# Advancing Video Object Segmentation with Spatio-Temporal Markov Random Fields

**Wang Ganyi**
School of Computing
National University of Singapore
Singapore
e1101663@u.nus.edu

**Li Xingchen**
School of Computing
National University of Singapore
Singapore
e0945876@u.nus.edu

**Zhang Rongqi**
School of Computing
National University of Singapore
Singapore
e1132299@u.nus.edu

**Yang Boran**
School of Computing
National University of Singapore
Singapore
boranyang@u.nus.edu

## 1 Introduction

Video Object Segmentation is a critical challenge in computer vision, which aims to accurately separate foreground object(s) from the background in consecutive video frames. This task requires not only the recognition of objects in a single frame, just like the typical image segmentation task, but also the continuous tracking of these object(s). In addition, in real scenarios, the video object segmentation task is affected by several factors, such as lighting conditions, occlusions, and the variety of object appearances and backgrounds in video streams, which significantly increase the complexity of the task.

There are multiple reasons that drive us to address the video object segmentation challenge. First, the video object segmentation task is the basis for many advanced applications that often require a high-level understanding of the video content. For example, in autonomous driving, which has become very popular in recent years, the ability to apply accurate segmentation and track objects is important for the safety and reliability of navigation systems. Similarly, in video content AI creation and editing, accurately recognising and processing objects in a scene is fundamental to enabling further complex editing. Therefore, any advancement in video object segmentation tasks in these related application areas not only pushes the boundaries of computer vision but also opens up new possibilities.

On the other hand, the nature of video object segmentation involves several other challenging domains in computer vision, including object recognition, motion analysis, and scene understanding. Attempting to solve the video object segmentation problem often requires researchers to have a broad understanding of how models interpret dynamic visual information. Any developments in these domains will likely provide new ideas for developing better approaches to solving video object segmentation tasks.

Furthermore, the video object segmentation task is very interesting due to its complexity and the creative solutions required. Many related works exist in this area, and we will introduce some interesting ideas and studies in Section 3. One notable benchmark in this domain is the DAVIS 2017 Challenge (Densely Annotated VIdeo Segmentation). This challenge provides a rigorously annotated dataset designed to test the robustness of segmentation algorithms under various challenging conditions. We chose to verify our method using the DAVIS 2017 Challenge due to its comprehensive and diverse set of video sequences, which include multiple objects per scene, rapid object movements, and frequent occlusions, making it an ideal testbed to evaluate the effectiveness and efficiency of our proposed solutions in real-world scenarios.

### 1.1 Motivation

Our method aims to improve the accuracy of video segmentation in complex environments by integrating convolutional neural networks (CNN) and Markov random fields (MRF). CNN is adopted for its powerful feature extraction capabilities, generating detailed maps of each frame, while MRF models the dependencies between these frames to maintain temporal and spatial continuity. This collaborative approach not only accounts for differences in frame-to-frame changes, especially in sequences with fast-moving objects but also ensures a realistic depiction of object trajectories and interactions. The implementation involves fine-tuning CNN and MRF models to work seamlessly together, bridging the gap between per-frame analysis and overall sequence-aware insights, thereby significantly improving video segmentation performance.

### 1.2 Related Work

Historically, video object segmentation research began with simpler, rule-based algorithms that relied heavily on colour and texture features to distinguish objects from the background. As artificial intelligence evolved, the advent of machine learning and, subsequently, deep learning opened up many new possibilities. This section reviews some interesting work related to video object segmentation.

### 1.3 Handcrafted Feature-based Methods

Optical flow, depicting object motion due to object or camera movement between frames, is a pivotal feature in video segmentation. By effectively capturing the temporal and spatial relationships of moving objects, it estimates motion vectors for pixel position changes across frames, aiding in understanding scene motion patterns and facilitating object segmentation. FlowNet [1] and its successor FlowNet2 [2], which is used in our method, significantly enhance optical flow estimation with Convolutional Neural Networks (CNNs), addressing traditional method limitations through deep learning. FlowNet includes two main architectures: FlowNetS, which stacks two consecutive frames for CNN processing, and FlowNetC, which uses a correlation layer for refined motion analysis between frames.

