Supplementary Material: One-Trimap Video Matting

Hongje Seong[1,⋆], Seoung Wug Oh[2], Brian Price[2], Euntai Kim[1], and Joon-Young Lee[2]

[1] Yonsei University, Seoul, Korea. {hjseong,etkim}@yonsei.ac.kr
[2] Adobe Research, San Jose, CA, USA. {seoh,bprice,jolee}@adobe.com

A Network Structure Details

In this section, we describe detailed network structures for trimap propagation, alpha prediction, and alpha-trimap refinement.

Trimap propagation network. Fig. S1 shows a detailed architecture of the trimap propagation network. The architecture is based on STM [11]. We employed two independent ResNet50 [2] encoders to embed memory and query. Here, the last layer (`res5`) is omitted to extract fine-scale features. The extracted memory and query features are embedded into keys and values via four independent 3×3 convolutional layers. Using the memory key and query key, the similarity is computed via non-local matching. Then the memory value is retrieved based on the computed similarity. The retrieved memory value and query value are concatenated along the channel dimension, and it is fed to the trimap decoder. In the trimap decoder, several residual blocks [3] and upsampling blocks [12,10] are employed. Finally, the propagated trimap is output from the trimap decoder.

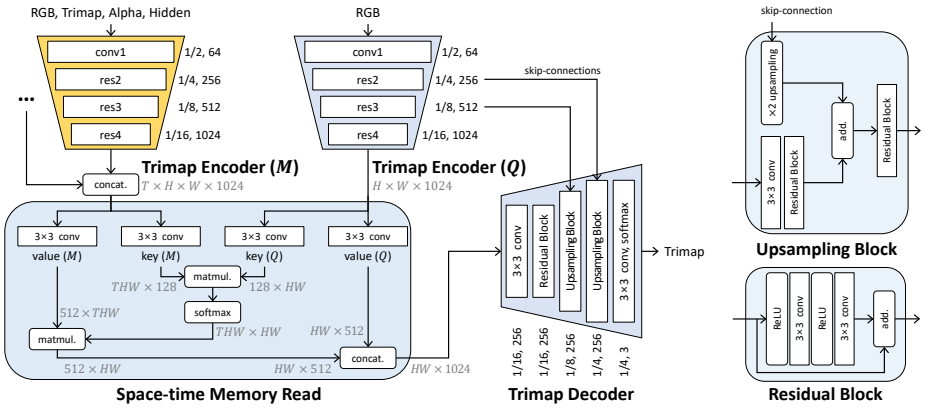

Fig. S1. A detailed illustration of the trimap propagation network. Next to each block in the figure, the relative spatial scale and channel dimension of the output are notated.

⋆ This work was done during an internship at Adobe Research.

Alpha prediction network. A detailed implementation of the alpha prediction network is given in Fig. S2. We follow the architecture of FBA [1]. The ResNet50 [2] with Group Normalization [16] and Weight Standardization [13] is used for the alpha encoder. The alpha encoder takes an RGB frame and (either a generated or user-provided) trimap. The three channels of the trimap are encoded into eight channels that are one channel for softmax probability of the foreground mask, one channel for softmax probability of the background mask, and six channels for three different scales of Gaussian blurs of the foreground and background masks [5]. In the encoder structure, the striding in the last two layers (`res4` and `res5`) is removed and the dilations of 2 and 4 are included, respectively [8]. The alpha decoder takes the resulting pyramidal features of the alpha encoder. In the alpha decoder, Pyramid Pooling Module (PPM) [19] is employed to increase the receptive field of the fine-scale feature. And then, several convolutional layers, leaky ReLU [9], and bilinear upsampling are followed. Finally, one channel of the alpha matte, three channels of the foreground RGB, three channels of the background RGB, and 64 channels of the hidden features are output from the alpha decoder.

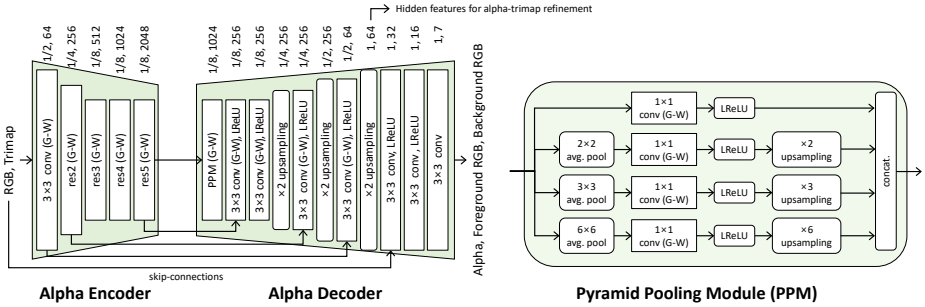

Fig. S2. A detailed illustration of the alpha prediction network. In each block, (G-W) indicates Group Normalization [16] with Weight Standardization [13] is used. LReLU denotes Leaky ReLU [9] with a negative slope of 0.01.

Alpha-trimap refinement module. We illustrate a detailed implementation of the alpha-trimap refinement module in Fig. S3. The module takes an RGB frame, trimap, predicted alpha matte, and hidden feature which is extracted from the alpha decoder. We employed two light-weight residual blocks with Group Normalization [16] and Weight Standardization [13]. The outputs are one channel of the refined alpha matte, three channels of the trimap, three channels of the foreground RGB, three channels of the background RGB, and 16 channels of the hidden features. All the outputs in the module will be used for the input of the trimap memory encoder.

B Loss Functions

For each predicted trimap, alpha matte, foreground RGB, and background RGB, we leverage several loss functions. In summary, we used cross-entropy loss for the

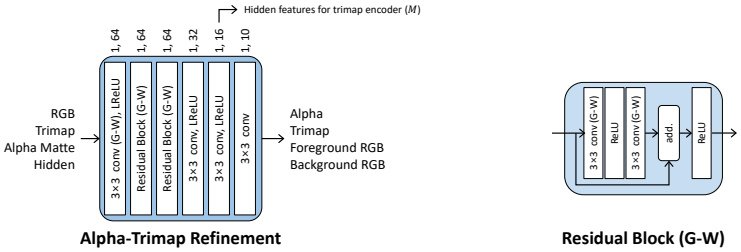

Fig. S3. A detailed illustration of the alpha-trimap refinement module.

predicted trimaps, as used in STM [11], and we used image matting losses used in FBA [1] and temporal coherence loss [14] for predicted alpha mattes, foreground RGB colors, and background RGB colors. In what follows, the specific definition of each loss function is described.

