Journal of the Brazilian Computer Society, 2026, 32:1, doi: 10.5753/jbcs.2026.5904

Memorizing Features Efficiently for Self-supervised Video Object Segmentation

Marcelo Mendonça [**Federal Institute of Bahia** | ***marcelomendonca@ifba.edu.br***]
Luciano Oliveira [**Federal University of Bahia** | ***lrebouca@ufba.br***]

Institute of Computing, Federal University of Bahia, Av. Milton Santos, s/n, Campus de Ondina, 40170-110, Salvador, BA, Brazil.

Received: 13 April 2025 • **Accepted:** 12 January 2026 • **Published:** 15 March 2026

Abstract. Video object segmentation (VOS) involves consistently identifying and classifying object pixels in video sequences, a task that traditionally depends on extensive, manually annotated datasets. In this work, we present SHLS (Superfeatures in a Highly Compressed Latent Space), a self-supervised VOS method that reduces reliance on both annotations and large training datasets. SHLS employs a metric learning framework combining superpixels and deep learning features, enabling effective training with just 10,000 unlabeled still images. Utilizing an efficient memory clustering mechanism, SHLS generates ultra-compact representations called superfeatures, which efficiently store and classify object information across video sequences. Experiments on the DAVIS dataset demonstrate SHLS's strong performance in multi-object scenarios, underscoring its potential as a robust and efficient alternative in self-supervised VOS.

Keywords: Video object segmentation, Superpixel segmentation, Metric Learning

1 Introduction

The primary objective of Video Object Segmentation (VOS) is to categorize all pixels within a sequence of frames into foreground and background regions. In the simplest scenario, single-object segmentation, distinguishing between multiple objects within the same region is unnecessary. Here, the entire foreground is assigned a single label, as is the background. The task becomes more complex in the multi-object scenario, where each object in the foreground must be labeled uniquely.

Supervised VOS methods try to learn this task based on fully annotated video datasets. During training, the loss is calculated by comparing the predicted results with manually created masks for each frame. However, a major limitation of this approach is its reliance on manual annotations, which are challenging and time-consuming to produce, especially for videos.

In contrast, self-supervised methods aim to eliminate the need for human-provided labels during training. By avoiding this dependency, these approaches can utilize vast amounts of unlabeled data available in both video and image formats. For videos, the core idea is to learn the segmentation task from inherent signals within the data itself, without requiring external guidance.

One example involves leveraging the smooth transitions that typically occur between consecutive frames. Some self-supervised methods capture this smoothness by traversing back and forth through a video and making predictions for subsequent frames. This closed-cycle strategy allows the calculation of an error signal by comparing the final prediction with the known initial frame. Another widely used approach employs data augmentation to generate pseudo-labels automatically, applicable to both videos and still images. This is the training approach utilized by our proposed VOS method, superfeatures in a highly compressed latent space (SHLS) [Mendonça *et al*., 2023]. As we will show, this methodology allows SHLS to be trained exclusively with unlabeled data, relying solely on a small dataset of $10k$ static images.

Although recent progress has been made, existing VOS methods — including self-supervised ones — typically depend on high-dimensional feature embeddings and the storage of extensive per-pixel representations, limiting scalability in long videos and multi-object scenarios. As a result, the field still lacks memory mechanisms that are both compact and capable of preserving long-range temporal consistency.

To address this limitation, SHLS has been conceived as a memory-efficient VOS method in a self-supervised setting. Building on our superpixel algorithm, ISEC [Mendonça and Oliveira, 2018], SHLS introduces *superfeatures* — ultra-compact representations derived from the combination of superpixels and convolutional feature descriptors. The compactness of these representations enables the efficient storage of temporal information by clustering superfeatures in a metric-learning latent space. As we will show, the ability to retrieve past information significantly enhances the segmentation capability of our method while drastically reducing memory requirements compared to existing memory-based VOS pipelines.

1.1 Contributions

Our main contributions are as follows:

- The **superfeature model**, an ultra-compact feature representation that decouples feature granularity from pixel resolution, drastically reducing memory requirements while preserving object-level structure;

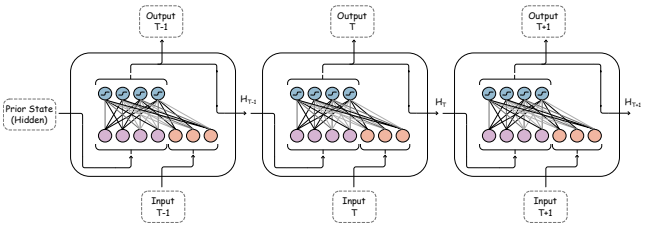

Figure 1. Vanilla RNN model over sequential time steps. The model is fed with the current input together with the hidden state H — a copy of the previous output generated by the model itself.

- A **compact long-range memory architecture** that stores and retrieves superfeatures in a latent space, enabling efficient temporal consistency across video frames;
- A **fully self-supervised training pipeline** that synthesizes multi-object pseudo-masks from static images, requiring no manual annotations during training;
- **Competitive performance** with significantly reduced training data requirements and memory footprint compared to state-of-the-art self-supervised VOS methods.

The remainder of the paper is organized as follows. Section 2 reviews prior work on memorizing features for VOS. Section 3 presents our method, including its learning paradigm, the training based on unlabeled static images, the superfeature model based on metric-learning, and the memory mechanism. Section 4 reports comparative results and ablation studies. Finally, Section 5 concludes the paper.

2 Background on memorizing features for VOS

Recent research on VOS has established that memory and attention mechanisms are essential to improve the models' capacity to recognize objects throughout a video sequence. Especially for long sequences, the ability to look back and retrieve only relevant information is critical. However, it is not trivial to provide an efficient way to maintain and access many features, accounting for their spatial dimensions and many channels (as is usually true in deep-learning models). In the case of video data, the challenge is even harder due to the additional temporal dimension. As a result, most VOS methods can only retrieve a small fraction of the available information from the frames.

2.1 VOS based on RNNs

Videos consist of sequences of images with strong interframe correspondence. Learning features that capture this correspondence is a central objective of VOS methods. Some studies have explored the use of 3D convolutions to process video volumes rather than individual frames, leveraging the additional temporal dimension provided by videos [Hou *et al*., 2017]. However, these methods failed to produce encouraging results to justify the significant increase in model size. As an alternative, recurrent neural networks (RNNs) are specifically designed to handle sequential data by storing information from previously observed inputs in their hidden states (Fig. 1).

