Attention-guided Temporally Coherent Video Object Matting

Yunke Zhang
Zhejiang University
yunkezhang@zju.edu.cn

Chi Wang
Zhejiang University
wangchi1995@zju.edu.cn

Peiran Ren
Alibaba Group
peiran.rpr@alibaba-inc.com

Miaomiao Cui
Alibaba Group
miaomiao.cmm@alibaba-inc.com

Hujun Bao
Zhejiang University
bao@cad.zju.edu.cn

Xuansong Xie
Alibaba Group
xingtong.xxs@taobao.com

Xian-Sheng Hua
Damo Academy, Alibaba Group
xiansheng.hxs@alibaba-inc.com

Qixing Huang
The University of Texas at Austin
huangqx@cs.utexas.edu

Weiwei Xu
Zhejiang University
xww@cad.zju.edu.cn

ABSTRACT
This paper proposes a novel deep learning-based video object matting method that can achieve temporally coherent matting results. Its key component is an attention-based temporal aggregation module that maximizes image matting networks’ strength for video matting networks. This module computes temporal correlations for pixels adjacent to each other along the time axis in feature space, which is robust against motion noises. We also design a novel loss term to train the attention weights, which drastically boosts the video matting performance. Besides, we show how to effectively solve the trimap generation problem by fine-tuning a state-of-the-art video object segmentation network with a sparse set of user-annotated keyframes. To facilitate video matting and trimap generation networks’ training, we construct a large-scale video matting dataset with 80 training and 28 validation foreground video clips with ground-truth alpha mattes. Experimental results show that our method can generate high-quality alpha mattes for various videos featuring appearance change, occlusion, and fast motion. Our code and dataset can be found at: https://github.com/yunkezhang/TCVOM

CCS CONCEPTS
• Computing methodologies → Image processing; Video segmentation; Neural networks; Supervised learning.

KEYWORDS
datasets, neural networks, video matting, attention mechanism

1 INTRODUCTION
The task of video object matting is to compute temporally coherent alpha mattes for a foreground video object at each frame. It is a fundamental task for many video editing applications, e.g., composing the foreground object into new background videos. The resulting alpha mattes represent the fractional opacity (between 0 and 1) of pixels. Such opacity mainly comes from the transparency or the partial coverage of background pixels around the foreground object boundaries. Specifically, the matting problem tries to solve for three types of unknowns at each pixel, i.e., the foreground color \(F\), the background color \(B\), and the alpha value \(\alpha\), based on the measured colors. Thus, it is critical to build priors to constrain the solution space [1, 11, 15, 32]. State-of-the-art (SOTA) image matting algorithms typically build on convolutional neural network (CNN). They improve the image matting results significantly by learning multi-scale features to predict alpha values for pixels in the unknown region (UR) [7, 10, 12, 24, 39, 54, 61]. Given an input video clip and its corresponding trimap for each frame, one can perform video matting with any image matting method by processing each video frame independently. However, this approach may lead to temporal incoherence in the obtained alpha mattes (e.g., flickering, shown in the third row of Figure 1). To improve temporal coherence, existing video matting methods exploit temporal correspondence between video frames, such as optical flow, to construct multi-frame alpha or color priors or compute temporal affinities to incorporate motion cues [3, 13, 14, 34, 74]. However, they rely on local color distributions as main features and may suffer from motion ambiguities at transparent pixels, resulting in flickering or blocky artifacts in the matting results.
Another challenge is generating trimaps for an input video clip. The central problem of video matting is how to obtain temporally coherent alpha mattes, e.g. foreground image [47], rough segmentation map or coarse alpha matte [67]. Others do not use any prior at all [37, 45, 70]. The most used image matting dataset in this line of research is provided by Xu et al. [61], and a larger dataset is proposed recently by Qiao et al. [45].

