electronics

MDPI

Article

Sequential Clique Optimization for Unsupervised and Weakly Supervised Video Object Segmentation

Yeong Jun Koh [1,*], Yuk Heo [2] and Chang-Su Kim [2]

[1] Department of Computer Science & Engineering, Chungnam National University, Daejeon 34134, Korea
[2] School of Electrical Engineering, Korea University, Seoul 02841 Korea
* Correspondence: yjkoh@cnu.ac.kr

Abstract: A novel video object segmentation algorithm, which segments out multiple objects in a video sequence in unsupervised or weakly supervised manners, is proposed in this work. First, we match visually important object instances to construct salient object tracks through a video sequence without any user supervision. We formulate this matching process as the problem to find maximal weight cliques in a complete k-partite graph and develop the sequential clique optimization algorithm to determine the cliques efficiently. Then, we convert the resultant salient object tracks into object segmentation results and refine them based on Markov random field optimization. Second, we adapt the sequential clique optimization algorithm to perform weakly supervised video object segmentation. To this end, we develop a sparse-to-dense network to convert the point cliques into segmentation results. The experimental results demonstrate that the proposed algorithm provides comparable or better performances than recent state-of-the-art VOS algorithms.

Keywords: video object segmentation; primary object segmentation; salient object detection; sequential clique optimization; convolutional neural networks

Citation: Koh, Y.J.; Heo, Y.; Kim, C.-S. Sequential Clique Optimization for Unsupervised and Weakly Supervised Video Object Segmentation. *Electronics* **2022**, *11*, 2899. https://doi.org/10.3390/electronics11182899

Academic Editor: Riccardo Bernardini

Received: 17 August 2022
Accepted: 9 September 2022
Published: 13 September 2022

1. Introduction

Video object segmentation (VOS) is the task of classifying each pixel in video frames into target objects or backgrounds. VOS can be categorized according to the level of user supervision: *unsupervised*, *weakly supervised*, *semi-supervised*, and *interactive* VOS. Unsupervised VOS, in general, attempts to segment out primary objects from the background without any user annotations, where a 'primary' object [1] refers to the most salient one in a video. In contrast, users can provide annotations to facilitate VOS and obtain desired segmentation results. In interactive VOS, a user provides annotations repeatedly to refine results. Semi-supervised VOS tracks and segments a target object manually annotated in the first frame. Finally, weakly supervised VOS requires weaker supervision (e.g., points [2], scribbles, boxes [3,4], and video tags [5,6]) than pixel-level accurate annotations for target objects.

Many VOS algorithms adopt deep learning models with advances in regularization techniques [7–10]. One early unsupervised VOS algorithm [11] uses motion and appearance information to produce segmentation results automatically, but it does not consider the appearance frequency of objects. In other words, it may fail to detect primary objects, which have less distinct features but appear frequently in the sequence. Some algorithms [12,13] address these problems by considering the frequently appearing characteristics of primary objects, but they are designed to extract only a single primary object. Thus, they have the common limitation that they cannot handle multiple primary objects systematically. The semi-supervised VOS algorithms in [14,15] can address the multi-object problem using annotations in the first frames, but they need to fine-tune the networks for each video sequence, which is computationally expensive. Even though recent semi-supervised algorithms [16,17] do not need fine-tuning, the semi-supervised approach still demands that annotators provide accurate pixel-level masks for target objects in the first frames, which is impractical.

In this paper, we extend the work in a conference paper [18] to develop a novel unsupervised VOS algorithm, which segments out multiple primary objects without any mask annotations, and a weakly supervised VOS algorithm, which may reduce the efforts for masking target objects in the first frame. For this purpose, we develop sequential clique optimization (SCO), which can be employed in both unsupervised and weakly supervised VOS. First, we extract object instances in each frame based on the instance-wise segmentation technique [19]. Then, we perform instance matching in order to construct salient object tracks. This is similar to finding multiple cliques in a complete k-partite graph [20] of object instances. Each clique should contain the instances over frames, corresponding to an identical object. Thus, the instances should be similar to one another. However, finding the optimal multiple cliques with maximal similarity weights is NP-hard. Hence, we develop the clique optimization process, called SCO, which considers both node and edge energies. SCO constructs the most salient object track by selecting one object instance from each frame, and multiple salient object tracks can be extracted by repeating the process. We convert these salient object tracks into VOS results. Then, we perform the segmentation refinement based on two-class Markov random field (MRF) optimization to improve the segmentation results. Furthermore, the proposed SCO can be adapted to perform weakly supervised VOS, which accepts only a number of point clicks on a target object in the first frame. To this end, we develop the sparse-to-dense network (SD-Net) to obtain dense segmentation masks from sparse points. The experimental results demonstrate that the proposed unsupervised and weakly supervised algorithms provide comparable or better performances than the recent state-of-the-art VOS algorithms on the DAVIS [21] benchmark dataset.

The main contributions of this paper are summarized as follows:

- We propose the SCO process to extract multiple primary objects effectively. It determines a clique efficiently with $O(NT^2)$ complexity, where T is the number of frames in a video, and N is the number of instances in each frame.
- The proposed algorithm can extract multiple primary objects effectively, whereas most conventional algorithms assume a single primary object.
- We extend the preliminary work [18] of this paper to achieve weakly supervised VOS using the SD-Net, which yields segmentation results using only a few point clicks instead of dense masks for target objects. The proposed SD-Net also improves the unsupervised VOS results of the preliminary work.
- We develop two segmentation refinement methods to improve the unsupervised VOS results based on MRF optimization and the SD-Net.

2. Related Work

2.1. Unsupervised VOS

The objective of unsupervised VOS is to separate foreground objects from the background in a video sequence without any user supervision, such as annotation masks. Many unsupervised VOS algorithms assume that there exists a primary object in a video sequence and attempt to extract such a single primary object. To this end, various cues, including motion boundaries [22], saliency maps [23], and object proposals [12], have been used to localize and segment a primary object. Papazoglou and Ferrari [22] constructed Gaussian mixture models (GMMs) for foreground and background from the regions delineated by motion boundaries and then used the models to segment out moving objects. By adopting the boundary before image boundary regions tend to become backgrounds, Jang [23] computed foreground and background probability maps. They also developed the alternate convex optimization to minimize a hybrid energy function, including the antagonistic energy. Object proposal techniques also have been employed to estimate the initial regions of a primary object. Koh [12] estimated the initial primary regions from object proposals and refined them based on the augmentation and reduction process.

Deep learning techniques have been employed for unsupervised VOS. Jain [11] proposed end-to-end networks to yield pixel-wise segmentation results. They considered

both appearance and motion information. Tokmakov [24] developed fully convolutional networks to learn motion patterns. They used synthetic video sequences to augment training data. In [13,25,26], differentiable attention models have been adopted to detect recurring primary objects. For instance, Lu [13] developed the co-attention Siamese network. Yang [25] computed dense correspondences between frames in an embedding space to segment out foreground objects. Zhou [26] proposed the motion-attentive transition network, which transforms appearance features into motion features at each convolutional stage. Additionally, human eye gaze has been predicted to initialize primary object regions [27]. Zhuo [28] achieved unsupervised online VOS by combining salient motion detection results and object proposals. Ventura [29] employed recurrent neural networks to consider the spatiotemporal features of primary objects. Moreover, object detection methods [30,31] can be adopted for unsupervised VOS.

2.2. Semi-Supervised VOS

Semi-supervised VOS segments and tracks target objects, which are annotated by users in the first frames. Recently, deep learning models have been adopted in semi-supervised VOS. For example, in [14], CNNs were fine-tuned by exploiting user annotations in the first frames. In [32], segmentation masks, propagated from previous frames, were refined by deep networks. Sun [33] performed reinforcement learning to obtain a reliable region of interest. For fast VOS, Yin [34] reduced training iterations to fine-tune the deep neural networks, which tune the target objects. Furthermore, some algorithms [16,17] perform segmentation without fine-tuning. Chen [16] classified pixels based on a nearest-neighbor criterion by employing features extracted from embedding networks. Voigtlaender [17] performed end-to-end embedding learning of the multiple object segmentation task with a cross-entropy loss.

3. Proposed Unsupervised VOS Algorithm

We segment out foreground objects in video frames $\mathcal{I} = \{I_1, \ldots, I_T\}$, where I_t is the tth frame, and T is the number of frames in the sequence. The output is the corresponding sequence of pixel-wise maps, which locate the foreground objects in the frames. Figure 1 shows an overview of the proposed algorithm. First, we generate the object instances in each frame. Second, we construct a complete k-partite graph using the set of object instances. Gray lines in Figure 1 represent positive edges that connect the object instances in different frames. Third, we extract the salient object tracks by finding cliques in the graph and refine each object track. Finally, we convert the salient object tracks into VOS results.

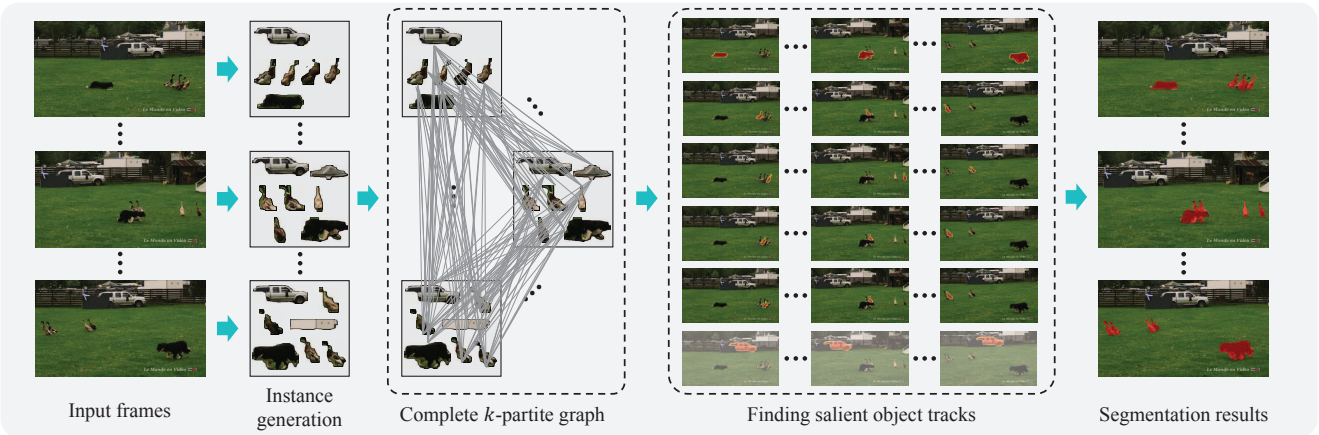


Figure 1. An overview of the proposed algorithm.

