APPENDIX

A. *Comparison with Active Object Tracking Methods*

Active Object Tracking (AOT) methods, as described in [29], utilize an end-to-end approach through reinforcement learning. These methods process raw video frames as input and generate camera movement actions as output. According to [29], there are seven discrete actions: *move-forward/backward, turn-left/right, move-forward-and-turn-left/right*, and *no-op*. However, existing person-following datasets, such as Chen's dataset [5] and our own dataset, only provide ground truth bounding boxes of the target person. For a valid comparison, we need to map these bounding boxes to the action space. As noted in [29], a two-stage method (combining single object tracking with a PID controller) can achieve a 100% success rate if the estimated target bounding box (bbox) is accurate. Therefore, we can deduce the ground truth actions from the ground truth target bboxes. This approach allows us to evaluate AOT methods within person-following datasets. According to [29], the objective of the camera action is to keep the target person's bbox centered in the image, maintaining the same size as initially observed. The ground truth action is generated according to the horizontal error $X_{err} = \frac{X_b - W/2}{W/2}$ and the size error $S_{err} = \frac{W_b \times H_b - W_{exp} \times H_{exp}}{W_{exp} \times H_{exp}}$ shown as in Fig. 4. According to [29], we have the following ground truth action mappings:

(1) *Move forward* if $\text{abs}(X_{err}) \leq 0.1$ and $S_{err} \leq -0.2$;
(2) *Move backward* if $\text{abs}(X_{err}) \leq 0.1$ and $S_{err} \geqslant 0.2$;
(3) *No-op* if $\text{abs}(X_{err}) < 0.1$ and $\text{abs}(S_{err}) < 0.2$;
(4) *Move forward and turn right* if $0.1 \leq X_{err} \leq 0.3$;
(5) *Turn right* if $X_{err} > 0.3$;
(6) *Move forward and turn left* if $-0.3 \leq X_{err} \leq -0.1$;
(7) *Turn left* if $X_{err} < -0.3$.

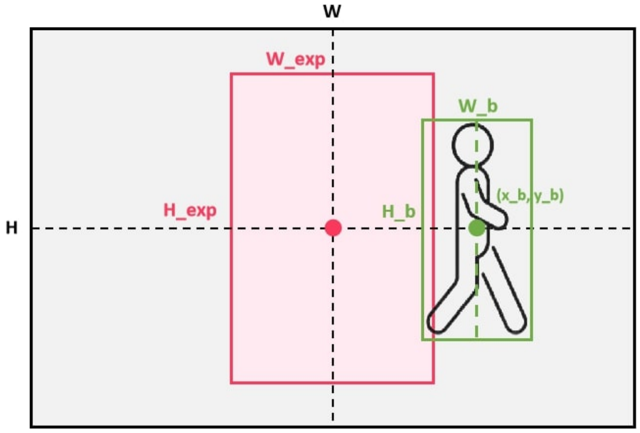
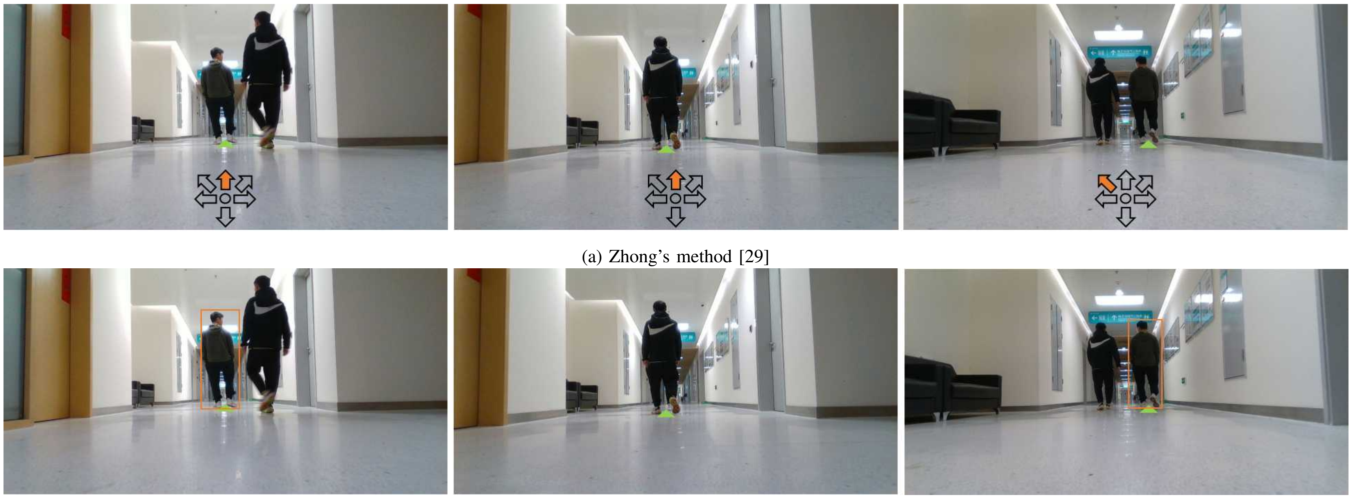