Video matting. The central problem of video matting is how to obtain temporally coherent alpha mattes. Chuang et al. [14] proposed to interpolate manually specified trimaps at key-frames using optical flow, estimate the background pixels, and then perform Bayesian matting at each frame with the estimated background. Motion cues and prior distributions for alpha values and multi-frame colors are widely used in video matting [3, 48, 49, 60]. In [13, 17, 31, 34], spatio-temporal edges between pixels are constructed to compute the alpha mattes for video frames simultaneously. These methods are the extension of affinity-based methods to video matting, which is time-consuming due to the Laplacian matrix’s fast-growing size. Zou et al. [74] proposed to select nonlocal neighbors through sparse
Attentive video matting method that focuses on temporal coherency. Yang et al. [36] and Ke et al. [27] proposed real-time CNN-based methods for trimap-less human portrait matting. However, both of the methods do not enforce the temporal consistency between video frames during training, which may lead to temporally incoherent results. Our method, on the other hand, is a trimap-based video matting method that can handle different types of objects with explicitly supervised temporal consistency. Concurrent to our work, Sun et al. [52] also proposed a CNN-based video matting method that focuses on temporal coherence.

### Attention mechanism in segmentation and matting.

Attention mechanism provides an effective way for neural networks to reinforce correlated features and suppress feature noise, leading to a performance boost in segmentation. There are two main variations of this mechanism. One is the channel-wise self-attention pioneered by Hu et al. [25]. Given an input feature tensor, it leverages the global average pooling and the fully-connected layer to infer a channel-wise weight vector to modulate feature maps. The other is the non-local block proposed by Wang et al. [59]. It computes the spatiotemporal correlation as the attention, reinforcing the consistency of feature maps effectively. The channel-wise attention approach is widely adopted in image segmentation [65, 66, 69].

Many methods [20, 63, 64, 68, 71] exploit variants of non-local attention modules to capture the spatial long-range dependency. For image matting tasks, the attention mechanism is mostly used for fusing high and low-level image features. Qiao et al. [45] adopted both channel-wise and spatial attention for trimap-free matting since high-level image features are the key to recognize a foreground object. GCA [35] also utilizes high-level image features as the attention map to guide the low-level alpha features, achieving SOTA performance in image matting. We thus employ GCA as one of the base matting network structures. Several recent VOS methods also utilize the attention mechanism to fuse features from different video frames for improving temporal consistency [41, 57]. Oh et al. [42] extended the memory network approach used in NLP to VOS, which is also a variation of the spatiotemporal attention mechanism. Yang et al. [62] extended this idea by matching both the foreground and background with multi-scale features in those frames, achieving SOTA performance. Our method also leverages the attention mechanism for temporally coherent matting. Nevertheless, our attention module is bi-directional, and we use additional temporal loss terms to supervise the network training.

### Temporal coherence

One standard solution to temporal coherence is the temporal smoothing filter, which considers the spatial and temporal adjacent pixels simultaneously [8, 9, 30, 43]. Another solution is to impose the temporal coherence in the post-processing, which is blind to image filters [5, 29]. In contrast, our method does not rely on temporal smoothing filter but the feature-space affinity to produce temporally coherent alpha mattes.

## 3 OUR METHOD

Given an input video, our method first runs trimap generation to propagate the user-annotated trimaps to the other frames. We then run a video matting network, formed by integrating temporal aggregation module (TAM) into an image-based matting network, to obtain a temporally coherent alpha matte at each frame (See Figure 2). When computing an alpha matte for frame \( I_t \) in testing stage, TAM only needs to aggregate the CNN features from three consecutive frames, i.e. \( I_{t-1}, I_t, I_{t+1} \). The choice of three consecutive frames offers great flexibility in network design while ensures computational efficiency. However, during training, our network takes five consecutive frames simultaneously as inputs, i.e. \( I_{t-2}, ..., I_{t+2} \), and predicts \( \alpha_{t-2}, \alpha_{t}, \alpha_{t+1} \) to facilitate the computation of loss functions. Note that we choose to integrate TAM into the base network at the decoder stage of output stride (OS) 8. It indicates that the resolution of the feature map should be \( H/8 \times W/8 \), where \( H, W \) is the input image resolution. This choice is to balance computational cost and feature level, and we empirically found that OS=8 is a good trade-off (see the supplementary material for the OS experiment).

In the following, we will first describe the design of TAM (Sec. 3.1), and proceed to describe the training loss (Sec. 3.2) and the training strategy of TAM (Sec. 3.3). Finally, we describe the details of trimap generation using STM [42] (Sec. 3.4) and our video object matting dataset (Sec. 3.5).