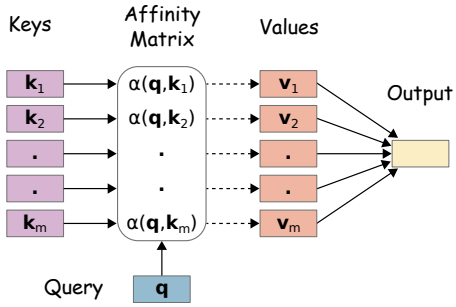

Figure 2. Attention-based memory. Past input representations are stored in the form of key-value pairs, where the keys are more compact representations. At the memory reading process, each key is matched with the current input representation (query), resulting in an affinity matrix that is used to select the values more correlated to that input.

For video processing, the most commonly used techniques are convolution-based recurrent models, such as conv-GRU and conv-LSTM. For example, Tokmakov *et al*. [2017] proposed a two-stream network featuring a conv-GRU module placed between the feature fusion and segmentation head stages. Similarly, Xu *et al*. [2018] employed conv-LSTM units as the core component of a single-object segmentation model. Later, Ventura *et al*. [2019] expanded the application of conv-LSTMs to the multi-object segmentation task. However, these experiments did not yield remarkable results. Nonetheless, they highlighted a growing focus on improving memory mechanisms in VOS research.

2.2 Attention-based memories

The STM method proposed by Oh *et al*. [2019] introduces an external memory bank to aid the video segmentation. This memory mechanism allows for storing information in the form of feature vectors, which are selected and retrieved according to the input (Fig. 2). The working principle is based on three main elements: the *values*, which are features extracted from past inputs and stored to compose the memory bank; the more relevant values to that input are retrieved when a new input comes in. Each value in the memory bank is associated with a *key*; the keys are more compact representations of the memory entries, working as memory indexes. To read the memory, each key is matched with a *query*, which is a feature representation of the current input. The matching process gives rise to an affinity matrix that is used to accomplish a weighted average on the values; in this case, the higher the correspondence between the key and query (and, consequently, between memory entry and input), the greater the weight. Mathematically, this matching process yields a memory-based representation I, given by

$$I(q,k,v) = \sum_i \frac{exp(A_{qk_i})}{\sum_j exp(A_{qk_i})}\, v_i\,, \qquad (1)$$

where A is the affinity matrix that correlates the *query* q with each *key* k present in the memory bank. The summation involving A defines a weighted average over the *values* v, which acts as an attention mechanism that modulates the memories according to the context given by q.

Since past information is explicitly stored in a memory bank rather than being implicitly learned by network parameters (as in RNNs), attention-based memories can retrieve a lot

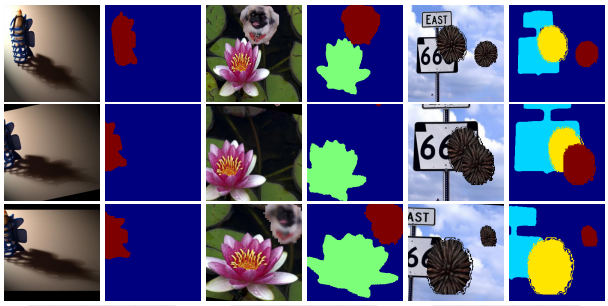

Figure 3. Examples of generated pseudo-sequences. The sequences are comprised of pairs of pseudo-frames and pseudo-masks containing a variable number of objects in different conditions: (a) single-object sequence with partial disappearance; (b) multi-object sequence with total disappearance and reappearance situation; and (c) multi-object sequence with a cloned foreground object.

more information to aid the segmentation process. In practice, however, there are constraints due to the cost of computing large affinity matrices. Despite that, many variants of the original approach have been proposed, making the memory mechanism a crucial component for most state-of-the-art VOS methods [Yang *et al*., 2020; Li *et al*., 2022; Cheng and Schwing, 2022; Xu *et al*., 2022].

3 Proposed method

This section describes the learning paradigm of SHLS and its main components, including self-supervised training based on still images, the superfeature model learned through a metric learning approach, the memory mechanism, and the segmentation refinement module.

3.1 Learning Paradigm

In the VOS literature, methods trained exclusively on unlabeled data – such as our proposed method, SHLS – are typically classified as *self-supervised*. However, even self-supervised approaches generally rely on a manually annotated mask of the first frame at inference time to indicate the target objects. This inference protocol is often referred to as *semi-supervised* segmentation, which may be misleading when discussing the self-supervised nature of the training procedure. To avoid ambiguity, in this paper we adopt the term *one-shot* to describe the inference setting.

Therefore, SHLS is a genuinely self-supervised VOS method with respect to training, and the training masks referenced throughout the paper are pseudo-masks generated automatically without human annotations.

3.2 Using static images for self-supervised VOS training

To learn VOS in a self-supervised fashion, we have devised a set of techniques that merge saliency detection with data augmentation to produce synthetic videos (Fig. 3). Our approach entails extracting foreground objects from various images using saliency maps [Nguyen *et al*., 2019] and aggregating them to form pseudo-frame sequences. Every frame in the generated sequences is accompanied by its corresponding object mask, which are completely free of manual annotations and solely rely on unlabeled still images as inputs.

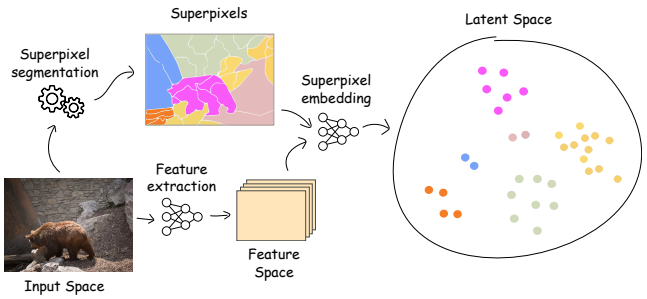

Figure 4. Latent space comprised of superfeatures. Given the superpixel segmentation of the input frame, and the convolutional feature maps extracted by a CNN, the latent space is constructed by a superfeature embedding model that generates superfeatures as a combination of superpixels and convolutional features.

Inspired by the work of Oh *et al*. [2018], our methodology involves three steps: first, a random image from the dataset is chosen as a template; second, the selected image and its foreground mask are replicated N times, where N is the sequence length, and each replica is an augmented version of the template; and third, a random number of different image-mask pairs are selected from the dataset, and their foreground pixels are extracted, augmented, and randomly pasted into each template instance.

With these augmentation techniques, we can create an unlimited number of pseudo-sequences to train our VOS method, SHLS. The training is based on the images from the MSRA10K dataset [Cheng *et al*., 2015], which contains 10,000 images. By doing so, we can train SHLS in a self-supervised manner.