Fig. 4. An example to illustrate errors is as follows: The goal of a correct action is to move the target person closer to the center of the image. Here, (W, H) represents the image's width and height. (W_{exp}, H_{exp}) denotes the size of the expected centered bounding box, initialized by the first bounding box of the target person. (W_b, H_b) represents the target person's bounding box size in the current frame, and (x_b, y_b) is its center point.

This paper aims to evaluate the algorithms' tracking performance, meaning true target person identification is the first priority. Therefore, besides evaluating accurate action estimation, we reduce the matching standards. Specifically, we consider an action to be true if this action tries to move the target person to the center of the image. For example, if the bbox of the target person is on the left of the image, *move backward, turn left*, and *move forward and turn left* are all considered as true actions. The results are reported in Table. I. An example is shown in Fig. 5. We observe that after a long occlusion by a visually similar person, Zhong's method outputs *move forward and turn left*, failing to re-identify the target person. In contrast, our method reliably re-identifies the target person even after long-term occlusion.

B. *Online Continual Learning Evaluation*

1) Metric: The evaluation of OCL [7] aims to assess how well the model remembers previous knowledge, which is essential for person ReID in RPF, as previous knowledge contains potentially matching experiences for future ReID. Additionally, incrementally remembering previous knowledge might result in a more generalized feature extractor. Specifically, we treat the OCL evaluation for person ReID as a classification task, where we assume that the true identity of the target person is known in each frame, and the model incrementally learns with known labels. For evaluation purposes, we divide each sequence into eight segments, each representing different levels of distribution drift. During incremental learning, after each segment is learned, the model is evaluated on previously seen segments. Similar to [7], we use the ReID



(a) Zhong's method [29]

(b) Ours

Fig. 5. An example comparing Zhong's method [29] and ours. From left to right, the sequence represents observations where a long-term occlusion occurs. (a) Zhong's method outputs *move forward and turn left* due to a failed re-identification of the target person. (b) Our method reliably re-identifies the target person even after long-term occlusion.

mean accuracy at the end of training (r-mEAcc) as our OCL evaluation metric:

$$\text{r-mEAcc} = \frac{1}{8}\sum_{j=0}^{8} a_{8,j}, \tag{6}$$

where $a_{8,j}$ represents the average accuracy on the j_{th} segment, with the model learned from all eight segments. Higher r-mEAcc values indicate that the model retains more of the previous knowledge during incremental learning.

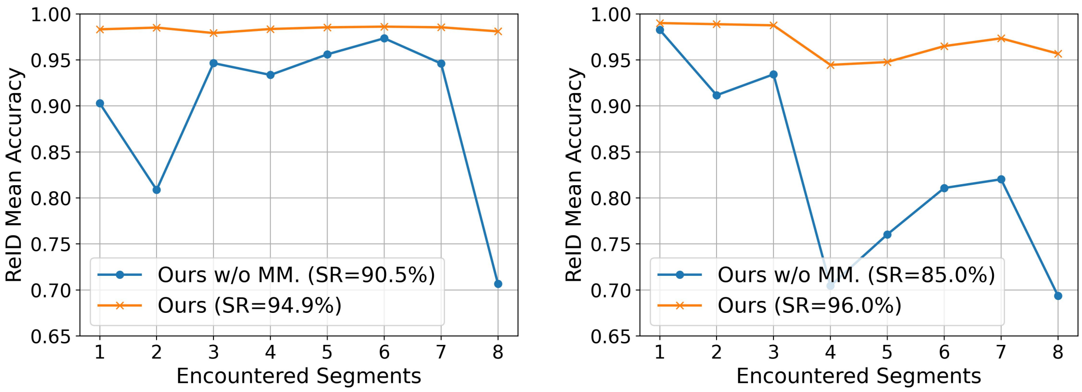
2) Experimental Results: The results are shown in Table III. The row of "ByteTrack [34] + RPF-ReID" utilizes a pre-trained and fixed feature extractor. In comparison to the original setup "ByteTrack + RPF-ReID + OCL, based on Reservoir [26]", it experiences a significant decline in performance, with reductions of 37.3% and 74.7% in r-mEAcc and t-mAcc, respectively, on *corridor2*. This underscores the importance of online optimization of the feature extractor with collected experiences to combat domain drift and enhance the system's ReID performance. Moreover, this approach to discriminative ReID modeling markedly improves person tracking efficacy, as evidenced by higher t-mAcc values.