Tsai, et al. [3] proposed a method that considers the optical flow estimation and video segmentation at the same time. Specifically, they compute the flow independently in the segmented regions and recompose the results to estimate the optical flow. They also design a multi-scale spatio-temporal objective function to transfer information between frames using the optical flow.

### 1.4 Classical Machine Learning Methods

Yu, et al. [4] introduced a video segmentation framework utilizing parametric graph partitioning (PGP), a method with minimal parameter requirements that works by identifying and eliminating edges between clusters to create clusters of nodes. PGP is known for its computational efficiency and ability to cluster spatio-temporal volume without the need to pre-determine the number of clusters.

Grundmann, et al. [5] proposed an approach for the spatiotemporal segmentation of long video sequences through a hierarchical graph-based method. The process is initiated by over-segmenting a volumetric video graph into space-time regions that are categorized based on their appearance. Subsequently, a "region graph" is formed from the segmentation, and this procedure is iteratively applied across several layers to generate a hierarchical tree of spatio-temporal segmentations.

Jang and Kim [6] proposed a method that employs short-term hierarchical segmentation (STHS) and couples it with frame-by-frame optimization using a Markov random field (MRF). STHS initially segments by moving through a window of frames, applying spatial agglomerative clustering to each frame, followed by inter-frame bipartite graph matching to create initial segments. Subsequently, they obtained the final segments for each frame by minimizing an MRF energy function that includes both unary and pairwise costs and refining the solution using Iterated Conditional Modes (ICM).

### 1.5 Deep Neural Network Models

Perazzi, et al. [7] presented the idea of using a convolutional network-based method in video object segmentation. The proposed model considers the results of the previous frame and guides the recognition of the interest object in the next frame.

Chandra, et al. [8] proposed a method that integrates neuronal decisions across both spatial and temporal dimensions. By leveraging the progress in deep Gaussian Conditional Random Fields (GCRFs), they demonstrated the capability for exact and efficient inference on a densely connected spatio-temporal graph.

DeepLabV3, developed by Chen, et al. [9], represents a significant advancement in semantic image segmentation. Utilizing atrous convolution allows the network to control the resolution at which feature responses are computed within Deep Convolutional Neural Networks. The atrous spatial pyramid pooling (ASPP) system also optimizes the model's ability to manage scene context by capturing objects at multiple scales. This architecture has been particularly effective in video segmentation tasks where multiple objects occur, facilitating precise multi-label segmentation and improving the ability to delineate object boundaries over consecutive frames. In our method, at the start of each iteration, DeepLabV3 performs a forward pass to compute the current image features and predict an initial mask. This predicted mask then serves as a starting point for the Iterated Conditional Modes (ICM) optimization, enhancing the refinement of segmentation results.

One-Shot Video Object Segmentation (OSVOS) [10] is a semi-supervised video object segmentation technique. OSVOS aims to separate an object from the background in a video sequence. By transferring this knowledge to the task of foreground segmentation and fine-tuning on a single annotated object in the test sequence, OSVOS achieves impressive results. OSVOS processes each video frame independently, leading to temporally coherent and stable segmentations without explicit temporal constraints. It is chosen as our baseline benchmark.