3.1. Generating Object Instances

To detect object instances without requiring user annotations, we employ the instance-aware semantic segmentation method, FCIS [19]. Figure 2b illustrates examples of object instances in a frame in the "Boxing-fisheye" sequence. Let $\mathcal{O}_t = \{o_{t,\theta} \mid \theta \in \mathbb{N}_{N_t}\}$ denote

the set of detected object instances in frame I_t, where $\mathbb{N}_m = \{1, 2, \ldots, m\}$ is the finite index set, and N_t is the number of object instances in frame I_t. In general, each frame contains a different number of object instances. The θth object instance $o_{t,\theta}$ in frame I_t has two attributes: the saliency score $s_{t,\theta}$ and feature vector $\mathbf{f}_{t,\theta}$.

(**a**) (**b**) (**c**)

Figure 2. Object instance generation in the "Boxing-fisheye" sequence: (**a**) input frame, (**b**) object instances, and (**c**) foreground distribution.

For the saliency score $s_{t,\theta}$, we adopt the boundary prior to estimating a foreground distribution map for frame I_t. More specifically, we divide I_t into SLIC superpixels [35]. Then, we construct a three-ring graph of the superpixels; two superpixels v_i and v_j are connected if there is a sequence $(v_i = v_{i_1}, v_{i_2}, \ldots, v_{i_k} = v_j)$ of boundary-sharing superpixels from v_i to v_j for $k \leq 4$. The edge weights between two connected superpixels are computed by summing up the difference between the average LAB colors and the difference between the average optical flow vectors [36]. By assigning nonzero restart probabilities to the superpixels on the image boundary, we obtain a background distribution map using the random walk with restart (RWR) simulation. Finally, we obtain the foreground distribution map by inverting the background distribution map, as illustrated in Figure 2c. We then determine $s_{t,\theta}$ by averaging the foreground probability values of the pixels within the instance $o_{t,\theta}$.

Moreover, we use the bag of visual words (BoW) method to define the feature vector $\mathbf{f}_{t,\theta}$. To obtain the bag of visual words, we extract the LAB colors from 40 video sequences and quantize them into 300 codewords using the K-means algorithm. We then build the histogram of the codewords for the pixels within $o_{t,\theta}$ and normalize it into the feature vector $\mathbf{f}_{t,\theta}$.

3.2. Finding Salient Object Tracks

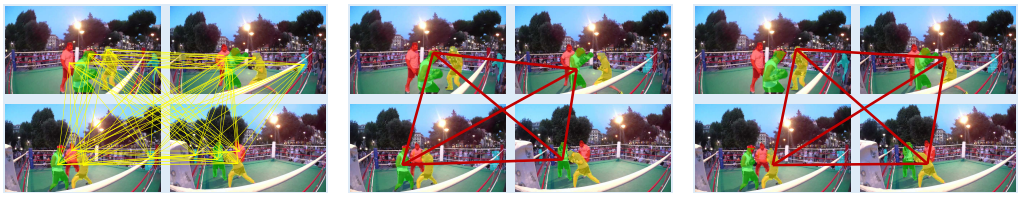
3.2.1. Problem Formulation

The set of all object instances from the video sequence, $\mathcal{V} = \mathcal{O}_1 \cup \mathcal{O}_2 \cup \cdots \cup \mathcal{O}_T$, includes many non-salient objects, as well as salient ones. From this set, we attempt to extract as many salient objects as possible, while excluding non-salient ones, assuming that a salient object should have distinct features and appear frequently throughout the entire video sequence.

We construct a complete k-partite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ using the set of object instances $\mathcal{V} = \mathcal{O}_1 \cup \mathcal{O}_2 \cup \cdots \cup \mathcal{O}_T$ as the node set [20]. Thus, each object instance becomes a node in the graph $\mathcal{G}$. Note that the sets $\mathcal{O}_1, \mathcal{O}_2, \cdots, \mathcal{O}_T$ form a partition of $\mathcal{V}$, since $\mathcal{O}_t \cap \mathcal{O}_\tau =$ for $t \neq \tau$. Moreover, we define the edge set as $\mathcal{E} = \{(o_{t,i}, o_{\tau,j}) \mid t \neq \tau\}$. In other words, every pair of object instances in different frames are connected by an edge in $\mathcal{E}$, whereas two object instances in the same frame are not connected in graph $\mathcal{G}$. As a result, $\mathcal{G}$ is complete k-partite [20], where $k = T$. For example, Figure 3a illustrates the complete k-partite graph for four frames, i.e., $k = 4$. We assign a weight to the edge $(o_{t,i}, o_{\tau,j})$ by

$$w(o_{t,i}, o_{\tau,j}) = \exp\left(-\frac{d_{\chi^2}(\mathbf{f}_{t,i}, \mathbf{f}_{\tau,j})}{\sigma^2}\right) \tag{1}$$

where d_{χ^2} denotes the chi-square distance, which is often used for comparing two histograms, and $\sigma^2 = 0.01$ is a scaling parameter.



(**a**) (**b**) (**c**)

Figure 3. Illustration of finding salient object tracks over four frames in the "Boxing-fisheye" sequence: (**a**) complete 4-partite graph, (**b**) 1st salient object track Θ_1, and (**c**) 2nd salient object track Θ_2.

We perform the instance matching in order to construct multiple salient object tracks, where each salient object track is obtained by selecting one object instance (one node) from each frame (each node subset $\mathcal{O}_t$). This process of finding object tracks is equivalent to finding multiple cliques in the complete k-partite graph $\mathcal{G}$. Notice that selecting one node from each frame satisfies the condition of a clique [20]: Every pair of nodes within the clique is connected. When the clique corresponds to the track of an identical object in the video sequence, the features of the member nodes should be similar to one another. Therefore, we select the clique to maximize the sum of the edge weights in Equation (1).

Let $\Theta_p = \{\theta_t\}_{t=1}^{T}$ denote the pth clique, which is represented by the sequence of node indices. Here, $\theta_t \in \mathbb{N}_{N_t}$ is the index of the selected node from the instance set $\mathcal{O}_t$ at the tth frame. Then, we define the similarity $E_{\text{similarity}}(\Theta_p)$ of the clique Θ_p as

$$E_{\text{similarity}}(\Theta_p) = \sum_{t=1}^{T} \sum_{\tau=1,\tau\neq t}^{T} w(o_{t,\theta_t}, o_{\tau,\theta_\tau}) \tag{2}$$

which is the sum of all edge weights in Θ_p. Assuming that the features of an identical object do not change drastically across the frames, the clique Θ_p should have a maximal similarity score. Moreover, object instances in the clique, representing a salient object track, should have high saliency scores. We hence define the saliency $E_{\text{saliency}}(\Theta_p)$ of the clique Θ_p as

$$E_{\text{saliency}}(\Theta_p) = \sum_{t=1}^{T} s_{t,\theta_t}. \tag{3}$$

We then attempt to find the set of maximal weight cliques $\mathbf{\Theta}^* = \{\Theta_p^*\}_{p=1}^{M}$ that maximizes the sum of the similarity scores,

$$\mathbf{\Theta}^* = \arg\max_{\mathbf{\Theta}} \sum_{p=1}^{M} E_{\text{similarity}}(\Theta_p) \tag{4}$$

subject to the constraint that Θ_p is more salient than Θ_q if $p < q$. However, even the unconstrained problem in Equation (4) is NP-hard [37]. There are $N_1 \times N_2 \times \cdots \times N_T$ possible cliques, which makes an exhaustive search unfeasible. Some approximation methods, e.g., the local search [37] and binary integer program [38], have been developed to obtain suboptimal cliques in complete k-partite graphs; however, these methods are still computationally expensive and do not consider node energy (e.g., E_{saliency} in this work). Instead, we develop an efficient optimization technique, called sequential clique optimization (SCO), to find the clique that considers both the node energy E_{saliency} and the edge energy $E_{\text{similarity}}$.

3.2.2. SCO

For efficient instance matching, we propose SCO, which extracts the most salient object track. It selects an object instance in each frame that corresponds to one identical salient object. Then, after removing all instances in the track from $\mathcal{V}$, we repeat the process to extract the next salient object track, and so on. In this section, we consider the finding of

one clique, Θ_p, which represents the most salient object track, and omit the subscript p from all notations for the sake of simplicity.

In SCO, we first initialize the clique $\Theta^{(0)}$ to maximize the saliency E_{saliency} in Equation (3). Specifically, the tth element in $\Theta^{(0)}$ is determined by

$$\theta_t^{(0)} = \arg\max_{\theta \in \mathbb{N}_{N_t}} s_{t,\theta}. \tag{5}$$

Then, at iteration κ, we update $\theta_t^{(\kappa)}$ by selecting the node that is the most similar to the nodes in the other frames,

$$\theta_t^{(\kappa)} = \arg\max_{\theta \in \mathbb{N}_{N_t}} \sum_{\tau=1, \tau \neq t}^{T} w(o_{t,\theta}, o_{\tau,\theta_\tau}), \tag{6}$$

and then set θ_t to be $\theta_t^{(\kappa)}$ for each t sequentially from one to T. We repeat this sequential update of the nodes in all frames until $\Theta^{(\kappa)} = \{\theta_t^{(\kappa)}\}_{t=1}^{T}$ is unaltered from $\Theta^{(\kappa-1)} = \{\theta_t^{(\kappa-1)}\}_{t=1}^{T}$. This process is theoretically guaranteed to converge since $E_{\text{similarity}}(\Theta^{(\kappa)})$ is a monotonically increasing function of κ. To summarize, SCO initializes the clique to maximize the saliency E_{saliency} and then refines it iteratively to achieve a local maximum of the $E_{\text{similarity}}$. Thus, at the initialization, the clique consists of salient object instances across the frames, which may not represent an identical object. However, as the iteration goes on, the clique converges to a salient object track, in which the nodes represent an identical object and thus exhibit high similarity weights in general. Algorithm 1 summarizes the proposed SCO process. In most cases, less than 10 iterations are required for the convergence.

