

# SUPPLEMENTARY DOCUMENT: EFFICIENT VIDEO MATTING ON HUMAN VIDEO CLIPS FOR REAL-TIME APPLICATION

*Chao-Liang Yu and I-Chen Lin*

College of Computer Science, National Yang Ming Chiao Tung University, Taiwan

Note: The images and video of the supplementary material have been substantially compressed to meet the size limitation of ICME'23.

## 1. METHOD DETAILS

### 1.1. ConvGRU

To ensure the temporal coherence across frames, and to prevent the matting results from flickering, we adopted the ConvGRU module in the decoder of our network, which is defined by following formulas:

$$\begin{aligned} z_t &= \sigma(w_{xz} * x_t + w_{hz} * h_{t-1} + b_z) \\ r_t &= \sigma(w_{xr} * x_t + w_{hr} * h_{t-1} + b_r) \\ h'_t &= \tanh(w_{xh} * x_t + w_{hh} * (r_t \odot h_{t-1}) + b_h) \\ h_t &= z_t \odot h_{t-1} + (1 - z_t) * h'_t \end{aligned} \quad (1)$$

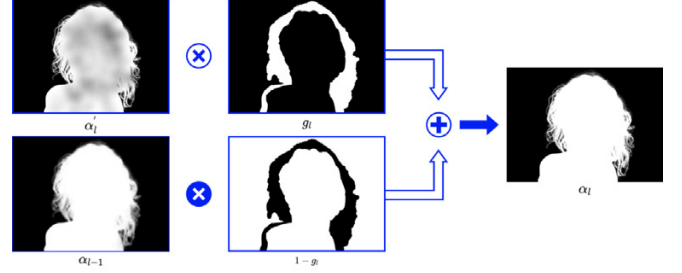
where  $x$  is the input features, and  $w$  and  $b$  are the weight and bias of convolution operations.  $z$  is the update gate for deciding how much of the past information needs to be passed to the next state, and  $r$  is the reset gate for deciding how much of the past information needs to be forgotten.  $*$  and  $\odot$  represents convolution and Hadamard product respectively.

At timestamp  $t$ ,  $h_t$  is the output of the ConvGRU module, and at the next timestamp, it is fed into the module again to act as hidden state  $h_{t+1}$ . At the beginning of a video sequence, where no past information is available, the hidden state  $h_0$  is initialized as an all zero tensor.

### 1.2. Progressive Refinement Module (PRM)

Both low-level structures and high-level details are important in matting, so by utilizing the fact that different layers of a CNN network focus on different size of images, we can let the network output alpha values at each scale, and combine them together to produce better results.

After obtaining outputs from different levels, at level  $l$ , the corresponding alpha outputs  $\alpha_l$  is upsampled to match the original resolution, and a self-guidance mask is defined with the following formula:



**Fig. 1:** Visualization of how PRM module works. At each level, a mask is generated according to Eq. 2, and the white parts from different output levels are combined.

$$g_l(x, y) = \begin{cases} 1, & \text{if } 0 < \alpha_{l-1}(x, y) < 1 \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

The intuition is similar to how we define error-prone regions in the aforementioned refiner: pixels with values 0 and 1 are regarded as confident regions, and those with values between 0 and 1 are regarded as non-confident regions. Given the raw alpha output  $\alpha'_l$  and the self-guidance mask  $g_l$ , refined alpha output  $\alpha_l$  is obtained with the following formula:

$$\alpha_l = \alpha'_l \odot g_l + \alpha_{l-1} \odot (1 - g_l) \quad (3)$$

Confident regions in the previous alpha output  $\alpha_{l-1}$  is kept, whereas non-confident regions are replaced with contents in the current alpha output  $\alpha'_l$  according to  $g_l$ . The above process is visualized in Fig. 1. During training, loss functions are applied only on non-confident regions. Therefore, the model can focus on non-confident regions each time, and the alpha output will be gradually refined. Note that our model outputs foreground images only at the output block, and they are not refined by the PRM module.

