End-to-end Video Matting with Trimap Propagation

Wei-Lun Huang
National Taiwan University
r09944040@csie.ntu.edu.tw

Ming-Sui Lee
National Taiwan University
mslee@csie.ntu.edu.tw

Abstract

The research of video matting mainly focuses on temporal coherence and has gained significant improvement via neural networks. However, matting usually relies on user-annotated trimaps to estimate alpha values, which is a labor-intensive issue. Although recent studies exploit video object segmentation methods to propagate the given trimaps, they suffer inconsistent results. Here we present a more robust and faster end-to-end video matting model equipped with trimap propagation called FTP-VM (Fast Trimap Propagation - Video Matting). The FTP-VM combines trimap propagation and video matting in one model, where the additional backbone in memory matching is replaced with the proposed lightweight trimap fusion module. The segmentation consistency loss is adopted from automotive segmentation to fit trimap segmentation with the collaboration of RNN (Recurrent Neural Network) to improve the temporal coherence. The experimental results demonstrate that the FTP-VM performs competitively both in composited and real videos only with few given trimaps. The efficiency is eight times higher than the state-of-the-art methods, which confirms its robustness and applicability in real-time scenarios. The code is available at https://github.com/csvt32745/FTP-VM.

1. Introduction

Image matting aims to estimate the alpha value of each pixel for a target in the input image. Unlike general segmentation, which generates binary values, matting outputs values between 0 and 1, meaning the degree of transparency. The output alpha mattes can describe semi-transparent objects with precise details. As shown in Eq. (1), each pixel color $C$ of an image is composited by the foreground color $F$, background color $B$, and an alpha value $\alpha$, where the background can be substituted to generate a matting dataset.

$$C = \alpha F + (1 - \alpha)B$$  (1)

Given a frame like Fig. 1a, a trimap Fig. 1b is the common requirement for image matting, which divides the pixels into three regions: foreground, background and unknown. Matting methods adopt such information to solve alpha values of unknown (gray) regions.

Video matting extracts an alpha matte of each frame of the given video. The resulting alpha mattes can be used for background replacement, which is decisive for video applications such as video conferencing and visual effects. It is intuitive to perform image matting on each frame of a video. However, severe flickering artifacts would occur in the resultant image sequence. In order to improve the robustness, considering spatial and temporal coherence is the main challenge for video matting as the temporal information helps infer the matting target from the previous frames. Another challenge is to provide a trimap for each frame, which is expected to be a massive cost to most users.

To tackle the above two issues, automatic matting and trimap propagation are addressed in this paper. Automatic matting captures specific targets without trimaps, but
the input videos with ambiguous scenarios highly affect the results. Fig. 1 shows an example of an in-street interview video where an interviewee and passengers occasionally appear simultaneously. While the human matting method [32] attempts to capture all the people, the ambiguity caused by inconspicuous passengers results in unsatisfactory output Fig. 1d. Thus we opt for trimap propagation to select targets more stably as shown in Fig. 1f.

While performing trimap propagation, the user is required to provide a pair of so-called memory frame Fig. 1a and memory trimap Fig. 1b, and this information is utilized to propagate throughout the video. Since we are predicting a sequence of three-class segmentation masks, the trimap propagation can be treated as video object segmentation. Recent studies [42, 47, 55] leverage STM (Space-Time Memory network), [35] an emerging video object segmentation model, to produce the trimaps successfully. However, the resultant trimaps usually contain flickers and lead to unsatisfactory results. As STM lacks temporal coherence, we append ConvGRUs [2] to the model to improve stability. Moreover, the previous approaches containing two full models make them unsuitable for interactive applications. We thus combine the two models into an end-to-end model to enhance the speed and performance. The main contributions of this work are summarized as follows.

• A novel end-to-end video matting model equipped with trimap propagation, called FTP-VM (Fast Trimap Propagation - Video Matting), is proposed. FTP-VM is faster than the previous two-model methods by a large margin while preserving competitive performance in different scenarios. While the frame rate of the previous methods is 5 FPS, the proposed method reaches 40 FPS on an NVIDIA RTX 2080Ti GPU.