Algorithm 1 (SCO) Sequential Clique Optimization

Input: Sets of object instances $\mathcal{V} = \mathcal{O}_1 \cup \mathcal{O}_2 \cup \cdots \cup \mathcal{O}_T$

Construct a complete k-partite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
for each frame I_t **do**
  Initialize the node index in clique Θ via
  $\theta_t \leftarrow \arg\max_{\theta \in \mathbb{N}_{N_t}} s_{t,\theta}$
end for
repeat
  for each frame I_t **do**
    Update the node index via
    $\theta_t \leftarrow \arg\max_{\theta \in \mathbb{N}_{N_t}} \sum_{\tau=1, \tau \neq t}^{T} w(o_{t,\theta}, o_{\tau,\theta_\tau})$
  end for
until node indices are unaltered

Output: Optimized clique $\Theta = \{\theta_1, \theta_2, \ldots, \theta_T\}$

Let Θ_1 denote the most salient object track, obtained by this SCO process. To extract the next track Θ_2, we exclude the nodes in Θ_1 from $\mathcal{G}$ and perform SCO again. This is repeated to yield the set of tracks $\{\Theta_1, \Theta_2, \ldots, \Theta_M\}$ until no node remains in $\mathcal{G}$. In general, if $p < q$, Θ_p is more salient than Θ_q. Thus, the subscript p in Θ_p is the saliency rank of the track. Figure 3b,c illustrate the first two tracks Θ_1 and Θ_2, respectively.

3.2.3. Salient Object Track Refinement

The track selection is greedy in the sense that, if an object instance is mistakenly included in a track Θ_p, it cannot be included in a later track Θ_q even when it indeed belongs to Θ_q. To alleviate this problem, we perform postprocessing to maximize the sum of the similarity scores

$$\sum_{p=1}^{M} E_{\text{similarity}}(\Theta_p) \tag{7}$$

as follows. In each frame I_t, we match the object instances in $\mathcal{O}_t$ to the tracks in $\{\Theta_p\}_{p=1}^{M}$. The matching cost $C(o_{t,i}, \Theta_p)$ between an instance $o_{t,i}$ in $\mathcal{O}_t$ and a track Θ_p is defined as the sum of the feature distances from $o_{t,i}$ to all object instances in Θ_p, except for the instance in the same frame I_t. After computing the matching costs, we find the optimal matching pairs using the Hungarian algorithm and update the tracks to include the matched instances. This is performed for all frames. As a result, we obtain the set of refined salient object tracks $\{\tilde{\Theta}_1, \tilde{\Theta}_2, \ldots, \tilde{\Theta}_M\}$.

3.2.4. Disappearance Detection

Next, we detect object disappearing events in each refined salient object track. Notice that, when an object disappears or is fully occluded in some frames, noisy objects are selected instead in those frames. For simple notations, let $\tilde{\Theta} = \{\tilde{\theta}_t\}_{t=1}^{T}$ denote a refined salient object track. We determine whether to discard $o_{t,\tilde{\theta}_t}$ at frame I_t from $\tilde{\Theta}$. To this end, for each $\tau \neq t$, we compare the edge weight $w(o_{\tau,\tilde{\theta}_\tau}, o_{t,\tilde{\theta}_t})$ against the average edge weight $\bar{w}$ in the track. Specifically, we count the number of object instances $o_{\tau,\tilde{\theta}_\tau}$ for $\tau \neq t$, which satisfies $w(o_{\tau,\tilde{\theta}_\tau}, o_{t,\tilde{\theta}_t}) < \bar{w}$. If the number is larger than $0.7T$, we declare $o_{t,\tilde{\theta}_t}$ as noisy and discard it. On the other hand, the proposed algorithm can detect reappearing objects automatically, since every pair of object instances in different frames are fully connected in the complete k-partite graph. In other words, reappearing object instances are connected to all object instances in different frames, so that they are selected by SCO, in general, without requiring any postprocessing.

Figure 4 shows an example of object disappearance and reappearance. In this example, a bicycle and its rider are the primary objects. In Figure 4c, the rider on the bicycle disappears in a frame due to occlusion by another human body, which is a noisy object. The proposed disappearance detection method identifies the noisy human body and excludes it from the salient object tracks. Moreover, in Figure 4d, the proposed SCO automatically recognizes the object's reappearance and declares the reappearing rider and bicycle as the primary objects.

(**a**) (**b**) (**c**) (**d**)

Figure 4. A rider on a bicycle disappears and reappears in the "BMX-bumps" sequence. (**a**) Frame 55; (**b**) Frame 58; (**c**) Frame 70; (**d**) Frame 78.

3.3. Segmentation Results

3.3.1. Object Track Selection

We develop four schemes to choose segmentation results from the object tracks in $\{\tilde{\Theta}_1, \ldots, \tilde{\Theta}_M\}$: SCO-F, SCO-M, SCO-OF, and SCO-OM.

- **SCO-F:** The first track $\tilde{\Theta}_1$ extracts the primary object in a video in general. Thus, SCO-F selects $\tilde{\Theta}_1$. However, it may fail to extract spatially connected objects. For example, given a motorbike and its rider, it may detect only one of them. Therefore, SCO-F additionally picks another salient object track $\tilde{\Theta}_p$, only when $\tilde{\Theta}_1$ and $\tilde{\Theta}_p$ are spatially adjacent in most frames in a video.
- **SCO-M:** To handle multiple primary objects, which may not be spatially connected, we choose multiple tracks from $\{\tilde{\Theta}_1, \tilde{\Theta}_2, \ldots, \tilde{\Theta}_M\}$. To this end, we compute the mean saliency score of the object instances in each track and discard the tracks whose mean scores are lower than a pre-specified threshold δ. We fix $\delta = 0.1$ in all experiments.
- **SCO-OF:** The aforementioned SCO-F is an offline approach that constructs the global T-partite graph for an entire video. In contrast, SCO-OF is an online approach that uses the t-partite graph for frames $I_1, \ldots, I_t$ to obtain the segmentation result for the

current frame I_t. In other words, SCO-O uses the information in the current and past frames only to achieve VOS.

- **SCO-OM:** SCO-OM is an online approach of SCO-M.

3.3.2. Pixel-Wise Segmentation Refinement

The foreground region masks of the object instances from FCIS are noisy in general since the pooling layers in FCIS cause the loss of object details. We hence refine the region masks of the object instances at the pixel level based on the MRF optimization in [32]. In this section, we consider the refinement of an object instance $o_{t,\theta}$ in frame I_t.

We design a weighted graph for MRF, by employing the pixel set $\mathcal{X}$ in frame I_t as the node set. We construct the edge set $\mathcal{H}$ by connecting each pixel $\mathbf{x} \in \mathcal{X}$ to its four neighbors. Then, we determine the segmentation label map $\mathcal{L}$ by dichotomizing each pixel $\mathbf{x}$ into either foreground ($\mathcal{L}(\mathbf{x}) = 1$) or background ($\mathcal{L}(\mathbf{x}) = 0$) based on the cost function

$$C(\mathcal{L}) = \sum_{\mathbf{x} \in \mathcal{X}} C_{\text{unary}}(\mathbf{x}, \mathcal{L}) + \gamma \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{H}} C_{\text{pariwise}}(\mathbf{x}, \mathbf{y}, \mathcal{L}) \tag{8}$$

where C_{unary} and C_{pariwise} are the unary and pairwise costs, respectively, and γ is a balance parameter.

To define the unary cost C_{unary}, we use two GMMs for the foreground and background. To build the GMMs, we use the RGB colors of the pixels in the initial foreground region $o_{t,\theta}$ and the initial background region $o^c_{t,\theta}$, respectively. Each GMM is a full-covariance Gaussian mixture with 10 components. Then, the unary cost is given by

$$C_{\text{unary}}(\mathbf{x}, \mathcal{L}) = \begin{cases} 0 & \text{if } \mathbf{x} \in \mathcal{F} \text{ and } \mathcal{L}(\mathbf{x}) = 1 \\ 0 & \text{if } \mathbf{x} \in \mathcal{B} \text{ and } \mathcal{L}(\mathbf{x}) = 0 \\ \min_k \{-\log p(\mathbf{x}; M_{\mathcal{L}(\mathbf{x}),k})\} & \text{otherwise} \end{cases} \tag{9}$$

where $p(\mathbf{x}; M_{\mathcal{L}(\mathbf{x}),k})$ denotes the probability distribution function of the kth Gaussian component $M_{\mathcal{L}(\mathbf{x}),k}$ of the foreground GMM (when $\mathcal{L}(\mathbf{x}) = 1$) or the background GMM (when $\mathcal{L}(\mathbf{x}) = 0$).

In Equation (9), $\mathcal{F}$ and $\mathcal{B}$ denote the sets of definite foreground pixels and definite background pixels, respectively. In this work, we attempt to refine only the segmentation labels of the pixels near object boundaries. To this end, we define the definite foreground set $\mathcal{F}$ and the definite background set $\mathcal{B}$ using the SLIC superpixels [35] in frame I_t. Specifically, $\mathcal{F}$ has the superpixels, which are fully included in $o_{t,\theta}$ as subsets. Similarly, $\mathcal{B}$ is composed of the superpixels, which are fully included in $o^c_{t,\theta}$. Then, we try to refine only the pixels in the other superpixels, which overlap with both $o_{t,\theta}$ and $o^c_{t,\theta}$. Note that, to preserve the foreground pixels in $\mathcal{F}$, $C_{\text{unary}}(\mathbf{x}, \mathcal{L})$ in Equation(9) has a zero cost if $\mathbf{x} \in \mathcal{F}$ is labeled as the foreground, i.e., $\mathcal{L}(\mathbf{x}) = 1$. In other words, by minimizing $C_{\text{unary}}(\mathbf{x}, \mathcal{L})$, the pixels in $\mathcal{F}$ are encouraged to be labeled as the foreground. Similarly, the pixels in $\mathcal{B}$ are likely to be labeled as the background.

Next, we define the pairwise cost to encourage neighboring pixels to have the same labels,

$$C_{\text{pairwise}}(\mathbf{x}, \mathbf{y}, \mathcal{L}) = \begin{cases} \exp(-d(\mathbf{x}, \mathbf{y})) & \text{if } \mathcal{L}(\mathbf{x}) \neq \mathcal{L}(\mathbf{y}) \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

The distance $d(\mathbf{x}, \mathbf{y}) = d_c(\mathbf{x}, \mathbf{y}) + d_o(\mathbf{x}, \mathbf{y})$ is the sum of the RGB colors' difference $d_c(\mathbf{x}, \mathbf{y})$ and the optical flow difference $d_o(\mathbf{x}, \mathbf{y})$ given by

$$d_c(\mathbf{x}, \mathbf{y}) = \|\mathbf{c}(\mathbf{x}) - \mathbf{c}(\mathbf{y})\|, \tag{11}$$

$$d_o(\mathbf{x}, \mathbf{y}) = \|\mathbf{u}(\mathbf{x}) - \mathbf{u}(\mathbf{y})\|, \tag{12}$$

where $\mathbf{c}(\mathbf{x})$ and $\mathbf{u}(\mathbf{x})$ are the RGB color and the optical flow vectors for pixel $\mathbf{x}$, respectively. We normalize both $d_c(\mathbf{x}, \mathbf{y})$ and $d_o(\mathbf{x}, \mathbf{y})$ into the range $[0, 1]$ to balance their magnitudes.

