Pattern Recognition 134 (2023) 109073

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Video Object Segmentation using Point-based Memory Network

Mingqi Gao[a,b], Jungong Han[b,c,*], Feng Zheng[a], James J.Q. Yu[a], Giovanni Montana[b]

[a] Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China
[b] WMG Data Science, University of Warwick, Coventry CV4 7AL, UK
[c] Department of Computer Science, Aberystwyth University, Aberystwyth SY23 3DB, UK

ARTICLE INFO

Article history:
Received 14 April 2022
Revised 8 August 2022
Accepted 25 September 2022
Available online 27 September 2022

Keywords:
Video object segmentation
Point-based feature matching
Adaptive matching module

ABSTRACT

Recent years have witnessed the prevalence of memory-based methods for Semi-supervised Video Object Segmentation (SVOS) which utilise past frames efficiently for label propagation. When conducting feature matching, fine-grained multi-scale feature matching has typically been performed using all query points, which inevitably results in redundant computations and thus makes the fusion of multi-scale results ineffective. In this paper, we develop a new Point-based Memory Network, termed as PMNet, to perform fine-grained feature matching on hard samples only, assuming that easy samples can already obtain satisfactory matching results without the need for complicated multi-scale feature matching. Our approach first generates an uncertainty map from the initial decoding outputs. Next, the fine-grained features at uncertain locations are sampled to match the memory features on the same scale. Finally, the matching results are further decoded to provide a refined output. The point-based scheme works with the coarsest feature matching in a complementary and efficient manner. Furthermore, we propose an approach to adaptively perform global or regional matching based on the motion history of memory points, making our method more robust against ambiguous backgrounds. Experimental results on several benchmark datasets demonstrate the superiority of our proposed method over state-of-the-art methods.

1. Introduction

Video Object Segmentation (VOS) is one of the fundamental problems in video understanding. As the main branch of VOS, semi-supervised video object segmentation (SVOS) aims to infer the object masks in every frame using only masks annotated in the first frame. SVOS focuses on target objects (the annotated objects) segmentation, which has significant value in several real-world applications, such as video editing, summarisation, and surveillance [1,2].

Recently, matching-based methods have dominated the SVOS field due to their robust and efficient implementations. The basic idea of these methods advocates the propagation of dense labels from the reference frames (past frames) to the query frame (the frame to be segmented). Such a propagation is implicitly guided by the similarity of their features. The earlier matching-based methods consider the first and previous frames as the reference to generate the results with spatiotemporal consistency [3,4]. Under the umbrella of memory-based methods, new variants also utilise the intermediate frames between the first and previous frames, enriching the object changes and further improving the state-of-the-art performance at the cost of more memory consumptions [5–13].

For both methods, feature matching plays a vital role in fine-grained label prediction, which has been one of the major goals in general segmentation tasks, but has not been fully explored in SVOS. Fig. 1(a) summarises the primary matching-based methods. It is observed that the feature matching at the coarsest scale (with a stride of 16) builds a single connection between frames, which guides the coarse label propagation. However, when refining the propagation, these methods only rely on high-resolution query features but ignore the reference frame labels on the same scales. Therefore, the final results may not be consistent with the target objects in the reference frames.

To improve the fine-grained label propagation, recent works advocated the use of multi-scale matching [12,14]. As shown in Fig. 1(b), the matching results on the finer scales (with strides of 4, 8) are measured and fused with the coarsest results. Therefore, the label propagation is simultaneously constrained by high-level semantic features and low-level detailed features. Although achieving high-quality results, these methods share a common limitation, i.e., all query feature points on all scales are considered during matching, resulting in a high degree of redundant computation. In re-

* Corresponding author at: Department of Computer Science, Aberystwyth University, Aberystwyth SY23 3DB, UK.
E-mail address: juh22@aber.ac.uk (J. Han).

https://doi.org/10.1016/j.patcog.2022.109073

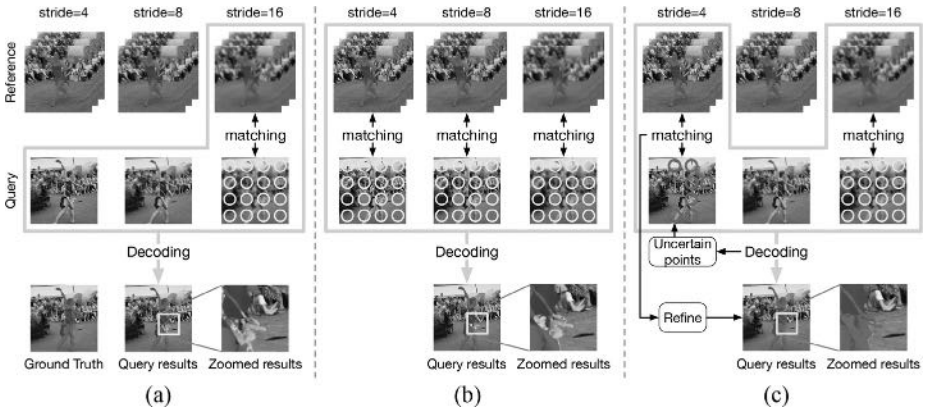

Fig. 1. Comparison of different matching-based SVOS methods, mainly in what scales of features, and what point sampling approaches are involved during inference. (a) Most existing methods utilise the coarsest reference features and multi-scale query features (highlighted with the light green rectangle). Matching is done on the coarsest scale only, where query features are densely sampled (yellow points). (b) Multi-scale matching-based methods perform reference/query feature matching on multiple scales. For each scale, query features are densely sampled. (c) Our method still performs matching on the coarsest scale, where all query features are involved. However, on the finest scale, we only sample the uncertain points (highlighted red circles, where segmentation results are error-prone) and consider them for fine-grained matching. The segmentation results are predicted by STM [5] (a), HMMN [12] (b), and ours (c). The ROIs highlighted by yellow boxes are zoomed in for better comparison in fine-grained prediction. Best viewed in colour. . (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

ality, only the points with high-frequency signals require different processes on a finer scale, whereas the others can achieve satisfactory results even if the matching is conducted at the coarsest scale. Moreover, considering all feature points distracts the learning from an effective fusion strategy between different scales of matching results since most video frame points are “easy samples” and do not require refinement.

Inspired by the above observations, we propose a point-based refinement module to reduce redundant computations and focus on “hard samples” only, i.e., the spatial regions or points with ambiguous features, which confuse most existing SVOS methods to make correct and confident predictions. As shown in Fig. 1(c), we first infer the initial masks from the coarsest matching results. Then, a set of uncertain points with potentially non-ideal results are derivated from the intermediate decoding outputs. During refinement, our approach only considers these uncertain points for fine-grained feature matching. In this way, these points can serve as hard samples to encourage segmentation modules to be more effective against ambiguous features. To improve the confidence of the uncertain points, we implement an uncertainty detection module with a lightweight CNN architecture, which generates uncertainty maps from the intermediate decoding outputs (with different scales).

To the best of our knowledge, the proposed SVOS algorithm is the first approach to implement the point-based refinement, which utilises detailed low-level features more efficiently than multi-scale matching [12,14]. Although AFB-URR [8] also computes the uncertain regions to refine the initial results, all query points are still involved. In addition, AFB-URR [8] only relies on the confident results to refine their neighbouring uncertain regions. Therefore, the effect of its refinement module is limited. Compared with the point-based refinement for other tasks, our proposed method remains competitive. Kirillov et al. [15] measures a set of uncertain regions from the ambiguous probabilities and refines them with fine-grained features. Zhang et al. [16] first predicts initial masks from the coarsest features and then improves the boundary regions with fine-grained features. Although these methods also resort fine-grained features for refinement, they mainly focus on the ambiguous boundaries, which come from the final predictions. In contrast, our method implements a learnable module to predict potential errors from different scales of intermediate decoding outputs. As a result, more hard samples can be mined to enhance the refinement module.

Besides the fine-grained feature matching, temporal consistency is also essential for high-quality segmentation results. However, most memory-based methods [5,6,8,9,13] ignore temporal information and only perform global matching between frames. Although achieving good robustness against occlusions and fast motion, the methods are sensitive to the regions similar to target objects. In more recent developments [10–12], this problem has been mitigated by incorporating local matching, which replies on a sensible assumption that objects move smoothly throughout video sequences. During local matching, recent frames could provide reasonable spatial-temporal constraints to filter out ambiguous regions. Given the complementary nature of global and local matching, it is evident that a fusion strategy would benefit to segmentation performance. However, this has been underexplored in existing methods. LCM [10] and HMMN [12] perform local matching on recent reference frames and global matching on distant reference frames. Although such a strategy can handle most cases, some challenges remain (e.g., the ambiguity when matching with distant reference frames). Alternatively, RMNet [11] performs local matching on all reference frames. However, since only short-term spatial-temporal constraints are applied (from optical flow), matching with distant reference frames would lose some informative correlations, thus requiring an additional mechanism for complements. Therefore, to handle more challenging videos efficiently, an adaptive and compact approach is required to fuse different kinds of matching schemes.