• A lightweight trimap fusion module is designed to replace an additional encoder in the STM-based model to make FTP-VM efficient and more powerful.

• Motivated by [38], the segmentation consistency loss from automotive segmentation is adapted to trimap segmentation. The final setting reaching the more satisfactory performance is determined by conducting comprehensive experiments.

2. Related Work

2.1. Image Matting

Non-deep-learning image matting includes two types of methods: The sampling-based approaches [12,16,18,44,48] construct the foreground and background priors, and infer the alpha values by solving the composition equation. The affinity-based methods [4, 5, 17, 24–26, 46] propagate the alpha values from known pixels to unknown ones according to similarities under the assumption of local smoothness. These classical methods rely on low-level representations and often require assistance to deal with complex scenes. For deep-learning methods, CNN (Convolution Neural Network) based methods [9, 15, 20, 23, 30, 34, 51] exploit high-level features and achieve state-of-the-art results. Vision-transformer-based methods [36] are emerging by combining global similarity and inductive bias of images. These methods build prior from the extracted features or the similarity to estimate the alpha values and preserve details by the UNet-like [40] architecture.

In order to annihilate the need for trimaps, automatic image matting combines segmentation and matting into one task. It aims to matte specific targets such as humans [21, 45] or salient objects [3, 28, 29, 37] without any auxiliary inputs. Mask-guided matting [54] takes a rough segmentation mask as an additional input followed by refinement. However, it cannot process semi-transparent objects due to the binary masks. Background-based methods [31, 41] require a background image instead of a trimap as an auxiliary input so that it needs to handle misaligned backgrounds better.

2.2. Video Matting

Temporal coherence plays an essential role in video matting. Most existing methods [1, 10, 27, 43] extend classical image matting methods to the temporal dimension and encounter a similar weakness. RVM [32] places ConvGRU hierarchically to maintain temporal coherency. It performs in real-time due to the efficiency of ConvGRU. DVM [47] uses deformable convolution to warp the feature temporally by estimated offsets but is limited by the kernel size of convolutions. TCVOM [55] leverages guided attention block to compute affinity between frames, and [49] analyzes inter-frame relationships by graph neural network. [21] reduces the flickers by applying post-processing tricks to the results, but the image matting methods still limit the performance.

For trimap propagation, [11] interpolates trimaps by optical flows and contains restrictions caused by weak low-level features from semi-transparent objects. Recent deep-learning-based methods [47, 55] leverage approaches from video object segmentation to perform trimap segmentation before video matting. Decoupling trimap propagation and video matting simplifies the problem and forms a two-stage method. However, the model sometimes needs to be finetuned based on the given trimaps to generate acceptable results. OTVM [42] further improves performance through multi-stage training and joint trimap and alpha prediction modeling. The joint modeling utilizes alpha mattes and trimaps to perform trimap propagation.
3. Method

The overall network architecture containing an encoder, trimap fusion module, bottleneck fusion module, and two corresponding decoders is illustrated in Fig. 2. The encoder extracts features from the memory frame \( F_m \) and query frame \( F_q \). The trimap fusion module encodes the memory trimap \( T_m \) and memory features into memory values. The bottleneck fusion module propagates the trimap information to the query frame and aggregates the features globally. The decoders contain segmentation and matting decoders to estimate the trimap and boundary matte, respectively. Finally, the resulting trimap and boundary matte are integrated into a complete matte.

3.1. Encoder

The MobileNetV3-Large [22] is adopted as the encoder for its efficiency. The encoder takes a single image as input and extracts features at strides of 2, 4, 8 and 16, respectively. The receptive fields are larger in deeper stages, which makes features at different strides focus on different tasks. The low-level features are for preserving details, while high-level ones are for inferring the matting target. This kind of hierarchy can be employed as an UNet-like structure [40] where it receives skip-connections of the features to avoid information diminishing due to the deeper networks. \( F_m \) and \( F_q \) are passed through the same encoder to reduce the size of the model to enhance efficiency. The extracted memory and query features are denoted as \( F_m^s \) and \( F_q^s \) where \( s \) represents the stride of the feature.