We repeat the following two steps. First, we optimize the label map $\mathcal{L}^*$ by minimizing the MRF cost function in Equation (8) via the graph-cut algorithm [39]. Second, we update the GMMs based on $\mathcal{L}^*$ and the corresponding cost function. We terminate the iteration when there is no change in $\mathcal{L}^*$. We then determine the refined segmentation region for the object instance $o_{t,\theta}$ using the converged optimal map $\mathcal{L}^*$.

3.4. Complexity Analysis

Let us analyze the computational complexity of the proposed SCO algorithm. For the convenience of analysis, we fix the number of object instances in each frame to N. Note that SCO has two steps: initialization and update. In the initialization in Equation (5), $N-1$ comparisons are made to find the maximum saliency in each frame, requiring $O(NT)$ comparisons in total. In the update step in Equation (6), $N(T-2)$ additions and $(N-1)$ comparisons are performed for each frame in one iteration. Thus, the update step demands $O(KNT^2)$ complexity, where K is the number of iterations and is restricted to be less than 10 in this work.

We repeat the SCO process N times to extract N object tracks. Thus, the complexity is $O(KN^2T^2)$. Then, in the refinement, the Hungarian matching of $O(N^3)$ complexity is performed for each frame. Hence the complexity of the refinement is $O(N^3T)$. Moreover, the complexity of the disappearance detection is $O(NT^2)$. Finally, the overall complexity of the proposed SCO algorithm can be approximated to $O(KN^2T^2)$, since T is larger than N in general. This complexity is significantly lower than the binary integer program in [38], which requires $O(2^{T^2N^2})$ complexity in the worst case because of the depth-first node selection.

4. Proposed Weakly Supervised Algorithm

In supervised VOS, a user can provide annotations about an object of interest, which is the target object to be segmented out. In particular, semi-supervised VOS [14,32] assumes that the accurate pixel-wise segmentation mask of a target object is available. Interactive VOS [40] repeatedly allows a user to check the segmentation results, select a frame with erroneous results, and give refining annotations, such as scribbles, for improving the results. For the annotations, point clicks also can be adopted. For example, the interactive image segmentation algorithm in [41] obtains an initial segmentation result when a user inputs the first click on a target object. After considering the segmentation result, the user provides a new click to refine the result. This refinement is performed recursively until the user stops clicking. Notice that point clicks are also used to refine the segmentation results in interactive VOS algorithms [16]. Moreover, the four extreme point clicks, composed of the left-most, right-most, top, and bottom pixels of a target object, are used for object segmentation in [2].

In this section, we propose a weakly supervised VOS algorithm, which takes point clicks on target objects in the first frame, as illustrated in Figure 5b, but does not require repetitive interaction to refine the results. In this regard, the proposed algorithm requires weaker supervision than the semi-supervised and interactive cases. To obtain the segmentation results in Figure 5c from the weakly supervised points, we train the sparse-to-dense network (SD-Net) for binary classification, which separates each target object from the background. In this work, the proposed SD-Net is adopted to achieve weakly supervised VOS and also to refine the segmentation results of unsupervised VOS.

(**a**) (**b**) (**c**)

Figure 5. Examples of weakly supervised VOS: (**a**) inputs, (**b**) point clicks, and (**c**) segmentation results.

4.1. Network Architecture

Figure 6 is the architecture of the proposed SD-Net, which has two encoders, a feature mixer, and a decoder. The two encoders are based on a Siamese structure and thus share parameters. For each encoder, we modify ResNet50 [42] to take a four-channel input: RGB images and a point map. The spatial resolution of the input is 384×640. The final block of the encoder is also modified to maintain the spatial resolution by employing the dilated convolution with stride 1. We adopt the Siamese structure to use features of the first frame and the annotated point map. Thus, the top encoder takes the first frame and its annotated point map, while the bottom encoder takes the current frame and its point map warped from the previous frame. Then, they extract feature vectors from their input, respectively.

In the feature mixer, the two features, extracted from the first frame and the current frame, are concatenated, and the concatenated feature passes through a 3×3 convolution layer and the ReLU activation. To mix features of the first and current frames more efficiently, we use two modules: the squeeze and excitation module [43] and the dilated convolution, in parallel. The squeeze and excitation module adaptively recalibrates channel-wise features to determine which channels are significant for the segmentation task. Additionally, the dilated convolution increases the receptive field of the concatenated feature vector. The larger receptive field is beneficial since the spatial locations of a target object in the first frame and the current frame are generally different from each other due to the movements of the target object.

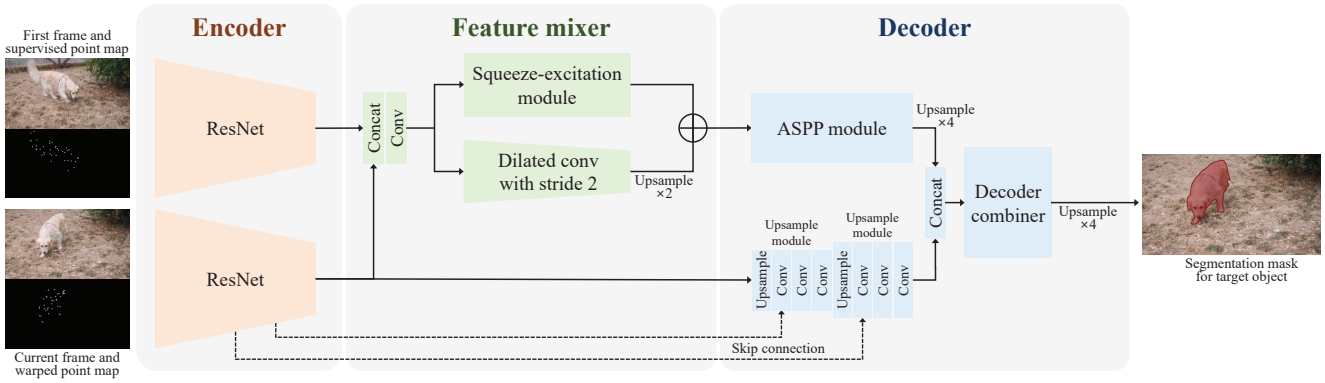


Figure 6. The architecture of the proposed SD-Net.

Next, the decoder draws segmentation inferences on the current frame from the blended feature of the feature mixer and the output of the bottom encoder. We design two kinds of decoder modules to consider both the blended feature of the first and current frames and the feature of the current frame only. First, the ASPP module [44] is adopted to extract multi-scale features with different receptive fields from the blended feature. Using the ASPP module, SD-Net can enlarge the receptive field and exploit various scale information without increasing the number of parameters or the number of computations. Moreover, the two upsample modules are used in series to extract effective multi-scale features from the current frame, where each upsample module consists of one upsample layer and three convolutional layers. The upsample modules take the multi-scale interme-

diate features of the bottom encoder through skip connections. Then, the output vectors of the ASPP module and the upsampling modules are concatenated and fed into the decoder combiner, which has three convolutional layers with the batch normalization and the ReLU activation. The combiner yields a segmentation probability map for the target object. By thresholding the map, a binarized segmentation result for the target object is obtained.

To train the proposed SD-Net, we use two datasets: (1) the YouTube2018 dataset [45], which is the largest VOS dataset, and (2) the training set in the DAVIS2017 dataset. YouTube2018 and DAVIS2017 are used for pre-training and fine-tuning, respectively. For each video, we randomly choose two frames that contain an identical target object. One of them becomes a reference frame, and the other becomes a target frame. Then, for each frame, we produce a point map by sampling points from the ground-truth mask for the target object. More specifically, we sample 50 points randomly from the mask in the reference frame, while we choose one point randomly from every 50 mask pixels in the target frame. We then use the reference frame and its point map as input to the top encoder. On the other hand, to mimic inaccurate optical flow warping, we deform the point map of the target frame using random rotation, scaling, translation, and erosion. Then, we input the target frame and its deformed point map to the bottom encoder. We adopt pixel-wise cross-entropy losses between a predicted probability map and the ground-truth binary mask. We use the Adam optimizer [46] with learning rates 10^{-4} and 10^{-5} for pre-training and fine-tuning, respectively. We decrease the learning rates by a factor of 0.1 every 20 epochs. The training is repeated for 50K iterations with an RTX 2080Ti GPU.

4.2. SD-Net for Weakly Supervised VOS

Given a number of point clicks (50 clicks in this work) on each target object in the first frame, we perform the segmentation of multiple target objects throughout all frames in a video sequence. For each object, an identical pair of the first frame and the corresponding point map are fed into both top and bottom encoders in the Siamese network in Figure 6. In this way, the segmentation results for the multiple objects in the first frame, $\mathcal{O}_1 = \{o_{1,p} \mid p \in \mathbb{N}_M\}$, are obtained from the decoder, where M is the number of the target objects. Then, we initialize object tracks $\{\Theta_p\}_{p=1}^{M}$, where $\Theta_p = \{\theta_1 = p\}$. In other words, each initial track Θ_p contains the single index of the target object in the first frame I_1.

From the second frame I_2, we extend the object tracks by employing warped segmentation masks from SD-Net, as well as object instance masks from FCIS. For each target object p, we generate a warped point map by randomly selecting one point for every 50 pixels in $o_{1,p}$ and then transferring those points from the first frame to the second frame using optical flow vectors [36]. Then, by employing SD-Net, we obtain the warped segmentation mask in the second frame. We then decide whether to add the warped segmentation mask to the set of object instances $\mathcal{O}_2$. After computing the intersection over union (IoU) ratios between the warped segmentation masks and the object instance masks, we find optimal matching pairs using the Hungarian algorithm. For each matching pair, if the IoU ratio is smaller than 0.6, we add the warped segmentation mask to the set of object instances $\mathcal{O}_2$. By adding these segmentation masks, we can boost the recall rate of target objects.