Trimap. We use the cross-entropy loss for propagated trimap ($\mathcal{L}^{tri}$) as follows:

$$\mathcal{L}^{tri} = y^{tri}_{t,i} \log(p^{tri}_{t,i}) \tag{S1}$$

where t and i indicate time and spatial pixel index, respectively; y^{tri} and p^{tri} are GT trimap and propagated trimap, respectively. The loss for refined trimap ($\widehat{\mathcal{L}}^{tri}$) is computed by simply replacing p^{tri} in Eq. (S1) with refined trimap $\widehat{p}^{tri}$. The total loss for the propagated and refined trimap is

$$\mathcal{L}^{tri}_{total} = \sum_{t=1}^{T} \sum_{i} \mathcal{L}^{tri} + \sum_{t=0}^{T} \sum_{i} \widehat{\mathcal{L}}^{tri} \tag{S2}$$

where the reference frame (where the GT trimap is provided as the input) is given at $t = 0$.

Alpha matte. With the GT alpha matte y^{α}, the predicted alpha matte p^{α} extracted from the alpha decoder, input RGB frame I, GT foreground RGB F, and GT background RGB B, we compute the $L1$ loss ($\mathcal{L}^{\alpha}_{L1}$), compositional loss ($\mathcal{L}^{\alpha}_{comp}$) [17], Laplacian pyramid loss ($\mathcal{L}^{\alpha}_{lap}$) [4], gradient loss ($\mathcal{L}^{\alpha}_{grad}$) [15], and temporal coherence loss ($\mathcal{L}^{\alpha}_{tc}$) [14] as follows:

$$\mathcal{L}^{\alpha}_{L1} = \left\| y^{\alpha}_{t,i} - p^{\alpha}_{t,i} \right\|_1, \tag{S3}$$

$$\mathcal{L}^{\alpha}_{comp} = \left\| I_{t,i} - p^{\alpha}_{t,i} F_{t,i} - (1 - p^{\alpha}_{t,i}) B_{t,i} \right\|_1, \tag{S4}$$

$$\mathcal{L}^{\alpha}_{lap} = \sum_{s=1}^{5} 2^{s-1} \left\| \mathcal{L}^{s}_{pyr}(y^{\alpha}_{t,i}) - \mathcal{L}^{s}_{pyr}(p^{\alpha}_{t,i}) \right\|_1, \tag{S5}$$

$$\mathcal{L}^{\alpha}_{grad} = \left\| \frac{\mathrm{d}y^{\alpha}_{t,i}}{\mathrm{d}i} - \frac{\mathrm{d}p^{\alpha}_{t,i}}{\mathrm{d}i} \right\|_1, \tag{S6}$$

$$\mathcal{L}_{tc}^{\alpha} = \left\| \frac{\mathrm{d}y_{t,i}^{\alpha}}{\mathrm{d}t} - \frac{\mathrm{d}p_{t,i}^{\alpha}}{\mathrm{d}t} \right\|_1, \tag{S7}$$

and the losses for the refined alpha matte ($\widehat{\mathcal{L}}_{L1}^{\alpha}, \widehat{\mathcal{L}}_{comp}^{\alpha}, \widehat{\mathcal{L}}_{lap}^{\alpha}, \widehat{\mathcal{L}}_{grad}^{\alpha}, \widehat{\mathcal{L}}_{tc}^{\alpha}$) are computed by replacing p^{α} in Eqs. (S3) to (S7) with refined alpha matte $\widehat{p}^{\alpha}$. The total loss for the predicted and refined alpha matte is defined as follows:

$$\begin{aligned}\mathcal{L}_{total}^{\alpha} = \sum_t \sum_i & \mathcal{L}_{L1}^{\alpha} + \mathcal{L}_{comp}^{\alpha} + \mathcal{L}_{lap}^{\alpha} + \mathcal{L}_{grad}^{\alpha} + \mathcal{L}_{tc}^{\alpha} \\ & + \widehat{\mathcal{L}}_{L1}^{\alpha} + \widehat{\mathcal{L}}_{comp}^{\alpha} + \widehat{\mathcal{L}}_{lap}^{\alpha} + \widehat{\mathcal{L}}_{grad}^{\alpha} + \widehat{\mathcal{L}}_{tc}^{\alpha}.\end{aligned} \tag{S8}$$

Foreground and background colors. The foreground and background RGB colors are predicted from the alpha decoder (p^F, p^B) and the alpha-trimap refinement module ($\widehat{p}^F, \widehat{p}^B$). Then, we compute the $L1$ losses ($\mathcal{L}_{L1}^{FB}, \widehat{\mathcal{L}}_{L1}^{FB}$), Laplacian losses ($\mathcal{L}_{lap}^{FB}, \widehat{\mathcal{L}}_{lap}^{FB}$), compositional losses ($\mathcal{L}_{comp}^{FB}, \widehat{\mathcal{L}}_{comp}^{FB}$), gradient exclusion losses ($\mathcal{L}_{excl}^{FB}, \widehat{\mathcal{L}}_{excl}^{FB}$), and temporal coherence losses ($\mathcal{L}_{tc}^{FB}, \widehat{\mathcal{L}}_{tc}^{FB}$). Here, the losses are computed only where the GT trimap's unknown regions ($\tilde{i} \in$ Unknown Region). Additionally, the losses for the predicted foreground color are not computed where the GT alpha value is 0 because the exact foreground color is not available in those regions. Each loss function is defined as follows:

$$\mathcal{L}_{L1}^{FB} = \left\| (y_{t,\tilde{i}}^{\alpha} > 0)(F_{t,\tilde{i}} - p_{t,\tilde{i}}^{F}) \right\|_1 + \left\| B_{t,\tilde{i}} - p_{t,\tilde{i}}^{B} \right\|_1, \tag{S9}$$

$$\mathcal{L}_{comp}^{FB} = \left\| I_{t,\tilde{i}} - y_{t,\tilde{i}}^{\alpha} p_{t,\tilde{i}}^{F} - (1 - y_{t,\tilde{i}}^{\alpha}) p_{t,\tilde{i}}^{B} \right\|_1, \tag{S10}$$

$$\begin{aligned}\mathcal{L}_{lap}^{FB} = \sum_{s=1}^{5} 2^{s-1} \Big(& \left\| (y_{t,\tilde{i}}^{\alpha} > 0)(\mathcal{L}_{pyr}^{s}(F_{t,\tilde{i}}) - \mathcal{L}_{pyr}^{s}(p_{t,\tilde{i}}^{F})) \right\|_1 \\ & + \left\| \mathcal{L}_{pyr}^{s}(B_{t,\tilde{i}}) - \mathcal{L}_{pyr}^{s}(p_{t,\tilde{i}}^{B}) \right\|_1 \Big),\end{aligned} \tag{S11}$$