Unlike existing methods [10–12], which apply the same matching scheme to all the feature points from the same reference frame, we propose to deal with them differently using appropriate matching schemes. To this end, we build point trajectories from reference frames to the query frame, based on the intermediate results during segmentation. For each reference feature point, its matching scheme depends on the changes it has experienced along the corresponding path. For example, if one point has undergone only slight changes between frames, it should be locally matched with query points, even if it comes from a distant reference frame. On the contrary, the global matching should be performed on the points experiencing drastic changes, even they come from recent reference frames. In this way, the proposed adaptive matching module can break the limitations of temporal distance and adapt feature matching to video contexts.

Our contribution can be summarised as follows:

(1) We propose a point-based refinement module, which resorts to multi-scale feature matching to improve segmentation results. But unlike the existing methods, the proposed module only considers the uncertain points rather than all the points when performing matching on the finer scales. Therefore, similar or even better results can be achieved with less computation.
(2) We propose an adaptive matching module, which flexibly assigns each memory feature point (spatially basic component in memory feature maps) with an appropriate matching scheme, according to its dynamic information throughout the video. Compared with the existing matching-based methods, which solely rely on either global or temporal distance-based matching schemes, the proposed module can better adapt to video contexts and achieve better complementary between different matching schemes.
(3) The proposed method (Point-based Matching Network, termed as PMNet) achieves the state-of-the-art performance on several benchmark datasets while retaining competitive efficiency.

2. Related work

Earlier methods perform SVOS mainly based on discriminative feature descriptors and motion information [17,18]. More recently, deep learning techniques have prompted SVOS performance considerably due to their robust feature representations. This section gives a brief overview of the deep learning-based SVOS methods by different strategies they utilise.

2.1. Online fine-tuning-based SVOS

This approach is firstly proposed in OSVOS [19], which fine-tunes segmentation networks with the first frame annotations, shifting the output domain from general objects to the annotated ones. Extended from OSVOS, OnAVOS [20] further fine-tuned networks with confident segmentation results. OSVOS-S [21] complements the segmentation results with the semantic information of the annotated objects. Despite achieving good results, online fine-tuning is rarely explored in recent works since it is time-consuming and easy to overfit.

2.2. Propagation-based SVOS

This approach is firstly proposed in MaskTrack [22], which assumes objects move smoothly throughout the sequence. Therefore, the objects predicted from the previous frame can well estimate the current segmentation. Due to its efficient implementation, mask propagation has been widely used in subsequent SVOS works. The representative improvement mainly lies in adapting the propagated masks to the current frame [23]. To mitigate the error accumulation, ARG-VOS [24] implemented two reinforcement learning-based models to adapt the previous results to the current frame context. Despite being efficient, these methods are vulnerable to occlusions and fast motion.

Besides the short-term propagation, the approaches dedicated to long-term spatiotemporal information propagation are also utilised for SVOS. For instance, ConvLSTM-based methods [25] and ConvGRU-based methods [6]. Theoretically, this approach can learn long-term dependency. However, limited by computation resources, their models can only be optimised with short video clips, which degrades the expected SVOS performance.

2.3. Matching-based SVOS

Unlike online fine-tuning, this approach segments the target objects by measuring cross-frame feature correspondence rather than fine-tuning network parameters. Therefore, more efficiency can be achieved during inference. Currently, there are mainly two matching strategies for SVOS: ROI matching and dense matching. The former tracks and segments the ROIs of either the whole object [26] or object parts [27] throughout the sequences. However, since the ROI-level matching is sensitive to partial loss, this approach cannot handle the sequences with heavy occlusions. In contrast, the latter utilises dense matching results to implicitly guide the label propagation between frames. The conventional dense matching-based SVOS methods initialise the reference with the first frame annotation and enrich it with subsequent confident results [28]. To suppress ambiguous backgrounds and achieve the results with temporal smoothness, other matching-based methods considered the previous frame during inference. For instance, RGMP [3] integrates previous frame masks into the matching between the first and query frames. FEELVOS [4] explicitly performs feature matching between the previous and query frames. The resulting local correspondence complements the global matching between the first and query frames well. Instead of focusing on the object regions only, CFBI [29] performs feature matching on both object and background, encouraging the feature embedding to be contrastive. As an extension of CFBI [29], CFBI+ [14] further improves the SVOS performance by multi-scale feature matching.

2.4. Memory-based SVOS

This approach was firstly proposed in STM [5], which enables the segmentation network to be more robust against object changes (e.g., scale and appearance) by considering additional past frames as the reference frames (memory frames). STM [5] outperformed state-of-the-art methods on all benchmark datasets at the time of publication and therefore attracted much attention in the community. Several recently proposed methods have attempted to improve specific aspects of STM:

(1) Temporal correspondence between memory frames, implemented in EGMN [6]. This approach proposes a graph-based scheme to highlight the background and frequently appearing objects in memory frames.
(2) Memory management, implemented in AFB-URR [8] and SwiftNet [9]. Instead of considering the whole past frames, these approaches only store useful and discriminative points in memory to avoid redundant computation and improve memory usage.
(3) Local feature matching, implemented in KMN [7], LCM [10], and RMNet [11]. These approaches perform feature matching within a local area of the memory and/or query frames based on temporal smoothness [10,11] or mutual matching [7].
(4) Multi-scale feature matching, implemented in HMMN [12]. This approach improves the quality of segmentation results by performing matching on feature maps at different resolutions.
(5) Efficient similarity metrics, implemented in STCN [13]. This approach reveals that the dot product in memory-based methods degraded feature utilisation and replaced it with a computationally efficient L2 distance. In addition, the architecture of feature encoders are simplified. These contributions introduced a considerable improvement in both accuracy and efficiency and set a new benchmark for memory-based SVOS.

Due to the good balance between SVOS accuracy and efficiency, we build our method upon a memory-based approach. From the above descriptions, it is observed that multi-scale feature matching has not been fully explored. Therefore, our method can achieve further improvement and inspire future research. In addition, the existing memory-based SVOS methods cannot well balance local and global matching between frames. Our adaptive matching strategy shrinks the gap by linking the memory feature points across frames.

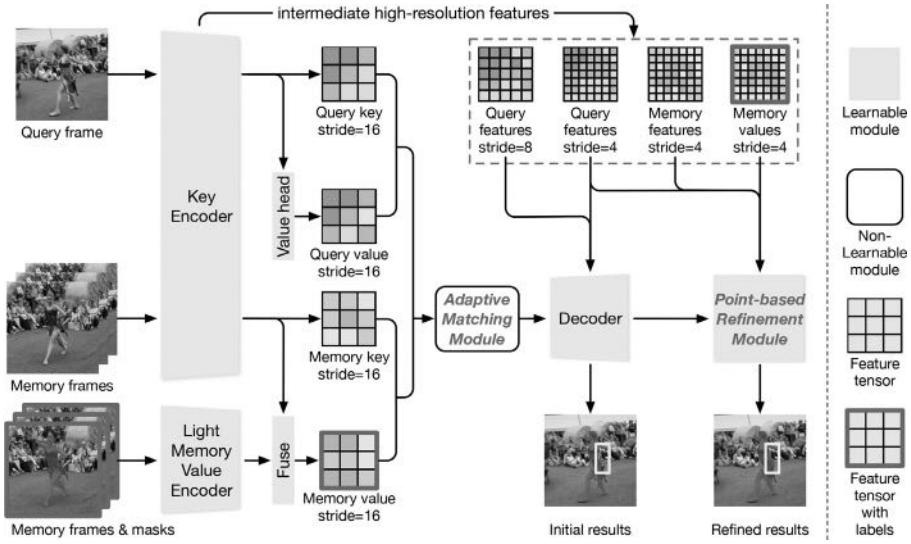

Fig. 2. An overview of our PMNet. The point-based refinement module improves decoding results with high-resolution query keys, memory keys (both from the key encoder) and memory values (from the memory value encoder). The adaptive matching module utilises the motion histories of memory points to suppress ambiguous backgrounds. Both contributions are highlighted in red bold fonts. Yellow boxes show the main difference between the initial and refined results. Please refer to Section 4.1 for more implementation details. Best viewed in colour. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

3. Method

We propose Point-based Memoty Network (PMNet) for fine-grained SVOS. Section 3.1 overviews the proposed architecture. In Section 3.2, we introduce the point-based refinement module. The adaptive matching module is presented in Section 3.3.

3.1. Overview

Fig. 2 illustrates the overall architecture of PMNet. Analogously to other memory-based methods, PMNet encodes video frames into keys and values, where keys are light-weight embeddings for similarity measurement; values are formed with more channels of information and used for feature aggregation and decoding. Specifically, our backbone network is designed based on STCN [13], which utilises the shared encoder for query/memory keys and query values, a lightweight encoder for memory values. Although the decoder considers high-resolution query features (with strides of 4 and 8) to generate fine-grained outputs, the memory labels on the same scales are ignored. In other words, STCN [13] and most existing memory-based SVOS methods [5–11,13] refine their results with coarse labels only. Therefore, the predicted objects might differ from the target in some details.

We address this issue with a refinement scheme, where high-resolution memory labels are taken into account in order to improve the initial results. To maintain competitive efficiency, we implement the scheme with point-based refinement module, where only uncertain results are involved. The module is presented in Section 3.2. In addition, we propose an adaptive matching module to suppress ambiguous backgrounds. During inference, the tracking confidence of the memory points determine whether the global matching or local matching is performed. More details can be found in Section 3.3.