### 3.1 Temporal Aggregation Module

Figure 3 illustrates the structure of TAM. It leverages the attention mechanism to aggregate features from \( I_{t-1} \) and \( I_{t+1} \) with features...
Finally, we formulate feature aggregation for \( I' \) as follows:

\[
\hat{F}'(i) = \begin{cases} 
S(i) + \hat{U}^{t-1}(i) + \hat{U}^{t+1}(i), & i \in \text{UR} \\
S(i), & \text{Otherwise}
\end{cases}
\]

where the modulated feature \( \hat{U}^{t-1} \in \mathbb{R}^{M \times C} \) is a weighted sum of all features inside the local patch using \( A^{t-1} \), and \( S \) indicates the features output by a 3×3 convolution layer at frame \( I' \). The features \( \hat{F}' \) are fed into next convolution layer to continue the workflow of the original network.

Note that the TAM’s design can be seen as sharing the weight between the “query” and “value” convolution in the conventional “KQV” structure. We choose this design because it achieves the best performance among other weight sharing configurations. The influence of the weight sharing will be investigated later in Sec. 4’s ablation study.

### 3.2 Loss function

The overall loss function is a composition of three different terms: the image matting term \( L_{im} \), the temporal coherence term \( L_{tc} \), and the target affinity term \( L_{af} \). In the following, we describe these three terms in details.

**Image matting term.** The image matting term \( L_{im} \) is directly inherited from the deep learning-based image matting methods. This term only considers the single frame prediction results w.r.t. its corresponding ground truth. Common choices of this term are the combination of \( L_1 \) alpha matte loss, \( L_4 \) alpha gradient loss, composition loss [61] and Laplacian loss [64]. In all our experiments, we set \( L_{im} \) the same as the image matting method we chose to use as the base network. Hence, for DIM [61] and Index [38], we use alpha matte loss, alpha gradient loss, and composition loss; for GCA [35], we only use alpha matte loss.

**Temporal coherence term.** We leverage the temporal coherence metric proposed by Efroeev et al. [18] as the temporal coherence term \( L_{tc} \), which can be expressed as follows:

\[
L_{tc}^{pq}(i) = \left| \alpha(i)_{t-1} - \alpha(i)_t \right| - \left| \alpha(i)_t - \alpha(i)_{t+1} \right|_1, \tag{3}
\]

where \( \alpha(i) \) and \( \bar{\alpha}(i) \) are the predicted and ground-truth alpha matte. \( i \) denotes a pixel in the UR of frame \( p \). This term is denoted by “dtSSD” in [18], which penalizes the temporal gradient of alpha mattes in our work. There is another temporal coherence metric, termed as “MESSDdt” in [18], which augments Eq. 3 with motion vectors for the purpose of better correspondence between frame \( t-1 \) and frame \( t \). However, we found that even computed with SOTA optical flow algorithms [55], there are severe motion noises for translucent pixels, which hurts the performance of the network when using “MESSDdt” as the temporal coherence loss. Thus, we do not use “MESSDdt” in our implementation. The quantitative result of using “MESSDdt” as a loss term can be found in the supplementary material.

**Target affinity term.** While the attention mechanism works well in many cases, Yu et al. [64] demonstrates that providing direct supervision for the attention weights can further boost the performance. Inspired by this work, we thus design the target affinity term \( L_{af} \). For a pixel \( i \) in \( I' \), the target probability of having a
large attention weight between \(i\) and a pixel \(j \in \mathcal{W}\) inside its local patch at a neighboring frame \(f\) can be formulated as:

\[
G_f(i, j) = \begin{cases} 
1 - s, & |\hat{a}_f - \hat{a}_j| < \theta \\
0, & \text{Otherwise}
\end{cases}, \quad (4)
\]

where \(f = t - 1\) or \(t + 1\), and \(\theta\) is set to be 0.3. In addition, we follow the label smoothing technique [53] to avoid over-confident decision by introducing the parameter \(s = 0.2\). The target probability is computed according to the ground-truth alpha mattes. The goal of this loss term is to make the network learn to assign small affinity values to pixels with large alpha value differences. Therefore, we model the target affinity term between pixel \(i\) and \(j\) as

\[
L_{af}^f(i, j) = \text{BCE}(\Phi(K(i) \cdot U_f(i, j)), G_f(i, j)) \quad (5)
\]

where BCE denotes the binary cross-entropy function, and \(\Phi\) denotes the sigmoid function. During training, the term \(L_{af}\) is calculated as

\[
L_{af} = \frac{1}{t}(L_{af}^{t-1} + L_{af}^{t+1}).
\]