3.2. Trimap Fusion Module

Fig. 3 shows the architecture of trimap fusion module. It integrates the extracted memory features \( F_m^s \) and the memory trimap \( T_m \) into the memory value \( F_m^{s/2} \), which will be used to perform memory matching in the bottleneck fusion module. In general video object segmentation methods, the memory value is generated through an additional encoder (e.g., ResNet18 [19]) with a 4-channel input. Introducing another encoder leads to more parameters and a longer training time. Thus we construct a lightweight module to reduce the computation cost and parameters. Illuminated by DeepFillV2 [53], the gated convolutions are utilized instead of vanilla convolutions [23]. DeepFillV2 performs inpainting and requires a multiple-channel mask as an additional input. A gated convolution is composed of two vanilla convolutions, where one extracts features, and the other predicts scales followed by a sigmoid function. The results are then combined by an element-wise multiplication. The gated convolution makes different channels concentrate on different regions and enhances the performance of vanilla convolutions. The downsampled trimaps \( T_m \), memory features \( F_m^s \) and output features \( F_m^{s/2} \) from the previous level are concatenated and passed through a gated convolution block \( g^* \):

\[
F_m^{s/2} = \begin{cases} 
  g^*([T_m^s, F_m^s]) & \text{if } s = 2 \\
  g^*([T_m^s, F_m^s, F_m^{s/2}]) & \text{if } s = 4, 8, 16
\end{cases}
\]  

where \( s \) represents the stride and \([\cdot, \cdot]\) denotes channel-wise concatenation. \( F_m^{s/2} \) is the final output memory value \( F_m^s \).

3.3. Bottleneck Fusion Module

The bottleneck fusion module consists of a memory matching module, a convolution block attention module (CBAM) [50] and a pyramid pooling module (PPM) [56]. Memory matching module applies the same mechanism of cross-attention as STCN [8]. The query feature \( F_q^s \) and memory feature \( F_m^{s/2} \) are first projected to the query key \( K_q \in \mathbb{R}^{ch_k \times hw} \) and memory key \( K_m \in \mathbb{R}^{ch_k \times hw} \) through a shared convolution. \( h \) and \( w \) are 1/16 of the original resolution, and the channel of keys \( ch_k \) is 32. The memory values \( F_m^{s/2} \) from the trimap fusion module are linearly combined based on the similarity score between \( K_q \) and \( K_m \) to generate the matched memory values \( F_m^{s/2} \). In addition, the CBAM [50] and PPM [56] are leveraged to integrate the features competently. CBAM scales the features spatially and channel-wisely to strengthen different features from \( F_q^{s/2} \) and \( F_m^{s/2} \). PPM aggregates the global context from sub-regions of different scales by GAPs (Global Average Pooling), where the scales are set to be 1, 2, 4, and 8, respectively.
3.4. Decoders

As GFM [28] claims that decoding semantic information and edge details simultaneously may harm the matting quality, the decoding part is split into two decoders, trimap segmentation and matting. For frame-to-frame propagation, ConvGRU [2] is efficient but lacks long-term context and probably loses track after occlusions. Thus we apply it with memory matching in the bottleneck fusion module to maintain both short-term and long-term temporal coherence. These decoders are designed differently to boost performance and efficiency as PSPM [13]. They are both composed of bilinear upsampling, convolution blocks and ConvGRUs. Each ConvGRU is placed after a convolution block and is only employed for half of the channels of the input feature maps.