Before proceeding, we provide the necessary definitions about PMNet here. As shown in Fig. 2, PMNet mainly consists of three parts: backbone (f_{backbone} with learnable parameters θ, including encoders, value head, fuse module, and decoder), point-based refinement module (including uncertainty detection module $f_{\text{uncertain}}$ and point processing module f_{point}, with learnable parameters ϕ and γ, respectively), and adaptive matching module (non-learnable). Given a query frame $\boldsymbol{I}^Q \in \mathbb{R}^{H\times W\times 3}$ and T memory frames $\boldsymbol{I}^M \in \mathbb{R}^{T\times H\times W\times 3}$, where H and W indicate the height and width dimensions, PMNet performs SVOS mainly in three steps:

(1) Prepare multi-scale features with f_{backbone}, including query features ($\boldsymbol{X}_4^Q \in \mathbb{R}^{H/4\times W/4\times 256}$, $\boldsymbol{X}_8^Q \in \mathbb{R}^{H/8\times W/8\times 512}$, and $\boldsymbol{X}_{16}^Q \in \mathbb{R}^{H/16\times W/16\times 1024}$) and memory features ($\boldsymbol{X}_4^M \in \mathbb{R}^{T\times H/4\times W/4\times 256}$ and $\boldsymbol{X}_{16}^M \in \mathbb{R}^{T\times H/16\times W/16\times 1024}$).

(2) Encode the coarsest query/memory keys ($\boldsymbol{X}_{\text{key},16}^Q \in \mathbb{R}^{H/16\times W/16\times 64}$, $\boldsymbol{X}_{\text{key},16}^M \in \mathbb{R}^{T\times H/16\times W/16\times 64}$) and values ($\boldsymbol{X}_{\text{value},16}^Q \in \mathbb{R}^{H/16\times W/16\times 512}$, $\boldsymbol{X}_{\text{value},16}^M \in \mathbb{R}^{T\times H/16\times W/16\times 512}$) with f_{backbone} and perform the adaptive matching to get the initial segmentation results.

(3) Employ $f_{\text{uncertain}}$ to detect uncertain points from the intermediate decoding outputs (generated when predicting the initial results). Then, utilise f_{point} to encode point-wise query/memory keys and values from $\boldsymbol{X}_4^Q$, $\boldsymbol{X}_4^M$, and $\boldsymbol{X}_{\text{v},4}^Q$, and leverage point-based matching to refine the initial results.

3.2. Point-based refinement module

The module aims to improve the fine-grained SVOS while keeping competitive efficiency. As shown in Fig. 3, it performs refinement in three steps: (1) uncertainty detection from decoding outputs (with $f_{\text{uncertain}}$); (2) fine-grained feature matching and aggregation on uncertain points (with f_{point}); (3) point-based refinement on uncertain points (with f_{point}). More details are in order.

3.2.1. Uncertainty detection

The module detects uncertain points where the initial results are error-prone. The existing SVOS method [8] achieves this only from the predicted probabilities. Despite being efficient, the uncertain regions mainly focus on object boundaries and have little response inside the object or background regions. The uncertain region here indicates a set of spatially connected uncertain points. an uncertain region could be a spatially isolated uncertain point or any number of the spatially connected uncertain points. Instead of relying on the probabilities, we utilise more clues to detect potentially erroneous results. As shown in Fig. 4, we propose a lightweight CNN-based module to generate the uncertainty map:

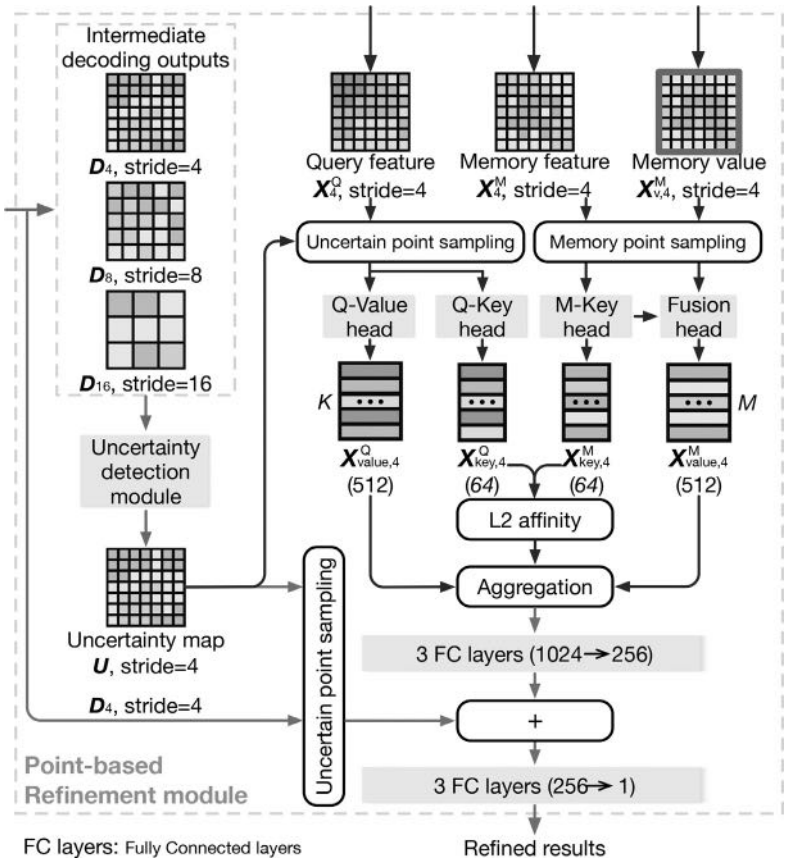

Fig. 3. Diagram of our point-based refinement module, which consists of two learnable modules: $f_{\text{uncertain}}$ (uncertainty detection module) and f_{point} (all other modules). At first, the uncertainty map is generated from the decoding outputs (Section 3.2.1, corresponding to the red data flow). Then, fine-grained feature matching and aggregation are performed on the uncertain points only (Section 3.2.2, corresponding to the blue data flow). Finally, the aggregated features are fused with the initial decoding results to further refine the initial predictions (Section 3.2.3, corresponding to the dark green data flow). All heads here are implemented with one FC layer. More details about the uncertainty detection module are shown in Fig. 4. Best viewed in colour. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

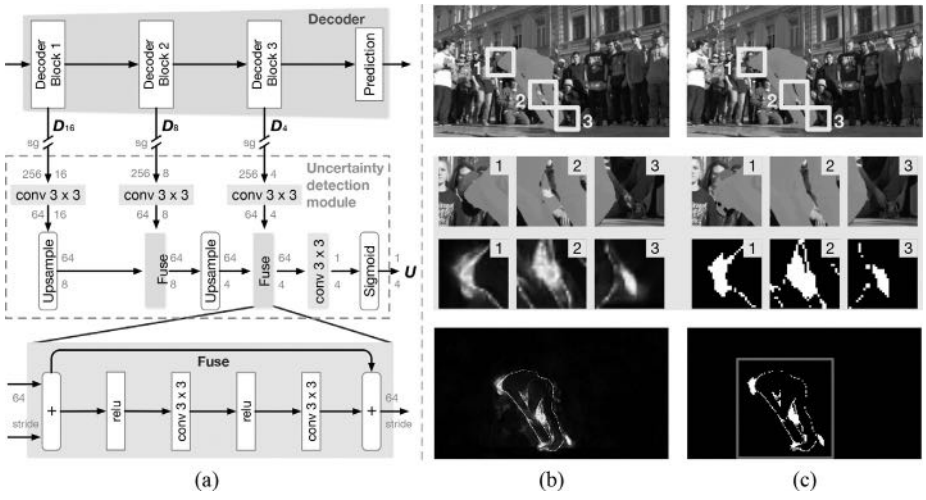

Fig. 4. (a) Diagram of our uncertainty detection module, which generates an uncertainty map $\boldsymbol{U} \in \mathbb{R}^{H/4 \times W/4}$ from the intermediate decoding outputs: ($\boldsymbol{D}_{16} \in \mathbb{R}^{H/16 \times W/16 \times 256}$, $\boldsymbol{D}_8 \in \mathbb{R}^{H/8 \times W/8 \times 256}$, and $\boldsymbol{D}_4 \in \mathbb{R}^{H/4 \times W/4 \times 256}$). H and W denote the height and width of video frames. The numbers around data flows (arrows) indicate the corresponding data dimensions and strides. We apply the stop gradient (sg) between decoder blocks and the module to focus our backbone network on the segmentation task only. (b) Top: The input video frame with the initial results (green mask). Yellow boxes are the ROIs for detailed analysis. Bottom: The generated uncertainty map $\boldsymbol{U} \in \mathbb{R}^{H/4 \times W/4}$, Middle: ROIs zoomed from initial results and the corresponding uncertainty map. (c) Top: The input video frame with the refined results (green mask), Bottom: Uncertain points sampled from the uncertainty map. The sampling is constrained by the red box, which contains the initial segmentation results (with 120% of the original height and width to avoid under-sampling), Middle: ROIs zoomed from the refined results and corresponding uncertainty points. Best viewed in colour. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

$\boldsymbol{U} = f_{\text{uncertain}}(\boldsymbol{D}, \phi)) \in \mathbb{R}^{H/4\times W/4}$, where $\boldsymbol{D} = \{\boldsymbol{D}_{16}, \boldsymbol{D}_8, \boldsymbol{D}_4\}$ are intermediate decoding outputs (with strides of 16, 8, and 4). The parameters ϕ in $f_{\text{uncertain}}$ are optimised by minimising a weighted smooth L1 loss:

$$\mathcal{L}_{\text{uncertain}} = \frac{1}{\sum_{i\in\Omega} w_i} \sum_{i\in\Omega} w_i \mathcal{L}_{\text{smoothL1}}(\boldsymbol{U}_i, \boldsymbol{U}_i^*), \tag{1}$$

where $\boldsymbol{U}_i$ is the uncertainty at the location i, $\boldsymbol{U}^* = |f_{\text{backbone}}(\boldsymbol{I}^Q, \boldsymbol{I}^M, \theta) - \boldsymbol{M}|$ is the difference between the predicted propabilities ($\in [0,1]^{H/4\times W/4}$) and the down-sampled ground truth mask $\boldsymbol{M} \in \{0,1\}^{H/4\times W/4}$. $w_i = 1$ if $\boldsymbol{U}_i$ belongs to the top ratio% uncertainty points and $w_i = 0$ otherwise.