In summary, our network is trained by the weighted average of these three terms:

\[
L = w_{lm}L_{lm} + w_{tc}L_{tc} + w_{af}L_{af}. \quad (6)
\]

where we set \(w_{lm} = 1\), \(w_{tc} = 0.5\), and \(w_{af} = 0.25\) in our experiments.

### 3.3 Training strategy

The training of our video matting network consists a pre-training stage and a main stage. In the following, we denote the video matting network as GCA+TAM, DIM+TAM or Index+TAM which corresponds to base methods GCA [35], DIM [61] and Index [38] respectively. For pre-training, we input three frames \(I^{t-1}, I^t, I^{t+1}\) along with trimaps and predict the center frame alpha matte \(\alpha'\) using the supervision from \(L_{lm}\) only. During pre-training, all layers before the TAM in the network are initialized and fixed using the pre-trained weight of a off-the-shelf image matting network. TAM and the rest of the decoder layers are randomly initialized and trained on the DIM dataset [61]. The dataset is augmented with random affine transformations (rotation, translation, and scaling) to generate sequences of three frames. Random flipping and cropping are also conducted for further augmentation. We pre-trained the network for 20 epochs using an input resolution of \(512 \times 512\) with the Adam optimizer [28]. For different base image matting methods, we used different batch sizes and learning rates. The batch size is set to 24, 16 and 24 respectively.

During main training, we use different batch sizes and learning rates. The batch size is set to 24, 16 and 24 respectively.

### 3.4 Video Trimap Generation

We leverage the SOTA VOS network STM [42] to segment each frame into FR, BR, and UR. That is, we let STM track FR and UR as two different objects in a video and classify the remaining pixels that do not belong to FR and UR as BR. To obtain the ground-truth labels, we label the translucent pixels obtained in the construction of VideoMatting108 without any dilation as UR, and the pixels with alpha value equals one as FR. Additionally, we give UR a higher weight (4.0 in training) to achieve class-balanced cross-entropy loss since UR generally has fewer pixels than FR and BR.

**Training parameters.** Same with STM, we also utilize the two-stage training strategy. First, the network is initialized from the weight pre-trained on ImageNet [16]. We then use the DIM dataset [61] augmented with random affine transformations (rotation, translation, scaling, and shearing) to pre-train the network. The network is pre-trained for 25 epochs. We then proceed to the main training stage. The only difference is that we use a larger maximum frame skip, which is 75 frames in our implementation since the videos in VideoMatting108 are much longer compared to the VOS dataset like DAVIS [44]. We train the network for 150 epochs, where every epoch consists of 850 iterations. The maximum frame skipping is gradually increased by one every two epochs. We also utilize the “stair” learning rate strategy with the base learning rate of \(10^{-5}, 5 \times 10^{-6}, 10^{-6}\) and \(5 \times 10^{-7}\) at 40, 80 and 120 epochs, respectively. We use the batch size of 4, input resolution of \(512 \times 512\), and Adam optimizer for all of our experiments.

**Inference strategy.** When generating trimaps for a new video that is not present in the training and validation sets, we found that online fine-tuning with the user-annotated trimaps at keyframes drastically improves the performance of the network in our case. During online fine-tuning, we treat the user-annotated trimap as the ground truth and use the same random affine transform technique to generate “fake” video sequences. Subsequently, we fine-tune the network on these sequences for 500-800 iterations with a constant learning rate of \(10^{-6}\). When there is more than one frame of user-defined trimaps, a bidirectional inference strategy is used to ensemble the prediction results. Please refer to our supplementary material for more details.