We also verify the necessity of our memory management (introduced in Sec. III-D) for mitigating domain drift. We use newly observed samples only to fine-tune the feature extractor without memory management. As shown in Fig. 6, this naive strategy leads to significant domain drift, resulting in a notable decrease in ReID accuracy across different segments. Such drift significantly undermines the RPF system's tracking efficiency, manifesting as t-mAcc reductions of 4.4% and 11.0% on *lab-corridor* and *corridor2*, respectively. This outcome highlights the value of our memory manager in preserving valuable long-term experiences. By replaying these experiences to mitigate domain drift, we ensure that our ReID features remain robust, thereby enhancing the RPF system's consistent tracking performance.

In summary, the above experiments demonstrate the effectiveness of optimizing the feature extractor online using collected experiences managed by our memory manager. This strategy effectively addresses domain drift, resulting in enhanced ReID performance within the RPF system. Another observation is that although the OCL ability of BioSLAM [9] is worse than Reservoir [26] with r-mEAcc of 79.0% vs. 94.0% on *lab-corridor*, its tracking accuracy only drops by 2.2%. This indicates that not all historical knowledge needs to be memorized for person ReID in some situations. However, we claim that maximizing the enhancement of ReID ability at a long-term scale is still necessary as it ensures a discriminative appearance representation for dealing with complex ReID situations.

TABLE III. Experiments on *corridor2* and *lab-corridor* are conducted to evaluate the ReID mean accuracy at the end of training (r-mEAcc, %) and success rate (SR, %). All r-mEAcc values are averaged across three runs.

Methods	**corridor2**		**lab-corridor**	
	r-mEAcc ↑	***SR*** ↑	***r-mEAcc***	***SR***
ByteTrack [34] + RPF-ReID	59.2 ± 0.0	20.2	31.7 ± 0.0	54.2
ByteTrack + RPF-ReID + OCL, based on BioSLAM [9]	94.9 ± 2.0	94.9	79.0 ± 22.5	93.8
ByteTrack + RPF-ReID + OCL, based on MIR [24]	94.7 ± 0.8	95.4	86.1 ± 14.3	96.1
ByteTrack + RPF-ReID + OCL, based on Reservoir [26]	96.5 ± 0.4	94.9	94.0 ± 0.7	96.0



(a) corridor2

(b) lab-corridor

Fig. 6. Plots of ReID mean accuracy w.r.t. encountered segments. "Ours w/o MM." indicates fine-tuning the feature extractor without memory management (introduced in Sec. III-D), using newly observed samples only. After fine-tuning from a new segment, the model is evaluated on segments it has encountered previously to determine its mean accuracy. This metric indicates the model's ability to retain knowledge from segments learned earlier.

C. Implementation Details

For all experiments, we set the following default parameters: memory sizes $|\mathbb{S}| = 64$ and $|\mathbb{L}| = 512$, a batch size of 64 for each replay including long-term and short-term relays, a regularization parameter $\lambda = 1.0$ for RR, a keyframe selection threshold $\delta_l = 0.02$, an id switch threshold $\delta_{\text{sw}} = 0.35$, a ReID threshold $\delta_{\text{reid}} = 0.7$ and a number of consecutive frames $\zeta_{\text{reid}} = 5$. In this paper, for representing the part-level features, we define ten parts: $\{$front, back$\}\times\{$head, torso, legs, feet, whole$\}$.

For orientation estimation, we employ MonoLoc[1] to infer the orientation using detected joint positions from AlphaPose[2]. These joint positions are also utilized to estimate the visible parts. We utilize YOLOX-S[3] for bounding-box detection. For the tracking module, we utilize ByteTrack as our tracking method. For our proposed ReID module, we use ResNet18 as our feature extractor, pre-trained on the MOT16 dataset [28]. During OCL for the ResNet18, only the layers after *conv3* are trainable (including *conv3*).

All evaluations are conducted on both a high-end PC and an onboard NUC. The high-end PC includes an Intel® Core™ i9-12900K CPU and NVIDIA GeForce RTX 3090. The onboard NUC is an Intel NUC 11 mini PC powered by a Core i7-1165G7 CPU and NVIDIA GeForce RTX2060-laptop GPU. This NUC is mounted on a Unitree Go1 quadruped robot to perform robot person following in the real world as shown in Fig. 1 and the submitted video. Besides the computer, a dual-fisheye Ricoh camera is mounted on the robot, providing cropped perspective images with a resolution of 640 × 480 and a frequency of 30Hz.

