Deep Image Matting with Flexible Guidance Input

Hang Cheng
ch20721335@shu.edu.cn
Shugong Xu*
shugong@shu.edu.cn
Xiufeng Jiang
XiufengJiang@shu.edu.cn
Rongrong Wang
wangrongrong@shu.edu.cn

Shanghai Institute for Advanced Communication and Data Science(SICS),
Shanghai University,
Shanghai,
200444,
China

Abstract

Image matting is an important computer vision problem. Many existing matting methods require a hand-made trimap to provide auxiliary information, which is very expensive and limits the real world usage. Recently, some trimap-free methods have been proposed, which completely get rid of any user input. However, their performance lag far behind trimap-based methods due to the lack of guidance information. In this paper, we propose a matting method that use **Flexible Guidance Input** as user hint, which means our method can use trimap, scribblemap or clickmap as guidance information or even work without any guidance input. To achieve this, we propose **Progressive Trimap Deformation(PTD)** scheme that gradually shrink the area of the foreground and background of the trimap with the training step increases and finally become a scribblemap. To make our network robust to any user scribble and click, we randomly sample points on foreground and background and perform curve fitting. Moreover, we propose **Semantic Fusion Module(SFM)** which utilize the Feature Pyramid Enhancement Module(FPEM) and Joint Pyramid Upsampling(JPU) in matting task for the first time. The experiments show that our method can achieve state-of-the-art results comparing with existing trimap-based and trimap-free methods. Our demo is at https://github.com/Charch-630/FGI-Matting.

1 Introduction

Image Matting is an important computer vision problem which aims to precisely predict an alpha matte to separate the foreground object from the background. The problem has already been studied by both academia and industry for years and has many applications in image processing and film production. Ordinarily, the Image Matting task is modeled to solve the following equation known as the Composition Equation.

$$I_i = \alpha_i F_i + (1-\alpha_i)B_i, \qquad \alpha \in [0,1] \tag{1}$$

* indicates the corresponding author.

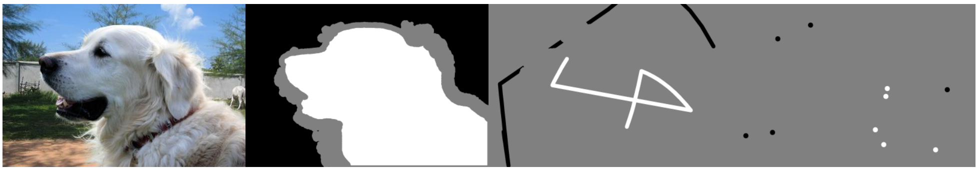

Figure 1: Different guidance input. Trimap, scribblemap and clickmap.

where I denotes the input Image, α refers to the alpha matte that shows the opacity of the foreground object, F and B represent the foreground and background respectively, i means per pixel location. As only the input RGB image is given and the algorithm has to solve the rest 7 values for each pixel, the problem is highly ill-posed.

In order to address this problem, most previous methods also require a hand-made trimap to provide auxiliary information. The trimap uses white($\alpha = 1$), gray($\alpha = 0.5$) and black($\alpha = 0$) to define the foreground, transition area and background respectively. In the past few years, many methods [1, 2, 7, 8, 9, 12, 19, 21] using trimap as input have achieved very good accuracy and very few trimap-free methods can surpass them. However, when it comes to application, drawing a suitable and correct trimap requires some skills and much time, which is hassle for users who don't have any prior knowledge about matting.

In the past few years, some trimap-free methods have been proposed. These methods [14, 20] hope to capture both semantic feature and texture details from the RGB input by end-to-end training on large-scale dataset. However, their performance still lag far behind trimap-based methods and these methods are controversial. The main reason is that without guidance input, the network is confused about which part of the input image is foreground area and thus affect subsequent detail feature extraction.

To solve this, we propose a matting method that use flexible guidance input as user hint, which means our method works not only for trimap input, but also for scribble and click input and even no guidance input(see Fig. 1). To achieve this, we introduce a data augmentation scheme called Progressive Trimap Deformation(PTD) that gradually shrink the area of the foreground and background of the trimap with the training step increases, and the shape of the foreground and background will eventually become scribbles. Moreover, to make our network robust to any user scribble and click, we propose to randomly sample points on foreground and background and perform curve fitting to simulate human input scribbles. As a result, experiments show that our method can achieve state-of-the-art results compared with previous trimap-based and trimap-free methods. Compared to previous trimap-based matting methods, ours reduces the complexity of the guidance input while ensuring the accuracy. When applied to real-world scene, for foreground objects of different difficulty, users can flexibly choose guidance input of different complexity levels. For example, for hard foreground objects, we can draw a trimap to give the network more details. For those of moderate difficulty, we can draw scribbles to save some time. For simple and salient objects, we just need a few clicks or even no guidance is needed. More importantly, our method can easily be applied to train other trimap-based methods, making them only require scribble or click input.