The trimap segmentation decoder estimates the probability maps with three classes and they are converted into a trimap by selecting the class with the highest probability for each pixel. Since the trimap usually retains fewer details, it is output at a stride of 4 and bilinearly upsampled to the original resolution for speed and stability. ConvGRUs are placed at the stride of 8 and 16. For the matting decoder, it estimates precise mattes of the foreground in the boundary with high resolution. Moreover, it is expected to focus on details, so it starts with features at the stride of 4. \( F_{T}^{16} \) is upsampled to the stride of 4 and concatenated with \( F_{T}^{4} \), where \( F_{T}^{s} \) denotes the feature obtained from the trimap decoder with a stride of \( s \). Then it is upsampled and receives skip-connection from \( F_{q}^{4} \) and \( F_{q}^{2} \) until the original resolution. Finally, the pixels in the unknown regions of the resulting trimaps are replaced with the corresponding alpha values in the boundary mattes to generate complete mattes.

3.5. Loss Function

**Segmentation Loss.** The focal loss [33] and consistency loss [38] are considered as the loss functions. The focal loss makes the training focus on the more challenging pixels. \( p_{c} \) and \( y_{c} \) are the predicted and ground truth probability that a pixel belongs to a class \( c \) in the trimap, and \( \gamma \) is set to be 2.

\[
\mathcal{L}_{\text{focal}} = -\sum_{c=1}^{3} y_{c}(1 - p_{c})^{\gamma} \log(p_{c}) \tag{3}
\]

The consistency loss penalizes pixels with different predictions in consecutive frames. Those pixels are required to be correct in at least one frame of the consecutive pair to avoid interruption in the basic segmentation training. This design ensures that the trimap decoder utilizes the temporal information more appropriately. While the loss in [38] is only used for the correct class, we apply it to all the classes regardless of correctness because we discovered that different weights assigned to correct and wrong classes further improve the performance. The segmentation consistency loss is defined as:

\[
\mathcal{L}_{\text{consis}} = \sum_{c=1}^{3} w_{t,h,w,c} \cdot \|p_{t,h,w,c} - p_{t+1,h,w,c}\|_1 \tag{4}
\]

where \( w_{t,h,w,c} = \begin{cases} 0.5 & \text{if } y_{t,h,w} = c \\ 0.25 & \text{else} \end{cases} \).
The final trimap segmentation loss is
\[ L_{\text{trimap}} = L_{\text{focal}} + L_{\text{consis}}. \] (5)

Matting Loss. Following RVM [32], the same loss functions are exploited to train the matting network. However, we additionally input masks of the unknown regions to make the matting decoder focus on the boundary more. For the predicted and ground truth alpha matte \( \alpha_t, \alpha^*_t \) at time \( t \) and the given mask \( M \), L1 loss \( L_{11} \) and pyramid Laplacian loss \( L_{\text{lap}} \) [15, 20] is considered to learn the alpha values and edge details. A temporal coherence loss \( L_{tc} \) [47] is adopted as well to reduce flickers.

\[ L_{11}(M) = \| M \cdot (\alpha_t - \alpha^*_t) \|_1 \] (6)

\[ L_{\text{lap}}(M) = \sum_{s=1}^{5} \frac{2^{s-1}}{5} \| M \cdot (L_{\text{pyr}}^s(\alpha_t) - L_{\text{pyr}}^s(\alpha^*_t)) \|_1 \] (7)

\[ L_{tc}(M) = \| M \cdot (\frac{d\alpha_t}{dt} - \frac{d\alpha^*_t}{dt}) \|_2 \] (8)

The overall alpha loss is defined as
\[ L_{\alpha}(M) = L_{11}(M) + L_{\text{lap}}(M) + 5 \cdot L_{tc}(M), \] (9)

and it is applied to the unknown region of the predicted trimap \( T_U \) and the ground truth trimap \( T_U^* \). The final matting loss is
\[ L_{\text{matting}} = L_{\alpha}(T_U) + L_{\alpha}(T_U^*). \] (10)

4. Experiment

4.1. Dataset, Metric and Implementation Detail

Training Dataset. The models are trained on the image matting dataset, D646 [37], the video matting dataset, VM108 [55], and the background image dataset, BG-20k [29]. D646 contains 546 training images, VM108 includes 80 clips, and BG-20k comprises 15000 background images without salient objects. The foregrounds from the matting datasets and the backgrounds from BG-20k and VM108 are composited to form the training data. Additionally, the video segmentation dataset, Youtube-VIS [52], is adopted to train the trimap segmentation to improve the generalizability of the model. Following RVM [32], the motion augmentation is applied, including affine transformation, color jittering, noise and blur, which differs gradually over time with an easing function. Frames of the matting datasets are randomly cropped to \( 480 \times 480 \) while frames of the segmentation dataset are cropped to \( 352 \times 352 \) for the larger batch size.