## 2 Problem Statement

We describe a discrete random field $\mathbf{X}$ which is defined over all pixels in the video sequence $\mathcal{V} = \{1, 2, ..., N\}$. Each variable $\mathbf{X}_i$ in $\mathbf{X}$ corresponds to a pixel $\mathbf{i}$ in $\mathcal{V}$ and selects a value $\mathbf{x}_i$ from a set of labels $\mathcal{L} = \{0, 1\}$. The video frame data is denoted by $\mathbf{D}$. In the field, a chunk is denoted by $\mathbf{c}$ and a variable in the chunk is denoted by $\mathbf{x}_c$. In the field, a chunk is denoted by $\mathbf{c}$, and the variable in the chunk is denoted by $\mathbf{x}_c$. The distribution of the random variable in the field is denoted by the product of the potential functions on the largest chunk [12]:

$$p(\mathbf{x} \mid \mathbf{D}) = \frac{1}{Z} \prod_c (\psi_c(\mathbf{x}_c \mid \mathbf{D})) = \frac{1}{Z} \prod_c (\exp\{-E_c(\mathbf{x}_c \mid \mathbf{D})\}) \tag{1}$$

Here, $\mathbf{Z}$ represents the normalization constant and $\mathbf{E}_c(\mathbf{x}_c \mid \mathbf{D})$ is the energy function associated with the potential function $\psi_c(\mathbf{x}_c \mid \mathbf{D})$. Our goal is to determine the maximum a posteriori (MAP) labelling $\mathbf{x}^*$ of the random field, as shown in Equation 2, which is equivalent to finding the solution $\mathbf{x}^*$ that matches the ground truth object mask, to label pixels as binary values.

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} \log p(\mathbf{x} \mid \mathbf{D}) = \arg\min_{\mathbf{x}} \sum_c E_c(\mathbf{x}_c \mid \mathbf{D}) \tag{2}$$

This can be achieved by defining the graph structure of the random field and its corresponding energy function, thus deriving the MAP label by minimizing the total energy in the field. We note that the energy functions in our model depend on the data $\mathbf{D}$, but for simplicity, we will omit $\mathbf{D}$ in the notation later.

## 3 Methodology

The following ideas are inspired by [11]. Our codes can be found in Github repository: `github.com/16261909/CS5340-Project`

## 3.1 Energy Function Design

The total energy of a state $\mathbf{x}$, crucial for understanding the frame's properties, is calculated using Equation 3:

$$E(\mathbf{x}) = \sum_{i\in\mathcal{V}} E_u(x_i) + \sum_{(i,j)\in\mathcal{N}_T} E_t(x_i, x_j) + \sum_{c\in\mathcal{S}} E_s(\mathbf{x}_c), \tag{3}$$

In this equation, $E_u$ represents monadic energy, while $E_t$ and $E_s$ are energies associated with temporal and spatial dependencies, respectively. The symbol $\mathcal{N}_T$ denotes the set of all temporal connections, while $\mathcal{S}$ denotes the set of all spatial parcels.

Detailed definitions of these components are provided below,

**Unary Energy** $E_u(x_i)$: This is defined as the negative log-likelihood of the labelling for each individual random variable, as $E_u(x_i) = -\theta_u \log p(X_i = x_i)$. The parameter $\theta_u$ is used to balance the weight of this term relative to other energy components. This energy component is derived from the OSVOS model, which provides a baseline estimate of the state of each pixel regardless of its context in the frame or video.

**Temporal Energy** $E_t(x_i, x_j)$: This component is calculated using an optical flow algorithm and evaluates energy based on the temporal consistency of pixel states across frames. By comparing the traditional OpenCV method and the advanced FlowNet technology, we selected FlowNet2, which has superior performance in terms of noise reduction and accuracy.

$$E_t(x_i, x_j) = \theta_t w_{ij}(x_i - x_j)^2, \tag{4}$$

The set of temporal connections $\mathcal{N}_T$, is established using semi-dense optical flow and restricted to reliable motion estimates between pixels in neighboring frames, ensured by a forward-backward consistency check [13, 15, 14]. We extend one-step temporal dependencies to k-step (with $k \leq 2$ and $k \neq 0$) by computing optical flow directly between frames spaced by k steps. This results in each pixel having up to four temporal connections, represented as $\{t \leftrightarrow t-2\}$, $\{t \leftrightarrow t-1\}$, $\{t \leftrightarrow t+1\}$, $\{t \leftrightarrow t+2\}$. The temporal energy function, defined in Equation 4, uses $\theta_t$ for balancing and $w_{ij}$ as a weight measuring the confidence of temporal connections between variables $X_i$ and $X_j$, promoting temporally consistent labelling where connections are confident.