3.3 Superfeatures

SHLS constructs a latent space by combining image features and superpixels. In this space, each data point represents the features extracted from all the pixels inside a superpixel, and we refer to this representation as a *superfeature*.

The latent space is created by combining the superpixels with the convolutional features extracted through a CNN (we use a lightweight ResNet-18 as backbone). Afterward, the superfeatures are embeddings of the convolutional features in the superpixel areas, as illustrated in Fig. 4.

Together with the extracted convolutional features, we encode and concatenate the size and position of the superpixels into three additional channels. The idea is to inform the network about the locations and sizes of the superpixels concerning the original image. Therefore, the added channels include: (i) the relative size of the superpixel (calculated by dividing the number of pixels in the superpixel by the total number of pixels in the image); and (ii) and (iii) the mean position of the superpixel regarding the x- and y-axes, respectively. These meta-data are repeated across the added channels for each pixel that belongs to the superpixel. From this process, we end up with two sets of feature maps: L1, with size H x W x (64 + 3), and L4, with size H/4 x W/4 x (256 + 3).

To generate the superfeatures, for each feature map channel we calculate the average value of the features contained

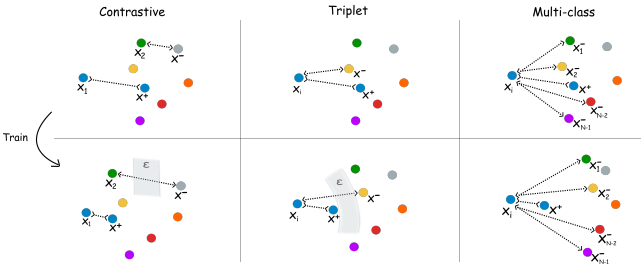

Figure 5. Contrastive losses. In the original contrastive function (left), each sample is contrasted with a single example at a time, whether it is a positive or negative example. In the triplet function (middle) each sample is always contrasted with a pair consisting of a positive and a negative example at the same time. In the multi-class function (right) each sample is contrasted with a positive and multiple negative examples simultaneously.

in the superpixel area. The result is a set of feature maps that are no longer related to the spatial dimensions of the input image but to the number of superpixels generated for the image, *i.e.*, N x C, where N is the number of superpixels and C is the number of channels of the feature map. These features are then passed, one by one, to network heads consisting of fully-connected layers, where each head produces a superfeature prototype of size S, which is then concatenated and passed through a 1x1 convolution to produce the final superfeature.

By generating superfeatures that accurately represent the content of the superpixels, our method can use the latent space to recover this content. In other words, it can use the superfeatures to reconstruct the superpixels that make up the object parts, and ultimately, the object pixels that make up the superpixels.

3.4 Deep Metric Learning based on multi-class contrastive objective

During the training, our model is encouraged to bring together superfeatures that belong to the same object while maximizing the distance between dissimilar samples, *i.e.*, it promotes inter-class contrast.

To achieve this, we employ the NT-Xent loss [Chen *et al.*, 2020]. This function, designed to maximize the agreement between positive pairs in a mini-batch, is a contrastive and multi-class objective. The function constructs a mini-batch $\mathcal{MB}$ by sampling N positive pairs, x_i and x_j, resulting in $\mathcal{MB} = x_{1i}, x_{1j}, ..., x_{Ni}, x_{Nj}$, where $|\mathcal{MB}| = 2N$. The remaining $2\,(N-1)$ samples are then assumed to be negative examples, and the function tries to maximize the cosine similarity between positive pairs, which is measured as

$$sim\,(x_i, x_j) = \frac{x_i \cdot x_j}{||x_i||\;||x_j||}\,, \tag{2}$$

where x_i and x_j are embedding representations of a sampled positive pair. The total loss for all positive pairs, including (i, j) and (j, i), is formalized as

$$\mathcal{L}_{\text{NT-Xent}} = -\frac{1}{N}\sum_{i,j\in\mathcal{MB}} \log \frac{\exp\left(sim\left(\mathbf{x}_i, \mathbf{x}_j\right)/\tau\right)}{\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} \exp\left(sim\left(\mathbf{x}_i, \mathbf{x}_k\right)/\tau\right)}\,, \tag{3}$$

where τ is a temperature hyper-parameter and the function $\mathbb{1}_{[k\neq i]} \in \{0, 1\}$ is 0 when $k = i$ to ignore similarities between the same data point, *i.e.*, $sim\,(x_i, x_k)_{[k=i]}$. Fig. 5 depicts the principle of a multi-class function in comparison to other common contrastive functions.

With the NT-Xent loss in hands, we are able to train the modules of our model described so far, *i.e.*, the feature extractor and the superfeature embedding. Fig. 6 provides an overview of the framework consisting of these elements at the training stage. The training process begins by using the input images to generate an N-length pseudo-sequence of frames and masks. The frames, along with the superpixels provided by ISEC, are passed one at a time to the model (feature extraction + superfeature embedding), which outputs the superfeatures. Meanwhile, the superpixels are combined with the corresponding masks to form the ground-truth labels. Each superfeature-label pair is then compared by the NT-Xent function, and the resulting loss is summed throughout the sequence. Finally, the gradient given by the accumulated loss is back-propagated to adjust the model parameters. The training cycle then restarts with a new generated sequence.

3.5 The need for memory

During the training process illustrated in Fig. 6, we monitor the learning progress of our model. To achieve this, we use the superfeatures and labels of the first frame to fit a KNN algorithm, which is used to classify the superfeatures of the subsequent frames. Based on the classification, we reassemble the superpixels to predict the object masks. However, we observe a significant issue when the model is required to process longer video sequences: the quality of the generated masks deteriorates rapidly as the frame index increases. This phenomenon is evidenced by the plot in Fig. 7.

The plot reveals the behavior of our model as time advances in terms of *segmentation self-consistency*. We have formulated this metric to measure the variation of the segmentation performance along the frame sequence. We measured the segmentation self-consistency in relation to the performance achieved by our method in the first prediction, regardless of the quality of that prediction. More specifically, for each frame sequence, we used the first frame-mask pair to fit the KNN classifier; the predicted mask of the second frame was taken as the reference mask; then, the self-consistency was calculated by dividing the segmentation accuracy for each predicted mask in the sequence by the accuracy of the reference mask. We access the accuracy in terms of intersection over union (IoU). The values plotted in Fig. 7 represent the average self-consistency considering 30 videos from the DAVIS-17 dataset [Pont-Tuset *et al.*, 2017] over an interval of 80 frames.