Given the set of object instances $\mathcal{O}_2$ in frame I_2, we extend the object tracks by modifying the SCO process. Specifically, for the track $\Theta_p = \{\theta_1 = p\}$ of target object p, its second element θ_2 is determined by

$$\theta_2 = \arg\max_{\theta \in \mathbb{N}_{N_2}} w(o_{2,\theta}, o_{1,\theta_1}) \tag{13}$$

to maximize the similarity of the clique in a greedy manner. Then, we have the extended track $\Theta_p = \{\theta_1 = p, \theta_2\}$. The selected instance o_{2,θ_2} is excluded from the set of object instances $\mathcal{O}_2$, and the track extension is performed for the next target object. This is repeated to extend the tracks for all target objects, i.e., $\{\Theta_p\}_{p=1}^{M}$. Then, the object track refinement in Section 3.2.3 is performed to yield the refined object tracks $\{\tilde{\Theta}_p\}_{p=1}^{M}$.

We sequentially perform this processing from the second frame to the last frame in the video sequence to extend the target object tracks. For frame I_t, $t \geq 2$, the selection rule in (13) is generalized to

$$\theta_t = \arg\max_{\theta \in \mathbb{N}_{N_t}} \sum_{\tau=1}^{t-1} w(o_{t,\theta}, o_{\tau,\theta_\tau}). \tag{14}$$

Finally, the refined track $\tilde{\Theta}_p$ contains the segmentation results of target p in the video sequence.

4.3. SD-Net for Segmentation Refinement in Unsupervised VOS

SD-Net is also adopted to refine the segmentation results in unsupervised VOS. For each frame, we generate two point maps by randomly choosing 50 points from the segmentation mask for the first point map and one point from every 50 mask pixels for the second point map. We then input the frame and the first point map to the top encoder, while we use the same frame and the second point map as input to the bottom encoder. Then, we obtain the output of SD-Net as the refined segmentation result.

5. Experimental Results

Given a video sequence, the proposed algorithm can yield segmentation results for each frame, which delineates target objects at the pixel level, in both unsupervised and weakly supervised scenarios. Target objects are automatically segmented in the unsupervised scenario, while they are extracted using point clicks in the first frames in the weakly supervised scenario. We compare the proposed algorithm with

- unsupervised VOS algorithms: FST [22], ACO [23], FSEG [11], ARP [12], AGS [27], LSMO [24], COSNet [13], AnDiff [25], MATNet [26], UOVOS [28], Zhao [47], FEM-Net [48], Wang [49], RVOS [29];
- weakly supervised VOS algorithms: BBOX [3], SiamMask [4], En [6].

We use the DAVIS dataset [21]. Note that the DAVIS dataset has two versions, DAVIS2016 and DAVIS2017. DAVIS2016 is a benchmark for evaluating VOS algorithms. It consists of 50 video sequences, which are divided into training and validation videos. These videos are challenging due to various factors, including appearance change, fast motion, and motion blur. Each video contains a single object or spatially connected objects, e.g., a motorbike and its rider, which appear repeatedly in the sequence. Spatially connected objects are also regarded as primary objects. DAVIS2016 was extended to DAVIS2017. It includes 90 train–validation sequences: 60 are training sequences, while 30 are validation sequences. We evaluate the proposed algorithm on the validation sets in DAVIS2016 and DAVIS2017 unless specified otherwise. Note that DAVIS2017 is more challenging than DAVIS2016, since multiple objects, which are not connected, correspond to different targets.

5.1. Ablation Studies

In Tables 1–5, we conduct various ablation studies on the validation sets in DAVIS2016 and DAVIS2017. To assess the proposed algorithm on DAVIS2016, we adopt SCO-F, SCO-M, and SCO-OF. On the other hand, we use only SCO-M and SCO-OM, which segment out multiple objects, for DAVIS2017, whose sequences contain multiple object instances. For the evaluation metrics, we employ the region similarity $\mathcal{J}$ and the contour accuracy $\mathcal{F}$ [21]. The region similarity $\mathcal{J}$ is defined as the IoU ratio $\mathcal{J} = \frac{|\mathcal{S}_\mathrm{p} \cap \mathcal{S}_\mathrm{gt}|}{|\mathcal{S}_\mathrm{p} \cup \mathcal{S}_\mathrm{gt}|}$, where $\mathcal{S}_\mathrm{p}$ and $\mathcal{S}_\mathrm{gt}$ are an estimated segment and the ground-truth, respectively. Additionally, the contour accuracy $\mathcal{F}$ is the F-measure, which is the harmonic mean of the contour precision and recall rates. In these metrics, there are two statistics: 'mean' measures the average score and 'recall' denotes the proportion of the frames whose scores are higher than 0.5.

Table 1. Ablation studies on the validation set in DAVIS2016 according to the refinement methods.

Algorithm	Refinement		Region Similarity $\mathcal{J}$		Contour Accuracy $\mathcal{F}$	
	MRF	SD-Net	Mean	Recall	Mean	Recall
SCO-F			0.788	0.958	0.757	0.867
	✓		0.809	0.957	0.770	0.877
		✓	0.815	0.962	0.796	0.885
	✓	✓	0.819	0.962	0.799	0.881
SCO-OF			0.751	0.885	0.736	0.852
	✓		0.772	0.887	0.749	0.855
		✓	0.776	0.892	0.766	0.855
	✓	✓	0.781	0.894	0.773	0.856
SCO-M			0.743	0.851	0.727	0.823
	✓		0.763	0.860	0.740	0.831
		✓	0.766	0.857	0.761	0.833
	✓	✓	0.771	0.865	0.764	0.837

Table 2. Ablation studies on the validation set in DAVIS2017 according to the refinement methods.

Algorithm	Refinement		Region Similarity $\mathcal{J}$		Contour Accuracy $\mathcal{F}$	
	MRF	SD-Net	Mean	Recall	Mean	Recall
SCO-M			0.518	0.615	0.552	0.640
	✓		0.532	0.620	0.557	0.644
		✓	0.533	0.618	0.567	0.646
	✓	✓	0.537	0.623	0.568	0.651
SCO-OM			0.474	0.561	0.510	0.577
	✓		0.483	0.562	0.514	0.580
		✓	0.486	0.565	0.523	0.589
	✓	✓	0.490	0.568	0.524	0.587

Table 3. Performances on the validation set in DAVIS2017 according to the saliency estimation methods: 'w/o OF' denotes that optical flow is not used in the proposed saliency estimation.

Algorithm	Saliency	Region Similarity $\mathcal{J}$		Contour Accuracy $\mathcal{F}$	
		Mean	Recall	Mean	Recall
SCO-M	CSF [50]	0.512	0.597	0.549	0.625
	w/o OF	0.509	0.594	0.543	0.618
	Proposed	0.537	0.623	0.568	0.651
SCO-OM	CSF [50]	0.485	0.557	0.523	0.592
	w/o OF	0.472	0.543	0.511	0.569
	Proposed	0.490	0.568	0.524	0.587

Table 4. Ablation studies on the validation set in DAVIS2017: 'w/o SOTR' and 'w/o DD' mean that the salient object track refinement and the disappearance detection are not used, respectively.

Algorithm	Setting	Region Similarity $\mathcal{J}$		Contour Accuracy $\mathcal{F}$	
		Mean	Recall	Mean	Recall
SCO-M	Proposed	0.537	0.623	0.568	0.651
	w/o SOTR	0.525	0.611	0.562	0.642
	w/o DD	0.522	0.607	0.559	0.637

Table 5. Performances on the validation set in DAVIS2017 according to the feature settings. The performance of the proposed setting is boldfaced.

Algorithm	Feature Type	Distance Metric	Region Similarity $\mathcal{J}$		Contour Accuracy $\mathcal{F}$	
			Mean	Recall	Mean	Recall
SCO-M	VGG16	Chi-square	0.504	0.580	0.551	0.616
	VGG16	Cosine	0.512	0.600	0.557	0.636
	ResNet50	Chi-square	0.514	0.597	0.552	0.623
	ResNet50	Cosine	0.516	0.603	0.560	0.639
	BoW	**Chi-square**	**0.537**	**0.623**	**0.568**	**0.651**
SCO-OM	VGG16	Chi-square	0.470	0.545	0.519	0.569
	VGG16	Cosine	0.469	0.543	0.514	0.571
	ResNet50	Chi-square	0.475	0.549	0.520	0.571
	ResNet50	Cosine	0.477	0.552	0.523	0.578
	BoW	**Chi-square**	**0.490**	**0.568**	**0.524**	**0.587**

Table 1 lists the $\mathcal{J}$ and $\mathcal{F}$ performances according to the refinement methods on DAVIS2016. The segmentation refinement can be performed using two methods: (1) MRF

in Section 3.3.2 and (2) SD-Net in Section 4.3. Without any refinement, SCO-F yields a mean $\mathcal{J}$ of 78.8% and a mean $\mathcal{F}$ of 75.7%. MRF increases these scores by 2.1% and 1.3%, while SD-Net increases them by 2.7% and 3.9%. Moreover, when both MRF and SD-Net are used sequentially, SCO-F provides the best $\mathcal{J}$ and $\mathcal{F}$ performances of 81.9% and 79.9%, respectively. The studies on SCO-OF and SCO-M exhibit similar improvements due to the refinement methods. Table 2 shows the ablation studies on DAVIS2017. Again, both MRF and SD-Net improve the segmentation accuracy of SCO-M and SCO-OM on DAVIS2017. Thus, in the following experiments, both MRF and SD-Net are used for the refinement, unless specified otherwise.

Table 3 compares the performances when different methods are used to compute the saliency scores of object instances. We replace the RWR-based saliency in Section 3.1 with the state-of-the-art salient object detection algorithm [50]. This salient object detection algorithm does not improve performance since it does not consider motion information. Additionally, we compute the RWR-based saliency without using the optical flow ('w/o OF'). Without the optical flow, unreliable saliency maps are obtained, degrading the performances severely.

In Table 4, we analyze the efficacy of each component of the proposed algorithm through two ablation studies. First, we measure the performance of SCO-M without the salient object track refinement. Second, we do not perform the disappearance detection. Let us refer to these settings as 'w/o SOTR' and 'w/o DD'. Table 4 provides the $\mathcal{J}$ and $\mathcal{F}$ scores on DAVIS2017 for these settings. In this test, we use only SCO-M since the two components are not applied in SCO-OM. Without track refinement or disappearance detection, the $\mathcal{J}$ and $\mathcal{F}$ scores are lowered. Thus, these components are essential in the proposed SCO-M.

Table 5 shows the $\mathcal{J}$ and $\mathcal{F}$ scores on DAVIS2017 according to the feature settings. In the proposed algorithm, a color-based BoW is employed to describe the feature of each instance. In this test, instead of the BoW, we use deep features extracted from VGG16 [51] and ResNet50 [42]. To generate a feature of an object instance, we feed the rectangular patch containing the object instance to each baseline network and extract the output of the last pooling layer. For the deep features, we use two metrics, i.e., the chi-square distance and cosine similarity, to compute edge weights in the graph. In Table 5, we observe that deep features degrade the performances regardless of the metrics. This is because deep semantic features yield high similarity weights between different objects in the same class. This is undesirable in VOS applications since different objects should be clearly distinguished from each other.