### 1.3. Loss Functions

Loss functions mentioned below are applied on all frames  $t \in [1, T]$ . For alpha outputs, given ground truth  $\alpha_t^*$  and prediction  $\alpha_t$ , we used L1 loss, Laplacian loss [1], and temporal coherence loss [2], where  $s$  is the index of layers in computing

Laplacian Pyramids.

$$L_{l1}^\alpha = \|\alpha_t - \alpha_t^*\|_1 \quad (4)$$

$$L_{lap}^\alpha = \sum_{s=1}^5 2^{s-1} \|L_{pyr}^s(\alpha_t) - L_{pyr}^s(\alpha_t^*)\|_1 \quad (5)$$

$$L_{tc}^\alpha = \left\| \frac{d\alpha_t}{dt} - \frac{d\alpha_t^*}{dt} \right\|_2 \quad (6)$$

For foreground outputs, loss were only computed where  $\alpha_t^* > 0$ . Given ground truth  $F_t^*$  and prediction  $F_t$ , we used L1 loss and temporal coherence loss.

$$L_{l1}^F = \|(\alpha_t^* > 0) * (F_t - F_t^*)\|_1 \quad (7)$$

$$L_{tc}^F = \|(\alpha_t^* > 0) * \left( \frac{dF_t}{dt} - \frac{dF_t^*}{dt} \right)\|_2 \quad (8)$$

For the semantic mask, we also used the same L1 loss and Laplacian loss as the alpha loss.

$$L^{mask} = L_{l1}^{mask} + L_{lap}^{mask} \quad (9)$$

Moreover, following [3], the L1 loss and Laplacian loss for the low-resolution alpha were applied to each output head of the PRM module, and were only computed on the non-confident regions. The total loss function for the low-resolution output without the refiner module can be formulated as:

$$L_{LR}^M = \sum_l w_l (L_{l1}^\alpha * g_l + L_{lap}^\alpha * g_l) + 5 * L_{tc}^\alpha + 2 * L_{l1}^F + 10 * L_{tc}^F + 0.25 * L^{mask} \quad (10)$$

where  $w_l$  is the weighting factor assigned to outputs at level  $l$ , and we use  $w_0 = \frac{1}{6}, w_1 = \frac{2}{6}, w_2 = \frac{3}{6}$ .

For the high-resolution output, we do not apply loss on the semantic mask, and the total loss function is:

$$L_{HR}^M = L_{l1}^\alpha + L_{lap}^\alpha + 5 * L_{tc}^\alpha + 2 * L_{l1}^F + 10 * L_{tc}^F \quad (11)$$

## 2. EXPERIMENT DETAILS

### 2.1. Datasets

To train our model, we applied VideoMatte240K (VM240K) [4] in the early stage of the training process, and the mixture of Adobe Image Matting (AIM) [5], Distinctions-646 (D646) [6], and Human-2K (H2K) [7]) in the later stage of the training process. VM240K contains a wide variety of action and movement, which is desirable for our model to learn long-term dependencies across frames. Although the other three datasets contain only still images, they have more fine-grained details like hair, and is also essential to the training procedure.

We followed the common approach to split VM240K into 479/5 clips for training/testing, and the training set contains 237510 frames in total. Moreover, we used only a subset of AIM and D646, excluding those with non-human foreground objects, and got 2500 frames along with H2K for training.

As matting datasets only provide foregrounds and alpha mattes, we still need backgrounds to compose images for training. For background videos, we used video clips provided by [2]; we selected clips that do not contain humans, and extract the first 100 frames from each clip. For background images, we used BG-20K [8], which does not contain salient objects and is therefore suitable for composing matting data.