Moreover, in matting tasks, advanced semantics from deep levels of the backbone indicate foreground category and profiles, while low-level features contain texture and detail information. High-level semantic features can guide low-level features to correctly predict the details of the foreground region. In order to extract advanced semantics efficiently, we propose Semantic Fusion Module(SFM) inspired by FPEM [16] and JPU [18]. The SFM module can enhance semantic features by extracting multi-scale information from the backbone, then the advanced semantics of different scales can be joint upsampled and fused.

To sum up, our contribution mainly includes the following points:

- We propose a matting method based on Flexible Guidance Input, which means our method can use trimap, scribblemap or clickmap as guidance information. Moreover, Our method can even work without any guidance input. The experiments verify that our method can achieve state-of-the-art results comparing with existing trimap-based and trimap-free methods.
- We propose a data augmentation scheme called Progressive Trimap Deformation(PTD) that gradually shrink the area of the foreground and background of the trimap during training and eventually become a scribblemap. To make our network robust to any user scribble and click, we randomly sample points on foreground and background and perform curve fitting to mimic human input scribbles.
- In order to extract advanced semantics efficiently, we propose Semantic Fusion Module(SFM) which utilize the FPEM [16] and JPU [18] modules in matting tasks for the first time. The SFM module extract multi-scale information from the backbone and then fuse and joint upsample them to enhance the advanced semantics.

2 Related Work

2.1 Trimap-based Matting methods.

Most existing matting methods require a trimap as an auxiliary input. Traditional matting methods can be categorized into two types: sampling based and propagation based. Sampling-based methods [4, 5, 6] first model foreground and background statistics through sampling pixels in the given foreground and background area, then solve the composition equation to get alpha matte. Propagation based methods [2, 8] propose to propagate the alpha value from the given foreground and background region to unknown area. In the past few years, deep learning based methods have been proved successful in solving image matting problems. Xu *et al.* [19] proposed an encoder-decoder structure with RGB image and a trimap as input to predict alpha matte, and created a matting dataset with various foregrounds composited to background images. Hou *et al.* [7] propose to use two decoder to predict foreground color and alpha simultaneously. Li *et al.* [9] propose a u-net structure with a guided contextual attention block and they achieved better results.

2.2 Trimap-free Matting methods.

Recently, some trimap-free matting approaches have emerged. Some of them propose to train on large-scale dataset to completely get rid of trimap. Zhang *et al.* [20] use two decoder branches to predict foreground and background classification respectively and then fuse them together. Its input is only an RGB image. Qiao *et al.* [14] propose to use spatial and channel attention to filter high-level semantics and appearance feature. Others find easier form of guidance input instead of trimap or use semantic information. Liu *et al.* [11] propose to use a course mask as guidance input. Chen *et al.* [3]propose to automatic generate trimap using semantic information. Sengupta *et al.* [15] and Lin *et al.* [10] propose to take another photo of the background as auxiliary input. Wei *et al.* [17] propose to use user clicks as foreground and background hints.

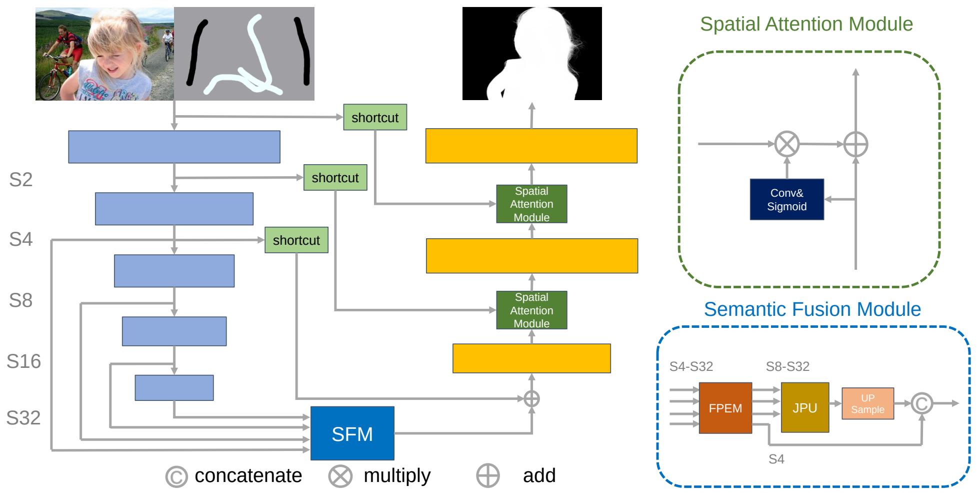

Figure 2: Our proposed network structure. Our network is based on a U-Net architecture [9].

