Hongsheng Li, Xiaogang Wang, and Xiaokang Yang. Cross-scene crowd counting via deep convolutional neural networks. In *CVPR*, pages 833–841, 2015.

[43] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. In *CVPR*, pages 589–597, 2016.

[44] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *ECCV*, pages 474–490, 2020.

[45] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.

[46] Xingyi Zhou, Tianwei Yin, Vladlen Koltun, and Philipp Krähenbühl. Global tracking transformers. In *CVPR*, pages 8771–8780, 2022.

[47] Zibin Zhou, Fei Wang, Wenjuan Xi, Huaying Chen, Peng Gao, and Chengkang He. Joint multi-frame detection and segmentation for multi-cell tracking. In *ICIG*, pages 435–446, 2019.