Additionally, as proposed by RVM [9], we also used datasets from the semantic segmentation task, including YouTubeVIS [10], COCO [11], and Supervisely Person Dataset (SPD), to assist the training process. Because training data for the matting task are synthetically generated, images may look fake or unnatural and cause the model to overfit on these artificial data. On the other hand, semantic segmentation is closely related to matting, especially human matting, and its datasets feature a great amount of natural images, which can help our model learn the distribution of real-world human poses. These data went through both the mask-prediction network and the matting network, and the last layer of the matting network was switched to have only one-channel output to fit the segmentation task.

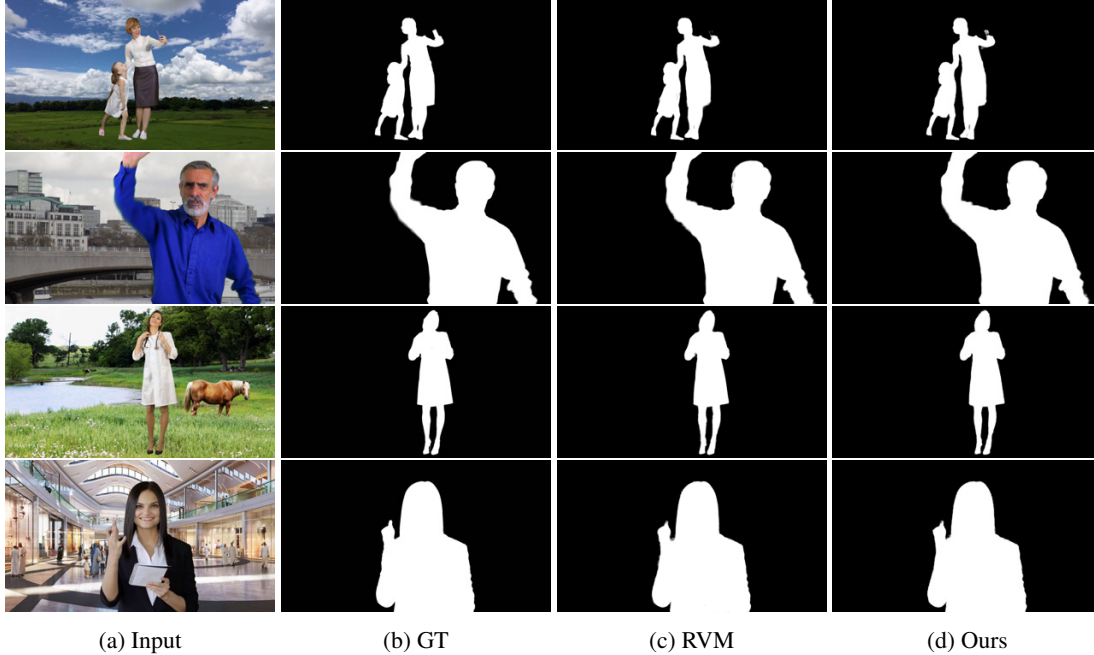
We used a wide variety of data augmentation techniques to increase the diversity of training data, including affine translation, noise, color jittering, and blur. They were also applied on datasets containing only still images to generate synthetic image sequences. Clip reversal, speed changes, random pausing, and frame skipping were used to make the model learn more complex temporal changes.

### 2.2. Training Procedures

Because we focus on video clips instead of individual frames, besides an extra dimension  $T$  is added to control the number of frames used for training, and the dimension of input data becomes  $[B, T, C, H, W]$ , where  $B$  is the batch size,  $C$  is the number of channels, and  $H$  and  $W$  represent the height and width of an image frame. The whole training procedure can be divided into three stages, and it took about 5 days on an NVIDIA RTX 3090 GPU.

Additionally, training for segmentation data took place after every iteration of training for matting data. Image segmentation (with COCO and SPD data) was trained after every odd iteration, whereas video segmentation data is trained after every even iteration.