The graph reveals a severe segmentation degradation by our model without memory, with accuracy dropping to less than 50% of the initial performance around frame #65. On the other hand, with the presence of memory, the performance loss is significantly reduced, remaining above 60% of the initial result throughout the entire interval. We achieve such improvement by introducing an innovative memory clustering mechanism.

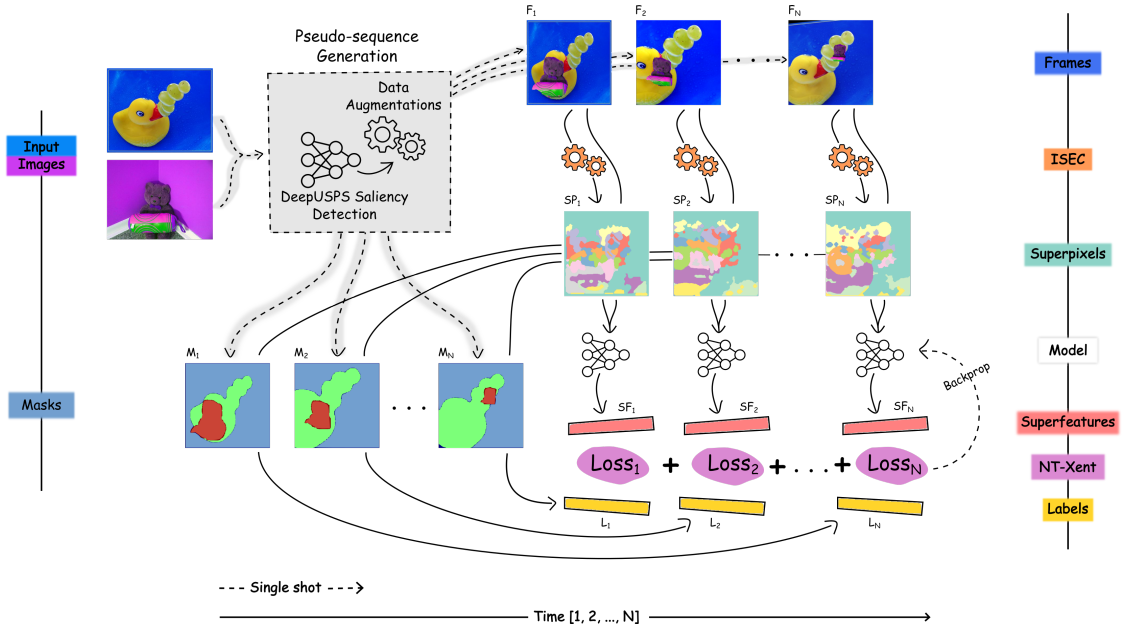

Figure 6. Training process of SHLS including the modules feature extraction and superfeature embedding. The process begins with applying saliency detection and data augmentation on the input images to create a pseudo-sequence. Each pseudo-frame and corresponding superpixels (provided by ISEC) are fed to the model, which extracts the features and generates the superfeatures. Simultaneously, the superpixels are labeled using the object pseudo-masks. The resulting superfeatures and labels are compared using the NT-Xent loss function, and the resulting gradient is back-propagated to adjust the parameters of the model.

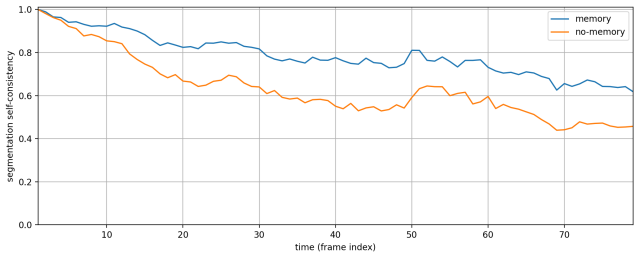

Figure 7. Self-consistency over time. The plot depicts the variation of segmentation performance of our VOS method along the frame sequence, comparing two scenarios: with a memory mechanism (blue) and without memory (orange). The graph highlights the benefits of including a memory mechanism in the proposed solution, as it improves segmentation self-consistency over time.

3.6 Memory clustering

Usually, the affinity matrix in Equation (1) is calculated over the entire key set, *i.e.*, the memory size. Additionally, computing A typically involves expensive matrix operations, which places serious constraints on methods regarding memory management.

We have overcome this constraint by designing a new mechanism that treats memory management as a clustering problem. This mechanism leverages the enormous compactness provided by superfeatures, allowing it to maintain information from virtually all past frames of videos from common VOS datasets without overhead. The idea combines two approaches to provide short- and long-term information through a memory structure that comprises three main stages: fitting, prediction, and update. Both approaches are based on similarity measures among the superfeatures in the latent space.

Short-term memory. The short-term mechanism aims to provide a quick response memory by incorporating information from more immediate changes in objects during short time intervals. This mechanism is based on KNN searches performed on the superfeature latent space. At the **fitting** stage, the superfeatures extracted from the first frame are associated with the classes given by the object labels present in the first mask (provided by the user). In this case, the class is taken from the object label that overlaps the superpixel corresponding to that superfeature the most. At the **prediction** stage, we compute the KNN distances between each query superfeature (*i.e.*, the unclassified superfeatures from the second frame onward) and its nearest labeled superfeatures. Finally, at the **update** stage, those samples for which the class is assigned with high confidence (*i.e.*, the similarity to the class is above a threshold) are incorporated into the search pool.

Long-term memory. Unlike the short-term mechanism, sudden changes in the video scene do not substantially affect the long-term mechanism. This mechanism aims to capture the general tendency that each object presents throughout the entire video sequence. Instead of measuring the similarity between the query and the neighbors, the long-term mechanism performs per-class clustering of the superfeatures at the **fitting** stage. Then, the query similarity with respect to the cluster centroid is measured at the **prediction** stage. The rationale is that the centroid changes gradually as the cluster incorporates new members, which is done at the **update** stage. Another particularity of this mechanism is that each object class present in the scene can be associated with a variable number of clusters. The reason for this is to account for the complexity presented by each object so that their distinct sub-parts can be assigned to different clusters. We define the

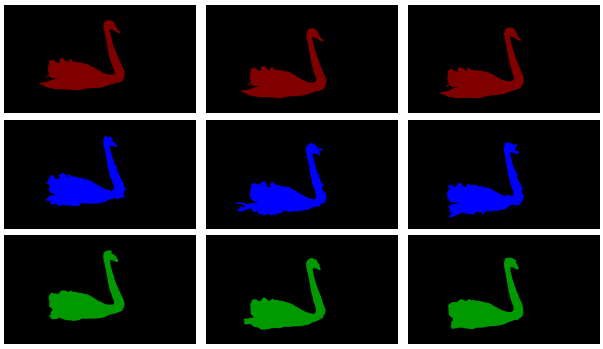

Figure 8. Preliminary segmentation comparison. The top row displays the ground-truth masks (red) for the object in frames number 10, 20 and 30 of a video sequence. The middle row displays the mask predictions (blue) obtained by classifying the superpixels directly. The bottom row displays the mask predictions (green) produced by the segmentation refinement module operating pixel-wisely.

number of clusters for a given class based on the number of superpixels generated by ISEC for the objects of that class in the first frame, which correlates to the object's complexity.