$$\mathcal{L}_{excl}^{FB} = \left\| (y_{t,\tilde{i}}^{\alpha} > 0) \frac{\mathrm{d}p_{t,\tilde{i}}^{F}}{\mathrm{d}\tilde{i}} \right\|_1 \left\| \frac{\mathrm{d}p_{t,\tilde{i}}^{B}}{\mathrm{d}\tilde{i}} \right\|_1, \tag{S12}$$

$$\mathcal{L}_{tc}^{FB} = \left\| (y_{t,\tilde{i}}^{\alpha} > 0)(\frac{\mathrm{d}F_{t,\tilde{i}}}{\mathrm{d}t} - \frac{\mathrm{d}p_{t,\tilde{i}}^{F}}{\mathrm{d}t}) \right\|_1 + \left\| \frac{\mathrm{d}B_{t,\tilde{i}}}{\mathrm{d}t} - \frac{\mathrm{d}p_{t,\tilde{i}}^{B}}{\mathrm{d}t} \right\|_1, \tag{S13}$$

and the losses for the predicted colors extracted from the alpha-trimap refinement module ($\widehat{\mathcal{L}}_{L1}^{FB}, \widehat{\mathcal{L}}_{lap}^{FB}, \widehat{\mathcal{L}}_{comp}^{FB}, \widehat{\mathcal{L}}_{excl}^{FB}, \widehat{\mathcal{L}}_{tc}^{FB}$) are computed by replacing p^F, p^B in Eqs. (S9) to (S13) with $\widehat{p}^F, \widehat{p}^B$, respectively. The total loss for the foreground and background RGB colors is defined by

$$\begin{aligned}\mathcal{L}_{total}^{FB} = \sum_t \sum_{\tilde{i}} & \mathcal{L}_{L1}^{FB} + \mathcal{L}_{comp}^{FB} + \mathcal{L}_{lap}^{FB} + \mathcal{L}_{excl}^{FB} + \mathcal{L}_{tc}^{FB} \\ & + \widehat{\mathcal{L}}_{L1}^{FB} + \widehat{\mathcal{L}}_{comp}^{FB} + \widehat{\mathcal{L}}_{lap}^{FB} + \widehat{\mathcal{L}}_{excl}^{FB} + \widehat{\mathcal{L}}_{tc}^{FB}.\end{aligned} \tag{S14}$$

Finally, all loss functions are summarized by

$$\mathcal{L}_{total} = \mathcal{L}^{tri}_{total} + \mathcal{L}^{\alpha}_{total} + 0.25\mathcal{L}^{FB}_{total}. \tag{S15}$$

C Trimap Input for the Trimap Encoder

Ideally, the hidden feature can subsume the trimap information. We study the effect of the trimap input for the trimap encoder, and the results are given in Table S1. We empirically found that explicitly providing the trimap input is helpful. We conjecture that trimap input facilitates the training of the trimap propagation module under the insufficient video training dataset.

Table S1. Ablation on trimap input for trimap encoder.

Trimap Encoder Inputs	SSDA-V	MSE-V	MAD-V	dtSSD-V	MESSDdt-V	SSDA	MSE	MAD	dtSSD	MESSDdt
trimap+alpha+hidden	**54.67**	**2.61**	**13.02**	**29.87**	**1.78**	**50.51**	**8.58**	**37.16**	**28.28**	**1.63**
alpha+hidden	75.08	9.42	22.03	31.95	2.98	67.50	18.88	50.21	29.37	2.67
hidden	130.54	31.09	43.84	32.88	3.84	77.30	37.70	69.36	28.92	2.71

D Visual Analysis of the Hidden Feature

To analyze what information is contained in the hidden features, we visualize the learned hidden feature through k-means clustering (k=8) in Fig. S4. For a fair comparison, we also apply k-means clustering to the predicted trimap. As shown in the figure, the hidden feature embeds more information than trimap: (1) it subdivides the unknown regions into several levels; (2) it includes semantic information in the background regions that would be helpful to estimate accurate background regions of the next frame.

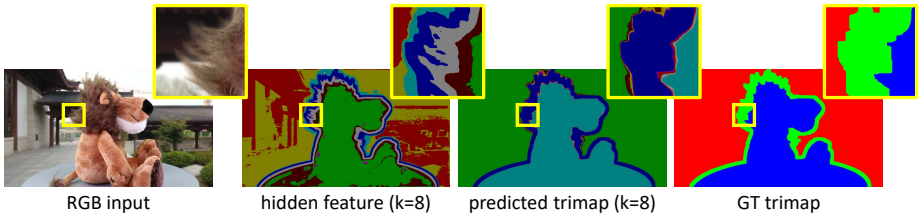

Fig. S4. Visualization of the hidden feature and trimap. For a fair comparison, we apply k-mean clustering to both hidden feature and predicted trimap.

E Effect of the Refinement Module on Trimap Estimation

To clearly show the effects of the refinement module, we measure a trimap performance of the output from the trimap propagation module in OTVM. The result is given in Table S2. We further show the effect of the refinement module qualitatively in Fig. S5. In the figure, the trimap propagation fails in the zoomed region due to motion blur, while the refinement module corrects it.

Table S2. Trimap performance. "-T" is presented to estimate trimap quality and denotes that the unknown region in GT trimap has been modified (see Sec. 4.2 in the main paper).

Method	Precision-T	Recall-T	Average
Decoupled STM [14,18]	96.98	93.58	95.28
OTVM (from trimap propagation)	98.00	95.29	96.65
OTVM (from refinement module)	**98.17**	**95.92**	**97.05**

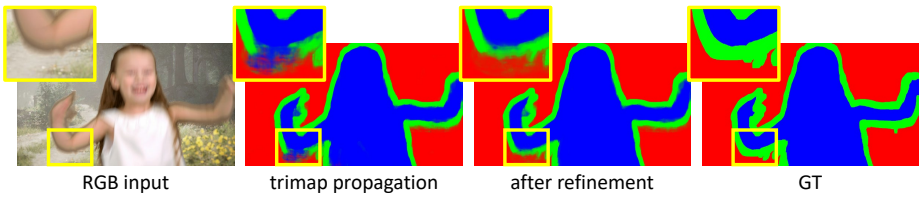

Fig. S5. Effect of the trimap refinement.