**Stage 1:** We first trained the model on VM240K dataset for 15 epochs, so it can learn the basic structure of human poses and the temporal relationship across frames. The resolution is  $512 \times 512$ , and the PointRend-based refiner is not



**Fig. 2:** Qualitative comparison on VM240K HD dataset. Our method does not produce unexpected artifacts at boundary regions like RVM does. Please zoom in to see the difference.

used in this stage. We set the learning rate of the mask-prediction network as  $5e-4$ , the encoder of the matting network as  $1e-4$ , and the rest of the network as  $2e-4$ . Batch size  $B$  and sequence length  $T$  is set to be 4 and 15 respectively.

**Stage 2:** The refiner module is used in this stage to learn on high-resolution data for 5 epochs. Due to memory limitation, we set batch size  $B = 2$  and sequence length  $T = 6$  for high-resolution data. The resolution is  $2048 \times 2048$ , and the downsampling factor  $s$  is 0.25. To prevent the model from overfitting on short sequences, we also jointly trained the model on low-resolution data, with the same  $B$  and  $T$  setting from Stage 1. We set the learning rate of the mask-prediction network as  $5e-5$ , the encoder of the matting network as  $1e-5$ , the decoder of the network as  $2e-5$ , and the refiner as  $2e-4$ .

**Stage 3:** For the model to learn more fine-grained details, we trained it on the combination of AIM, D646 and H2K for 8 epochs. The refiner module uses  $1e-4$  learning rate for the first 5 epochs, and  $2e-5$  for the last 3 epochs, while the rest of the network uses  $2e-6$ .

### 2.3. Additional Evaluation

Fig. 2 shows the visual comparison between our method and RVM, which uses [12] for upsampling and filtering on high-resolution input.

We also compared our method against two auxiliary-free methods (MODNet and RVM) on real-world data. Cellphone videos in Fig. 3 were from [13], while webcam videos in Fig.

4 were from [14]. These real-world data is more challenging than the synthetic data used for training, thus making the matting process harder. It is clear that our method produces less classification errors, and is more visually pleasing.