From $\boldsymbol{U}$, we sample uncertain points with top K uncertainty values. K depends on the size of the corresponding object. As shown in Fig. 4, we first generate a box containing the initial object mask, with 120% of original height and width to avoid under-sampling. Then, we select the points with top 20% uncertainty from $\boldsymbol{U}$, i.e., K is 20% of the box's area.

3.2.2. Point-based feature matching and aggregation

Based on the uncertain points, we select K query feature vectors from X_4^Q and feed them into Q-Key/Value heads to encode query keys ($X^Q_{\text{key},4} \in \mathbb{R}^{K\times 64}$) and values ($X^Q_{\text{value},4} \in \mathbb{R}^{K\times 512}$). To achieve a better balance between the SVOS accuracy and efficiency, we only sample M memory feature points from the first and previous frames for point-based matching and refinement. Specifically, we sample $H\times W/4s^2_{\text{global}}$ points from the first frame, where s_{global} is the interval for stridden sampling. For the previous frame, we only consider a local window ($w_{\text{local}} \times w_{\text{local}}$) around each query feature point. Therefore, $M = H\times W/(4s_{\text{global}})^2 + w^2_{\text{local}}$ for each query feature point. We utilise M-Key/Fusion heads to generate $X^M_{\text{key},4} \in \mathbb{R}^{M\times 64}$ and $X^M_{\text{value},4} \in \mathbb{R}^{M\times 512}$ from X_4^M and $X^M_{\text{v},4}$. For each location i in $X^Q_{\text{key},4}$, its L2 similarity with M corresponding fine-grained memory features $X^M_{\text{key},4,i}$ is measured by:

$$S_{i,j} = ||\boldsymbol{X}^Q_{\text{key},4,i} - \boldsymbol{X}^M_{\text{key},4,i,j}||^2. \tag{2}$$

$S \in \mathbb{R}^{K\times M}$ can be efficiently estimated via the simplified approach in Cheng et al. [13], which formulates L2 distance as the tensor multiplication. Next, the affinity is processed by softmax and used as weights to sum the memory values:

$$\boldsymbol{X}^M_{\text{sum},4} = \text{softmax}(S)_{\text{dim}=1} \times \boldsymbol{X}^M_{\text{value},4}. \tag{3}$$

Then, $\boldsymbol{X}^M_{\text{sum},4} \in \mathbb{R}^{K\times 512}$ is concatenated with $\boldsymbol{X}^Q_{\text{value},4} \in \mathbb{R}^{K\times 512}$ to aggregate the fine-grained query and memory features: $\boldsymbol{X}_{\text{agg},4} \in \mathbb{R}^{K\times 1024}$.

3.2.3. Point-based refinement

As shown in Fig. 3, $\boldsymbol{X}_{\text{agg},4} \in \mathbb{R}^{K\times 1024}$ is processed by three FC layers to further enhance the aggregation and reduce the feature dimensions from 1024 to 256. Then, it is added to the uncertain decoding outputs $Uncertain(\boldsymbol{D}_4) \in \mathbb{R}^{K\times 256}$. Finally, the refinement is achieved by feeding the added features into another module with three FC layers.

3.2.4. Optimisation

As mentioned above, we define all heads and FC layers as a unified point processing module f_{point}, whose parameters γ are optimised together with backbone parameters θ by minimising a combined loss:

$$\begin{aligned}\mathcal{L} \quad &= \tfrac{1}{\sum_{i\in\Omega} w_i} \textstyle\sum_{i\in\Omega} w_i \mathcal{L}_{\text{coarse}}(f_{\text{backbone}}(\boldsymbol{I}^Q, \boldsymbol{I}^M, \theta)_i, \boldsymbol{M}_i) \\ &+\lambda \tfrac{1}{K} \textstyle\sum_{k\in topK(\boldsymbol{U})} \mathcal{L}_{\text{fine}}(f_{\text{point}}(\boldsymbol{D}_4, \boldsymbol{X}, \gamma)_k, \boldsymbol{M}_k),\end{aligned} \tag{4}$$

where $\boldsymbol{X} = \{\boldsymbol{X}_4^Q, \boldsymbol{X}_4^M, \boldsymbol{X}^M_{\text{v},4}\}$ are the fine-grained features, $topK(\boldsymbol{U})$ is a set with K uncertain points. $\mathcal{L}_{\text{coarse}}$ and $\mathcal{L}_{\text{fine}}$ are both cross-entropy loss. $w_i = 1$ if the loss on the location i belongs to the top ratio% losses and $w_i = 0$ otherwise.

3.3. Adaptive matching module

The module aims to mitigate the ambiguity issue when matching between memory and query features. In most memory-based SVOS methods, all memory feature points are considered equally, making these methods vulnerable to ambiguous backgrounds. By contrast, our method builds trajectories from memory to query points. With the trajectories and corresponding confidences, each query feature point will be matched with the relevant memory feature points only, potentially suppressing the ambiguity within memory features.

Given the embedded key features of memory and query frames: $\boldsymbol{X}^M_{\text{key}} \in \mathbb{R}^{T\times H\times W\times 64}$ and $\boldsymbol{X}^Q_{\text{key}} \in \mathbb{R}^{H\times W\times 64}$, existing memory-based methods firstly compute the point-wise similarities between them:

$$S_{p,q} = \text{s}(\boldsymbol{X}^M_{\text{key},p}, \boldsymbol{X}^Q_{\text{key},q}), \tag{5}$$

where H, W, 64, and T represent the height, width, channel, and temporal dimensions. p and q are memory and query feature points. $\text{s}(\cdot)$ usually measures cosine similairty or L2 distance. Upon the similarities, the methods propagate the information implying labels from memory frames to the query frame. For each query point q, the propagation is implemented by aggregating the value features of all points. This procedure can be formulated as:

$$\boldsymbol{X}^M_{\text{agg},q} = \sum_p \left(\frac{W_{p,q}\cdot \exp(S_{p,q})}{\sum_p W_{p,q}\cdot \exp(S_{p,q})} \cdot \boldsymbol{X}^M_{\text{value},p}\right), \tag{6}$$

where $W_{p,q}$ is the parameter to weigh point-wise similarities, used to formulate different matching schemes in a uniform way. The global matching-based methods [5,6,8,9,13] usually fix $W_{p,q}$ as a constant, which indicates all similarities are considered equally during inference. As mentioned in Section 1, these methods cannot well handle ambiguous regions. To mitigate this problem, some recent methods [10,12] consider the spatial-temporal distance between points to define $W_{p,q}$, which increases when p and q are approaching. However, such constraint gradually fades for distant memory frames during inference since the increase of temporal distance makes $W_{p,q}$ related to these frames closer together. This makes sense because the foreground objects from remote frames probably experience heavy changes in location. However, the ambiguity problem remains when matching with remote frames.