Evaluation Dataset. The experiments are conducted on three matting datasets: VM108 [55], VM240k [31], and Real Human dataset [49]. VM108 contains 28 composited HD (1920 × 1080) clips with about 23000 frames in the validation set. Due to the variety of objects and the longer length of the sequences, it is chosen as the benchmark in ablation studies. VM240k is a composited human video matting dataset that includes 20 4K (3840 × 2160) videos, each containing 100 frames. Real Human has 19 real HD videos about humans having about 7000 frames in total, with only 1/10 frames annotated. By default, the trimap of the first frame is given to perform trimap propagation and the input resolution is set to be 1024 × 576. All trims are produced by discretizing the dilated ground truth alpha mattes where \( 25 \times 25 \) is chosen as the dilation kernel.

Evaluation Metric. The same metrics as in RVM [32] are reported. MAD (mean absolute difference) represents the basic alpha error, MSE (mean square error) focuses on segmentation error, Grad [39] denotes the gradient error, Conn [39] (connectivity) depicts the completeness and dtSSD (mean square error of direct temporal gradient) [14] evaluates the temporal coherence such as flickering. For all metrics, the lower the better. In addition, the number of parameters and FPS (frames per second) of each model are also presented to evaluate efficiency. FPS is measured as FP32 tensor throughput and represents the speed.

Implementation Detail. The codes for training are referenced to [7]. All models are trained by an Adam optimizer with a Cosine Annealing learning rate scheduler. The learning rate is set to 1e-4 initially. It takes 120000 iterations in total to train the model. 30000 iterations are required for the image matting dataset, and the remaining are for the video matting dataset. The video segmentation training is interleaved between every 30000 iterations for 10000 iterations. Training is performed on an NVIDIA RTX A6000 and inference is conducted on an NVIDIA RTX 2080Ti GPU.

4.2. Comparative Result

We compared FTP-VM against previous methods with trimap propagation, including STM+TCVOM [35, 55] and OTVM [42]. STM+TCVOM first produces trimaps via STM and performs video matting by TCVOM. The GCA decoder is applied in TCVOM. In addition, STM is finetuned on the memory trimap to generate better results for reference, denoted as STM(ft). OTVM adopts the memory trimap and predicted alpha mattes to perform trimap propagation. Moreover, RVM [32], a human video matting method, is compared on the human matting datasets, including VM240k and Real Human dataset. The model weights from the original implementation are
applied, which are trained on the VM240k training set and a human segmentation dataset. For methods with trimap propagation, the numbers of parameters and speed are reported in Tab. 1. As previous approaches use two complete models to achieve different tasks, much more parameters are introduced. Instead, the FTP-VM combines two models along with a lightweight trimap fusion module so that the model size is significantly reduced, demonstrating that the FTP-VM outperforms other methods by a large margin in terms of efficiency.