### 3.5 A New Video Matting Dataset

Lacking training data is a massive barrier to deep learning-based video matting methods. For instance, the most commonly used video matting dataset from videomatting.com [18] has only ten test clips and three clips with ground-truth mattes, which is not enough for network training. To this end, we propose our video matting dataset, VideoMatting108. We rely on green screen video footage to extract ground-truth alpha mattes, which is not enough for network training. To this end, we propose our video matting dataset, VideoMatting108. We rely on green screen video footage to extract ground-truth alpha mattes, which is not enough for network training. To this end, we propose our video matting dataset, VideoMatting108. We rely on green screen video footage to extract ground-truth alpha mattes, which is not enough for network training. To this end, we propose our video matting dataset, VideoMatting108. We rely on green screen video footage to extract ground-truth alpha mattes, which is not enough for network training. To this end, we propose our video matting dataset, VideoMatting108.
In total, our dataset consists of 108 video clips, all in 1080p resolution. The average length of the video clip is 821 frames, significantly longer than other datasets. The foreground objects cover a wide range of categories, such as human, fluffy toys, cloth (net, lace, and chiffon), smoke and plants. The background footage usually has 3D camera motion or complex scenery, adding more challenge to our dataset. We split the dataset with 80 clips in the training set and 28 clips in the validation set. Trimaps are generated by dilating transparent pixels with three different kernel sizes: 11 × 11 for narrow trimaps, 25 × 25 for medium trimaps, and 41 × 41 for wide trimaps.

4 EXPERIMENTAL RESULTS

In this section, we present the evaluation of our approach on the VideoMatting108 dataset. We also evaluate our trimap generation algorithm. Our computing platform for video matting related experiments was a 4 V100 GPU server with 2 Intel 6148 CPUs. Trimap generation related experiments were conducted on a 4 1080Ti GPU server with 2 E5-2678v3 CPUs.

Evaluation metrics. We employ SSDA (average sum of squared difference) and two temporal coherence metrics, namely diSSD (mean squared difference of direct temporal gradients) and MEStDdt (mean squared difference between the warped temporal gradient) from [18] to evaluate the accuracy of the predicted video alpha mattes and their temporal coherence. Besides, we also report “MSE” (mean squared error) and “SAD” (sum of absolute difference) to verify the pixel-wise accuracy of alpha values at each frame. Lower evaluation metrics correspond to better video matting results.

Table 2: Comparison between GCA+TAM and GCA [35] on the 10 test clips from videomatting.com [18] with different trimaps. The best result is in bold. Please see our supplementary material for quantitative results of each test video clip.

| Method | Trimap | SSDA | dtSSD | MEStDdt |
|--------|--------|------|-------|---------|
| GCA+F  | Narrow | 39.40| 30.83 | 1.43    |
| GCA+TAM| Narrow | 36.95| 26.37 | 1.12    |
| GCA+F  | Medium | 44.74| 33.42 | 1.74    |
| GCA+TAM| Medium | 42.17| 28.81 | 1.35    |
| GCA+F  | Wide   | 50.45| 36.71 | 2.14    |
| GCA+TAM| Wide   | 49.23| 32.94 | 1.76    |

Quantitative comparisons. Table 1 shows the quantitative comparisons between our video matting networks and single image matting networks on the VideoMatting108 validation set. For fair comparisons, we fine-tune single image matting networks on VideoMatting108 using each video frame as the image matting training data. The learning rate and input resolution are kept the same as we train our video matting network. The results are averaged over all 28 test video clips. It can be seen that our method (denoted “+TAM”) with “\(I_{im} + I_{tc} + I_{af}\)” in the table consistently outperform the baseline image matting networks (denoted “GCA+F”, “Index+F” and “DIM+F”) on all metrics. More comparison results with other methods, such as KNN Video Matting [34] and DVM [52], can be found in our supplementary material.

Furthermore, in Table 2, the GCA+TAM network also outperforms GCA on the test dataset from videomatting.com [18]. This verifies the ability of the proposed TAM that is designed to aggregate temporal information for better video object matting results. Since GCA+TAM achieves much lower metric numbers comparing to DIM+TAM and Index+TAM, we use GCA+TAM as the default for evaluating video matting, if not mentioned otherwise.