We propose an adaptive matching module to alleviate the ambiguity problem while achieving robustness against occlusions and fast motion. Instead of using the constant or time-conditioned weights, our module generates weights for each memory point individually, based on its dynamic property from the original frame to the query frame. Here, memory point indexes the spatially basic component in the memory feature map. Specifically, the weight between a memory point p and query point q is obtained by:

$$W_{p,q} = f_{\text{weight}}(d(p^Q, q), \delta(p^Q)), \tag{7}$$

where p^Q is the tracking location of point p in the query frame. $d(\cdot,\cdot)$ measures the spatial distance between points. $W_{p,q}$ increases when p^Q and q are approaching. For each memory point p, the related $W_{p,q}$ form a matrix: $W_p \in \mathbb{R}^{H\times W}$, which corresponds to the weights between p and all points in the query frame. Due to the dynamic nature of videos, the tracking location p^Q might not always be perfect. Therefore, we measure the tracking confidence $\delta(p^Q)$ to control the distribution of W_p. Specifically, the distribution becomes "sharp" when p^Q is confident. As a result, the weights corresponding to the query points close to p^Q are much

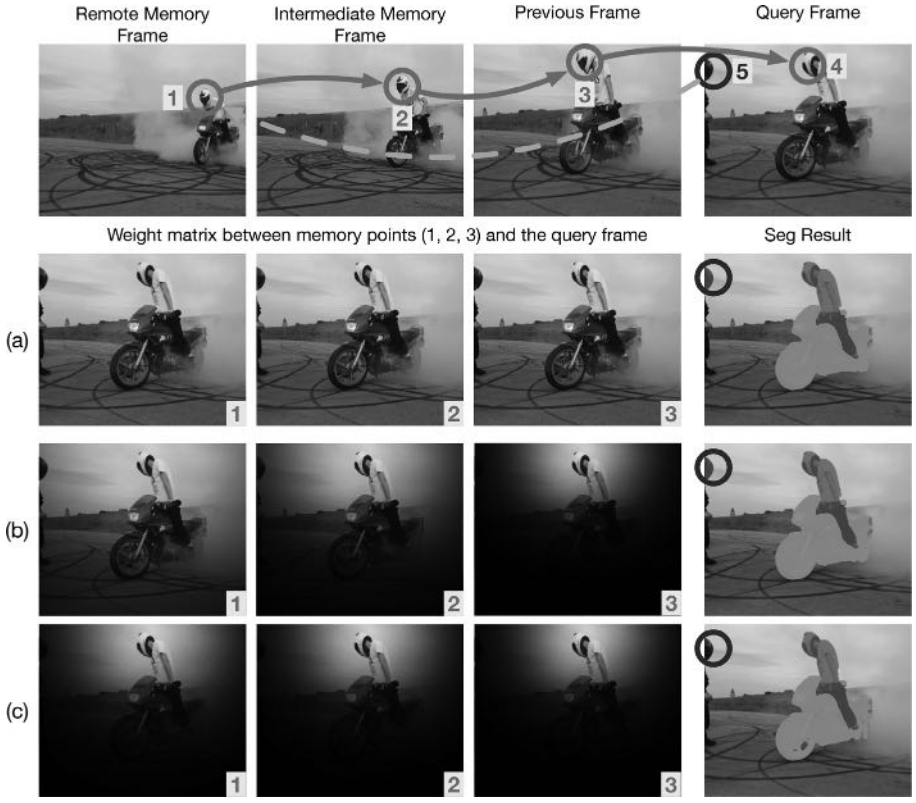

Fig. 5. The main idea of our adaptive matching module and its difference from the existing methods. **Top row**: The red arrows form a point trajectory from a remote memory frame to the query frame (point 1 → point 2 → point3 → point 4). Point 5 (highlighted in blue) looks similar to point 1 (belong to the target object). Therefore it is an ambiguous point. **(a)**: The global matching-based methods (whose $W_{p,q}$ is constant) cannot handle this since they consider all similarities equally. Therefore, when matching with the remote memory frame, the high similarity between points 1 and 5 leads to false label propagation. **(b)**: The existing local matching-based methods (whose $W_{p,q}$ is time-conditioned) cannot handle this since they only apply the distance-based constraint to the recent memory frames. With temporal distance increasing, the weight matrices of remote memory points gradually tend to consider all similarities equally. Therefore, they cannot stop the label propagation from the remote memory frames to the ambiguous query point. **(c)**: By contrast, our adaptive matching module can handle this since we generate $W_{p,q}$ based on the dynamic property of each memory point individually. If a memory point (e.g., point 1) can be tracked confidently to the query frame, the module would apply a distance-based constraint to the corresponding weights even if they come from remote frames. As shown in the first column in (c), the ambiguous point is filtered out by the generated weights. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

higher than others. By contrast, less confident p^{Q} leads to a "soft" distribution, making all the weights in W_p closer together.

From the above weight distributions, it is concluded that we perform local matching for the memory points with confident tracking locations and global matching for the others. The underlying principle is that: For each memory point p, p^{Q} essentially provides a candidate location of p in the query frame. If $\delta(p^{\mathrm{Q}})$ is large, it implies that the semantic element of p probably appears around p^{Q} in the query frame. Therefore, focusing on the local area around p^{Q} is enough to match p. Conversely, p might undergo heavy occlusions or fast motion, resulting in small p^{Q}. In this case, it is hard to locate p from only the local area around p^{Q}. Instead, a non-local region is required. Since the proposed module assigns matching schemes for each memory point individually, we can achieve much more flexibility and adaptivity than most existing methods, which usually use a similar matching plan for all memory points within a frame. Fig. 5 illustrates the proposed module and its difference from the existing methods.

We implement the adaptive matching on the coarsest scale only for computation efficiency. For each memory point, a 2D Gaussian kernel map is generated to represent its weight matrix since the map perfectly matches the properties of weight distributions. Therefore, given a memory point p, we have:

$$W_p = \mathrm{G}_{2\mathrm{d}}(p^{\mathrm{Q}}, \delta(p^{\mathrm{Q}})), \tag{8}$$

where p^{Q} and $\delta(p^{\mathrm{Q}})$ control the centre location and distribution of the weight matrix, respectively. For each memory point p, we generate p^{Q} and $\delta(p^{\mathrm{Q}})$ by accumulating the local correspondence between its original frame and the query frame. Assuming p comes from the tth frame, we first illustrate how to compute the tracking location for p in the $(t+1)$th frame and the related confidence:

$$\begin{cases} p^{t+1} = \operatorname{argmin}_{p'} s(\boldsymbol{X}^{t}_{\mathrm{key},p}, \boldsymbol{X}^{t+1}_{\mathrm{key},p'}) \\ \delta(p^{t+1}) = \beta \exp\left(1 - \frac{s(\boldsymbol{X}^{t}_{\mathrm{key},p}, \boldsymbol{X}^{t+1}_{\mathrm{key},p'})_{1\mathrm{st}}}{s(\boldsymbol{X}^{t}_{\mathrm{key},p}, \boldsymbol{X}^{t+1}_{\mathrm{key},p'})_{2\mathrm{nd}}}\right) \end{cases}, \tag{9}$$

where $s(\boldsymbol{X}^{t}_{\mathrm{key},p}, \boldsymbol{X}^{t+1}_{\mathrm{key},p'})$ measures the key feature similarity between p and p'. $p' \in w$ is the point within a local window w in the $(t+1)$th frame. β is a constant scaling parameter. It is observed that p^{t+1} is located by retrieving the point most similar to it within w. $\delta(p^{t+1})$ corresponds to the uncertainty, measured by the ratio between the first and second highest similarities. The higher $\delta(p^{t+1})$ is, the less the confidence of p^{t+1}. When tracking p across multiple frames, we generate p^{Q} by concatenating the short-term correspondence from all pairs of adjacent frames, which are located between the tth and query (Qth) frames. Then, we select the most uncertain point from the established track and consider its uncertainty as $\delta(p^{\mathrm{Q}})$, i.e., $\delta(p^{\mathrm{Q}}) = \max\{\delta(p^{i+1})\}_{i=t+1}^{\mathrm{Q}}$.

With the generated p^{Q} and $\delta(p^{\mathrm{Q}})$, our proposed matching module can derive more adaptive and flexible $W_{p,q}$ by (8) and therefore boost the memory-based information propagation in (6).

Table 1
Quantitative comparison of different methods on DAVIS-2017 validation and test-dev sets. "–": Not given. The methods marked "*" considered 600p instead of the standard 480p as the input resolution during inference on the test-dev set.

Methods	Years	2017 validation set			2017 test-dev set			FPS
		$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$	
STM [5]*	2019	81.7	79.2	84.3	72.2	69.3	75.2	10.2
KMN [7]*	2020	82.8	80.0	85.6	77.2	74.1	80.3	<8.4
EGMN [6]	2020	82.8	80.2	85.2	–	–	–	<2
LWL [39]	2020	81.6	79.1	84.1	–	–	–	<6.0
AFB-URR [8]	2020	74.6	73.0	76.1	–	–	–	4
CFBI+ [14]*	2021	82.9	80.1	85.7	75.6	71.6	79.6	5.6
LCM [10]	2021	83.5	80.5	86.5	78.1	74.4	81.8	~9.2
RMNet [11]	2021	83.5	81.0	86.0	75.0	71.9	78.1	<11.9
SwiftNet [9]	2021	81.1	78.3	83.9	–	–	–	25
HMMN [12]	2021	84.7	81.9	87.5	78.6	74.7	82.5	8
STCN [13]	2021	85.4	82.2	88.6	76.1	72.7	79.6	20.2
PMNet	**2022**	**86.0**	**82.7**	**89.4**	**78.5**	**74.3**	**82.6**	**14.1**

4. Experiments

We introduce the network structure, training and inference details in Section 4.1. Section 4.2 compares PMNet and state-of-the-art SVOS methods. The relative contribution of each module is studied in Section 4.3.

4.1. Implementation details

Framework We build PMNet on top of STCN [13], where key and value encoders are implemented with the first 4 blocks of ResNet-50 (with a 3×3 convolutional layer) and ResNet-18 [30] (with a 3×3 convolutional layer). Value head is a 3×3 convolutional layer. Fuse module consists of CNN layers and a Convolutional Block Attention Module (CBAM [31]). We employ CNN layers and fully-connected layers to construct the uncertainty detection module and point processing module, respectively. The high-resolution intermediate features are selected from both key and value encoders. The selection is based on the stride. Specifically, the query features (stride = 8/4) are the output of the Block-3/2 of ResNet-50. The memory values (stride = 4) are the output of the Block-2 of ResNet-18. The first layer of ResNet-18 is modified to accept 4-channel input data (video frame + mask).