Quantitative and Qualitative Results. Tab. 2 demonstrates the quantitative results on different datasets. For VM108, OTVM achieves the best and FTP-VM is the second best in the five metrics. Both STM+TCVOM and the finetuned one perform unsatisfactorily in this dataset. An example is illustrated in Fig. 4. Although FTP-VM produces false predictions in the boundary region, it is considered comparative due to the merit of speed.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>MSE ↓</th>
<th>MAD ↓</th>
<th>Grad ↓</th>
<th>dtSSD ↓</th>
<th>Conn ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TCVOM* [55]</td>
<td>31.54</td>
<td>48.98</td>
<td>10.37</td>
<td>3.77</td>
<td>28.60</td>
</tr>
<tr>
<td></td>
<td>OTVM [42]</td>
<td>2.92</td>
<td>13.38</td>
<td>2.46</td>
<td>2.09</td>
<td>7.28</td>
</tr>
<tr>
<td></td>
<td>FTP-VM (Ours)</td>
<td>5.19</td>
<td>20.74</td>
<td>3.92</td>
<td>2.46</td>
<td>12.13</td>
</tr>
<tr>
<td></td>
<td>TCVOM* [55]</td>
<td>1.51</td>
<td>2.98</td>
<td>3.01</td>
<td>1.96</td>
<td>1.44</td>
</tr>
<tr>
<td></td>
<td>OTVM [42]</td>
<td>0.56</td>
<td>4.55</td>
<td>1.63</td>
<td>1.32</td>
<td>0.74</td>
</tr>
<tr>
<td></td>
<td>RVM [12]</td>
<td>3.02</td>
<td>7.71</td>
<td>4.27</td>
<td>1.83</td>
<td>2.64</td>
</tr>
<tr>
<td></td>
<td>FTP-VM (Ours)</td>
<td>0.45</td>
<td>4.48</td>
<td>1.54</td>
<td>1.30</td>
<td>0.70</td>
</tr>
<tr>
<td></td>
<td>TCVOM* [55]</td>
<td>11.72</td>
<td>16.46</td>
<td>11.20</td>
<td>8.64</td>
<td>9.58</td>
</tr>
<tr>
<td></td>
<td>OTVM [42]</td>
<td>18.13</td>
<td>22.79</td>
<td>8.65</td>
<td>8.24</td>
<td>13.49</td>
</tr>
<tr>
<td></td>
<td>RVM [73]</td>
<td>2.38</td>
<td>5.87</td>
<td>4.86</td>
<td>5.15</td>
<td>3.29</td>
</tr>
<tr>
<td></td>
<td>FTP-VM (Ours)</td>
<td>2.80</td>
<td>5.87</td>
<td>5.29</td>
<td>5.29</td>
<td>3.27</td>
</tr>
</tbody>
</table>

Table 2. The quantitative comparison. TCVOM denotes STM+TCVOM and * represents the finetuned model.

For VM240k, FTP-VM outperforms other methods in most metrics except MAD. Although RVM is trained on the VM240k dataset, it reveals limited advantage. Fig. 5 compares the qualitative results. STM+TCVOM fails to provide a reasonable result in this example but the finetuned version greatly improves. OTVM occasionally produces unsatisfactory results. Both RVM and FTP-VM generate complete mattes but FTP-VM contains more details.

The Real Human dataset which contains real-world videos is also evaluated. As seen from Tab. 2 and Fig. 6, the performance of OTVM drops drastically. Since OTVM includes larger models and requires more training data, it might lead to overfitting the composited dataset, resulting in a lack of generalizability. RVM and FTP-VM both provide pleasing results with a slight difference. In Fig. 6, STM(ft)+TCVOM can produce an acceptable result but the details are missing. OTVM fails to infer the complete shape of the target and the hair part needs to be correctly captured. If the hair parts are zoomed in, it is apparent that FTP-VM describes the details better than RVM. In summary, OTVM outperforms others on VM108 but its performance drops on the other two datasets. On the contrary, RVM provides the best performance on the Real Human dataset but not on others. Overall speaking, FTP-VM reaches competitive results in all the experiments, demonstrating that the proposed method retains generalizability at a low cost.

Trimap Updating Period. In this experiment, various trimap updating periods are tested to evaluate the efficiency of trimap propagation. Fig. 7 depicts the mechanism of
trimap updates. For example, with period = 30, the trimap is updated every thirty frames. With period = 1, the trimap is given at each frame, meaning that the models only concern matting performance without any trimap propagation. This is the setting where the best performance is expected. On the contrary, if only the first trimap is provided, it will be propagated through the whole video. The bidirectional inference strategy is used in trimap propagation as [55].