3 Methodology

In this Section, we explain the details of our method. We first elaborate our Progressive Trimap Deformation scheme, and then we introduce our Semantic Fusion Module.

3.1 Progressive Trimap Deformation

To find a simpler form of guidance input than trimap, Wei *et al.* [17] use clickmap to train the network. The clickmap use white or black circles of radius r to indicate foreground or background hint respectively. Intuitively, the definition of the clickmap is similar to trimap. Both of them divide the image into absolute foreground, absolute background and transition area. The difference is that clickmap provides far less guidance information than trimap. Wei *et al.* [17] directly use clickmap to train the network and the result is better than trimap-free methods but worse than trimap-based methods. Based on the above analysis, using clickmap is much easier than trimap but the result is less accurate. This motivates us to train a network that works for all kinds of guidance input between the trimap and clickmap(see Fig. 1). We leave the trade-off between accuracy and difficulty to user, enabling flexible guidance input. The more accurate the guidance input is, the better result it can achieve. To do this, we gradually shrink the foreground and background area of the trimap with the training step increases. In this way, the network can learn to leverage guidance information rather than being restricted to the domains of trimap or clickmap. During the shrinking process, we slowly reduce FG and BG area to make the network better adapt to less guidance information, and this gentle process can make the network converge better. In our method, we shrink the trimap to scribblemap during training. Scribblemaps are more varied in shape compared with clickmaps, which can bring more challenge to the network.

3.1.1 Network Architecture

Our network is based on a U-Net architecture [9] (see Fig. 2) and we use ResNet34 as our backbone. The input RGB image is concatenated with a one-channel guidance map, which can be a trimap, scribblemap or clickmap. In the encoder part, the stride-1, stride-2 and stride-4 output of the encoder are fed into shortcut blocks. The shortcut blocks consist of

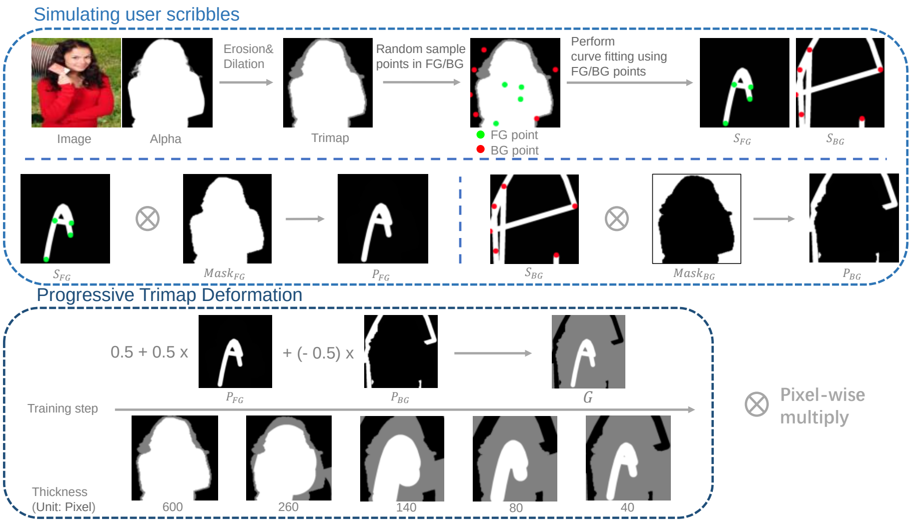

Figure 3: The process of our proposed Progressive Trimap Deformation.

two stride-1 convolution layers followed by batch normalization and Relu. These shortcut blocks are used to process low-level texture features. To leverage multi-scale information from the backbone, stride-4, stride-8, stride-16 and stride-32 output of the encoder are fed to our proposed SFM module to enhance high-level semantic features. In the decoder part, we use dilated convolutions to upsample the feature map. Since the high-level semantic feature indicate foreground category and profiles, we can use it to filtrate redundant low-level texture feature. Thus we use spatial attention module to process low-level features from the shortcut blocks.