F Runtime and GPU Memory Consumption

In Fig. S6, we show the inference time and memory consumption at each frame. We used high-resolution (1920×1080) video and tested with one NVIDIA GeForce 1080 Ti GPU. As shown in the figure, OTVM slows down and consumes more memory for every 10 frames because we store the intermediate frames for trimap propagation. Instead of keeping all intermediate memory frames, we avoid this from the 30th frame by limiting our maximum memory size to 5 frames and storing only the first frame, the last frame, and up to three latest intermediate frames to the memory.

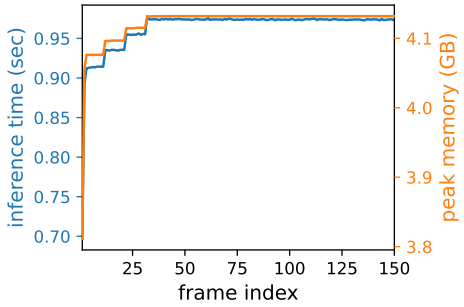

Fig. S6. Inference time and peak GPU memory.

G Temporal Stability

To demonstrate the superiority of OTVM in terms of temporal stability, we show per frame comparison in Fig. S7. In the figure, we did not cherry-pick the results and show results in all sequences of VideoMatting108 validation set. As shown in Fig. S7, decoupled approach, *i.e.*, STM+FBA, is extremely unstable in sequences

(4), (10), (15), (26), and (28). In contrast, OTVM achieves temporally stable results in most sequences.

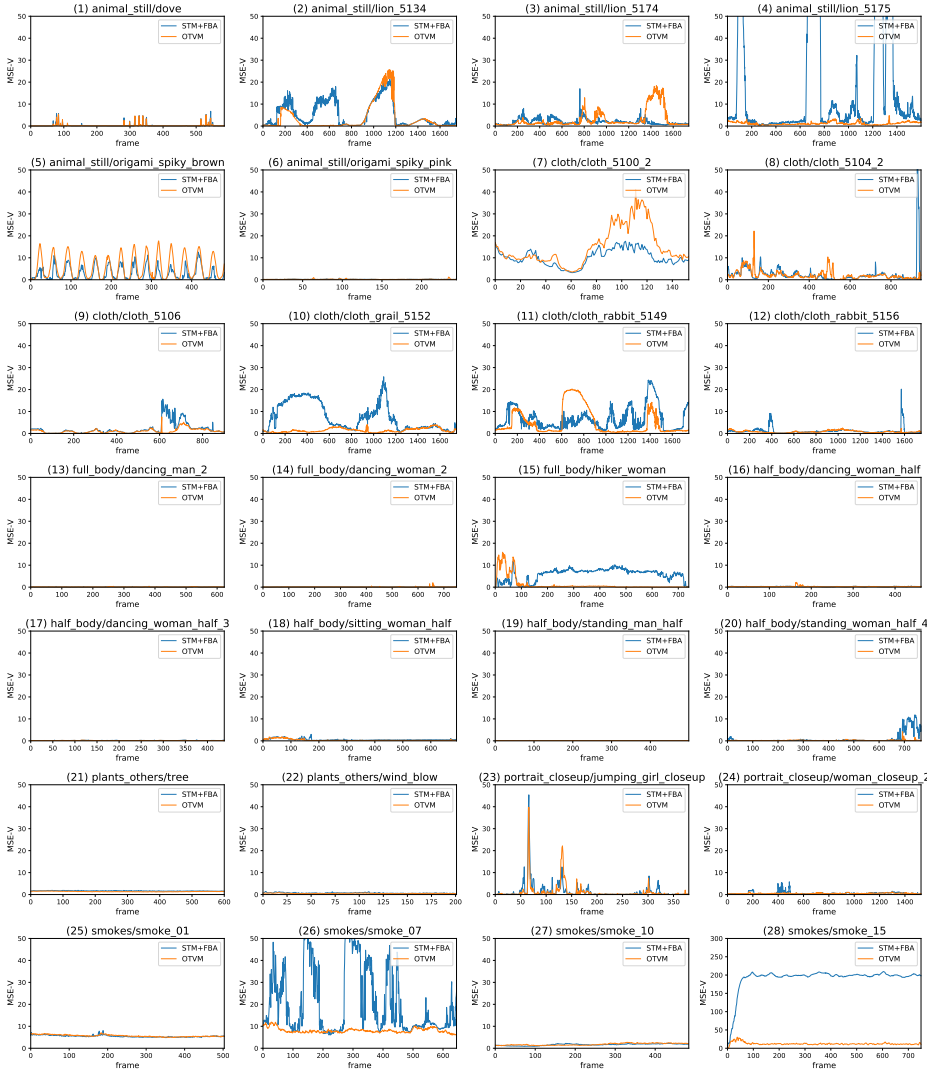

Fig. S7. Per frame comparison. The results are obtained on VideoMatting108 validation set with medium trimap setting. We plot all sequences in the validation set. Note that the lower MSE-V is better.

H More Quantitative Results

To encourage comparison for future works, we present additional results by measuring errors in a different way from the tables in the main paper. Specifically, we computed errors on the full-frame to capture the errors that occurred by inaccurate trimap propagation, and we denoted it with "-V" in Table 1 of the main paper. We re-measure by computing errors only on the unknown regions according to the official metric and report in Table S3. In contrast to those, Table 2 in the main paper is computed errors only on the unknown regions for fair comparisons with previous works. We re-measure of our implemented methods by computing errors on the full-frame and report in Table S4.

Furthermore, following [18], we show results with narrow and wide trimap settings in Tables S5 and S6. As shown in the tables, OTVM always outperforms the state-of-the-art methods in any trimap settings.

Table S3. Analysis experiments on VideoMatting108 validation set. For all experiments, we use 1-trimap setting where GT trimap is given only at the first frame.

(a) **Joint modeling**.

Model	Training method	SSDA	MSE	MAD	dtSSD	MESSDdt
STM+FBA	decoupled	70.63	**20.18**	**51.20**	31.00	2.86
	joint	**68.93**	21.10	51.57	**28.18**	**2.56**

(b) **Stage-wise training**.

Model	Training method	SSDA	MSE	MAD	dtSSD	MESSDdt
OTVM	joint	66.08	17.71	47.84	**28.28**	2.53
	joint + stage-wise	**50.51**	**8.58**	**37.16**	**28.28**	**1.63**

(c) **Ablation on training stages**.

Train stages				SSDA	MSE	MAD	dtSSD	MESSDdt
Stage 1	Stage 2	Stage 3	Stage 4					
			✓	73.66	24.30	55.81	30.39	2.72
✓			✓	66.81	18.24	48.91	28.20	2.58
✓	✓		✓	67.96	18.63	50.33	28.88	2.66
✓	✓	✓	✓	**50.51**	**8.58**	**37.16**	**28.28**	**1.63**

(d) **Ablation on modules**.