Ablation studies. We first investigate the influence of the weight sharing in TAM (see the second to the fourth row in Table 1). Different from the widely utilized “KQV” structure without any weight sharing, in our case we empirically found out that sharing “query” and “value” weights achieves the best result compared with other weight sharing configurations. Note that the conventional “KQV” structure without weight sharing performs worse than the baseline method without TAM on the validation set. We speculate that separating all weights causes over-fitting during the training, since it does achieve lower training loss compared with our design.

Table 1: Result on VideoMatting108 validation set. GCA [35], Index [38] and DIM [61] are used as the base image matting network structures to verify the effectiveness of the TAM. The best result is in bold, the second best is underlined. “+F” indicates the single image matting method is fine-tuned on our training dataset. “+TAM” denotes we add TAM for video matting. “+TAMshare” and “+TAMsep” denote we share / separate all convolutions in TAM, respectively. “MSEdt” denotes “MEStDdt”.

| Method | Loss | Narrow | Medium | Wide |
|--------|------|--------|--------|------|
|        | SSDA | dtSSD  | MSE    | SAD  |
| GCA+F  | 56.54| 38.01  | 17.31  | 1.56 |
| +TAM   | 57.91| 30.75  | 20.18  | 1.48 |
| +TAMshare| 58.53| 35.03  | 21.85  | 1.96 |
| +TAMsep| 59.13| 35.75  | 21.56  | 2.18 |
| +TM    | 59.74| 36.71  | 22.11  | 2.38 |
| +TAMshare| 60.45| 37.56  | 22.85  | 2.50 |
| +TAMsep| 61.16| 38.21  | 23.36  | 2.62 |
| +TM    | 61.87| 39.01  | 23.91  | 2.73 |
| +TAMshare| 62.58| 39.71  | 24.46  | 2.85 |
| +TAMsep| 63.29| 40.41  | 25.01  | 2.97 |

Table 3: Ablation study on the temporal window size \(W\) and the number of aggregated frames used for GCA+TAM. “nF” denotes \(n\)-frames are aggregated in TAM. 2F: only uses \(I_{im}^{1-1}\); 3F: uses \(I_{im}^{1-1}\) and \(I_{im}^{2}\); 5F: uses \(I_{im}^{1-1}\), \(I_{im}^{2}\), \(I_{im}^{3}\) and \(I_{im}^{4}\).

| Method | Trimap | SSDA | dtSSD | MEStDdt |
|--------|--------|------|-------|---------|
| GCA+F  | Narrow | 39.40| 30.83 | 1.43    |
| GCA+TAM| Narrow | 36.95| 26.37 | 1.12    |
| GCA+F  | Medium | 44.74| 33.42 | 1.74    |
| GCA+TAM| Medium | 42.17| 28.81 | 1.35    |
| GCA+F  | Wide   | 50.45| 36.71 | 2.14    |
| GCA+TAM| Wide   | 49.23| 32.94 | 1.76    |

For evaluating video matting, if not mentioned otherwise.

Ablation studies. We first investigate the influence of the weight sharing in TAM (see the second to the fourth row in Table 1). Different from the widely utilized “KQV” structure without any weight sharing, in our case we empirically found out that sharing “query” and “value” weights achieves the best result compared with other weight sharing configurations. Note that the conventional “KQV” structure without weight sharing performs worse than the baseline method without TAM on the validation set. We speculate that separating all weights causes over-fitting during the training, since it does achieve lower training loss compared with our design.
We proceed to assess the influence of different loss terms (see the fifth to the seventh row in Table 1). By only adding the temporal coherence term $L_{tc}$, we obtain performance boost across all metrics, except pixel-wise alpha value accuracy in “narrow” trimaps. Since the UR of a narrow trimap mainly consists of transparent pixels, we speculate that the motion ambiguities at these pixels are the main reason for the drop of alpha value accuracy. By only adding the target affinity term $L_{af}$, the temporal metrics “dtSSD” and “MESSDdt” are slightly improved while the alpha value accuracy metrics are comparable to the baseline. By combining both terms, the network achieved much improved results. For the “narrow” trimaps in particular, the direct supervision in $L_{af}$ could suppress the erroneous affinity values, thus improving performance.
Table 4: Ablation study for trimap generation on VideoMatting108 validation set using the mIoU metric. All metrics are averaged with a per-video basis. “nFT” denotes how many keyframe trimaps are used in fine-tuning. 1FT: first frame as keyframe. 2FT: first+last frame as keyframes. 3FT: first+last+100-th frame as keyframes.