The comparison on VM108 with different trimap updating periods is illustrated in Fig. 8. It is clear that the shorter the period, the better the performance. While the performance of both FTP-VM and OTVM is not significantly affected by the period, OTVM performs the best overall. Regarding MAD, FTP-VM reveals more advantages when the period becomes longer. However, STM(ft)+TCVOM surpasses FTP-VM with a shorter period, which indicates that the matting ability limits the performance of the proposed method. As for dtSSD, FTP-VM also delivers its merit with fewer flickers when the period is more extended, confirming the stability of the proposed trimap propagation.

4.3. Ablation Study
Ablation on Network Design. In order to justify the network design of FTP-VM, the results of models with different concepts are displayed in Tab. 3. Two-stage model is the baseline which has similar architecture to the previous works. After combining trimap propagation and matting into a single model, the performance slightly drops but the speed is boosted. Separating the decoder into trimap segmentation and matting decoders, as GFM [28] does, both MSE and MAD are enhanced but Grad and dtSSD degrade. For the same decoders utilizing the single features, it leads to fair stability. The best performance comes from the specially designed decoders, which is shown at the bottom of Tab. 3. The dtSSD does not outperform others, explaining that decoupling the decoder still harms the stability in some sense.

<table>
<thead>
<tr>
<th>Method</th>
<th>Param ↓</th>
<th>FPS ↑</th>
<th>MSE ↓</th>
<th>MAD ↓</th>
<th>Grad ↓</th>
<th>dtSSD ↓</th>
<th>Conn ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-stage network</td>
<td>8.46</td>
<td>31.71</td>
<td>8.80</td>
<td>30.00</td>
<td>5.87</td>
<td>2.90</td>
<td>18.18</td>
</tr>
<tr>
<td>Single network</td>
<td>4.82</td>
<td>50.07</td>
<td>10.21</td>
<td>30.16</td>
<td>3.98</td>
<td>2.30</td>
<td>17.24</td>
</tr>
<tr>
<td>Same decoders</td>
<td>5.44</td>
<td>37.33</td>
<td>9.72</td>
<td>28.42</td>
<td>5.53</td>
<td>2.80</td>
<td>17.51</td>
</tr>
<tr>
<td>Different decoders</td>
<td>5.13</td>
<td>40.16</td>
<td>5.19</td>
<td>20.74</td>
<td>3.92</td>
<td>2.46</td>
<td>12.13</td>
</tr>
</tbody>
</table>

Table 3. Ablation on the network designs. Param denotes the number of parameters (M).

Ablation on the Trimap Fusion Module. The trimap fusion module is designed to avoid introducing an additional backbone, which encodes memory frames and trimaps from scratch. As can be seen from Tab. 4, the additional backbone introduces the most parameters. The vanilla convolution improves efficiency and performance by integrating features from different levels. The gated convolution enables different channels to focus on distinct regions, so it outperforms others in all metrics with slightly increased parameters.

<table>
<thead>
<tr>
<th>Method</th>
<th>Param ↓</th>
<th>MSE ↓</th>
<th>MAD ↓</th>
<th>Grad ↓</th>
<th>dtSSD ↓</th>
<th>Conn ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>Additional backbone</td>
<td>7.59</td>
<td>11.34</td>
<td>30.75</td>
<td>5.72</td>
<td>2.82</td>
<td>17.81</td>
</tr>
<tr>
<td>Vanilla convolution</td>
<td>4.94</td>
<td>8.12</td>
<td>24.7</td>
<td>4.87</td>
<td>2.76</td>
<td>13.75</td>
</tr>
<tr>
<td>Gated convolution</td>
<td>5.13</td>
<td>5.19</td>
<td>20.74</td>
<td>3.92</td>
<td>2.46</td>
<td>12.13</td>
</tr>
</tbody>
</table>

Table 4. Ablation on trimap fusion module. Param denotes the number of parameters (M). FPS is not reported since the trimap fusion does not operate at every frame.