Training PMNet requires three learnable parameters: (1) backbone network (θ); (2) uncertainty detection module (ϕ); (3) point processing module (δ). Note that we only perform adaptive matching during inference since weighting features would distract the embedding learning. During training, the backbone network and point processing module are learned together, and we train them and the uncertainty detection module alternatively. Specifically, we freeze ϕ to train θ and δ and freeze θ and δ to train ϕ. Similar to other memory-based methods, we pre-train PMNet on image-based datasets [32–36] and then perform the main training on video datasets [37,38]. Since the uncertainty detection and point processing modules rely on the decoding outputs, only the backbone network is optimised during pre-training. During the main training, we initially measure uncertainty from the predicted probabilities until the performance of the uncertainty detection module remains stable. We employ the weighted cross-entropy loss for both pre-training and main training, where the ratios in Eqn 1 and Eqn 4 are set to 100% during initial iterations and then linearly reduced to 15% within subsequent iterations.

Inference Following [13], PMNet segments each video frame sequentially. For each query frame, the memory frames are the past/segmented frames, whose features have been encoded and stored in the memory bank. We consider the first frame and intermediate frames (sampling interval is 5) as the memory frames for the coarsest feature matching. During the point-based refinement, we perform stridden global matching (s_{global} is 2) on the first frame and local matching (within a 15×15 window, therefore $w_{\text{local}} = 15$) on the previous frame. As mentioned in Section 3.2.2, we keep K as the 20% of the corresponding object's area during training and inference. In the adaptive matching module, we compute local correspondence between adjacent frames within a 9×9 ($w = 9$) window on the coarsest feature map. The scaling parameter β in Eq. (9) is (8).

4.2. Comparison with state-of-the-art

This section compares PMNet with state-of-the-art SVOS methods on DAVIS 2017 (validation and test-dev sets [37]) and YouTube-VOS (2018 and 2019 validation sets [38]), the most frequently used testbeds for SVOS evaluation.

DAVIS (Densely Annotated VIdeo Segmentation) 2017 This dataset [37] consists of videos with high-resolution and dense annotations (all video frames are annotated with pixel-level labels), most of which contain multiple target objects and challenges, e.g., occlusion and appearance changes. There are 150 videos in this dataset, where 60 videos form the training set, the other 90 videos are evenly split into the validation, test-dev, and test-challenge sets. In most earlier methods, the validation set is the only DAVIS-2017 subset for SVOS evaluation. Recently, many methods also take the test-dev set into account since it consists of more challenging videos. In this section, we compare our PMNet and state-of-the-art methods on both validation and test-dev sets.

Like other SVOS methods, we use Jaccard–Index (abbreviated as $\mathcal{J}$, the Intersection over Union between object masks) and F-measure (abbreviated as $\mathcal{F}$, distances between contour points) for evaluation. Table 1 demonstrates the quantitative results of our PMNet and state-of-the-art methods on DAVIS-2017 validation and test-dev sets. Besides accuracy, the table also compares the segmentation efficiency (FPS, Frames segmented Per Second) of each method.

The comparison results on the validation set show that our PMNet outperforms the state-of-the-art methods in both region- (+0.5%) and contour-based (+0.8%) measurements, which validate the performance improvement brought by the point-based refinement module. In addition, the adaptive matching module further enhances the overall performance by suppressing the distractions from ambiguous regions. On the test-dev set, our PMNet can achieve competitive results with good computational efficiency. It is observed that HMMN [12] performs slightly better than ours (0.1%). This is mainly because many test-dev videos consist of objects with small areas and detailed structures, potentially increasing the demands for multi-scale feature analysis. Since our PMNet only extracts and utilises fine-grained features for uncer-

Fig. 6. Qualitative comparison between our method and other multi-scale matching-based methods (CFBI+ [14] and HMMN [12]) and our baseline model STCN [13]. Blue boxes highlight the main difference between methods. Numbers deonte the indices of video frames.

tain regions only, its performance improvement on this set is limited. Better results can be computed ($\mathcal{J}\&\mathcal{F}$: 78.8, $\mathcal{J}$: 75.0, $\mathcal{F}$: 82.8) when considering more uncertain regions (e.g., improve K from 20% to 40%). However, such improvement is achieved at the cost of more computation (FPS drops from 14.1 to 12.2). Therefore, we keep K as 20% to better balance the SVOS accuracy and efficiency.

The qualitative results on DAVIS-2017 are shown in Fig. 6. It is observed that our PMNet can handle not only the ambiguous regions but also the challenging details with complex context. To some extent, these results owe to our point-based refinement module, which is mainly learned from hard samples and therefore more robust against uncertain (challenging) regions.

YouTube-VOS Like DAVIS, this dataset [38] also consists of high-resolution videos. However, the number of videos, frames and annotations in YouTube-VOS is much larger than DAVIS. Therefore, YouTube-VOS can activate and evaluate the performance of SVOS methods in long-term modelling and generalisation. Currently, there are two versions of YouTube-VOS datasets: version 2018 (3471 training videos, 474 validation videos) and 2019 (3471 training videos, 507 validation videos), where YouTube-VOS-2019 is extended from YouTube-VOS-2018 by adding more challenging videos and annotations. In this section, we compare our PMNet and state-of-the-art methods on both versions.

To evaluate the generalisation performance, YouTube-VOS divides its validation set into two groups based on object categories: "seen" and "unseen". The object belonging to "seen" categories residents in both training and validation sets, and the ones belonging to "seen" categories only residents in the validation set. Therefore, the metrics in Tables 2 and 3 are still grouped into two subsets: $\mathcal{J}_{\text{seen}}$, $\mathcal{F}_{\text{seen}}$, $\mathcal{J}_{\text{unseen}}$, and $\mathcal{F}_{\text{unseen}}$. From two tables, more significant improvement in $\mathcal{F}_{\text{seen}}$ can be observed than $\mathcal{J}_{\text{seen}}$, which further demonstrates the consistency of the point-based refinement module across different datasets. However, the improvement in $\mathcal{F}_{\text{unseen}}$ is limited; this suggests that the generalisation of our point-based refinement module can be improved further.

Fig. 7 shows the qualitative comparisons on YouTube-VOS. As mentioned in the DAVIS part, the proposed point-based refinement module and adaptive matching module enable our PMNet to handle the videos with different challenges, e.g., detailed structures, complex context, and ambiguous appearance.

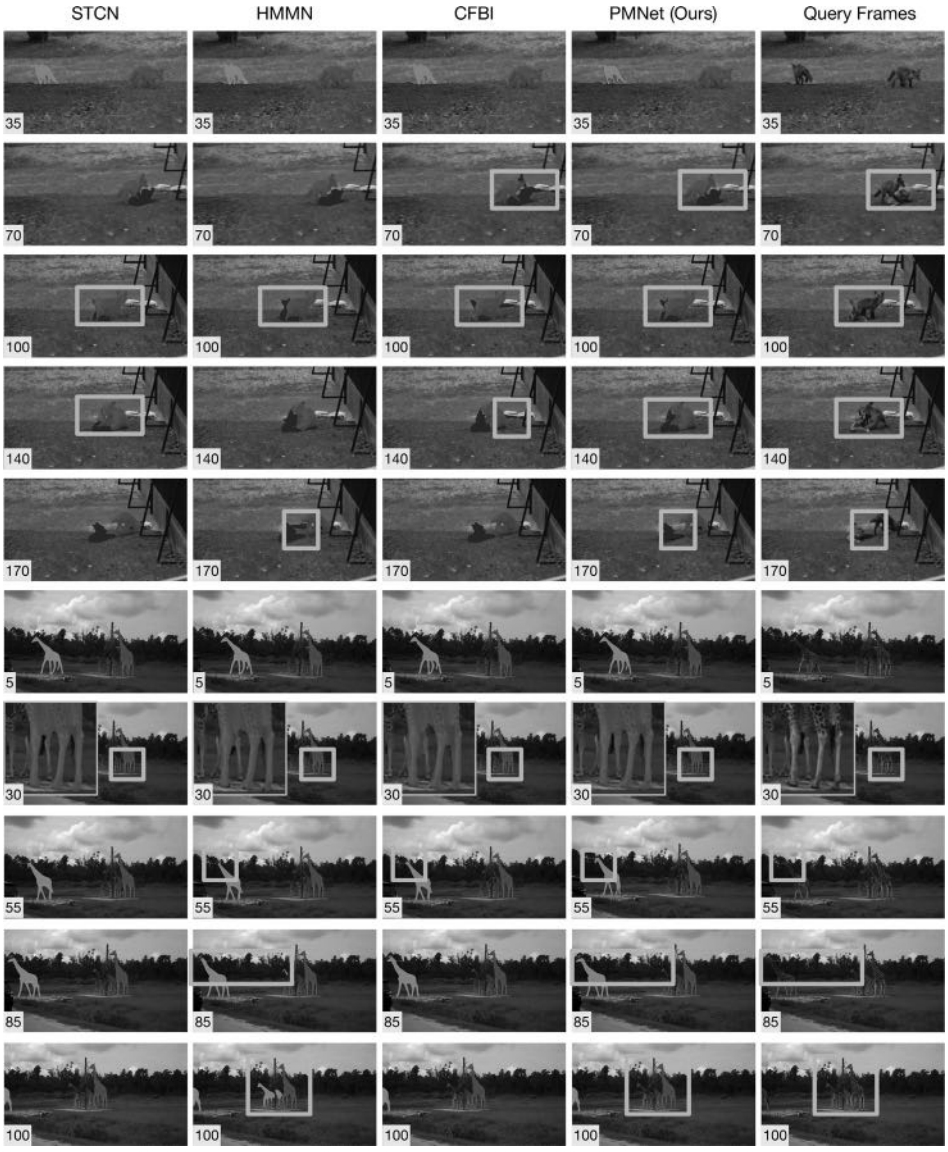

Fig. 7. Qualitative comparison between our method and other multi-scale matching-based methods (CFBI+ [14] and HMMN [12]) and our baseline model STCN [13]. Blue boxes highlight the main difference between methods. Numbers deonte the indices of video frames. Note that YouTube-VOS does not provide ground truth masks for the validation set. Therefore, we list raw video frames only.