5.2. Assessment of Unsupervised VOS Algorithm

Table 6 compares the proposed algorithm with the conventional unsupervised VOS algorithms on the validation set in DAVIS2016. The scores of the conventional algorithms are from the DAVIS dataset's website [21]. Note that the proposed SCO-F achieves comparable performances to the recent state-of-the-art VOS algorithms AnDiff [25] and MATNet [26]. In particular, SCO-F yields the highest recall score of the region similarity $\mathcal{J}$, which is as high as 96.2%. As compared with SCO-F, SCO-M yields lower performances, since it selects non-primary objects, as well as primary ones, in some videos. Additionally, the online version SCO-OF even surpasses the offline approach LSMO [24], as well as the online UOVOS [28].

Table 6. Comparison of the proposed SCO algorithm with the conventional unsupervised VOS algorithms on the validation set in DAVIS2016. The best results are boldfaced, and the second best ones are underlined.

	Region Similarity $\mathcal{J}$		Contour Accuracy $\mathcal{F}$	
Algorithm	**Mean**	**Recall**	**Mean**	**Recall**
FST [22]	0.558	0.649	0.511	0.516
FSEG [11]	0.707	0.835	0.653	0.738
ARP [12]	0.762	0.911	0.706	0.835
LSMO [24]	0.782	0.891	0.759	0.847
AGS [27]	0.797	0.911	0.774	0.858
COSNet [13]	0.805	0.931	0.795	0.895
AnDiff [25]	0.817	0.909	0.805	0.851
MATNet [26]	**0.824**	0.945	**0.807**	**0.902**
UOVOS [28]	0.739	0.885	0.680	0.806
Zhao [47]	0.634	0.703	0.602	0.627
FEM-Net [48]	0.799	0.939	0.769	0.883
Wang [49]	0.816	-	0.797	-
SCO-F	0.819	**0.962**	0.799	0.881
SCO-OF	0.781	0.894	0.773	0.856
SCO-M	0.771	0.865	0.764	0.837

The video sequences in DAVIS2016 have multiple attributes that describe the challenging factors. In Table 7, we analyze the performances according to the nine attributes: low resolution (LR), scale variation (SV), fast motion (FM), camera shake (CS), dynamic background (DB), motion blur (MB), occlusions (OCC), out of view (OV), and appearance change (AC). For the evaluation, we compute the average of the mean $\mathcal{J}$ and mean $\mathcal{F}$ scores (mean $\mathcal{J}\&\mathcal{F}$) on the validation set in DAVIS2016. For the LR, FM, CS, and AC attributes, SCO-F experiences no or negligible performance losses as compared with the overall $\mathcal{J}\&\mathcal{F}$ score of 80.9%, which is computed by averaging the mean $\mathcal{J}$ and mean $\mathcal{F}$ scores of SCO-F in Table 6. However, the DB and MB attributes decrease the performances of SCO-F since the refinement methods are less effective in the presence of motion blur and dynamic background. Nevertheless, except for the OCC attribute, the proposed SCO-F provides better performances than the conventional algorithms.

Table 7. Attribute-based performance comparison on the validation set in DAVIS2016.

	Mean $\mathcal{J}\&\mathcal{F}$				
Attr.	**ARP**	**FSEG**	**LSMO**	**AGS**	**SCO-F**
LR	0.722	0.712	0.772	0.815	**0.861**
SV	0.698	0.603	0.724	0.744	**0.779**
FM	0.728	0.660	0.734	0.775	**0.805**
CS	0.754	0.756	0.817	0.831	**0.845**
DB	0.693	0.482	0.556	0.640	**0.697**
MB	0.689	0.607	0.721	0.735	**0.751**
OCC	0.725	0.628	0.767	**0.769**	0.768
OV	0.728	0.629	0.696	0.754	**0.766**
AC	0.759	0.674	0.766	0.800	**0.840**

Table 8 compares the proposed algorithm with RVOS [29], which yields multiple segment tracks, on the validation set in DAVIS2017. We see that the proposed SCO-OM and SCO-M outperform RVOS. The experimental results in Tables 6–8 indicate that the proposed algorithm is more effective than the existing unsupervised VOS algorithms at segmenting both a single primary object and multiple primary objects.

Table 8. Comparison of the proposed SCO algorithm with the conventional unsupervised algorithms on the validation set in DAVIS2017.

	Region Similarity $\mathcal{J}$		Contour Accuracy $\mathcal{F}$	
Algorithm	**Mean**	**Recall**	**Mean**	**Recall**
RVOS [29]	0.368	0.402	0.457	0.464
SCO-OM	0.490	0.568	0.524	0.587
SCO-M	**0.537**	**0.623**	**0.568**	**0.651**

5.3. Assessment of Weakly Supervised VOS Algorithm

Table 9 compares the proposed weakly supervised algorithm with existing weakly supervised algorithms on the validation sets in DAVIS2016 and DAVIS2017. The scores of the conventional algorithms [2–4,6] are from their respective papers. In Table 9, 'Annotation' denotes the types of annotations, which are provided in the first frame. We compare the proposed weakly supervised algorithm with two existing weakly supervised VOS algorithms in [3,4] that take the bounding box annotation. Even though the point-click annotation in the proposed algorithm requires more human effort than the bounding box annotation in [3,4], the proposed weakly supervised algorithm achieves more accurate VOS and provides the best performances on both DAVIS2016 and DAVIS2017. Moreover, the proposed algorithm outperforms [2,6], which take four points for a target object per frame and category labels, respectively.

Table 9. Comparison of the proposed weakly supervised algorithm with the conventional weakly supervised algorithms on the validation sets in DAVIS2016 and DAVIS2017 according to annotation types: '# of targets' denotes the number of target objects.

		Region Similarity $\mathcal{J}$		Contour Accuracy $\mathcal{F}$	
Algorithm	**Annotation**	**Mean**	**Recall**	**Mean**	**Recall**
A. DAVIS2016					
BBOX [3]	Box	0.803	0.952	–	–
SiamMask [4]	Box	0.717	0.868	0.678	0.798
DEXTR [2]	4 points per frame	0.795	–	–	–
En [6]	category label	0.769	0.927	0.720	0.870
Proposed	50 points	0.815	0.955	0.795	0.886
B. DAVIS2017					
SiamMask [4]	Box	0.543	0.628	0.585	0.675
Proposed	# of targets	0.537	0.630	0.566	0.650
Proposed	50 points	0.658	0.781	0.688	0.804

Table 9 also shows the performance of the proposed algorithm on DAVIS2017, when only the number of target objects is provided without point clicks. In other words, supervision is limited to the number of target objects. Then, the proposed algorithm selects as many salient object tracks as the provided number of targets. Even with this minimal supervision, the proposed algorithm achieves comparable performances to SiamMask [4]. Figure 7 shows the mean $\mathcal{J}\&\mathcal{F}$ scores on DAVIS2017 according to the number of point clicks in the inference. We see that the performance generally increases as the number of point clicks becomes larger, but it is saturated when more than 30 points are used. We fix the number of point clicks to 50 in this work. Moreover, the proposed algorithm outperforms SiamMask [4] using only five point clicks.

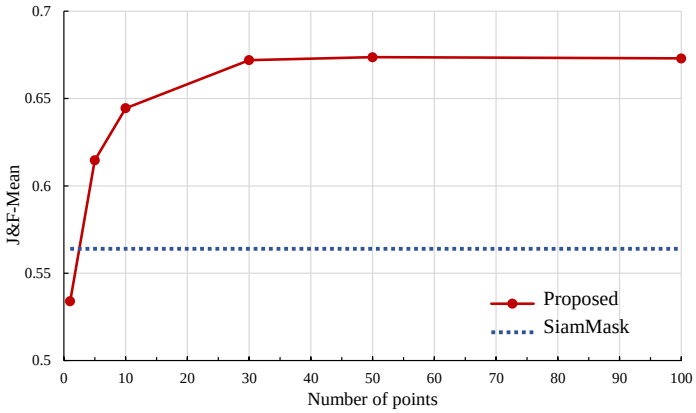


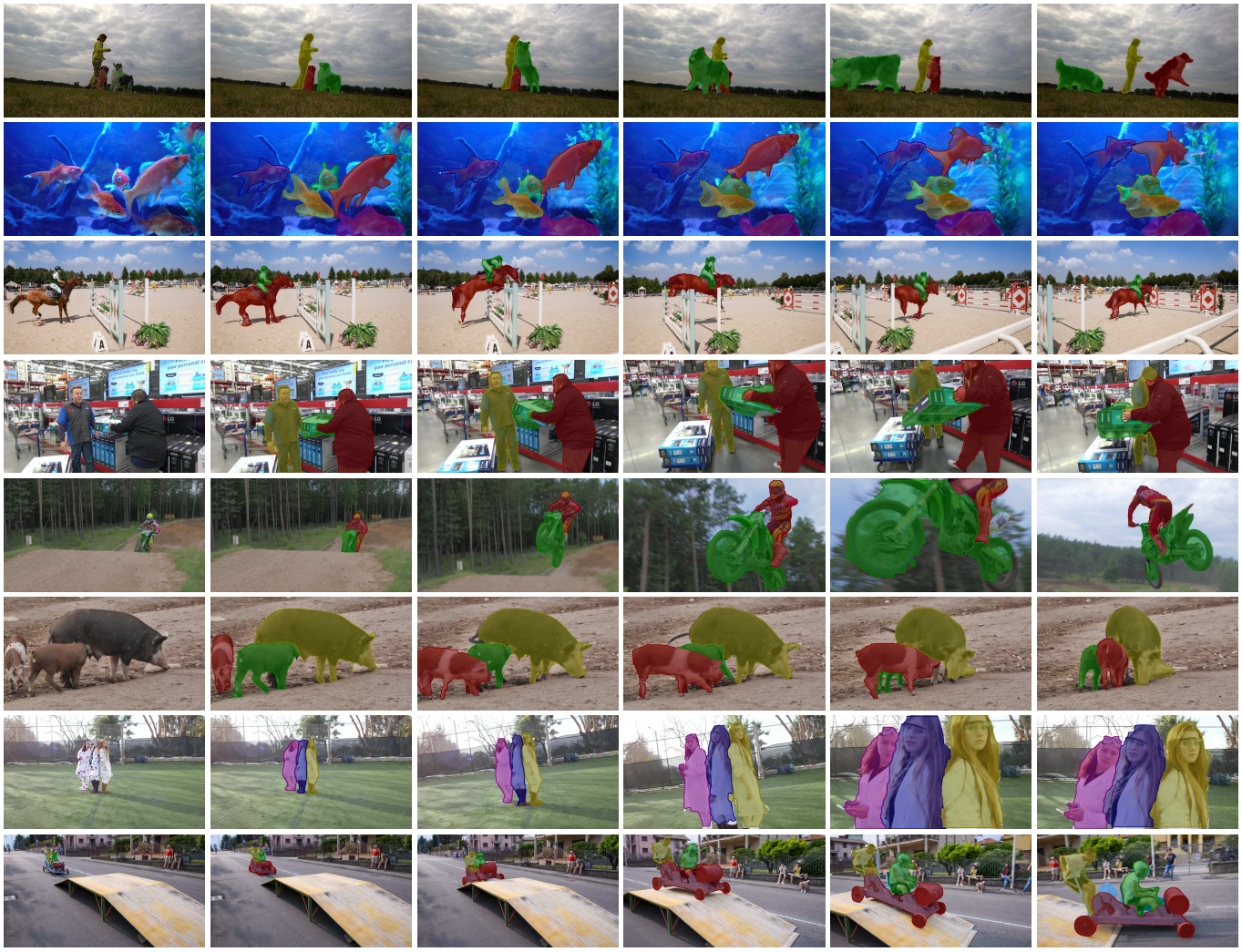
Figure 7. The mean $\mathcal{J}\&\mathcal{F}$ performances on DAVIS2017 according to the number of point clicks [19].