Refinement module (output)		Trimap module (input)		SSDA	MSE	MAD	dtSSD	MESSDdt	Time (sec/frame)
Alpha	Trimap	Alpha	Hidden						
				69.64	29.38	59.46	27.90	2.53	0.799
✓				70.95	27.68	57.59	28.52	2.57	0.951
	✓			69.73	22.41	53.05	28.22	2.61	0.946
✓	✓			66.96	17.96	48.24	28.62	2.57	0.952
✓	✓	✓		52.64	14.16	42.89	27.78	**1.59**	0.955
✓	✓	✓	✓	**50.51**	**8.58**	**37.16**	**28.28**	1.63	0.964

(e) **Different image matting backbones**.

Backbone	Model	SSDA	MSE	MAD	dtSSD	MESSDdt
DIM [17]	STM+DIM	88.49	**25.36**	66.98	41.64	4.40
	OTVM	**86.83**	25.66	**64.92**	**37.47**	**3.88**
GCA [6]	STM+GCA	84.82	24.38	66.01	36.14	3.60
	OTVM	**77.75**	**23.24**	**60.13**	**33.47**	**3.12**

Table S4. Comparison with state-of-the-art methods on public benchmarks. The trimap setting indicates how many GT trimaps are given as input, *i.e.*, "full-trimap" for all frames, "20/40-frame" for every 20/40th frames, "1-trimap" for only at the first frame.

(a) **Comparison on VideoMatting108 validation set**. In this experiment, we use the medium trimap setting. † denotes our reproduced results using our training setup.

Trimap Setting	Methods	SSDA-V	MSE-V	MAD-V	dtSSD-V	MESSDdt-V
full-trimap	TCVOM (GCA) [18]	50.41	2.14	12.80	27.28	1.48
	TCVOM (FBA)† [18]	**39.76**	**1.41**	**10.56**	**22.93**	**1.06**
1-trimap	STM + TCVOM (GCA) [18]	89.93	11.04	24.09	37.51	3.46
	STM + TCVOM (FBA)† [18]	81.97	10.24	22.22	34.68	3.17
	STM + FBA† [1]	83.61	10.62	22.12	36.31	3.45
	OTVM	**54.67**	**2.61**	**13.02**	**29.87**	**1.78**

(b) **Comparison on DVM validation set**.

Trimap Setting	Methods	SAD-V	MSE-V	Grad-V	Conn-V	dtSSD-V	MESSDdt-V
20-frame	OTVM	40.61	0.005	19.55	38.81	27.53	0.08
40-frame	OTVM	41.28	0.005	19.74	39.44	27.69	0.08
1-trimap	OTVM	50.64	0.007	20.88	48.45	27.75	0.10

Table S5. Comparison on VideoMatting108 validation set. We compute the error in all regions of the trimap

(a) **Narrow trimap setting**.

Trimap Setting	Methods	SSDA-V	MSE-V	MAD-V	dtSSD-V	MESSDdt-V
full-trimap	TCVOM (GCA) [18]	45.39	1.88	12.02	24.37	1.28
	TCVOM (FBA)† [18]	**37.03**	**1.24**	**9.87**	**21.09**	**0.93**
1-trimap	STM + TCVOM (GCA) [18]	86.73	10.87	23.40	35.60	3.29
	STM + TCVOM (FBA)† [18]	81.02	10.16	21.72	33.48	3.07
	STM + FBA† [1]	80.91	10.21	21.28	34.45	3.15
	OTVM	**57.58**	**2.91**	**13.13**	**29.24**	**1.72**

(b) **Wide trimap setting**.

Trimap Setting	Methods	SSDA-V	MSE-V	MAD-V	dtSSD-V	MESSDdt-V
full-trimap	TCVOM (GCA) [18]	54.35	2.38	13.56	29.60	1.69
	TCVOM (FBA)† [18]	**46.52**	**1.79**	**12.15**	**26.60**	**1.35**
1-trimap	STM + TCVOM (GCA) [18]	97.94	12.08	26.21	39.21	3.78
	STM + TCVOM (FBA)† [18]	88.97	11.12	24.54	36.67	3.51
	STM + FBA† [1]	90.78	11.48	24.21	38.62	3.89
	OTVM	**74.92**	**7.25**	**18.25**	**31.44**	**2.00**

Table S6. Comparison on VideoMatting108 validation set. We compute the error in unknown regions of the trimap

(a) **Narrow trimap setting**.

Trimap Setting	Methods	SSDA	MSE	MAD	dtSSD	MSDdt
full-trimap	DIM [17]	56.40	10.46	51.76	31.77	2.56
	IndexNet [7]	52.75	9.78	50.90	29.49	1.97
	GCA [6]	49.99	8.32	46.86	27.91	1.80
	TCVOM (GCA) [18]	45.39	7.30	44.01	24.37	1.28
	TCVOM (FBA)† [18]	**37.03**	**4.53**	**34.57**	**21.09**	**0.93**
1-trimap	STM + TCVOM (GCA) [18]	72.29	23.41	66.22	29.87	2.78
	STM + TCVOM (FBA)† [18]	66.94	21.52	60.52	28.05	2.59
	STM + FBA† [1]	66.79	21.76	60.37	29.01	2.65
	OTVM	**48.83**	**12.34**	**48.50**	**26.88**	**1.52**

(b) **Wide trimap setting**.

Trimap Setting	Methods	SSDA	MSE	MAD	dtSSD	MSDdt
full-trimap	DIM [17]	67.15	10.25	41.88	37.64	3.21
	IndexNet [7]	64.49	9.73	41.22	36.39	2.73
	GCA [6]	60.69	8.41	38.59	34.83	2.50
	TCVOM (GCA) [18]	54.35	6.98	34.81	29.60	1.69
	TCVOM (FBA)† [18]	**46.52**	**4.84**	**30.45**	**26.60**	**1.35**
1-trimap	STM + TCVOM (GCA) [18]	85.05	24.31	56.65	34.61	3.22
	STM + TCVOM (FBA)† [18]	77.59	**22.17**	52.62	32.51	3.01
	STM + FBA† [1]	78.57	22.71	52.44	33.98	3.18
	OTVM	**61.69**	24.84	**50.85**	**29.93**	**1.83**

I More Qualitative Results

We present additional qualitative results on real-world videos in Figs. S8 and S9, results on VideoMatting108 [18] with medium width trimap in Figs. S10 to S12, and results on DVM [14] in Figs. S13 and S14. We provided user-annotated (or GT) trimap only at the first frame. For all qualitative results in this section, we further provide full-frame results online: https://youtu.be/qkda4fHSyQE.