3.1.2 Simulating User Scribbles.

As mentioned above, we propose to shrink the trimap to scribblemap during training. To implement this process, we first need to use trimap to generate the target scribblemap(see Fig. 3). In each training step, the same as the previous trimap-based matting methods, we generate the trimap by using dilation and erosion operation on groundtruth alpha matte. Then we randomly select a total of up to 10 points on foreground and background area of each trimap. To avoid the points being too close to each other, we set a threshold of 50 pixels between each two points. After that, we use FG points and BG points to do curve fitting respectively. In detail, for FG points, we iteratively retrieve 3 points from the sampled FG points at a time. Then we use a cubic function to fit the curve through the 3 points. Finally we draw all fitted curves with a certain thickness on one graph to get S_{FG}. Using the same way, we can get S_{BG}. Now we can make sure that most part of the scribbles are in FG or BG area respectively, but we still have to deal with the excess part. So we simply use $Mask_{FG}$ and $Mask_{BG}$ from the trimap to restrict S_{FG} and S_{BG} using function $P_{FG} = S_{FG} * Mask_{FG}$ and $P_{BG} = S_{BG} * Mask_{BG}$. P_{FG} and P_{BG} are the final generated FG scribbles and BG scribbles.

3.1.3 Foreground and Background Deformation.

After the foreground scribble mask and background scribble mask has been generated, we then generate the scribblemap using the function below.

$$G = 0.5 + 0.5 * P_{FG} + (-0.5) * P_{BG} \tag{2}$$

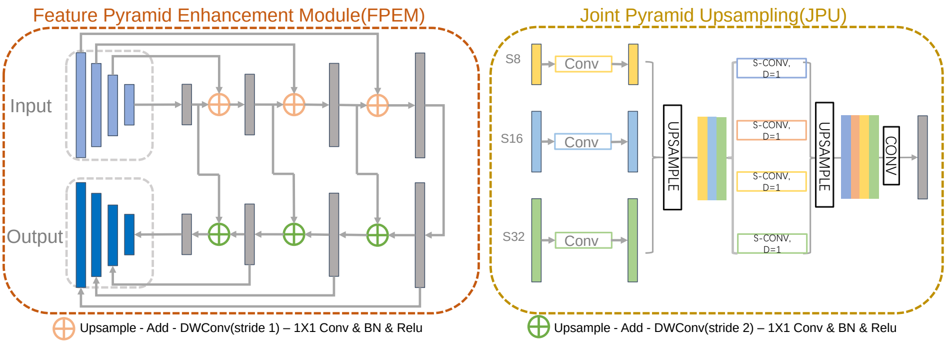
Feature Pyramid Enhancement Module(FPEM)
Input
Output
Joint Pyramid Upsampling(JPU)
S8
S16
S32
Conv
UPSAMPLE
S-CONV, D=1
CONV
Upsample - Add - DWConv(stride 1) – 1X1 Conv & BN & Relu
Upsample - Add - DWConv(stride 2) – 1X1 Conv & BN & Relu

Figure 4: The architecture of FPEM [16] and JPU [18] module.

To gradually shrink the trimap to scribblemap, we can simply change the thickness of the scribbles during training. At the beginning of training, the scribbles are thick enough to cover all FG and BG regions, which makes our guidance map almost the same as the trimap. As the training step increases, we gradually decrease the thickness of the scribbles. As we can see in Fig. 3, the area of FG and BG are gradually reduced. In this way, we gradually give less guidance information to the network during training which enhance the network's ability to distinguish the foreground from the background.

3.1.4 Loss function.

Our loss function is based on [17, 20]. We use three types of loss functions.

$$L(\hat{\alpha},\alpha) = \frac{1}{|K|}\sum_{i\in K}(\hat{\alpha}_i - \alpha_i)^2 + \frac{1}{|T|}\sum_{i\in T}|\hat{\alpha}_i - \alpha_i| + L_{grad}(\hat{\alpha},\alpha) \tag{3}$$

Where K denotes foreground and background region and T denotes transition region. $\hat{\alpha}$ and α indicate the predicted alpha matte and ground-truth. We apply ℓ_2 loss in foreground and background region and apply ℓ_1 loss in transition region. The ℓ_2 loss in foreground and background region improves the prediction of object contour and the ℓ_1 loss in transition area helps the detail prediction. At the beginning of training, ℓ_2 loss is more sensitive than ℓ_1 loss, which makes the network focus on foreground and background areas. As loss begins to converge, ℓ_1 loss will become more sensitive than ℓ_2 loss, the network will focus on the details in transition area.

$$L_{grad}(\hat{\alpha},\alpha) = \frac{1}{|I|}\sum_{i\in I}|\bigtriangledown(\hat{\alpha}_i) - \bigtriangledown(\alpha_i)| \tag{4}$$

L_{grad} is defined as ℓ_1 loss on the gradient of $\hat{\alpha}$ and α. I represents the whole image area. L_{grad} makes the network produce sharper alpha mattes. We compute the gradient by convolving the alpha matte with first-order Gaussian derivative filter.