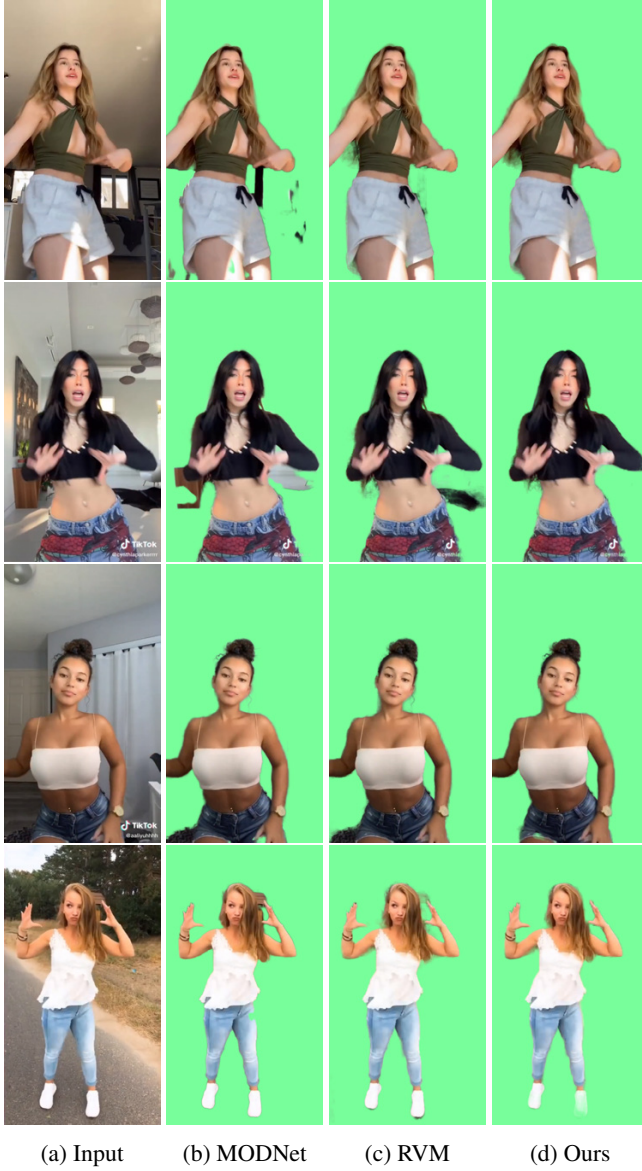
### 2.4. Figures of Ablation Studies

We conducted ablation studies to discuss the effectiveness of each part of our network. We first removed the mask-prediction network, and passed zero tensors into the ConvGRU modules to take away temporal information. As shown in Fig. 5, the results produced without the mask-prediction network cannot handle fine-grained details in certain regions, whereas the results produced without temporal information fail with fast-moving objects.

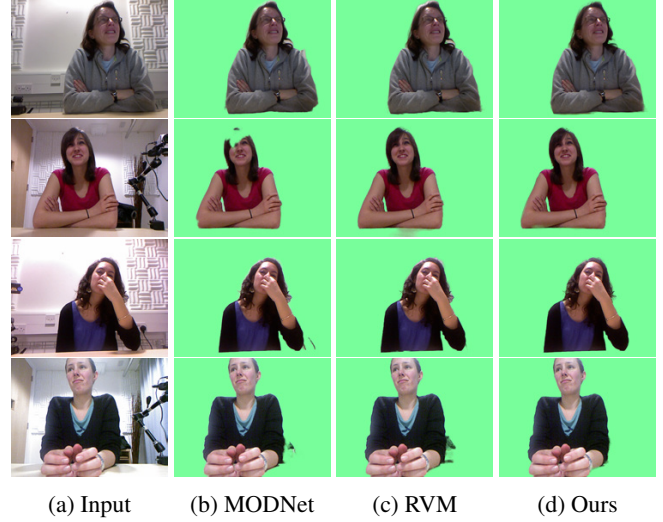
We also compared different criterion strategies for selecting error-prone regions for refinement, as discussed earlier in the main paper. Fig. 6 shows that our method can produce the sharpest boundary regions. Fig. 7 visualizes the regions selected by different strategies.

## 3. REFERENCES

- [1] Qiqi Hou and Feng Liu, “Context-aware image matting for simultaneous foreground and alpha estimation,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4130–4139.
- [2] Yanan Sun, Guanzhi Wang, Qiao Gu, Chi-Keung Tang, and Yu-Wing Tai, “Deep video matting via spatio-



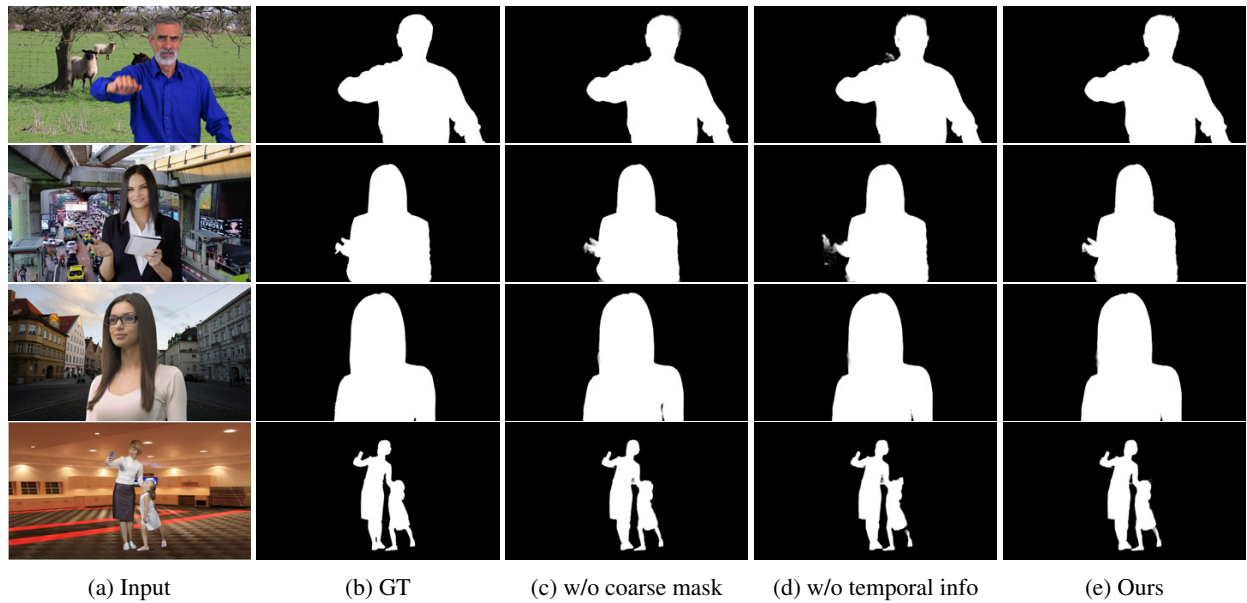
**Fig. 3:** Qualitative comparison on cellphone videos.