The temporal confidence weighting $w_{ij}$ in Equation 4 is determined by a decaying function based on the frame index, expressed as $w_{ij} = \xi_{t_i}\xi_{t_j}$. Here, $\xi_t$ is defined as $\max(0.96^{t-1}, 0.4)$ for each frame t, ranging from 1 to the total number of frames in a video. $\theta_u$ is defined as the energy balancing parameters.

In order to avoid erroneous information caused by generated unstable optical flow, we use sigma detection to filter out unreliable streams, thereby obtaining the difference from this frame to the next frame. For each position of each frame, we split the coordinates **x** and **y** out and calculate it separately, using Equation 5, where $\mu$ represents original optical flow and $\mu^*$ represents reliable optical flow.

$$\mu^*_{i,n,d} = \begin{cases} x_{i,n,d} & \text{if } |x_{i,n,d} - \mu_{i,d}| \leq 2\sigma_{i,d} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

**Spatial Energy** $E_s(\mathbf{x}_c)$: For spatial Energy, we define every pixel in a frame as a clique, where each pixel's labelling depends on all other pixels within the same frame. To establish a spatial energy function across all pixels in a frame, we design an energy function $f(\cdot)$, which assesses the quality of a complete mask $\mathbf{x}_c$. Ideally, if the ground-truth mask $x^*_c$ is known, constructing $f(\cdot)$ is straightforward. For example, the function can be defined as: $f(\mathbf{x}_c) = \frac{1}{2}\|\mathbf{x}_c - x^*_c\|^2$, which assigns lower energies to masks more closely resembling the ground-truth mask.

However, since $\mathbf{x}^*_c$ is unknown and the target of our solution, we utilize a feed-forward CNN to approximate $\mathbf{x}^*_c$. The function is then defined as: $f(\mathbf{x}_c) = \frac{1}{2}\|\mathbf{x}_c - g_{\text{CNN}}(\mathbf{x}_c)\|^2$, where $g_{\text{CNN}}(\cdot)$ is a mask refinement CNN that takes a given mask $\mathbf{x}_c$ as input and outputs a refined mask. The operator $g_{\text{CNN}}(\cdot)$ represents a feed-forward pass of the CNN, favoring masks that, when passed through $g_{\text{CNN}}(\cdot)$, remain more similar to the original. When using CNN for object segmentation, we then optimize a loss function. We understand the loss function as a spatial energy function, and the continuous optimization process is approximate to the initial loss.

This approach enables the function $f(\cdot)$ to assign lower energies to better masks. A well-trained $g_{\text{CNN}}(\cdot)$, capable of refining a coarse mask to a better one without altering a good mask, has been shown to be trainable in a two-stage manner using the first frame of a video, performing reliably during inference on subsequent frames [7].

We define the spatial energy as: $E_s(\mathbf{x}_c) = \eta_t \theta_s f(\mathbf{x}_c)$, where $\theta_s$ is a balancing parameter. This approach offers greater expressive power compared to traditional pairwise smoothness energies [16], higher-order energies enforcing label consistency, or energies that promote labels to conform to learned patterns [17]. Similar to the temporal energy, we also multiply the decay parameter based on the frame index $\eta_t = max(0.98^{t-1}, 0.4)$, this is defined as the energy balancing parameter.

However, inference in the Markov Random Field (MRF) with the CNN-based energy is complex, and the following section discusses an efficient approximate algorithm for this inference.