Attention maps. After analyzing the pool of superfeatures for a given frame, the similarity measures provided by the memory clustering mechanism are used to create a set of attention maps. For each object in the frame, two pairs of positive-contrastive maps are generated, one related to short-term memory and the other related to long-term memory. The positive maps in each pair display the similarity measure between the query superfeature and the most similar reference belonging to that class (k-neighbors or centroid). On the other hand, the contrastive maps display the similarity measure between the query superfeature and the most similar reference belonging to any other class. This arrangement of information considers not only a sample's proximity to a class but also its distance from other classes, which helps to resolve potential ambiguities.

3.7 Segmentation refinement

The attention maps allow us to predict object masks for each frame. To create these masks, we assign pixel labels based on the superpixel they belong to. Specifically, the label of the *ith* pixel p belonging to the *jth* superpixel P, with $p_i \subset P_j \;\; \forall\, i \in 1..I_j$ and $\; j \in 1..N$, is estimated as

$$f(p_{i,k}) = \mathsf{S}_j^k + \mathsf{L}_j^k - (\mathsf{S}_j^l + \mathsf{L}_j^l) \;\; \forall \;\; k, l \in 1..C \text{ and } k \neq l\,,$$
$$p_i = \underset{k}{argmax}(f(p_{i,k}))\,, \tag{4}$$

where N is the total of superpixels generated for the frame, C is the number of classes present in the video, S and L are the attention maps from the short- and long-term memories, respectively. Fig. 8 allows us to compare the ground-truth (first row, in red) and the segmentation masks (second row, in blue) obtained from Equation (4).

In Fig. 8 (bottom, in green) are the segmentation masks generated by the refinement module that we have included in our framework. The final results are more regular object masks, with smoother contours that resemble the shape delineated in the ground-truth more closely. To achieve these results, the proposed refinement module was designed as a CNN architecture that accomplishes four primary stages: (i) ROI selection; (ii) feature modulator; (iii) feature decoder; and (iv) multi-object re-aggregation.

ROI selection. Although our VOS method is intended for multi-object segmentation, during the refinement stage we divide this task into a series of single-object tasks. The motivation is to allow the refinement network to focus on learning a simpler, more specific objective. We then use the object masks obtained from Equation 4 to select a region of interest (ROI) for each foreground object. The ROI is determined by the bounding box that encloses the object pixels in the pre-segmentation mask, plus a margin of error. The coordinates of the ROI are then used to select the inputs for the feature modulator stage for each object, as shown in Fig. 9 (top).

Feature modulator. This is a CNN stage that receives the ROI-based L1 and L4 feature maps from the feature extractor, as well as the attention maps from the memory clustering. Its purpose is to function as a gate mechanism, allowing or filtering out features in the feature maps based on the object priors given by the attention maps. We feed the modulator network with features and attention maps from the current frame as well as the last segmented frame in the sequence. This helps the network learn to segment the current frame as a smooth transformation of the previous segmentation. As shown in Fig. 9 (middle), the architecture of the feature modulator includes a branch containing the softmax function, which acts as the gate that modulates features from the parallel branch. The resulting feature map (M4) is then passed to the next stage, the feature decoder.

Feature decoder. This stage is responsible for bringing the features back to the spatial dimensions of the input frame while simultaneously reducing their channels towards the final prediction. In the proposed architecture shown in Fig. 9 (bottom), the main components are two refinement blocks situated in the center of the diagram. These blocks, first introduced by Oh *et al*. [2018], allow for merging features from branches at different scales. In this case, the first merged branch is fed with the mask prediction of the previous frame, which is concatenated with the L1 features and manual annotation from the first frame. As in Yang *et al*. [2020], we found it useful always to include data from the first frame in the feature decoder since it is the most reliable information we have (thanks to the provided manual annotation of the first frame). Thus, the second merged branch includes the L4 features from the first frame along with the modulated features from the modulator stage. Before entering the first refinement block, these features undergo convolutions and residual blocks [He *et al*., 2016] to adjust their number of channels. Following this, we have the second refinement block, which implements skip connections by receiving already used inputs comprising the L1 features of the first frame, as well as the attention maps of the current frame. Finally, the output of this refinement block passes through a last convolution that generates a 2-channel object prediction.