Note that the proposed algorithm requires 50 point clicks on each target object in the first frame of a video sequence. Then, SD-Net in Figure 6 uses and propagates the information to segment the object in all frames in the sequence. We can generalize this weakly supervised algorithm straightforwardly to perform interactive VOS, in which repetitive user interactions are provided to refine the segmentation results. In the first interactive segmentation round, given 50 point clicks for each object in the first frame, the proposed SD-Net obtains the segmentation results for all frames. In the next round, we find the frame with the worst segmentation result and then provide additional point clicks to SD-Net to refine the inaccurate result. Then, SD-Net propagates the refined result bi-directionally to both ends of the sequence. This is repeated until a desired level of segmentation is achieved. Table 10 shows the segmentation performances according to the number of interaction rounds. The performances increase quickly and saturate at approximately the fifth round. Figure 8 shows the qualitative results of the proposed weakly supervised algorithm on DAVIS2017. The first column illustrates the point clicks, annotated by users on the first frames, and the other columns are the corresponding segmentation results in the first and subsequent frames. We see that multiple target objects, either connected or not, are segmented out well using the point clicks.

Table 10. Performance on the validation set in DAVIS2017 according to the number of interaction rounds.

	The Number of Interaction Rounds					
Metric	**1**	**2**	**3**	**4**	**5**	**6**
Mean $\mathcal{J}$	0.655	0.676	0.685	0.696	0.713	0.718
Mean $\mathcal{F}$	0.690	0.707	0.719	0.735	0.750	0.758
$\mathcal{J}\&\mathcal{F}$	0.673	0.692	0.702	0.716	0.732	0.738

Table 11 shows the performance on the SegTrack v2 dataset [52]. SegTrack v2 contains 14 low-resolution videos with 24 generic foreground objects. We perform the experiments on the full videos in SegTrack v2 using the proposed weakly supervised algorithm with 30 initial point clicks. Notice that the proposed SD-Net is trained on DAVIS2017 and YouTube2018 and is not fine-tuned on SegTrack v2. In Table 11, the mean $\mathcal{J}$ score is averaged across all instances. The scores of the conventional algorithms are from their respective papers. We see that, despite requiring weaker supervision, the proposed algorithm achieves a higher score than RGMP [53] and a comparable score to [54–57].



(**a**) (**b**) (**c**) (**d**) (**e**) (**f**)

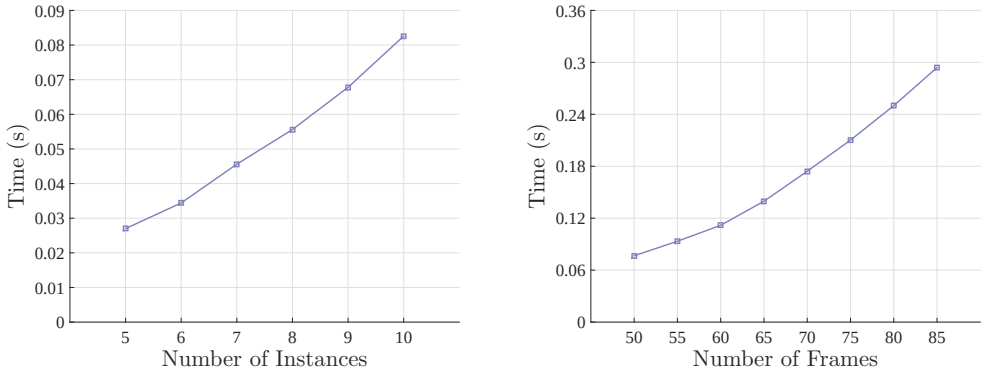
Figure 8. VOS results of the proposed algorithm in the weakly supervised scenario on the DAVIS2017 dataset: (**a**) point interaction of target objects in 1st frames, (**b**) segmentation results in 1st frames, and (**c**–**f**) segmentation results in subsequent frames. From top to bottom, "Dogs-jump", "Gold-fish", "Horsejump-high", "Loading", "Motocross-jump", "Pigs", "Lab-coat", and "Soapbox".

Table 11. Comparison of the proposed weakly supervised algorithm with the conventional semi-supervised algorithms on SegTrack v2.

	Semi-Supervised					**Weakly Supervised**
	RGMP [53]	**DMM-Net** [54]	**DIPNet** [55]	**NPMCA-net** [56]	**SiamPolar** [57]	**Proposed**
Mean $\mathcal{J}$	0.711	0.767	0.738	0.761	0.728	0.726

5.4. Running Time Analysis

We measure the running time of the proposed SCO algorithm for finding cliques in a complete k-partite graph. In this test, we use the "Boxing-fisheye" sequence in the DAVIS2017 dataset. We use a computer with a 2.6GHz CPU. The running time of SCO is affected by two factors: (1) the number N of object instances in a frame and (2) the number T of frames in a sequence. Figure 9a shows the running times according to N, when T is fixed to 50. Figure 9b plots the running times according to T, when N is limited to 10. The proposed algorithm is faster than the binary integer program in [38], which consumes about 1 s when $N = 10$ and $T = 50$.



(**a**) (**b**)

Figure 9. The running times according to (**a**) the number of object instances and (**b**) the number of frames.

Table 12 analyzes the running times of SCO-F. The proposed algorithm performs FCIS [19] for generating the object instances and also the optical flow estimation [36], saliency estimation, and feature extraction in each frame. Then, it performs SCO for the global optimization. In this analysis, the number of frames is 80, and the number of object instances in each frame is 10. SCO takes 0.21 s for the entire sequence, which is negligible. Then, the proposed algorithm also performs two segmentation refinement methods based on MRF and SD-Net in each frame. In total, the proposed algorithm takes 1.79 s per frame (SPF). It is comparable to MATNet [26] (0.75 SPF) and faster than UOVOS [28] (9.96 SPF).

Table 12. Running times in seconds per frame (SPF).

	FCIS	Optical Flow	Saliency Estimation	Feature Extraction	MRF	SD-Net	Total
Time	0.24	0.19	0.30	0.15	0.76	0.15	1.79

6. Conclusions

We proposed a novel algorithm to segment out objects in a video sequence in both unsupervised and weakly supervised scenarios by solving the problem of finding cliques in a complete k-partite graph. We first generated the object instances in each frame. Then, we chose a salient instance from each frame to construct the salient object track. For this purpose, we developed the SCO technique using both the saliency and similarity energies. By applying SCO repeatedly, we obtained multiple salient object tracks. Finally, we transformed these tracks into VOS results. For weakly supervised VOS, we adapted SCO and developed SD-Net to produce segmentation results by exploiting point clicks on the target objects in the first frame. The experimental results showed that the proposed algorithm provides comparable or better performances than the state-of-the-art VOS algorithms on the DAVIS2016 and DAVIS2017.

In spite of its achievements, the proposed algorithm still has a limitation. To obtain the set of object instances, the proposed algorithm uses FCIS, which is trained on still-image instance segmentation. In future work, we plan to develop an instance segmentation network for video sequences that takes advantage of the temporal context information in different frames. Instead of frame-by-frame instance segmentation, we expect that the instance segmentation results from multiple frames can provide more reliable salient instance segments to perform the proposed SCO process. Moreover, we will improve SD-Net by adding the transformer in the feature mixer to fuse the two features of the first frame and the current frame more effectively.

Author Contributions: Conceptualization, Y.J.K.; formal analysis, Y.H.; investigation, Y.J.K. and C.-S.K.; data curation, Y.H.; writing–original draft preparation, Y.J.K.; writing—review and editing, Y.J.K. and C.-S.K.; supervision, Y.J.K.; funding acquisition, Y.J.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partly supported by the National Research Foundation of Korea (NRF) grants funded by the Korea government (MSIT) (NRF-2021R1A4A1031864), Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2022R1I1A3069113), Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. RS-2022-00155857, Artificial Intelligence Convergence Innovation Human Resources Development (Chungnam National University)), and Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2020-0-01441, Artificial Intelligence Convergence Research Center (Chungnam National University)).

Conflicts of Interest: The authors declare that they have no known competing interests.