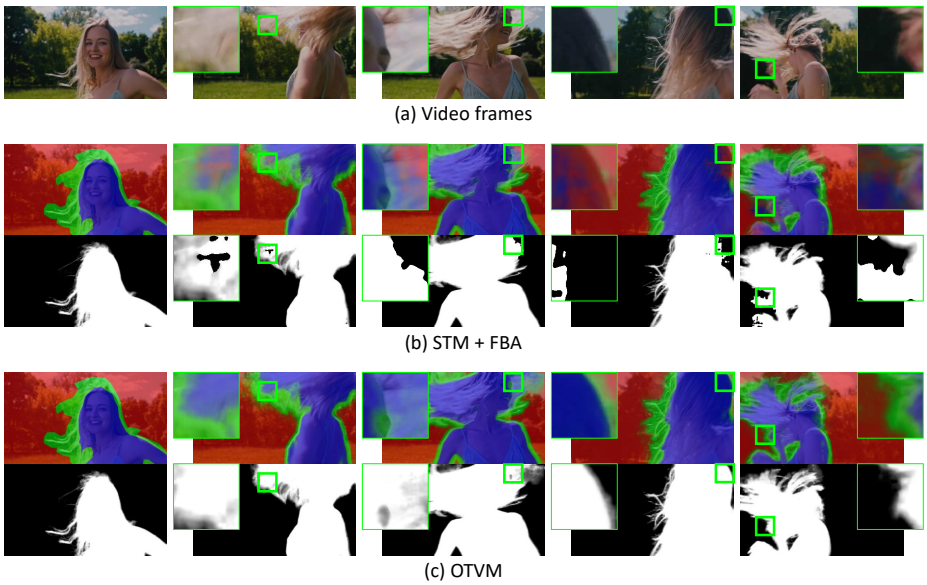

Fig. S8. Qualitative results on a real-world video.

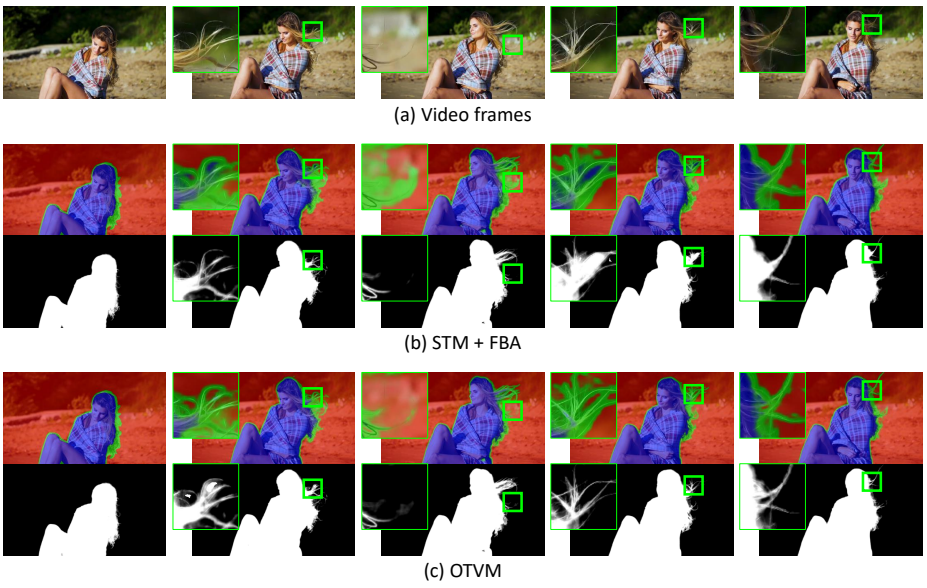

Fig. S9. Qualitative results on a real-world video.

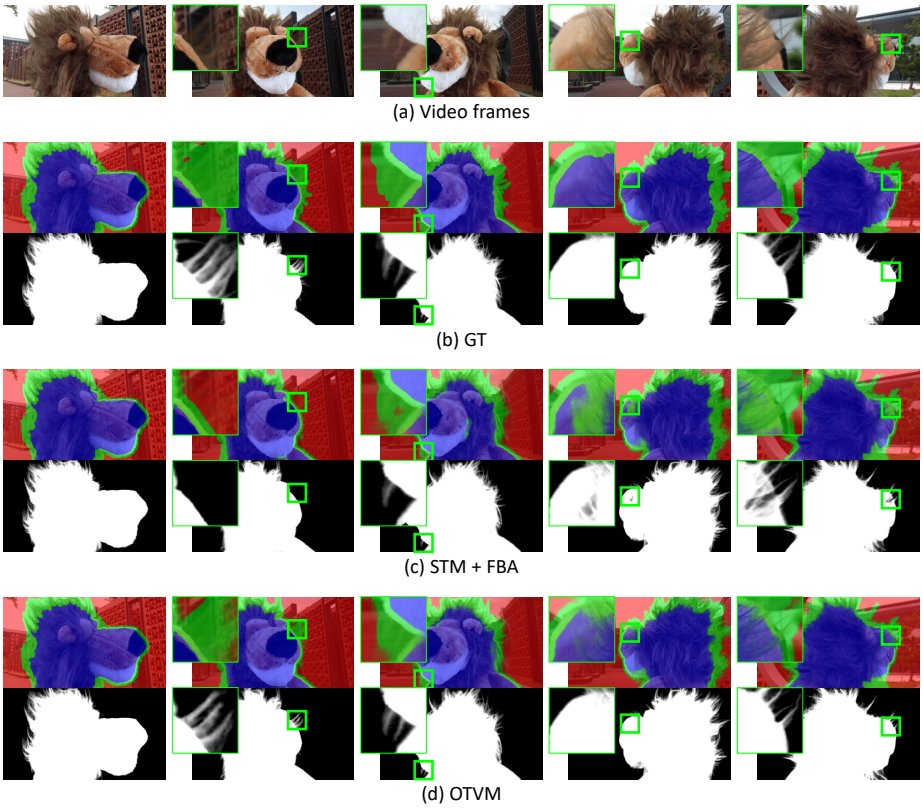

Fig. S10. Qualitative results on VideoMatting108 validation set.