3.2 Semantic Fusion Module

In matting tasks, advanced semantics from deep levels of the backbone indicate foreground category and profiles, which can guide low-level features to correctly predict the details of the foreground region. In order to extract advanced semantics efficiently, we propose to leverage multi-scale information from the backbone. Another problem is that in order to obtain a high-resolution output alpha matte, the decoder usually use dilated convolution to

Table 1: Results on Composition-1k test set. We use dashlines to divide these methods into three categories, from top to button: trimap-based, trimap-free and ours.

Methods	SAD	MSE	Grad	Conn
KNN Matting[2]	175.4	0.010	124.1	176.4
ClosedForm[8]	168.1	0.091	126.9	167.9
Alphagan[13]	90.9	0.018	93.9	95.2
DIM[19]	48.8	0.008	31.0	50.3
IndexNet Matting[12]	45.8	0.013	25.9	43.7
AdaMatting[1]	41.7	0.010	16.9	-
Context-Aware Matting[7]	35.8	0.082	17.3	33.2
GCA Matting[9]	35.3	0.009	16.9	32.5
GCA Matting(Scribblemap_test)	48.7	0.025	22.9	39.5
GCA Matting(Clickmap_test)	53.2	0.029	24.7	41.4
Late Fusion[20]	58.3	0.011	41.6	59.7
HAttMatting[14]	44.0	0.007	29.2	46.4
Ours(Trimap_test)	30.19	0.0061	13.07	26.66
Ours(Scribblemap_test)	32.86	0.0090	14.18	29.09
Ours(Clickmap_test)	34.67	0.0112	15.45	30.96
Ours(No_guidance_test)	36.36	0.0141	15.23	32.76

upsample the feature map, which brings heavy computation. To solve these two problems, we propose Semantic Fusion Module(SFM) which is composed of two modules, Feature Pyramid Enhancement Module(FPEM) [16] and Joint Pyramid Upsampling(JPU) [18](see Fig. 4). For the first time, we use the FPEM and JPU modules in matting tasks. The FPEM is a cascadable U-shaped module consists of up-scale enhancement and down scale enhancement using the feature pyramid. By fusing the low-level and high-level information, FPEM is able to enhance multi-scale features. In JPU module, four separable convolutions with different dilation rates are used to extract information from multi-level feature map. The JPU module is designed to obtain a feature map similar to the result of using dilated convolution but with less computation cost. In SFM module, we simply cascade FPEM and JPU together(see Fig. 2). Thus we can first use FPEM to extract multi-scale information, then use JPU to upsample the feature maps with less computational complexity and better performance.

4 Experiments

In this section, we report the test results of our method. We compare our method with existing matting methods on DIM dataset [19] and conduct an ablation study of our method.

4.1 Implementation Details

We simply follow GCA-matting[9]'s training strategy and apply their data augmentation methods during training. The data augmentation operations include random affine transformation, random cropping, random jitters and random resize and our PTD scheme. The input resolution of the network is $512 * 512$. During training, we set the initial thickness of the curves to be 800 and gradually decrease to 40 before the step of 530k. The relationship between the thickness and step is an exponential function which makes the thickness decreases slower at the later stage of training to avoid performance drop on trimap. After that, we then train the network with thickness of 40 for 70k steps. The base learning rate is set to $5 * 10^{-4}$ with cosine learning rate scheduler. We set the β_1 and β_2 of the Adam optimizer to 0.5 and 0.999. We use batch size of 10 in total on 2 GPUs.