## 3.2 Inference

Performing exact Maximum A Posteriori (MAP) inference in general [18] MRF models is known to be NP-hard. Efficient approximate algorithms like belief propagation or mean-field approximation still face challenges in minimizing Equation 3 due to the presence of very high-order spatial cliques. Each evaluation of the total energy in the MRF requires a feed-forward CNN pass for each video frame, leading to prohibitively high computational costs as the number of energy evaluations increases.

To manage this complexity, we separate the temporal energy ($E_t$) and spatial energy ($E_s$) by introducing an auxiliary variable $\mathbf{y}$. We then aim to minimize an approximation of Equation (3), given by:

$$\hat{E}(x, y) = \sum_{i \in \mathcal{V}} E_u(x_i) + \sum_{(i,j) \in \mathcal{N}_T} E_t(x_i, x_j) + \frac{\beta}{2} \|\mathbf{x} - \mathbf{y}\|_2^2 + \sum_{c \in S} E_s(\mathbf{y}_c), \tag{6}$$

where $\beta$ is a penalty parameter ensuring $\mathbf{y}$ closely approximates $\mathbf{x}$. This approximation can be minimized through iterative steps, alternating updates between $\mathbf{x}$ and $\mathbf{y}$:

1. With $\mathbf{y}$ held fixed, update $\mathbf{x}$ by solving:

$$\mathbf{x}^{(k)} \leftarrow \arg\min_{\mathbf{x}} \hat{E}(\mathbf{x}, \mathbf{y}^{(k-1)}), \tag{7}$$

This step essentially regularizes the total energy by disregarding spatial dependencies.

2. With $\mathbf{x}$ held fixed, update $\mathbf{y}$ by solving:

$$\mathbf{y}^{(k)} \leftarrow \arg\min_{\mathbf{y}} \hat{E}(\mathbf{x}^{(k)}, \mathbf{y}), \tag{8}$$

This step focuses solely on the spatial dependencies for each frame.

The solution to the first step involves a large quadratic integer programming problem with a Laplacian matrix size of $N \times N$, where $N$ is the number of pixels in a video. For practical efficiency, we adopt the Iterated Conditional Modes (ICM) approach for an approximate solution as Algorithm 1, performing updates for a fixed number of iterations.

In the second step, each frame $c$ updates independently:

$$\mathbf{y}_c^{(k)} \leftarrow \arg\min_{\mathbf{y}_c} \left( \frac{\beta}{2} \|\mathbf{x}_c^{(k)} - \mathbf{y}_c\|_2^2 + E_s(\mathbf{y}_c) \right). \tag{9}$$

Given the highly non-convex nature of the CNN-based energy function $E_s$, direct optimization is challenging. Instead, we approximate the update using a CNN function $g_{CNN}$:

$$\mathbf{y}_c^{(k)} \leftarrow g_{CNN}(\mathbf{x}_c^{(k)}), \tag{10}$$

**Algorithm 1** Inference Algorithm

```
1: Input: Number of outer iterations $K$, number of inner iterations $L$, number of pixels $N$, number of frames $C$.
2: Initialization: Initial labeling $x^{(0)} = y^{(0)}$.
3: for $k = 1$ to $K$ do
4:     $x^{(k,0)} \leftarrow x^{(k-1)}$
5:     for $l = 1$ to $L$ do
6:         for $i = 1$ to $N$ do
7:             $x_i^{(k,l)} \leftarrow \arg\min_{x_i} \left( \frac{\beta}{2}(x_i - y_i^{(k-1)})^2 + E_u(x_i) + \sum_{(i,j)\in NT} E_t(x_i, x_j^{(k,l-1)}) \right)$
8:         end for
9:     end for
10:    $x^{(k)} \leftarrow x^{(k,L)}$
11:    for $c = 1$ to $C$ do
12:        $y_c^{(k)} \leftarrow g_{CNN}(x_c^{(k)})$
13:    end for
14: end for
15: Output: Segmentation mask $y^{(K)}$.
```