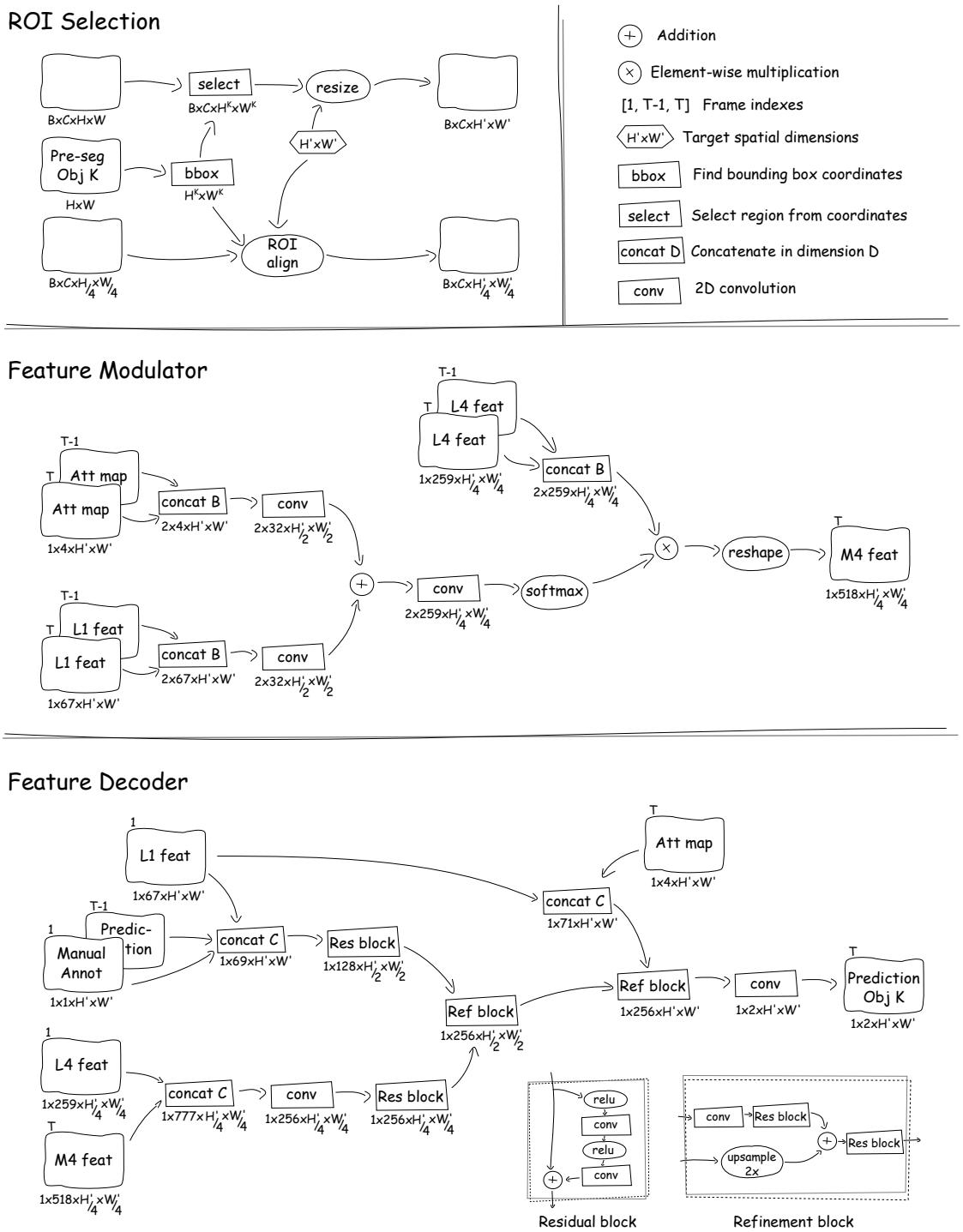

Figure 9. Segmentation refinement module. The ROI selection stage (Fig. 9, top) provides two possible pathways: for inputs with unchanged scale, the ROI is selected and the height and width are resized to the target dimensions; for down-sampled inputs, the ROI align function of the Mask R-CNN [He *et al*., 2017] is applied. The feature modulator stage (middle) combines the L1 features and the attention maps in a softmax-based gate mechanism that modulates the L4 features. Finally, the feature decoder stage (bottom) relies on residual and refinement blocks to reduce the channels and increase the spatial dimensions of the features in order to predict the object masks.

Multi-object re-aggregation. As described, the networks that make up the segmentation refinement module run for each object individually. Therefore, after obtaining all the predictions related to a frame, it is necessary to combine them into a unified prediction to return to the multi-object context. We achieve this by employing the same soft-aggregation function as in Oh *et al*. [2018]. This function combines multiple instance probabilities softly while constraining them to be positive and sum to 1, and it is given by

$$p_{i,k} = \sigma\left(\text{logit}(\hat{p}_{i,k})\right) = \frac{\hat{p}_{i,k}/(1-\hat{p}_{i,k})}{\sum_{j=0}^{K}\hat{p}_{i,j}/(1-\hat{p}_{i,j})}, \quad (5)$$

where σ and logit represent the softmax and logit functions, respectively, $\hat{p}_{i,k}$ is the network output probability of the object k at the pixel location i for a total of K object classes ($k = 0$ indicates the background). In this case, the probability of the background is estimated by subtracting from 1 the merged probabilities of all foreground instances. At the training, the multi-object prediction given by Equation (5) is used to compute a pixel-wise cross-entropy loss, which drives the adjustment of the refinement module parameters.

4 Experimental analysis

VOS methods are typically evaluated based on two metrics: region Jaccard similarity ($\mathcal{J}$), which is equivalent to the mean IoU; and boundary F-measure ($\mathcal{F}$), which evaluates the accuracy of the boundary localization. The overall performance is calculated as the mean of both metrics, denoted by $\mathcal{J}\&\mathcal{F} = (\mathcal{J} + \mathcal{F})/2$.

We compared the performance of SHLS against representative self-supervised VOS baselines that reflect the state of the art in this research field over recent years up to the present. The selected methods provide publicly reported results on the standard DAVIS 2017 benchmark for multi-object segmentation, ensuring a fair comparison.

- **VidColor** [Vondrick *et al*., 2018]
- **CorrFlow** [Lai and Xie, 2019]
- **CycleTime** [Wang *et al*., 2019]
- **JTTC** [Li *et al*., 2019]
- **RPM-Net** [Kim *et al*., 2020]
- **MAST** [Lai *et al*., 2020]
- **MUG** [Lu *et al*., 2020]
- **CRW** [Jabri *et al*., 2020]
- **DUL** [Araslanov *et al*., 2021]
- **TWIAA** [Zhu *et al*., 2021]
- **STT** [Li and Liu, 2023]
- **MAMP** [Miao *et al*., 2022]
- **PLR** [Guo *et al*., 2025]

All experiments with SHLS were conducted on a computing environment equipped with eight NVIDIA Tesla V100 GPUs (16 GB each). Training and inference were performed using PyTorch, with mixed-precision enabled to optimize memory usage and throughput. No cloud infrastructure was used; all experiments were executed locally on this multi-GPU setup.

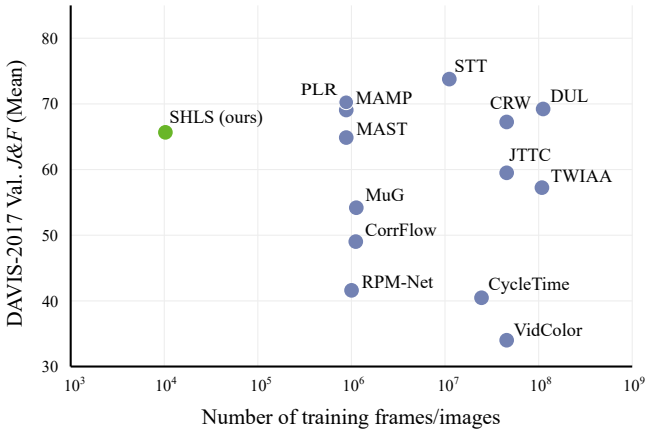

Figure 10. Self-supervised VOS methods' overall segmentation performance ($\mathcal{J}\&\mathcal{F}$) on the DAVIS-2017 validation split in terms of the number of images and/or video frames used for training. The results for the compared methods correspond to the values reported in their original publications. Our method presents competitive results even being trained with at least 10^2 orders of magnitude fewer frames/images than other self-supervised approaches.