[1] https://github.com/vita-epfl/monoloco
[2] https://github.com/MVIG-SJTU/AlphaPose
[3] https://github.com/Megvii-BaseDetection/YOLOX

D. Memory Examples

Several replayed examples of our short-term and long-term memories are shown in Fig. 5. For more visual examples of short-term and long-term memories, please refer to the supplementary video.

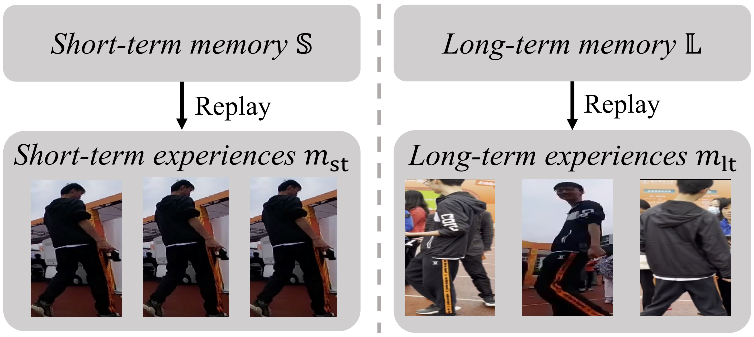

Short-term memory 𝕊
Replay
Short-term experiences m_{st}
Long-term memory 𝕃
Replay
Long-term experiences m_{lt}

Fig. 7. Both short-term and long-term experiences $(m_{\text{st}} \cup m_{\text{lt}})$ are responsible for training the feature extractor with m_{st} and m_{lt} being sampled from short-term memory $\mathbb{S}$ and long-term memory $\mathbb{L}$, respectively. $\mathbb{L}$ contains sparse yet valuable historical samples, maintained by the *memory manager*. Besides, the target classifier is trained with m_{st} sampled from $\mathbb{S}$, which stores the most recently observed samples, representing the latest knowledge.

E. Target-ReID Lifecycle

Algorithm 1: Target-ReID Lifecycle

```
Input: Current image $\mathbf{I}$ and tracked people $\{\mathbf{B}, \mathbf{p}\}_i$ representing bounding boxes and positions, target person's identity $id$, target confidence $s$,
       short-term memory $\mathbb{S}$, long-term memory $\mathbb{L}$, feature extractor $f$ and target classifier $g$
Output: Target person's position $\{\mathbf{p}\}_{id}$ in the current frame
Extract image patches $\mathbf{M}$ from $\mathbf{I}$ and $\mathbf{B}$;
Construct the observation set $\{\mathbf{M}, \mathbf{y}\}_i$ where $\mathbf{y} = 1$ if $i == id$, otherwise $\mathbf{y} = 0$;
Extract features $\mathbf{F}$ from $\mathbf{M}$ with $f$;
if $id \in \{i\}$ then
    Estimate $s$ of the target person based on Eq. 3;
    if $s > \delta_{\text{sw}}$ then
        Consider $\{\bar{i}\}$ as identities of negative tracks;
        $\{\mathbf{M}, \mathbf{y}\}_{id} \rightarrow \mathbb{S}$, $\{\mathbf{M}, \mathbf{y}\}_{\bar{i}} \rightarrow \mathbb{S}$ based on FILO rule;
        Sample $m_{\text{st}}$ from $\mathbb{S}$;
        Train $g$ with $m_{\text{st}}$ based on Eq. 4;
        ### Separate Thread ###
        $\{\mathbf{M}, \mathbf{y}\}_{\bar{i}} \rightarrow \mathbb{L}$ based on FILO rule;
        $\{\mathbf{M}, \mathbf{y}\}_{id} \rightarrow \mathbb{L}$ if it is a keyframe based on Eq. 5;
        Consolidate $\mathbb{L}$ with OCL techniques if $\mathbb{L}$ is full;
        Sample $m_{\text{lt}}$ from $\mathbb{L}$;
        Train $f$ with $m_{\text{st}}$ and $m_{\text{lt}}$ based on Eq. 1;
        ### Separate Thread ###
        Return target position $\{\mathbf{p}\}_{id}$;
    else
        Let $id = -1$, indicates id switch between the target person and other people;
        Return $\emptyset$;
else
    Estimate $s$ of the $i_{th}$ person based on Eq. 3;
    if $s > \delta_{\text{reid}}$ for consecutive $\zeta_{\text{reid}}$ frames then
        Let $id = i$, indicates successful target person ReID;
        Return target person's position $\{\mathbf{p}\}_{id}$;
    else
        Return $\emptyset$;
```