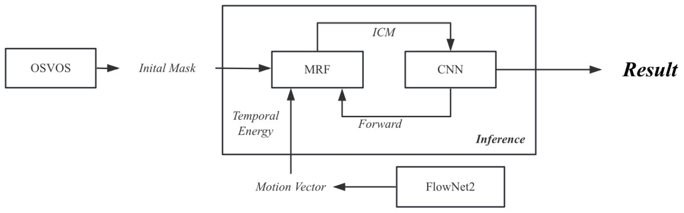

Figure 1: Computational flow of the iterative inference process

The CNN operator $g_{CNN}$ in our algorithm takes a 4-channel input (RGB image plus a coarse mask) and outputs a refined mask. The cross-entropy loss function is used to compare the model's output with the ground truth mask. The training of $g_{CNN}$ involves a two-stage process. Initially, an offline model is trained using available object segmentation data Imagenet. In the second stage, this model is fine-tuned using ground-truth masks from the first video frame.

For multi-label tasks, we compute the possible mask values of each pixel and update them using the value with the minimal energy value. The CNN we used can generate multi-label masks given a multi-object image so that this method can handle multi-label tasks.

In our experiments, we assessed the time complexity of Algorithm 1. The original method increases the computational load by categorizing each object into two classes individually to select the highest probability category. The complexity of our algorithm is $O(K(f_\theta(N) + N \cdot dT \cdot L \cdot B))$, where $f_\theta$ is the neural network's time complexity function, $dT$ is the number of frames referenced in Equation 4, and $B$ is the number of objects in the video.

The update used in the origin inference algorithm is shown in Equation 11. In order to optimize this problem, we designed a parallel computing strategy (Equation 12). Since there are no dependencies in computation between different frames, our parallel processing significantly improves operational efficiency when processing inferences.

$$x_j^{\tau+1} = \arg\min_{x_j} \hat{E}\left(x_j, \mathbf{y} \mid x_1^{\tau+1}, \ldots, x_{j-1}^{\tau+1}, x_{j+1}^{\tau}, \ldots, x_N^{\tau}\right) \tag{11}$$

$$x_j^{\tau+1} = \arg\min_{x_j} \hat{E}\left(x_j, \mathbf{y} \mid x_1^{\tau}, \ldots, x_{j-1}^{\tau}, x_{j+1}^{\tau}, \ldots, x_N^{\tau}\right) \tag{12}$$

## 4 Experiments

### 4.1 Experimental Setup

In this subsection, we would like to discuss the hyper-parameter settings in our experiments. As mentioned in Section 3.2, the time complexity of the inference could be very high. In order to trade-off accuracy and efficiency, we set the total number of iterations to 3, and the number of iterations for each individual MRF inference to 5. Based on the agreement between the current pixel value and the initial segmentation mask obtained from the OSVOS, we set the probability of a pixel being part of the foreground object(s) is 0.7.

To measure the relative importance of the unary, temporal, and spatial terms, we set the weights as follows: $\theta_u = 1$, $\theta_t = 3$, and $\theta_s = 1.5$. The range of frames is set to 2, which means that the temporal term only considers the previous two frames and the next two frames. And it allows for capturing temporal dependencies while maintaining computational efficiency.

Additionally, we introduce a dynamic coefficient for the spatial term, considering that the similarity between the later frames and the first frame decreases over time. Since the ground truth is only available for the first frame during the fine-tuning of the CNN-based model, the spatial coefficient is set to 0.96 and will be gradually reduced over time. This allows the spatial term to have a stronger influence in the early frames while reducing its impact in later frames as the appearance of the object changes.

For the model that is responsible for the spatial term in MRF, we use the DeepLabV3Plus model with a pre-trained ResNet-34 backbone. The model is trained with the help of the Adam optimizer and a learning rate scheduler (step size = 30 and gamma = 0.9).