Table 2
Quantitative comparison of different methods on YouTube-VOS-2018 validation set.

Methods	Years	$\mathcal{G}$	$\mathcal{J}_{\text{seen}}$	$\mathcal{F}_{\text{seen}}$	$\mathcal{J}_{\text{unseen}}$	$\mathcal{F}_{\text{unseen}}$
STM [5]	2019	79.4	79.7	84.2	72.8	80.9
KMN [7]	2020	81.4	81.4	85.6	75.3	83.3
AFB-URR [8]	2020	79.6	78.8	83.1	74.1	82.6
LWL [39]	2020	80.2	78.3	82.3	75.6	84.4
CFBI+ [14]	2021	82.0	81.2	86.0	76.2	84.6
LCM [10]	2021	82.0	82.2	86.7	75.7	83.4
RMNet [11]	2021	81.5	82.1	85.7	75.7	82.4
SwiftNet [9]	2021	77.8	77.8	81.8	72.3	79.5
HMMN [12]	2021	82.6	82.1	87.0	76.8	84.6
STCN [13]	2021	83.0	81.9	86.5	77.9	85.7
PMNet	**2022**	**83.6**	**82.5**	**87.6**	**78.4**	**85.9**

Table 3
Quantitative comparison of different methods on YouTube-VOS-2019 validation set. Methods marked "*" indicate their scores come from the non-original works.

Methods	Years	$\mathcal{G}$	$\mathcal{J}_{\text{seen}}$	$\mathcal{F}_{\text{seen}}$	$\mathcal{J}_{\text{unseen}}$	$\mathcal{F}_{\text{unseen}}$
STM [5]*	2019	79.4	79.8	83.8	73.0	80.5
KMN [7]*	2020	80.0	80.4	84.5	73.8	81.4
CFBI+ [14]	2021	82.9	80.6	85.2	78.9	86.8
HMMN [12]	2021	82.5	81.7	86.1	77.3	85.0
STCN [13]	2021	82.7	81.1	85.4	78.2	85.9
PMNet	**2022**	**83.2**	**81.9**	**85.7**	**78.7**	**86.6**

4.3. Ablation studies

This section demonstrates the effect of each module in PMNet on segmentation accuracy and efficiency. We choose STCN [13] as the baseline. The ablation studies are performed on the DAVIS-2017 validation and test-dev sets [37].

At first, we verify the effectiveness of the point-based refinement module and compare different methods for uncertain region detection. As shown in Table 4, the module improves both region-based and contour-based performance with acceptable overhead. Compared with the validation set, the module achieves more sig-

Table 4
The effectiveness of the point-based refinement and adaptive matching modules, evaluated on both DAVIS-2017 validation and test-dev sets. "Point", "Probs", "Learn", and "Adapt" indicate the point-based refinement, probability-based uncertain point sampling, learnable uncertain point sampling, and the adaptive matching module, respectively.

DAVIS sets	Point	Probs	Learn	Adapt	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$	FPS
validation	✗	✗	✗	✗	85.4	82.2	88.6	20.2
	√	✗	✗	✗	83.7	81.5	85.9	9.6
	√	√	✗	✗	85.6	82.2	89.0	15.4
	√	✗	√	✗	85.7	82.3	89.2	15.2
	✗	✗	✗	√	85.8	82.6	89.1	14.7
	√	✗	✗	√	84.6	81.7	87.5	8.8
	√	√	✗	√	85.9	82.5	89.3	14.5
	√	✗	√	√	86.0	82.7	89.4	14.1
test-dev	✗	✗	✗	✗	76.1	72.7	79.6	20.2
	√	✗	✗	✗	74.3	72.0	76.6	9.6
	√	√	✗	✗	77.1	72.9	81.3	15.4
	√	✗	√	✗	77.2	73.1	81.4	15.2
	✗	✗	✗	√	78.1	74.4	81.8	14.7
	√	✗	✗	√	76.0	73.4	78.6	8.8
	√	√	✗	√	78.3	74.2	82.3	14.5
	√	✗	√	√	78.5	74.3	82.6	14.1

Table 5
The effect of the point-based refinement module parameters on the segmentation performance, evaluated on the DAVIS-2017 validation (v) and test-dev (t) sets.

Values	Stride (s_{global})			Window size (w_{local})			Sampling ratio (K)		
	1	2	3	13	15	17	10%	20%	40%
$\mathcal{J}\&\mathcal{F}$ (v)	85.3	86.0	85.7	85.8	86.0	85.9	85.8	86.0	86.0
$\mathcal{J}\&\mathcal{F}$ (t)	77.9	78.5	78.2	78.2	78.5	78.3	78.1	78.5	78.8
FPS	12.8	14.1	14.7	14.0	14.1	14.3	14.9	14.1	12.2

Table 6
The effect of the adaptive matching module parameters on the segmentation performance, evaluated on the DAVIS-2017 validation (v) and test-dev (t) sets.

Values	Window size (w)					Scaling factor (β)				
	7	8	9	10	11	6	7	8	9	10
$\mathcal{J}\&\mathcal{F}$ (v)	85.8	85.9	86.0	85.9	85.7	85.8	85.9	86.0	86.0	85.9
$\mathcal{J}\&\mathcal{F}$ (t)	78.1	78.4	78.5	78.3	78.3	77.9	78.2	78.5	78.3	78.2
FPS	14.1	14.1	14.1	14.1	14.1	14.1	14.1	14.1	14.1	14.1

nificant improvement on the test-dev set since it consists of more challenging videos. On both sets, the improvement in $\mathcal{F}$ is higher than $\mathcal{J}$. This is because the point-based refinement module focuses on fine-grained-level feature analysis, which is beneficial for the objects with detailed structures. Although generating uncertainty maps directly from probabilities is more efficient, this method focuses more on object contours, limiting the performance improvement in uncertain regions away from contours. By contrast, the uncertainty detection module in our method is lightweight and only adds the overhead marginally. No matter whether using the adaptive matching module, the learnable module can bring better performance. Therefore, we keep using the learnable module to detect uncertain regions. In addition, we also evaluate the SVOS performance without uncertain region detection, i.e., all feature points on the finest scale are considered during matching. It is observed that both the accuracy and efficiency drop significantly, which shows that in most cases, the coarsest scale can bring good results. By contrast, the compulsive multi-scale fusion probably encourages segmentation models to focus more on the fine-grained features, which have fewer semantic clues and are prone to be misled by similar appearances.

Next, we verify the effectiveness of the adaptive matching module, which is designed to suppress the distractions from ambiguous regions. This module mainly improves the region-based accuracy $\mathcal{J}$, as shown in Table 4. Compared with the validation set, the module achieves more significant improvement on the test-dev set since more distracting scenes are involved in the test-dev videos, which form the main factor causing ambiguous regions. With the point-based refinement module, the overall performance is enhanced further while maintaining good segmentation efficiency.

Table 7
The impact of different training strategies on the segmentation performance, evaluated on the DAVIS-2017 validation (v) and test-dev (t) sets. "Epochs" indicates from which epoch the uncertainty detection module starts to serve the subsequent refinement procedure. The module is trained for a total of 150 epochs. Therefore, the last column means the uncertainty only comes from the predicted probabilities.

Epochs	0	50	100	120	150
$\mathcal{J}\&\mathcal{F}$ (v)	85.1	85.7	86.0	85.8	85.8
$\mathcal{J}\&\mathcal{F}$ (t)	77.6	78.1	78.5	78.2	78.3

Finally, we probe the choice of hyper-parameters in PMNet. Specifically, we analyse the hyper-parameters from the point-based refinement and adaptive matching modules in Tables 5 and 6, respectively. We also analyse the incorporation between the uncertainty detection module and the segmentation model in Table 7, and the choice of similairty function in Table 8. For Tables 5 and 6, the results illustrate that most assignments can generate bet-

Table 8
The impact of different distance measurements on the segmentation performance, evaluated on the DAVIS-2017 validation (v) and test-dev (t) sets.