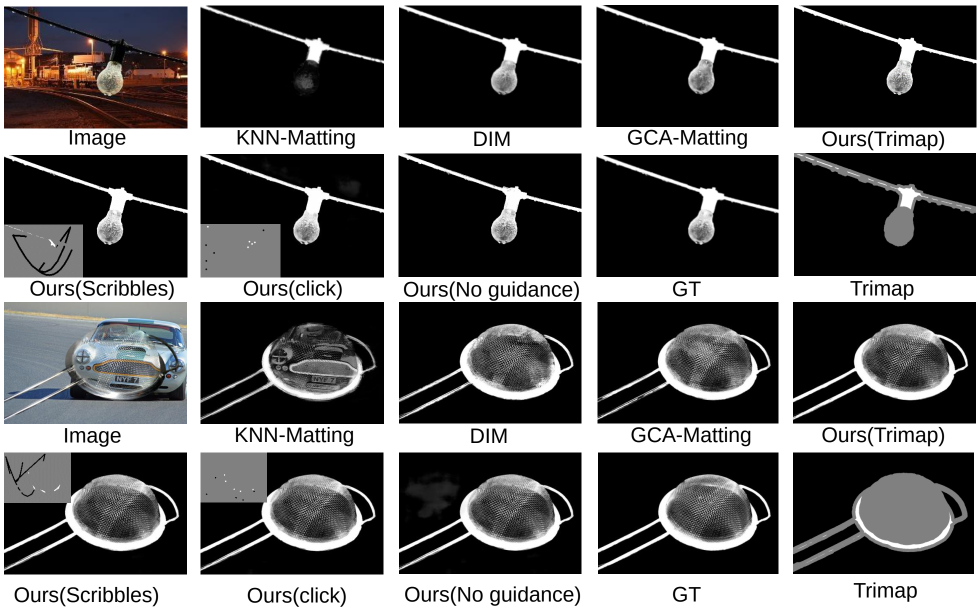

Figure 5: Visual comparisons on Composition-1k

4.2 Experiments on DIM Dataset

The DIM dataset[19] contains 43100 synthetic image with 431 unique foreground objects. The Composition-1k test set consists of 1000 synthetic images. In addition to testing with trimap, we also use the trimap in Composition-1k to generate scribblemap test set and clickmap test set in the same way as in our PTD. To obtain clickmaps, we simply draw circles of diameter 40 with 1 and 0 on foreground and background sampled points respectively. We also test our network without guidance information by using one-channel tensor with a value of 0.5 on each pixel as guidance input.

We follow the previous methods to evaluate the results by using Sum of Absolute Differences(SAD), Mean Squared Error(MSE), Gradient(Grad) and Connectivity(Conn) metrics, and all metrics are to be minimized. As shown in Table. 1, compared with previous trimap-based methods, our method outperforms all previous methods when tested with trimap. When using clickmap as guidance information, our method is still superior to GCA-Matting in the SAD metric. Moreover, the previous trimap-based methods all need an accurate trimap, while ours can get a better result with just a few clicks or scribbles. We also show the results of GCA-matting tested on our generated scribblemap test set and clickmap test set in Table. 1. When both tested on scribblemap or clickmap, our method can outperform GCA-Matting on all metrics. This is largely due to our PTD scheme, while GCA-matting is trained only on trimap. Compared with previous trimap-free methods, ours(No Guidance test) surpasses previous state-of-the-art methods by a large margin. The reason is mainly because previous trimap-free methods didn't use any guidance information in the training stage while ours gradually reduce guidance information.

When comparing our method using different guidance input, we can find that the results get worse as the guidance information decreases. The experiments prove that our method

Table 2: Our method tested using different Scribblemap test sets and Clickmap test sets.

Scribblemap					Clickmap				
Num	SAD	MSE	Grad	Conn	Num	SAD	MSE	Grad	Conn
1	32.86	0.0090	14.18	29.09	1	34.67	0.0112	15.45	30.96
2	33.08	0.0093	14.22	29.32	2	34.45	0.0109	15.24	30.74
3	32.70	0.0086	14.10	28.91	3	34.46	0.0110	15.27	30.74
σ	0.1557	0.00029	0.0498	0.1678	σ	0.1014	0.00012	0.0927	0.1037

Table 3: Ablation study of our method. Baseline: ResNet34 U-net with FPN structure.

Num	Method	SAD	MSE	Grad	Conn
1	Baseline + Trimap_train + Trimap_test	31.87	0.0068	13.48	28.55
2	Baseline + FPEM + Trimap_train + Trimap_test	32.94	0.0069	14.42	29.85
3	Baseline + JPU + Trimap_train + Trimap_test	33.48	0.0072	14.89	30.49
4	Baseline + SFM + Trimap_train + Trimap_test	30.99	0.0064	12.81	27.54
5	Baseline + SFM + Trimap_train + Scribblemap_test	44.33	0.0203	16.40	39.12
6	Baseline + SFM + Trimap_train + Clickmap_test	48.53	0.0250	17.84	42.92
7	Baseline + SFM + Trimap_train + No_guidance_test	52.97	0.0323	18.56	47.43
8	Baseline + PTD + Trimap_test	32.86	0.0069	14.01	29.92
9	Baseline + PTD + Scribblemap_test	35.96	0.0118	14.79	32.92
10	Baseline + PTD + Clickmap_test	37.44	0.0138	15.41	34.50
11	Baseline + PTD + No_guidance_test	38.67	0.0165	15.34	35.91
12	Baseline + SFM + PTD + Trimap_test	30.19	0.0061	13.07	26.66
13	Baseline + SFM + PTD + Scribblemap_test	32.86	0.0090	14.18	29.09
14	Baseline + SFM + PTD + Clickmap_test	34.67	0.0112	15.45	30.96
15	Baseline + SFM + PTD + No_guidance_test	36.36	0.0141	15.23	32.76