During the training process, we used various data augmentation techniques. The data augmentation pipeline was implemented using the PyTorch framework and the Torchvision library. The training images and corresponding masks are processed by a series of transformations, including resizing to a fixed size of 256 * 256 pixels, random horizontal and vertical flipping, random rotation within a range of (-20, 20) degrees, random affine transformations (translation, scaling, and shearing), and random cropping. On the other hand, the validation data is only resized since we want to evaluate the performance of the model on data that is similar to the original inputs. By applying these data augmentation techniques, we aim to improve the model's generalization ability on unseen data.

### 4.2 Results and Discussion

In video object segmentation tasks, two widely used evaluation metrics are the Region Similarity Jaccard ($\mathcal{J}$) and Contour Accuracy $\mathcal{F}$ measure ($\mathcal{F}$). The $\mathcal{J}$ measure, also known as Intersection over Union (IoU), measures the overlap between the predicted segmentation mask and the ground truth segmentation mask. The predicted mask is denoted as $M$, and the ground truth mask is denoted as $G$. The value of $\mathcal{J}$ should be between 0 and 1, with higher values indicating better accuracy, given by the Equation 13:

$$\mathcal{J} = \frac{|M \cap G|}{|M \cup G|} \tag{13}$$

The $\mathcal{J}$ Recall is calculated by counting only the frames where the $\mathcal{J}$ value exceeds a certain threshold (0.5) and computing the arithmetic mean of these frames. The final result is the average of all sequences. The $\mathcal{J}$ Decay is determined by subtracting the $\mathcal{J}$ value of the last frame from that of the first frame (ignoring the ground truth frame for semi-supervised VOS), with the final result being the average across all sequences.

The $\mathcal{F}$ measure focuses on the quality of the contour alignment between the predicted and ground truth masks. It is calculated by evaluating the precision ($P_c$) and recall ($R_c$) of the contour points between the predicted mask $M$ and the ground truth $G$. The F measure, denoted as $\mathcal{F}$, is the harmonic mean of precision and recall, given by the Equation 14:

$$\mathcal{F} = \frac{2P_cR_c}{P_c + R_c} \tag{14}$$

This measure assesses the accuracy of the segmentation boundaries. Like $\mathcal{J}$, the $\mathcal{F}$ measure also includes metrics like $\mathcal{F}$ Mean, $\mathcal{F}$ Recall, and $\mathcal{F}$ Decay, computed following the same principles.

Based on the visualization from Figure 2, our method demonstrates a successful noise reduction capability. However, our method achieves lower performance than the OSVOS based on the result in Table 1. One potential reason for the lower quantitative performance could be the limitation of computational resources, which constrained the input image size to 224 * 224 pixels during the experiments. And the prediction mask will be resized back to 854 * 480 pixels to compare with the ground truth annotations. This resizing operation may have resulted in a loss of fine-grained details. Another factor could be the integration of the simple CNN-based model. While the CNN-based model plays a key role in capturing spatial information and effectively reducing noise in the segmentation output, it may not have performed as well as expected due to using a model with a simple backbone (ResNet-34).

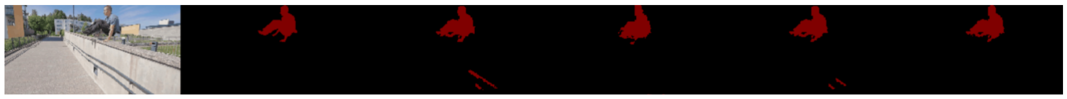

Figure 2: Comparison of segmentation results, from left to right: original image, ground truth, $\hat{E} = E_u$, $\hat{E} = E_t$, $\hat{E} = E_u + E_t$, $\hat{E} = E_u + E_t + E_s$

.

Interestingly, our method shows significantly lower decay values for both the Global $\mathcal{J}$ Decay and Global $\mathcal{F}$ Decay metrics than OSVOS. This indicates that our approach is more stable and consistent over time.