| Method     | FR  | BR  | UR  | Average |
|------------|-----|-----|-----|---------|
| STM [42]   | 81.43 | 95.58 | 85.04 | 90.23   |
| STM+1FT    | 85.92 | 96.62 | 82.75 | 88.43   |
| STM+2FT    | 87.73 | 97.91 | 84.90 | 90.18   |
| STM+3FT    | 87.72 | 97.93 | 85.04 | 90.23   |

In Table 3, we analyze the influence of the temporal window size (Sec. 3.1) parameter $W$ and the number of frames used in TAM. To reduce the computational cost, we conduct experiments on half of the VideoMatting108 training and validation set with “medium” ground-truth trimaps. We can see that the network performance achieves the best balance when $W = 7$. In contrast, the network performance degrades when $W = 9$. The reason may come from the difficulties of suppressing a large number of unrelated features from adjacent frames through attention. Thus, we choose $W = 7$ in all of our experiments. To verify the bi-directional design, we conduct experiments on the number of aggregated frames (fourth and fifth rows). As seen in Table 3, our bi-directional design (second row, 3F) outperforms all other configurations.

**STM-based trimap generation.** We use the medium trimaps from VideoMatting108 as the ground truth to evaluate the performance of STM [42] on trimap generation using the mIoU metric. As the video sequences in VideoMatting108 are long, we only use the first 200 frames in this experiment. Specifically, we choose the first, the last, and the 100-th frame as keyframes during online fine-tuning. We also gradually add their corresponding ground-truth trimaps to show their impact on online fine-tuning. The input resolution is set to 768 × 768, and we fine-tune the network for 500 iterations. The average time for fine-tuning STM is 7.5 minutes on one GPU. As shown in Table 4, online fine-tuning with the first or other frames improved the result by a large margin, especially in FR. Adding more ground-truth trimaps improves the results further. In conclusion, fine-tuning is necessary to adapt the network to user-specified keyframe trimaps, which improve the quality of generated trimaps. Table 4 also shows that adding more keyframes improves the mIoU score marginally, which indicates that a sparse set of annotated keyframe trimaps is enough for video trimap generation. We use around three frames in all our experiments. On the other hand, inaccurate FR/BR heavily affects the matting result. As shown in the third row of Figure 6(a), the pixels inside the gap between the two legs have erroneous high alpha values, despite the UR being roughly the same in the trimaps. In (b) and (c), the excessive URs lead to artifacts in the final alpha matte.

**Qualitative evaluations.** In Figure 1 and Figure 4, we show that our method could improve the temporal coherence comparing to single image matting networks. In the “plant” clip, GCA [35] produces flickering in the alpha matte although the foreground object is nearly static. As seen in the “lion” clip, GCA produces an erroneous white blob between the fur in the blowup. In contrast, our method could mitigate this discontinuity by aggregating temporal features. The same effect can be seen in the “dancing woman” clip and “standing man” clip when using Index [38] and DIM [61] as the base network, respectively. All examples validate the effectiveness of our TAM. In Figure 5, we qualitatively compare GCA+TAM with GCA on three Internet video clips. The trimaps are generated by the fine-tuned STM [42]. In the “fashion” clip, our method can alleviate the artifacts in the gap between the model’s arm and body. In the “xmas-dog” and the “sofa” clip, not only our method can produce more detailed result (#9 of “xmas-dog” and #92 of “sofa”), it is also more robust to inaccurate URs in the trimap (#22 of “xmas-dog” and #23 of “sofa”). The lengths of these three clips are 89, 100 and 93 frames respectively. More results can be found in the supplementary materials.

5 CONCLUSION

We have developed a deep video object matting method to achieve temporally coherent video matting results. Its key feature is an attention-based temporal aggregation module to compute the temporal affinity values in feature space, which are robust to appearance changes, fast motions, and occlusions. The temporal aggregation module can be easily integrated into image matting networks to enhance video object matting performance. We constructed a video matting dataset to enable the training of video object matting and trimap generation networks. This dataset has 80 training and 28 validation foreground video sequences with ground truth alpha mattes. In the future, we plan to investigate weakly supervised video object matting methods to reduce the workload of creating high-quality video matting training data.