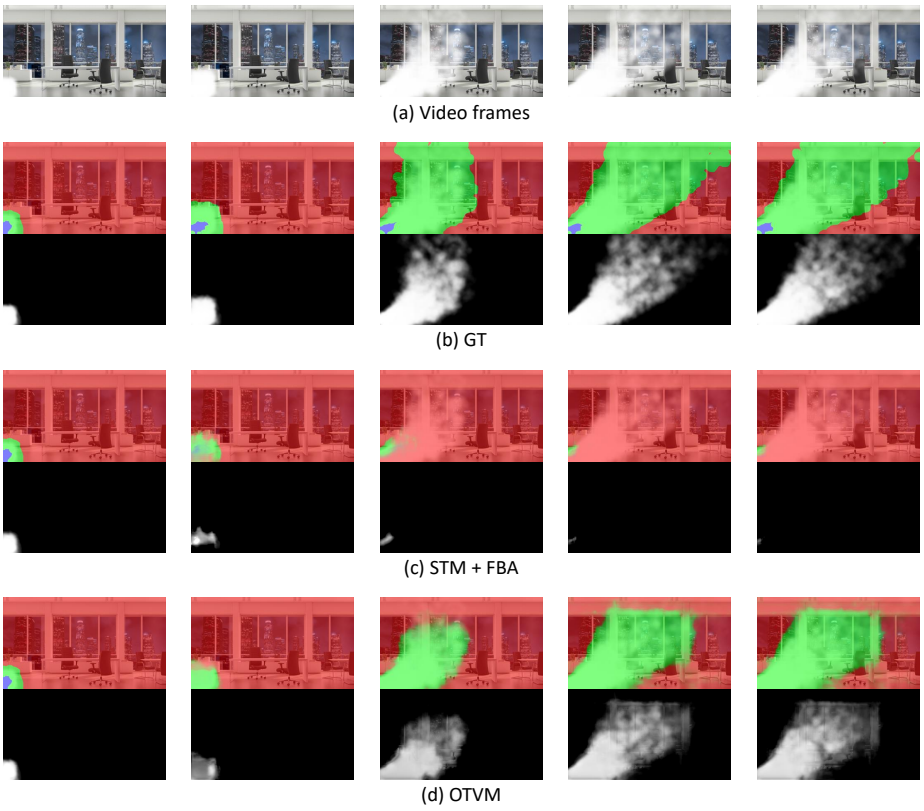

Fig. S11. Qualitative results on VideoMatting108 validation set.

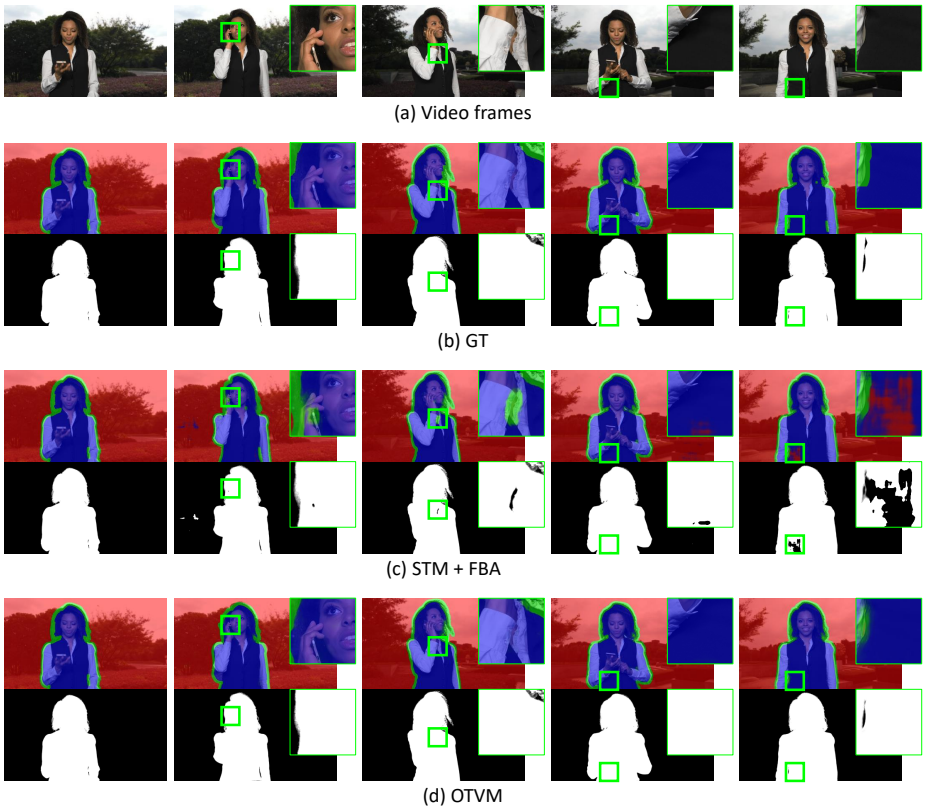

Fig. S12. Qualitative results on VideoMatting108 validation set.

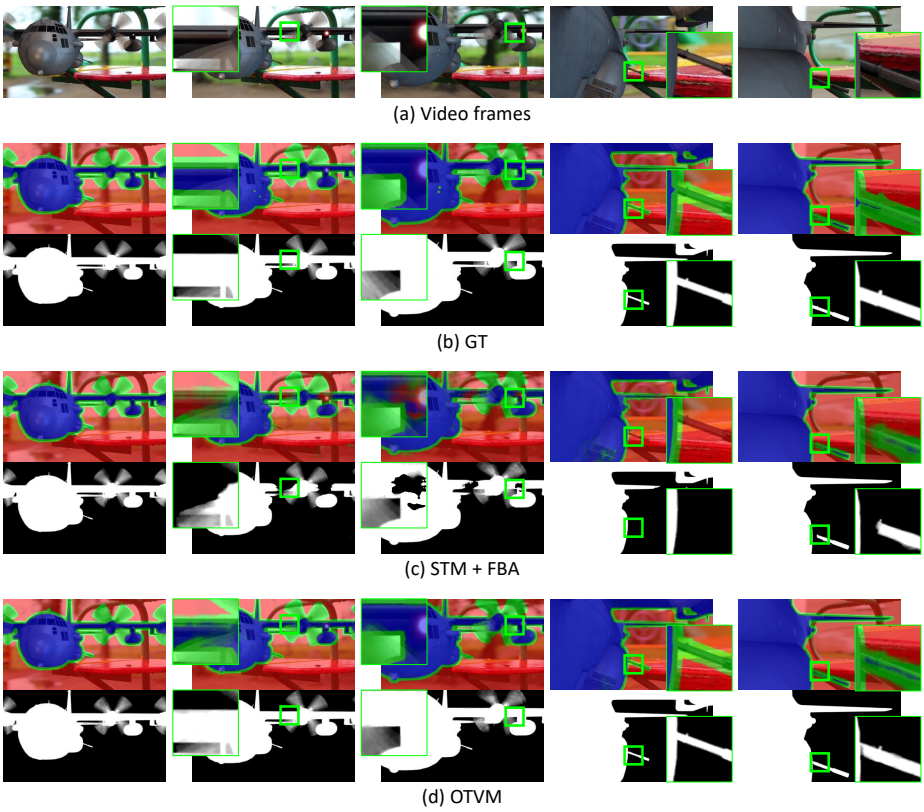

Fig. S13. Qualitative results on DVM validation set.