Ablation on Segmentation Consistency Loss. The segmentation consistency loss attempts to encourage the trimap segmentation employing information from previous frames. The results shown in Tab. 5 clearly exhibit the

<table>
<thead>
<tr>
<th>Method</th>
<th>Param ↓</th>
<th>MSE ↓</th>
<th>MAD ↓</th>
<th>Grad ↓</th>
<th>dtSSD ↓</th>
<th>Conn ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>Additional backbone</td>
<td>7.59</td>
<td>11.34</td>
<td>30.75</td>
<td>5.72</td>
<td>2.82</td>
<td>17.81</td>
</tr>
<tr>
<td>Vanilla convolution</td>
<td>4.94</td>
<td>8.12</td>
<td>24.7</td>
<td>4.87</td>
<td>2.76</td>
<td>13.75</td>
</tr>
<tr>
<td>Gated convolution</td>
<td>5.13</td>
<td>5.19</td>
<td>20.74</td>
<td>3.92</td>
<td>2.46</td>
<td>12.13</td>
</tr>
</tbody>
</table>

Table 5. Ablation on segmentation consistency loss.
improvement of temporal consistency. If consistency loss is applied to all the classes, the performance degrades slightly. However, with the strategy of weighting the loss according to the correctness, the results are further enhanced at little cost in sharpness (Grad) and stability (dtSSD).

<table>
<thead>
<tr>
<th>Method</th>
<th>MSE ↓</th>
<th>MAD ↓</th>
<th>Grad ↓</th>
<th>dtSSD ↓</th>
<th>Conn ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>w/o consistency</td>
<td>11.66</td>
<td>28.54</td>
<td>4.43</td>
<td>2.40</td>
<td>17.35</td>
</tr>
<tr>
<td>correct class only</td>
<td>6.19</td>
<td>23.33</td>
<td>3.75</td>
<td>2.28</td>
<td>13.45</td>
</tr>
<tr>
<td>all classes</td>
<td>8.75</td>
<td>24.64</td>
<td>4.08</td>
<td>2.45</td>
<td>14.26</td>
</tr>
<tr>
<td>all classes w/ weight</td>
<td>5.19</td>
<td>20.74</td>
<td>3.92</td>
<td>2.46</td>
<td>12.13</td>
</tr>
</tbody>
</table>

Table 5. Ablation on the trimap segmentation consistency loss.

4.4. Discussion

In Fig. 9, suppose the memory trimap aims at the left person. FTP-VM provides a correct result but OTVM attempts to matte all the people. This is due to overfitting on the composited dataset where salient objects are selected as the foreground. Moreover, STM-based methods such as OTVM store the resulting masks to maintain temporal coherence during memory matching. In other words, memory matching operates globally and cannot regularize the pixels spatially, which results in weaker coherence. FTP-VM adopts ConvGRU to preserve coherence and performs better.

Figure 9. The qualitative comparison between OTVM and FTP-VM on the target selection of the human video.

However, there remain some challenging cases as shown in Fig. 10. In the memory frame, there are four people in total; the leftmost is out of the screen. With the memory trimap aiming at three people on the right side, the model mattes them successfully at \( t = 55 \) (Fig. 10b). However, the video is zoomed in as considered a scene change at \( t = 65 \) (Fig. 10c), which breaks the temporal coherence in ConvGRUs so that the result is unsatisfactory. The error will be reduced over time when reconstructing the temporal coherence gradually. Furthermore, the leftmost person at \( t = 200 \) (Fig. 10d) cannot be well removed. Since the memory frame only contains his jacket, the memory cannot recognize his face and clothes. As a result, the ambiguity degrades the result. It is known that dealing with unseen objects and scene changes is challenging for memory matching, requiring further discussions and explorations.

5. Conclusion

FTP-VM, a novel end-to-end video matting model with trimap propagation, is proposed in this paper. The designed lightweight trimap fusion module is introduced to propagate the given trimap in the memory matching. Moreover, the segmentation consistency loss is adopted to fit trimap segmentation better and improve the performance. The proposed FTP-VM reaches 40 FPS, eight times faster than the cutting-edge two-model methods, TCVOM and OTVM. Extensive experiments are conducted to confirm the competitive performance in various scenarios. Its robustness and generalizability provide the foundation for more advanced developments and broader applications.
References