4.1 Multi-object segmentation test

The DAVIS-2017 [Pont-Tuset *et al*., 2017] validation split is considered the gold standard for multi-object VOS, as every method published after its release has reported results on this dataset.

Fig. 10 shows the overall performance of the compared methods on the DAVIS-2017 validation set. The results are plotted in terms of the number of images and/or video frames used for training. Among the compared methods, STT demonstrated the best performance, achieving impressive 74.1% of $\mathcal{J}\&\mathcal{F}$, which surpasses the performance of all non-memory-based supervised methods. After STT, there is a group of methods that have achieved $\mathcal{J}\&\mathcal{F}$ values greater than 65%. This group includes MAMP, DUL, CRW, MAST, and the proposed SHLS method, which is the only one trained exclusively with still images. As shown in the plot, our method is competitive even being trained with at least 10^2 orders of magnitude less data than top-performance approaches.

4.2 Ablative evaluation

We performed an ablative analysis to investigate how SHLS is influenced by different superpixel segmentation methods. We also examined the effect of the memory mechanism across different memory sizes, and assessed the benefit of the segmentation refinement module. These evaluations were based on the DAVIS-2017 validation set.

Superpixel segmentation: ISEC x SLIC. Here, we compare SHLS using the standard ISEC-based setup with a configuration based on SLIC [Achanta *et al*., 2012]. The plot in Fig. 11 indicates that SHLS achieves a better balance between segmentation performance and the number of superpixels when paired with ISEC instead of SLIC. Specifically, SHLS + ISEC achieves an average of 66.6% $\mathcal{J}\&\mathcal{F}$ with around 1,000 superpixels, while SHLS + SLIC requires an average of about 1,400 superpixels to reach the same $\mathcal{J}\&\mathcal{F}$.

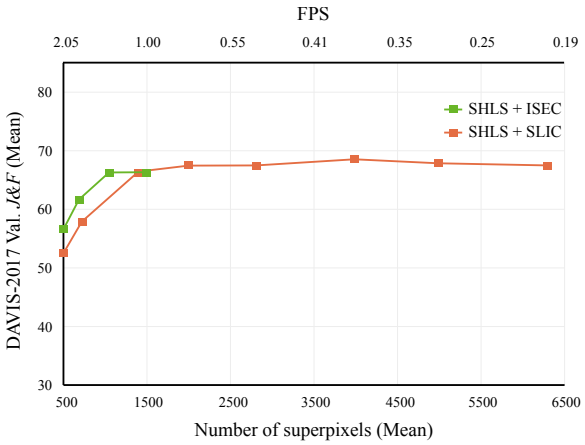

Figure 11. Comparison between SHLS performance by using ISEC and SLIC. The plot comprises the overall segmentation performance ($\mathcal{J}\&\mathcal{F}$) on the DAVIS-2017 validation split as a function of the number of generated superpixels. The upper axis shows the corresponding frame processing rate in terms of frames per second (FPS).

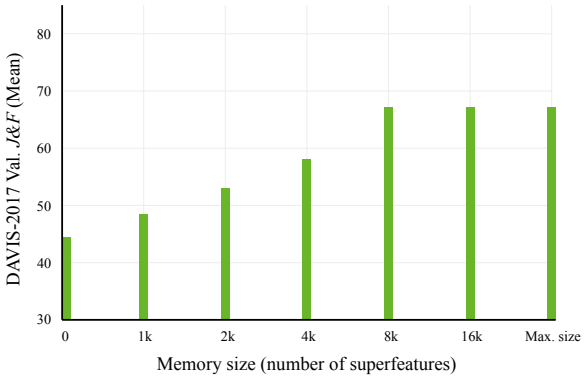

Figure 12. Impact of the memory size in SHLS performance. The plot comprises the overall segmentation performance ($\mathcal{J}\&\mathcal{F}$) on the DAVIS-2017 validation split as a function of the maximum number of superfeatures stored in the memory. Maximum size means no memory limitation, *i.e.*, all generated superfeatures are allowed to be stored in the memory.

Impact of the memory size. In this test we ran SHLS while varying the memory size from zero (*i.e.*, without memory) and then progressively increased the size towards the maximum size (corresponding to all generated superfeatures for a video). The results are plotted in Fig. 12. The result reinforces the idea that the memory mechanism is a crucial component of our method, as its segmentation performance is significantly impacted when no memory is used, and progressively improves with the memory size, up to a maximum performance at memory size of ∼8k superfeatures.

Benefit of the segmentation refinement module. To assess the effect of this module on SHLS performance, we tested our method in three different configurations: (i) without refinement, *i.e.*, the final result is given by the pre-segmentation directly; (ii) without ROI selection, *i.e.*, the feature modulator and feature decoder sub-modules are applied to the entire spatial area of the feature maps; and (iii) the segmentation refinement module is used integrally, *i.e.*, the feature modulator and feature decoder are focused on the ROI given by the attention maps. The results in Table 1 show that without the segmentation refinement module, SHLS achieves an overall performance of ∼60.5%. The refinement process involves applying the feature modulator and feature decoder networks. Firstly, on the entire feature maps, resulting in a raise of 3 percentage points of $\mathcal{J}\&\mathcal{F}$; and then, focusing on the ROI, leading to an additional 3 percentage points in performance.

Pre-seg (Eq. 4)	Feat. modulator + Feat. decoder	ROI selection	$\mathcal{J}\&\mathcal{F}$
✓			60.5
✓	✓		63.5
✓	✓	✓	66.5

Table 1. Ablation study of the segmentation refinement module.

5 Conclusion

In this paper, we introduced SHLS, an innovative self-supervised VOS method built on highly compressed superpixel-based representations known as superfeatures. By employing a memory clustering mechanism, SHLS effectively organizes superfeatures into per-object clusters, enabling efficient retrieval of past frame information. Despite being trained on only 10k still images, the innovative memory mechanism allows SHLS to deliver remarkable performance in multi-object scenarios on the DAVIS dataset. These results underscore the efficiency of our fully self-supervised training approach. However, SHLS still faces limitations, such as reduced performance in scenes with rapid appearance changes or very fine-grained object boundaries due to the superpixel-based feature compression. Additionally, the current inference pipeline assumes a known foreground at the first frame. As future work, we plan to integrate automatic foreground detection during inference, paving the way for SHLS to operate within the zero-shot VOS framework, further enhancing its versatility and practical applicability.

Declarations

Authors' Contributions

Mendonça, M. is the main contributor and writer of this manuscript. Oliveira, L. contributed to the conception of this study and also performed supervision and writing (review and editing). All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Availability of data and materials

The datasets and softwares generated during the current study are available in: `https://github.com/IvisionLab/SHLS`.