can enhance the network's ability to make full use of the guidance information to distinguish the foreground from the background. The more accurate the guidance input is, the better result it can achieve. Even without any guidance information, our method still shows great robustness. We provide some comparison results in Fig. 5. We also test our method by using real images, the results are shown in Fig. 6.

We also test our method using different generated scribblemap test sets and clickmap test sets of Composition-1k. The results are in Table. 2. The scribblemaps in different scribblemap test sets of the same image have the same thickness but different shapes, so do clickmaps. The standard deviation of all metrics in "scribblemap" column are very small, and "clickmap" column has the same phenomenon, proving that our method is robust to guidance maps of the same level(e.g. scribblemap, clickmap) but with different shape.

4.3 Ablation Studies

To show the effectiveness of our SFM and PTD, we conduct ablation studies in Table. 3. Compare 1,2,3 and 4, we can find that the SFM module can improve the network's performance on trimap, while using only FPEM or JPU will make the performance worse. Compare 8,9,10,11 and 12,13,14,15, we can also find that SFM can enhance the performance when training with PTD. Compare 4 to 12, the results show that when both tested on trimap, training with PTD is slightly better than training with trimap. This is probably because our PTD scheme can enhance the network's ability to leverage guidance information. Moreover, when testing with scribbles, clicks and no guidance information, our PTD scheme(13,14,15) can achieve far better results than training with trimap(5,6,7). The results show that our PTD scheme can ensure the performance on trimap, and can still achieve high accuracy with reduced guidance information. Note that we only shrink the trimap to scribblemap, but the performance on clickmap and no guidance is also enhanced. This is probably because during the shrinking process, the gray areas continue to expand, and the network is forced to leverage all given FG and BG information and improve the performance in gray areas. When

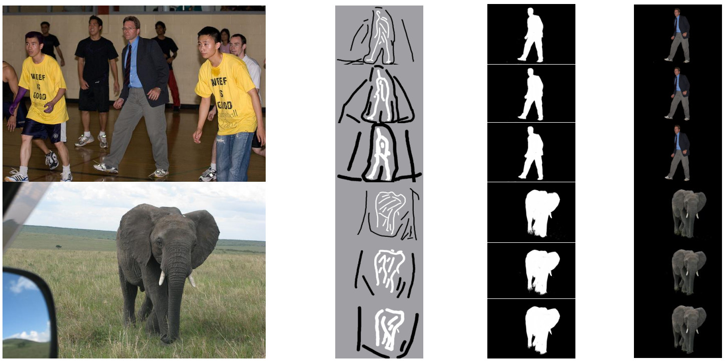

Figure 6: Visual results of real images. Each column from left to right: input images, guidance map, predicted alpha matte, foreground object result.

Table 4: Results of our PTD applied on GCA and DIM matting. Note that the GCA and DIM in this table are based on our own training results.

	GCA				DIM			
	SAD	MSE	Grad	Conn	SAD	MSE	Grad	Conn
Trimap_test	35.03	0.0086	16.54	30.78	56.77	0.0166	27.34	50.37
Scribblemap_test	46.92	0.0205	19.92	36.95	81.98	0.0523	33.59	65.90
Clickmap_test	52.57	0.0251	21.35	39.46	87.02	0.0596	35.17	68.83
No_guidance_test	60.15	0.0359	22.86	44.06	90.10	0.0654	35.52	71.32
	GCA+PTD				DIM+PTD			
	SAD	MSE	Grad	Conn	SAD	MSE	Grad	Conn
Trimap_test	32.71	0.0082	14.01	27.78	52.50	0.0150	29.42	46.11
Scribblemap_test	34.08	0.0111	13.93	28.65	61.80	0.0298	31.07	51.79
Clickmap_test	37.70	0.0172	14.90	30.98	64.12	0.0336	31.83	53.08
No_guidance_test	41.05	0.0242	15.63	32.57	65.67	0.0370	31.63	54.16

tested with less guidance information, the network will still make full use of the given hints and try to predict the alpha matte in gray areas.