Table 1: Performance Comparison of OSVOS and Our Methods

| | OSVOS | Ours |
|---|---|---|
| Global Mean $\mathcal{J}$ | 0.6543 | 0.5883 |
| Global Mean $\mathcal{F}$ | 0.7143 | 0.6471 |
| Global $\mathcal{J}$ Recall | 0.7355 | 0.6635 |
| Global $\mathcal{F}$ Recall | 0.7826 | 0.7261 |
| Global $\mathcal{J}$ Decay | 0.2973 | -0.0027 |
| Global $\mathcal{F}$ Decay | 0.3155 | -0.0536 |

## 5 Conclusion and Future Work

In this study, we explore a video object segmentation approach that integrates CNNs and MRFs to improve segmentation accuracy in complex video streams. Experiments on the DAVIS 2017 dataset demonstrated the method's effectiveness in reducing noise and achieving consistent segmentation results over time. However, the quantitative performance was lower than the OSVOS. The reason could be computational resource limitations, insufficient hyper-parameter tuning and the use of a simple CNN backbone.

Future work may focus on conducting experiments with higher computational resources and more hyper-parameters tuning, exploring advanced CNN architectures, refining the CNN-MRF integration, and also investigating the approach's performance on other challenging datasets.

## References

[1] Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., & Brox, T. (2017). Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2462-2470).

[2] Ilg, E., et al. (2017). Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

[3] Y. H. Tsai, M. H. Yang, & M. J. Black. (2016). Video segmentation via object flow. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3899-3908).

[4] C. P. Yu, H. Le, G. Zelinsky, & D. Samaras. (2015). Efficient video segmentation using parametric graph partitioning. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 3155-3163).

[5] M. Grundmann, V. Kwatra, M. Han, & I. Essa. (2010, June). Efficient hierarchical graph-based video segmentation. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 2141-2148). IEEE.

[6] W. D. Jang, & C. S. Kim. (2016). Streaming video segmentation via short-term hierarchical segmentation and frame-by-frame Markov random field optimization. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI 14* (pp. 599-615). Springer International Publishing.

[7] F. Perazzi, A. Khoreva, R. Benenson, B. Schiele, & A. Sorkine-Hornung. (2017). Learning video object segmentation from static images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2663-2672).

[8] S. Chandra, C. Couprie, & I. Kokkinos. (2018). Deep spatio-temporal random fields for efficient video segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 8915-8924).

[9] Chen, L.-C., Papandreou, G., Schroff, F., & Adam, H. (2017). Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*.

[10] Caelles, S., Maninis, K. K., Pont-Tuset, J., Leal-Taixé, L., Cremers, D., & Van Gool, L. (2017). One-shot video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 221-230).

[11] Bao, L., Wu, B., & Liu, W. (2018). CNN in MRF: Video object segmentation via inference in a CNN-based higher-order spatio-temporal MRF. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5977-5986).

[12] C. M. Bishop. Pattern recognition and machine learning. springer, 2006.

[13] L. Bao, Q. Yang, and H. Jin. Fast edge-preserving patch-match for large displacement optical flow. In CVPR, 2014.

[14] Y.-G. Jiang, Q. Dai, W. Liu, X. Xue, and C.-W. Ngo. Human action recognition in unconstrained videos by explicit motion modeling. IEEE Trans. on Image Processing, 24(11):3781–3795, 2015.

[15] Y.-G. Jiang, Q. Dai, X. Xue, W. Liu, and C.-W. Ngo. Trajectory-based modeling of human actions with motion reference points. In ECCV, 2012.

[16] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. International Journal of Computer Vision, 81(1):2–23, 2009.

[17] S. Roth and M. J. Black. Fields of experts: A framework for learning image priors. In CVPR, 2005.

[18] D. Koller and N. Friedman. Probabilistic graphical models: principles and techniques. MIT press, 2009.