References

1. Koh, Y.J.; Jang, W.D.; Kim, C.S. POD: Discovering primary objects in videos based on evolutionary refinement of object recurrence, background, and primary object models. In Proceedings of the IEEE Conference on Computer Vision and Pattern Cecognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 1068–1076.
2. Maninis, K.K.; Caelles, S.; Pont-Tuset, J.; Van Gool, L. Deep extreme cut: From extreme points to object segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 616–625.
3. Zhang, Z.; Hua, Y.; Song, T.; Xue, Z.; Ma, R.; Robertson, N.; Guan, H. Tracking-assisted Weakly Supervised Online Visual Object Segmentation in Unconstrained Videos. In Proceedings of the 26th ACM International Conference on Multimedia, Seoul, Korea, 22–26 October 2018; pp. 941–949.
4. Wang, Q.; Zhang, L.; Bertinetto, L.; Hu, W.; Torr, P.H.S. Fast Online Object Tracking and Segmentation: A Unifying Approach. In Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 1328–1338.
5. Wei, L.; Lang, C.; Liang, L.; Feng, S.; Wang, T.; Chen, S. Weakly Supervised Video Object Segmentation via Dual-attention Cross-branch Fusion. *ACM Trans. Intell. Syst. Technol.* **2022**, *13*, 1–20. [CrossRef]
6. En, Q.; Duan, L.; Zhang, Z. Joint Multisource Saliency and Exemplar Mechanism for Weakly Supervised Video Object Segmentation. *IEEE Trans. Image Process.* **2021**, *30*, 8155–8169. [CrossRef]
7. Zheng, Q.; Yang, M.; Yang, J.; Zhang, Q.; Zhang, X. Improvement of generalization ability of deep CNN via implicit regularization in two-stage training process. *IEEE Access* **2018**, *6*, 15844–15869. [CrossRef]
8. Zhao, M.; Chang, C.H.; Xie, W.; Xie, Z.; Hu, J. Cloud shape classification system based on multi-channel cnn and improved fdm. *IEEE Access* **2020**, *8*, 44111–44124. [CrossRef]
9. Jin, B.; Cruz, L.; Gonçalves, N. Deep facial diagnosis: Deep transfer learning from face recognition to facial diagnosis. *IEEE Access* **2020**, *8*, 123649–123661. [CrossRef]
10. Zheng, Q.; Zhao, P.; Li, Y.; Wang, H.; Yang, Y. Spectrum interference-based two-level data augmentation method in deep learning for automatic modulation classification. *Neural Comput. Appl.* **2021**, *33*, 7723–7745. [CrossRef]
11. Jain, S.D.; Xiong, B.; Grauman, K. FusionSeg: Learning to Combine Motion and Appearance for Fully Automatic Segmentation of Generic Objects in Videos. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 3664–3673.
12. Koh, Y.J.; Kim, C.S. Primary Object Segmentation in Videos Based on Region Augmentation and Reduction. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 3442–3450.
13. Lu, X.; Wang, W.; Ma, C.; Shen, J.; Shao, L.; Porikli, F. See More, Know More: Unsupervised Video Object Segmentation With Co-Attention Siamese Networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 3618–3627.
14. Caelles, S.; Maninis, K.K.; Pont-Tuset, J.; Leal-Taixe, L.; Cremers, D.; Van Gool, L. One-Shot Video Object Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 221–230.
15. Voigtlaender, P.; Leibe, B. Online adaptation of convolutional neural networks for video object segmentation. *arXiv* **2017**, arXiv:1706.09364.
16. Chen, Y.; Pont-Tuset, J.; Montes, A.; Van Gool, L. Blazingly Fast Video Object Segmentation with Pixel-Wise Metric Learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 1189–1198.

17. Voigtlaender, P.; Chai, Y.; Schroff, F.; Adam, H.; Leibe, B.; Chen, L.C. FEELVOS: Fast end-to-end embedding learning for video object segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 9481–9490.
18. Koh, Y.J.; Lee, Y.Y.; Kim, C.S. Sequential clique optimization for video object segmentation. In Proceedings of the European Conference on Computer Vision (ECCV), Seoul, Korea, 22–26 October 2018; pp. 537–556.
19. Li, Y.; Qi, H.; Dai, J.; Ji, X.; Wei, Y. Fully Convolutional Instance-Aware Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2359–2367.
20. Chartrand, G.; Zhang, P. *Chromatic Graph Theory*; CRC Press: Boca Raton, FL, USA, 2008.
21. Perazzi, F.; Pont-Tuset, J.; McWilliams, B.; Van Gool, L.; Gross, M.; Sorkine-Hornung, A. A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 724–732.
22. Papazoglou, A.; Ferrari, V. Fast object segmentation in unconstrained video. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 1777–1784.
23. Jang, W.D.; Lee, C.; Kim, C.S. Primary Object Segmentation in Videos via Alternate Convex Optimization of Foreground and Background Distributions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 696–704.
24. Tokmakov, P.; Schmid, C.; Alahari, K. Learning to segment moving objects. *Int. J. Comput. Vis.* **2019**, *127*, 282–301. [CrossRef]
25. Yang, Z.; Wang, Q.; Bertinetto, L.; Bai, S.; Hu, W.; Torr, P.H.S. Anchor Diffusion for Unsupervised Video Object Segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Long Beach, CA, USA, 15–20 June 2019; pp. 931–940.
26. Zhou, T.; Wang, S.; Zhou, Y.; Yao, Y.; Li, J.; Shao, L. Motion-Attentive Transition for Zero-Shot Video Object Segmentation. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020.
27. Wang, W.; Song, H.; Zhao, S.; Shen, J.; Zhao, S.; Hoi, S.C.; Ling, H. Learning unsupervised video object segmentation through visual attention. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019 ; pp. 3064–3074.
28. Zhuo, T.; Cheng, Z.; Zhang, P.; Wong, Y.; Kankanhalli, M. Unsupervised online video object segmentation with motion property understanding. *IEEE Trans. Image Process.* **2019**, *29*, 237–249. [CrossRef]
29. Ventura, C.; Bellver, M.; Girbau, A.; Salvador, A.; Marques, F.; Giro-i Nieto, X. RVOS: End-to-end recurrent network for video object segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 5277–5286.
30. Qi, G.; Zhang, Y.; Wang, K.; Mazur, N.; Liu, Y.; Malaviya, D. Small Object Detection Method Based on Adaptive Spatial Parallel Convolution and Fast Multi-Scale Fusion. *Remote. Sens.* **2022**, *14*, 420. [CrossRef]
31. Lazarow, J.; Xu, W.; Tu, Z. Instance Segmentation With Mask-Supervised Polygonal Boundary Transformers. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–24 June 2022; pp. 4382–4391.
32. Jang, W.D.; Kim, C.S. Online Video Object Segmentation via Convolutional Trident Network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5849–5858.
33. Sun, M.; Xiao, J.; Lim, E.G.; Xie, Y.; Feng, J. Adaptive ROI Generation for Video Object Segmentation Using Reinforcement Learning. *Pattern Recog.* **2020**, *106*, 107465. [CrossRef]
34. Yin, Y.; Xu, D.; Wang, X.; Zhang, L. AGUnet: Annotation-guided U-net for fast one-shot video object segmentation. *Pattern Recog.* **2021**, *110*, 107580. [CrossRef]
35. Achanta, R.; Shaji, A.; Smith, K.; Lucchi, A.; Fua, P.; Süsstrunk, S. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 2274–2282. [CrossRef]
36. Sun, D.; Yang, X.; Liu, M.Y.; Kautz, J. PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8934–8943.
37. Zamir, A.R.; Dehghan, A.; Shah, M. GMCP-Tracker: Global multi-object tracking using generalized minimum clique graphs. In Proceedings of the European Conference on Computer Vision, Providence, RI, USA, 16–21 June 2012 ; pp. 343–356.
38. Dehghan, A.; Modiri Assari, S.; Shah, M. GMMCP Tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 4091–4099.
39. Boykov, Y.; Veksler, O.; Zabih, R. Fast approximate energy minimization via graph cuts. In *Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence*; IEEE: New York, NY, USA, 1999; Volume 1, pp. 377–384.
40. Fan, Q.; Zhong, F.; Lischinski, D.; Cohen-Or, D.; Chen, B. JumpCut: Non-successive mask transfer and interpolation for video cutout. *ACM Trans. Graphics* **2015**, *34*, 195:1–195:10. [CrossRef]
41. Jang, W.D.; Kim, C.S. Interactive Image Segmentation via Backpropagating Refinement Scheme. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 5297–5306.
42. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
43. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141.

44. Chen, L.C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 834–848. [CrossRef]
45. Xu, N.; Yang, L.; Fan, Y.; Yang, J.; Yue, D.; Liang, Y.; Price, B.; Cohen, S.; Huang, T. YouTube-VOS: Sequence-to-sequence video object segmentation. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018 2018; pp. 585–601.
46. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization; *arXiv* **2015**, arXiv:1412.6980.
47. Zhao, Z.; Zhao, S.; Shen, J. Real-time and light-weighted unsupervised video object segmentation network. *Pattern Recognit.* **2021**, *120*, 108120. [CrossRef]
48. Zhou, Y.; Xu, X.; Shen, F.; Zhu, X.; Shen, H.T. Flow-edge guided unsupervised video object segmentation. *IEEE Trans. Circuits Syst. Video Technol.* **2021**. [CrossRef]
49. Wang, Y.; Choi, J.; Chen, Y.; Li, S.; Huang, Q.; Zhang, K.; Lee, M.S.; Kuo, C.C.J. Unsupervised video object segmentation with distractor-aware online adaptation. *J. Vis. Commun. Image Represent.* **2021**, *74*, 102953. [CrossRef]
50. Gao, S.H.; Tan, Y.Q.; Cheng, M.M.; Lu, C.; Chen, Y.; Yan, S. Highly Efficient Salient Object Detection with 100K Parameters. In Proceedings of the European Conference on Computer Vision (ECCV), Online Conference, 23–28 August 2020.
51. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2015**, arXiv:1409.1556.
52. Li, F.; Kim, T.; Humayun, A.; Tsai, D.; Rehg, J.M. Video segmentation by tracking many figure-ground segments. In Proceedings of the IEEE International Conference on Computer Vision, Portland, OR, USA, 23–28 June 2013; pp. 2192–2199.
53. Oh, S.W.; Lee, J.Y.; Sunkavalli, K.; Kim, S.J. Fast Video Object Segmentation by Reference-Guided Mask Propagation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7376–7385.
54. Zeng, X.; Liao, R.; Gu, L.; Xiong, Y.; Fidler, S.; Urtasun, R. DMM-Net: Differentiable mask-matching network for video object segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27–28 October 2019; pp. 3929–3938.
55. Hu, P.; Liu, J.; Wang, G.; Ablavsky, V.; Saenko, K.; Sclaroff, S. Dipnet: Dynamic identity propagation network for video object segmentation. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Snowmass Village, CO, USA, 2–5 March 2020; pp. 1904–1913.
56. Yu, S.; Xiao, J.; Zhang, B.; Lim, E.G.; Zhao, Y. Fast pixel-matching for video object segmentation. *Signal Process. Image Commun.* **2021**, *98*, 116373. [CrossRef]
57. Li, Y.; Hong, Y.; Song, Y.; Zhu, C.; Zhang, Y.; Wang, R. SiamPolar: Semi-supervised realtime video object segmentation with polar representation. *Neurocomputing* **2022**, *467*, 491–503. [CrossRef]