ACKNOWLEDGMENTS

We would like to thank anonymous reviewers for their constructive comments. Weiwai Xu is partially supported by National Key R&D Program of China (2017YFB1002600) and NSFC (No. 61732016). Qixing Huang would like to acknowledge the support from NSFIIS-2047677 and NSF HDR TRIPODS-1934932.
Paul Voigtlaender, Yuning Chai, Florian Schroff, Hartwig Adam, Bastian Leibe, and Liang-Chieh Chen. 2019. FEELVOS: Fast End-To-End Embedding Learning for Video Object Segmentation. In IEEE Conf. Comput. Vis. Pattern Recog.

Jue Wang, Michael F Cohen, et al. 2008. Image and video matting: a survey. Foundations and Trends® in Computer Graphics and Vision 3, 2 (2008), 97–175.

Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. 2018. Non-local neural networks. In IEEE Conf. Comput. Vis. Pattern Recog. 7794–7803.

Jiangjian Xiao and Mubarak Shah. 2005. Accurate motion layer segmentation and matting. In CVPR, Vol. 2. 698–703.

Ning Xu, Brian L Price, Scott Cohen, and Thomas S Huang. 2017. Deep Image Matting. In IEEE Conf. Comput. Vis. Pattern Recog., Vol. 2. 4.

Zongxin Yang, Yunchao Wei, and Yi Yang. 2020. Collaborative video object segmentation by foreground-background integration. In Eur. Conf. Comput. Vis.

Changqian Yu, Yifan Liu, Changxin Gao, Chunhua Shen, and Nong Sang. 2020. Representative Graph Neural Network. In Eur. Conf. Comput. Vis.

Changqian Yu, Jingbo Wang, Changxin Gao, Gang Yu, Chunhua Shen, and Nong Sang. 2020. Context Prior for Scene Segmentation. In IEEE Conf. Comput. Vis. Pattern Recog. 12416–12425.

Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. 2018. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In Eur. Conf. Comput. Vis. 325–341.

Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. 2018. Learning a discriminative feature network for semantic segmentation. In IEEE Conf. Comput. Vis. Pattern Recog. 1857–1866.

Qihang Yu, Jianming Zhang, He Zhang, Yilin Wang, Zhe Lin, Ning Xu, Yutong Bai, and Alan Yuille. 2020. Mask Guided Matting via Progressive Refinement Network. arXiv:2012.06722 [cs.CV]

Yuhui Yuan, Xilin Chen, and Jingdong Wang. 2020. Object-Contextual Representations for Semantic Segmentation. In Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part VI (Lecture Notes in Computer Science, Vol. 12351). Springer, 173–190. https://doi.org/10.1007/978-3-030-58539-6_11

Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Anirsh Tyagi, and Amit Agrawal. 2018. Context encoding for semantic segmentation. In IEEE Conf. Comput. Vis. Pattern Recog. 7151–7160.

Yunke Zhang, Lixue Gong, Lubin Fan, Peiran Ren, Qixing Huang, Hujun Bao, and Weisi Xu. 2019. A Late Fusion CNN for Digital Matting. In IEEE Conf. Comput. Vis. Pattern Recog.

Hengshuang Zhao, Yi Zhang, Shu Liu, Jianping Shi, Chen Change Loy, Dahua Lin, and Jiajia Jia. 2018. Psanet: Point-wise spatial attention network for scene parsing. In Eur. Conf. Comput. Vis. 267–283.

Ferfen Zhou, Yingjie Tian, and Zhiqian Qi. 2020. Attention Transfer Network for Nature Image Matting. IEEE Transactions on Circuits and Systems for Video Technology (2020).

Jieie Zhu, Miao Liao, Ruigang Yang, and Zhigeng Pan. 2009. Joint depth and alpha matte optimization via fusion of stereo and time-of-flight sensor. In CVPR. 453–460.

Dongqing Zou, Xiaowu Chen, Guangying Cao, and Xiaogang Wang. 2020. Unsupervised Video Matting via Sparse and Low-Rank Representation. PAMI 42, 6 (2020), 1501–1514.