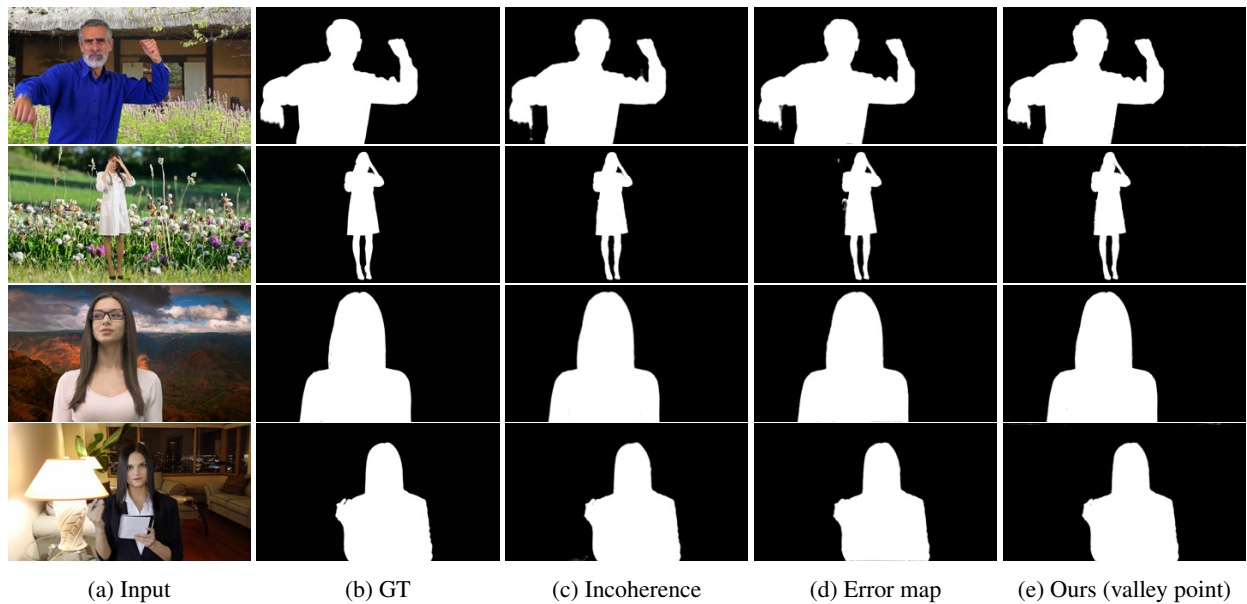
**Fig. 4:** Qualitative comparison on webcam videos.

temporal alignment and aggregation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6975–6984.