References

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Süsstrunk, S. (2012). Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*. DOI: 10.1109/TPAMI.2012.120.

Araslanov, N., Schaub-Meyer, S., and Roth, S. (2021). Dense unsupervised learning for video segmentation. In *Advances in Neural Information Processing Systems*.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*.

Cheng, H. K. and Schwing, A. G. (2022). Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *Computer Vision – ECCV 2022*. DOI: 10.1007/978-3-031-19815-1$_3$7.

Cheng, M.-M., Mitra, N. J., Huang, X., Torr, P. H. S., and Hu, S.-M. (2015). Global contrast based salient region detection. *IEEE TPAMI*. DOI: 10.1109/TPAMI.2014.2345401.

Guo, P., Zhang, W., Li, X., Fan, J., and Zhang, W. (2025). Self-supervised video object segmentation via pseudo label rectification. *Pattern Recogn.*. DOI: 10.1016/j.patcog.2025.111428.

He, K., Gkioxari, G., Dollar, P., and Girshick, R. (2017). Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV.2017.322.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Identity mappings in deep residual networks. In *Computer Vision – ECCV 2016*. DOI: 10.1007/978-3-319-46493-0$_3$8.

Hou, R., Chen, C., and Shah, M. (2017). An end-to-end 3d convolutional neural network for action detection and segmentation in videos. 10.48550/ARXIV.1712.01111.

Jabri, A., Owens, A., and Efros, A. A. (2020). Space-time correspondence as a contrastive random walk. *Advances in Neural Information Processing Systems*.

Kim, Y., Choi, S., Lee, H., Kim, T., and Kim, C. (2020). Rpm-net: Robust pixel-level matching networks for self-supervised video object segmentation. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*. DOI: 10.1109/WACV45572.2020.9093294.

Lai, Z., Lu, E., and Xie, W. (2020). MAST: A memory-augmented self-supervised tracker. In *IEEE Conference on Computer Vision and Pattern Recognition*. DOI: 10.1109/CVPR42600.2020.00651.

Lai, Z. and Xie, W. (2019). Self-supervised learning for video correspondence flow. In *BMVC*.

Li, M., Hu, L., Xiong, Z., Zhang, B., Pan, P., and Liu, D. (2022). Recurrent dynamic embedding for video object segmentation. In *Conference on Computer Vision and Pattern Recognition*. DOI: 10.1109/CVPR52688.2022.00139.

Li, R. and Liu, D. (2023). Spatial-then-temporal self-supervised learning for video correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2279–2288. DOI: 10.1109/CVPR52729.2023.00226.

Li, X., Liu, S., De Mello, S., Wang, X., Kautz, J., and Yang, M.-H. (2019). Joint-task self-supervised learning for temporal correspondence. In *Advances in Neural Information Processing Systems*.

Lu, X., Wang, W., Shen, J., Tai, Y., Crandall, D. J., and Hoi, S. H. (2020). Learning video object segmentation from unlabeled videos. In *Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR42600.2020.00898.

Mendonça, M., Fontinele, J., and Oliveira, L. (2023). SHLS: Superfeatures learned from still images for self-supervised vos. In *34th British Machine Vision Conference BMVC, Aberdeen, UK*.

Mendonça, M. and Oliveira, L. (2018). ISEC: Iterative over-segmentation via edge clustering. *Image and Vision Computing*, 80:45–57. DOI: 10.1016/j.imavis.2018.09.015.

Miao, B., Bennamoun, M., Gao, Y., and Mian, A. (2022). Self-supervised video object segmentation by motion-aware mask propagation. In *International Conference on Multimedia and Expo (ICME)*. DOI: 10.1109/ICME52920.2022.9859966.

Nguyen, D. T., Dax, M., Mummadi, C. K., Ngo, T. P. N., Nguyen, T. H. P., Lou, Z., and Brox, T. (2019). *DeepUSPS: Deep Robust Unsupervised Saliency Prediction with Self-Supervision*. Curran Associates Inc.

Oh, S. W., Lee, J.-Y., Sunkavalli, K., and Kim, S. J. (2018). Fast video object segmentation by reference-guided mask propagation. In *Conference on Computer Vision and Pattern Recognition*. DOI: 10.1109/CVPR.2018.00770.

Oh, S. W., Lee, J.-Y., Xu, N., and Kim, S. J. (2019). Video object segmentation using space-time memory networks. In *Proceedings of the International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV.2019.00932.

Pont-Tuset, J., Perazzi, F., Caelles, S., Arbeláez, P., Sorkine-Hornung, A., and Van Gool, L. (2017). The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*. 10.48550/arXiv.1704.00675.

Tokmakov, P., Alahari, K., and Schmid, C. (2017). Learning video object segmentation with visual memory. In *International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV.2017.480.

Ventura, C., Bellver, M., Girbau, A., Salvador, A., Marques, F., and Giro-i Nieto, X. (2019). Rvos: End-to-end recurrent network for video object segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR.2019.00542.

Vondrick, C., Shrivastava, A., Fathi, A., Guadarrama, S., and Murphy, K. (2018). Tracking emerges by colorizing videos. In *Computer Vision – ECCV 2018: 15th European Conference*. DOI: 10.1007/978-3-030-01261-8$_2$4.

Wang, X., Jabri, A., and Efros, A. A. (2019). Learning correspondence from the cycle-consistency of time. In *CVPR*. DOI: 10.1109/CVPR.2019.00267.

Xu, N., Yang, L., Fan, Y., Yang, J., Yue, D., Liang, Y., Price, B., Cohen, S., and Huang, T. (2018). Youtube-vos: Sequence-to-sequence video object segmentation. In *Computer Vision – ECCV 2018 - 15th European Conference, 2018, Proceedings*. DOI: 10.1007/978-3-030-01228-1$_3$6.

Xu, X., Wang, J., Li, X., and Lu, Y. (2022). Reliable propagation-correction modulation for video object segmentation. *Proceedings of the AAAI Conference on Artificial Intelligence*. DOI: 10.1609/aaai.v36i3.20200.

Yang, Z., Wei, Y., and Yang, Y. (2020). Collaborative video object segmentation by foreground-background integration. In *Computer Vision – ECCV 2020: 16th European Conference*. DOI: 10.1007/978-3-030-58558-7$_2$0.

Zhu, W., Meng, J., and Xu, L. (2021). Self-supervised video object segmentation using integration-augmented attention. *Neurocomput.*. DOI: 10.1016/j.neucom.2021.04.090.