Distances	L2	Dot product	Cosine
$\mathcal{J}\&\mathcal{F}$ (v)	86.0	84.5	84.3
$\mathcal{J}\&\mathcal{F}$ (t)	78.5	77.6	77.3

ter results than the baseline method (85.4 on the validation set, 76.0 on the test-dev set), further validating the effectiveness of the proposed modules in PMNet. Table 7 probes the best time to incorporate the uncertainty detection module into the segmentation model. The results show that performing incorporation in the middle or later stages can better leverage the uncertainty detection module. In Table 8, it is observed that L2 distance can bring better performance to both the SVOS method based on coarse matching [13] and the one based on multi-scale matching (ours), further validating its effectiveness in the memory-based SVOS.

5. Conclusion

In this paper, we have proposed PMNet for fine-grained SVOS. Compared with other methods based on multi-scale feature matching, our point-based refinement achieves a better balance between SVOS accuracy and efficiency. In addition, the adaptive matching further improves the overall performance by fusing multiple matching schemes. Experimental results on DAVIS and YouTube-VOS show that our method outperforms the state-of-the-art methods. In the future, we can extend this work to achieve further performance improvement, such as more elaborate strategy for uncertainty detection.

Data availability

Video Object Segmentation using Point-based Memory Network (Mendeley Data).

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] T. Hussain, K. Muhammad, W. Ding, J. Lloret, S.W. Baik, V.H.C. de Albuquerque, A comprehensive survey of multi-view video summarization, Pattern Recognit. 109 (2021) 107567.
[2] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, A. Sorkine-Hornung, A benchmark dataset and evaluation methodology for video object segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 724–732.
[3] S.W. Oh, J.-Y. Lee, K. Sunkavalli, S.J. Kim, Fast video object segmentation by reference-guided mask propagation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7376–7385.
[4] P. Voigtlaender, Y. Chai, F. Schroff, H. Adam, B. Leibe, L.-C. Chen, FEELVOS: fast end-to-end embedding learning for video object segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 9481–9490.
[5] S.W. Oh, J.-Y. Lee, N. Xu, S.J. Kim, Video object segmentation using space-time memory networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 9226–9235.
[6] X. Lu, W. Wang, M. Danelljan, T. Zhou, J. Shen, L. Van Gool, Video object segmentation with episodic graph memory networks, in: Proceedings of the European Conference on Computer Vision, Springer, 2020, pp. 661–679.
[7] H. Seong, J. Hyun, E. Kim, Kernelized memory network for video object segmentation, in: Proceedings of the European Conference on Computer Vision, Springer, 2020, pp. 629–645.
[8] Y. Liang, X. Li, N. Jafari, Q. Chen, Video object segmentation with adaptive feature bank and uncertain-region refinement, Advances in Neural Information Processing Systems, 2020.
[9] H. Wang, X. Jiang, H. Ren, Y. Hu, S. Bai, SwiftNet: real-time video object segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2021, pp. 1296–1305.
[10] L. Hu, P. Zhang, B. Zhang, P. Pan, Y. Xu, R. Jin, Learning position and target consistency for memory-based video object segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2021, pp. 4144–4154.
[11] H. Xie, H. Yao, S. Zhou, S. Zhang, W. Sun, Efficient regional memory network for video object segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2021, pp. 1286–1295.
[12] H. Seong, S.W. Oh, J.-Y. Lee, S. Lee, S. Lee, E. Kim, Hierarchical memory matching network for video object segmentation, in: Proceedings of the IEEE International Conference on Computer Vision, 2021, pp. 12889–12898.
[13] H.K. Cheng, Y.-W. Tai, C.-K. Tang, Rethinking space-time networks with improved memory coverage for efficient video object segmentation, Advances in Neural Information Processing Systems, 2021.
[14] Z. Yang, Y. Wei, Y. Yang, Collaborative video object segmentation by multi-scale foreground-background integration, IEEE Trans. Pattern Anal. Mach. Intell. 44 (2021) 4701–4712.
[15] A. Kirillov, Y. Wu, K. He, R. Girshick, Pointrend: Image segmentation as rendering, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 9799–9808.
[16] G. Zhang, X. Lu, J. Tan, J. Li, Z. Zhang, Q. Li, X. Hu, RefineMask: towards high-quality instance segmentation with fine-grained features, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2021, pp. 6861–6869.
[17] Q. Fan, F. Zhong, D. Lischinski, D. Cohen-Or, B. Chen, JumpCut: non-successive mask transfer and interpolation for video cutout, ACM Trans. Graph. 34 (6) (2015) 1–10. 195
[18] W. Wang, J. Shen, F. Porikli, R. Yang, Semi-supervised video object segmentation with super-trajectories, IEEE Trans. Pattern Anal. Mach. Intell. 41 (4) (2018) 985–998.
[19] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, L. Van Gool, One-shot video object segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 221–230.
[20] P. Voigtlaender, B. Leibe, Online adaptation of convolutional neural networks for video object segmentation, in: Proceedings of the British Machine Vision Conference, 2017.
[21] K.-K. Maninis, S. Caelles, Y. Chen, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, L. Van Gool, Video object segmentation without temporal information, IEEE Trans. Pattern Anal. Mach. Intell. 41 (6) (2018) 1515–1530.
[22] F. Perazzi, A. Khoreva, R. Benenson, B. Schiele, A. Sorkine-Hornung, Learning video object segmentation from static images, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2663–2672.
[23] J. Han, L. Yang, D. Zhang, X. Chang, X. Liang, Reinforcement cutting-agent learning for video object segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 9080–9089.
[24] M. Sun, J. Xiao, E.G. Lim, Y. Xie, J. Feng, Adaptive ROI generation for video object segmentation using reinforcement learning, Pattern Recognit. 106 (2020) 107465.
[25] N. Xu, L. Yang, Y. Fan, J. Yang, D. Yue, Y. Liang, B. Price, S. Cohen, T. Huang, YouTube-VOS: sequence-to-sequence video object segmentation, in: Proceedings of the European Conference on Computer Vision, 2018, pp. 585–601.
[26] Y. Yin, D. Xu, X. Wang, L. Zhang, AGUnet: annotation-guided U-net for fast one-shot video object segmentation, Pattern Recognit. 110 (2021) 107580.
[27] J. Cheng, Y.-H. Tsai, W.-C. Hung, S. Wang, M.-H. Yang, Fast and accurate online video object segmentation via tracking parts, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7415–7424.
[28] Y. Chen, J. Pont-Tuset, A. Montes, L. Van Gool, Blazingly fast video object segmentation with pixel-wise metric learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1189–1198.
[29] Z. Yang, Y. Wei, Y. Yang, Collaborative video object segmentation by foreground-background integration, in: Proceedings of the European Conference on Computer Vision, Springer, 2020, pp. 332–348.
[30] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[31] S. Woo, J. Park, J.-Y. Lee, I.S. Kweon, CBAM: convolutional block attention module, in: Proceedings of the European Conference on Computer Vision, 2018, pp. 3–19.
[32] J. Shi, Q. Yan, L. Xu, J. Jia, Hierarchical image saliency detection on extended CSSD, in: IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 38, 2015, pp. 717–729.
[33] L. Wang, H. Lu, Y. Wang, M. Feng, D. Wang, B. Yin, X. Ruan, Learning to detect salient objects with image-level supervision, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 136–145.
[34] X. Li, T. Wei, Y.P. Chen, Y.-W. Tai, C.-K. Tang, FSS-1000: a 1000-class dataset for few-shot segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2020.
[35] Y. Zeng, P. Zhang, J. Zhang, Z. Lin, H. Lu, Towards high-resolution salient object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2019.
[36] H.K. Cheng, J. Chung, Y.-W. Tai, C.-K. Tang, CascadePSP: toward class-agnostic and very high-resolution segmentation via global and local refinement, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2020.

[37] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, L. Van Gool, The 2017 davis challenge on video object segmentation, arXiv preprint arXiv:1704.00675(2017).
[38] N. Xu, L. Yang, Y. Fan, D. Yue, Y. Liang, J. Yang, T. Huang, YouTube-VOS: a large-scale video object segmentation benchmark, arXiv preprint arXiv:1809.03327(2018).
[39] G. Bhat, F.J. Lawin, M. Danelljan, A. Robinson, M. Felsberg, L.V. Gool, R. Timofte, Learning what to learn for video object segmentation, in: Proceedings of the European Conference on Computer Vision, Springer, 2020, pp. 777–794.

Mingqi Gao is currently a joint Ph.D. candidate with the University of Warwick and Southern University of Science and Technology (SUSTech). His research interests include computer vision and video analysis.

Jungong Han is currently a Chair Professor and the Director of Research of Computer Science, Aberystwyth University, U.K. He also holds an Honorary Professorship with the University of Warwick, U.K. His research interests include computer vision, artificial intelligence, and machine learning.

Feng Zheng is currently an Associate Professor with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China. His research interests include machine learning, computer vision, and human–computer interaction.

James J. Q. Yu is currently an Assistant Professor with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China. His research interests include smart cities development and deep learning techniques.

Giovanni Montana is currently a Chair Professor with the University of Warwick, U.K. His research interests include data science, machine learning, and digital healthcare.