- [3] Qihang Yu, Jianming Zhang, He Zhang, Yilin Wang, Zhe Lin, Ning Xu, Yutong Bai, and Alan Yuille, “Mask guided matting via progressive refinement network,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1154–1163.
- [4] Shanchuan Lin, Andrey Ryabtsev, Soumyadip Sengupta, Brian L Curless, Steven M Seitz, and Ira Kemelmacher-Shlizerman, “Real-time high-resolution background matting,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8762–8771.
- [5] Ning Xu, Brian Price, Scott Cohen, and Thomas Huang, “Deep image matting,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2970–2979.
- [6] Yu Qiao, Yuhao Liu, Xin Yang, Dongsheng Zhou, Mingliang Xu, Qiang Zhang, and Xiaopeng Wei, “Attention-guided hierarchical structure aggregation for image matting,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 13676–13685.
- [7] Yuhao Liu, Jiake Xie, Xiao Shi, Yu Qiao, Yujie Huang, Yong Tang, and Xin Yang, “Tripartite information mining and integration for image matting,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 7555–7564.
- [8] Jizhizi Li, Jing Zhang, Stephen J Maybank, and Dacheng Tao, “Bridging composite and real: towards end-to-end deep image matting,” *International Journal of Computer Vision*, vol. 130, no. 2, pp. 246–266, 2022.
- [9] Shanchuan Lin, Linjie Yang, Imran Saleemi, and

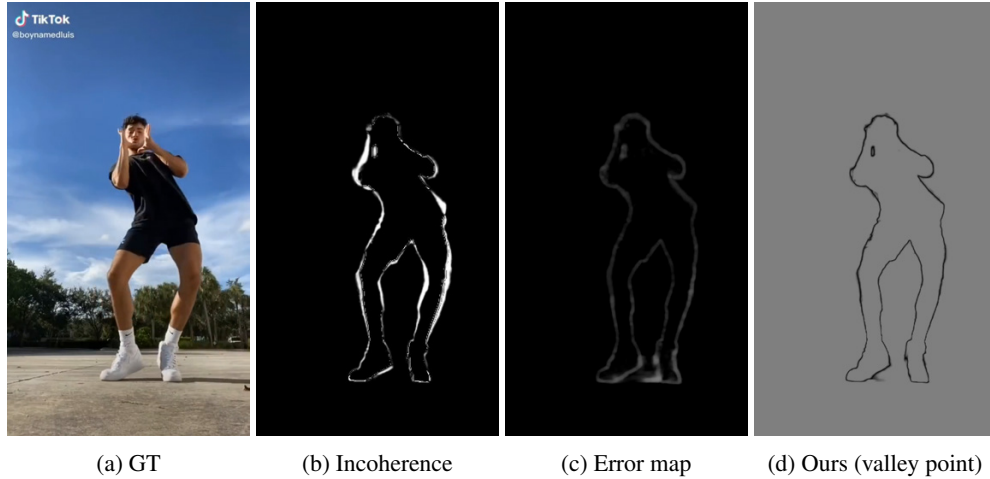


**Fig. 5:** Visual comparison on removing different parts of the network. Results produced by the full model contain less semantic-level errors.



**Fig. 6:** Visual comparison on different strategies for selecting error-prone regions. Our method can produce sharper boundary regions.





**Fig. 7:** Error-prone regions for refinement selected by different strategies.

Soumyadip Sengupta, “Robust high-resolution video matting with temporal guidance,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 238–247.

- [10] Linjie Yang, Yuchen Fan, and Ning Xu, “Video instance segmentation,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 5188–5197.
- [11] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick, “Microsoft coco: Common objects in context,” in *European conference on computer vision*. Springer, 2014, pp. 740–755.
- [12] Huikai Wu, Shuai Zheng, Junge Zhang, and Kaiqi Huang, “Fast end-to-end trainable guided filter,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1838–1847.
- [13] Yasamin Jafarian and Hyun Soo Park, “Learning high fidelity depths of dressed humans by watching social media dance videos,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12753–12762.
- [14] Marwa Mahmoud, Tadas Baltrušaitis, Peter Robinson, and Laurel D Riek, “3d corpus of spontaneous complex mental states,” in *International Conference on Affective Computing and Intelligent Interaction*. Springer, 2011, pp. 205–214.