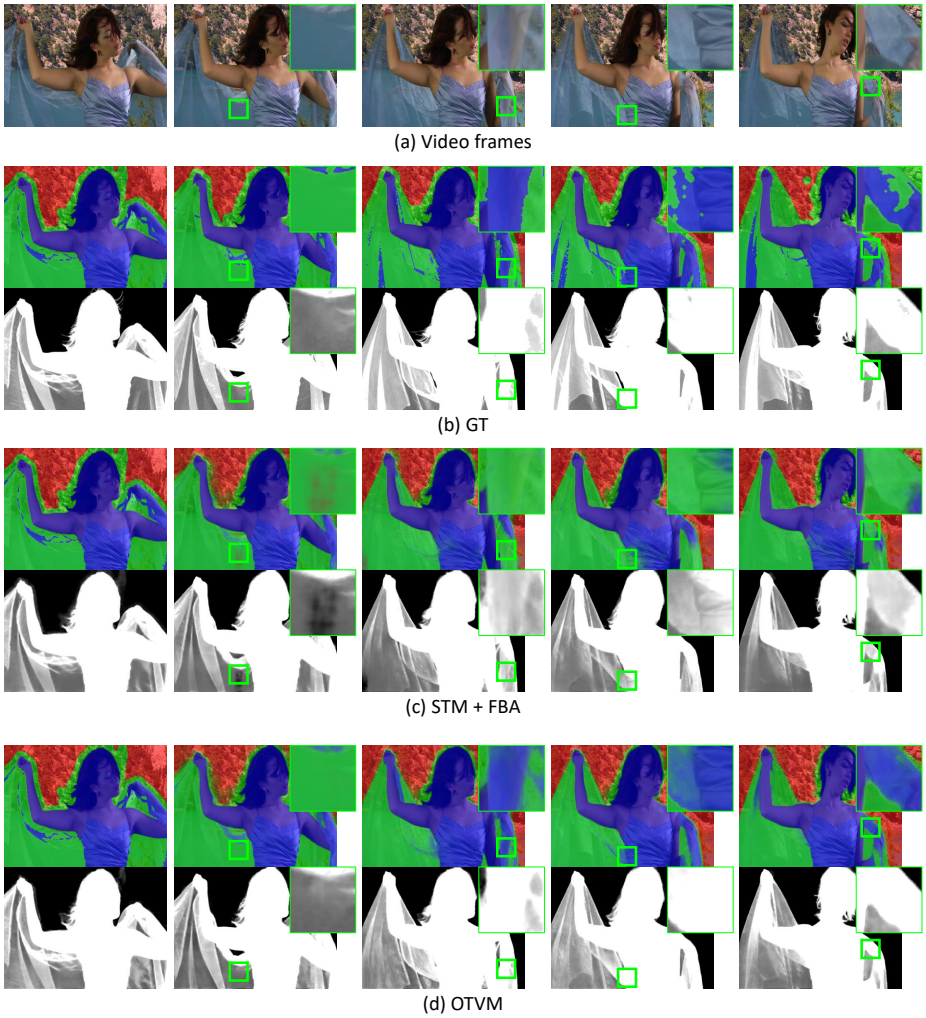

Fig. S14. Qualitative results on DVM validation set.

References

1. Forte, M., Pitié, F.: f, b, alpha matting. arXiv preprint arXiv:2003.07711 (2020) 2, 3, 9, 10
2. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016) 1, 2
3. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: ECCV. pp. 630–645. Springer (2016) 1
4. Hou, Q., Liu, F.: Context-aware image matting for simultaneous foreground and alpha estimation. In: ICCV. pp. 4130–4139 (2019) 3
5. Le, H., Mai, L., Price, B., Cohen, S., Jin, H., Liu, F.: Interactive boundary prediction for object selection. In: ECCV. pp. 18–33 (2018) 2
6. Li, Y., Lu, H.: Natural image matting via guided contextual attention. In: AAAI (2020) 8, 10
7. Lu, H., Dai, Y., Shen, C., Xu, S.: Indices matter: Learning to index for deep image matting. In: ICCV. pp. 3266–3275 (2019) 10
8. Lutz, S., Amplianitis, K., Smolic, A.: Alphagan: Generative adversarial networks for natural image matting. In: BMVC (2018) 2
9. Maas, A.L., Hannun, A.Y., Ng, A.Y., et al.: Rectifier nonlinearities improve neural network acoustic models. In: ICML (2013) 2
10. Oh, S.W., Lee, J.Y., Sunkavalli, K., Kim, S.J.: Fast video object segmentation by reference-guided mask propagation. In: CVPR. pp. 7376–7385 (2018) 1
11. Oh, S.W., Lee, J.Y., Xu, N., Kim, S.J.: Video object segmentation using space-time memory networks. In: ICCV (October 2019) 1, 3
12. Pinheiro, P.O., Lin, T.Y., Collobert, R., Dollár, P.: Learning to refine object segments. In: ECCV. pp. 75–91. Springer (2016) 1
13. Qiao, S., Wang, H., Liu, C., Shen, W., Yuille, A.: Weight standardization. arXiv preprint arXiv:1903.10520 (2019) 2
14. Sun, Y., Wang, G., Gu, Q., Tang, C.K., Tai, Y.W.: Deep video matting via spatio-temporal alignment and aggregation. In: CVPR. pp. 6975–6984 (2021) 3, 6, 10
15. Tang, J., Aksoy, Y., Oztireli, C., Gross, M., Aydin, T.O.: Learning-based sampling for natural image matting. In: CVPR. pp. 3055–3063 (2019) 3
16. Wu, Y., He, K.: Group normalization. In: ECCV. pp. 3–19 (2018) 2
17. Xu, N., Price, B., Cohen, S., Huang, T.: Deep image matting. In: CVPR. pp. 2970–2979 (2017) 3, 8, 10
18. Zhang, Y., Wang, C., Cui, M., Ren, P., Xie, X., Hua, X.s., Bao, H., Huang, Q., Xu, W.: Attention-guided temporal coherent video object matting. In: ACM MM (2021) 6, 8, 9, 10
19. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: CVPR. pp. 2881–2890 (2017) 2