We also apply our PTD scheme on other trimap-based methods to enhance their performance on multiple kinds of guidance inputs. In Table. 4, we show the results of applying our PTD on GCA and DIM matting. Experiments show that our PTD can improve the performance of GCA and DIM on trimap, and significantly enhance the performance on scribblemap, clickmap and no extra guidance.

5 Conclusion

In this paper, we propose a matting method that use flexible guidance input as user hint. Our method can use trimap, scribblemap or clickmap as guidance information or even work without any guidance input. To achieve this, we introduce our Progressive Trimap Deformation scheme and Semantic Fusion Module. The experiments show that our method can achieve state-of-the-art results compared with existing trimap-based and trimap-free methods.

Acknowledgments. This work was supported by the National Natural Science Foundation of China, No.61871262.

References

[1] Shaofan Cai, Xiaoshuai Zhang, Haoqiang Fan, Haibin Huang, Jiangyu Liu, Jiaming Liu, Jiaying Liu, Jue Wang, and Jian Sun. Disentangled image matting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8819–8828, 2019.

[2] Qifeng Chen, Dingzeyu Li, and Chi-Keung Tang. Knn matting. *IEEE transactions on pattern analysis and machine intelligence*, 35(9):2175–2188, 2013.

[3] Quan Chen, Tiezheng Ge, Yanyu Xu, Zhiqiang Zhang, Xinxin Yang, and Kun Gai. Semantic human matting. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 618–626, 2018.

[4] Yung-Yu Chuang, Brian Curless, David H Salesin, and Richard Szeliski. A bayesian approach to digital matting. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 2, pages II–II. IEEE, 2001.

[5] Xiaoxue Feng, Xiaohui Liang, and Zili Zhang. A cluster sampling method for image matting via sparse coding. In *European Conference on Computer Vision*, pages 204–219. Springer, 2016.

[6] Eduardo SL Gastal and Manuel M Oliveira. Shared sampling for real-time alpha matting. In *Computer Graphics Forum*, volume 29, pages 575–584. Wiley Online Library, 2010.

[7] 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*, pages 4130–4139, 2019.

[8] Anat Levin, Dani Lischinski, and Yair Weiss. A closed-form solution to natural image matting. *IEEE transactions on pattern analysis and machine intelligence*, 30(2):228–242, 2007.

[9] Yaoyi Li and Hongtao Lu. Natural image matting via guided contextual attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11450–11457, 2020.

[10] Shanchuan Lin, Andrey Ryabtsev, Soumyadip Sengupta, Brian Curless, Steve Seitz, and Ira Kemelmacher-Shlizerman. Real-time high-resolution background matting. *arXiv preprint arXiv:2012.07810*, 2020.

[11] Jinlin Liu, Yuan Yao, Wendi Hou, Miaomiao Cui, Xuansong Xie, Changshui Zhang, and Xian-Sheng Hua. Boosting semantic human matting with coarse annotations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8563–8572, 2020.

[12] Hao Lu, Yutong Dai, Chunhua Shen, and Songcen Xu. Indices matter: Learning to index for deep image matting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3266–3275, 2019.

[13] Sebastian Lutz, Konstantinos Amplianitis, and Aljosa Smolic. Alphagan: Generative adversarial networks for natural image matting. *BMVC*, 2018.

[14] 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*, pages 13676–13685, 2020.

[15] Soumyadip Sengupta, Vivek Jayaram, Brian Curless, Steven M Seitz, and Ira Kemelmacher-Shlizerman. Background matting: The world is your green screen. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2291–2300, 2020.

[16] Wenhai Wang, Enze Xie, Xiaoge Song, Yuhang Zang, Wenjia Wang, Tong Lu, Gang Yu, and Chunhua Shen. Efficient and accurate arbitrary-shaped text detection with pixel aggregation network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8440–8449, 2019.

[17] Tianyi Wei, Dongdong Chen, Wenbo Zhou, Jing Liao, Hanqing Zhao, Weiming Zhang, and Nenghai Yu. Improved image matting via real-time user clicks and uncertainty estimation. *arXiv preprint arXiv:2012.08323*, 2020.

[18] Huikai Wu, Junge Zhang, Kaiqi Huang, Kongming Liang, and Yizhou Yu. Fastfcn: Rethinking dilated convolution in the backbone for semantic segmentation. *arXiv preprint arXiv:1903.11816*, 2019.

[19] Ning Xu, Brian Price, Scott Cohen, and Thomas Huang. Deep image matting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2970–2979, 2017.

[20] Yunke Zhang, Lixue Gong, Lubin Fan, Peiran Ren, Qixing Huang, Hujun Bao, and Weiwei Xu. A late fusion cnn for digital matting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7469–7478, 2019.

[21] Yuanjie Zheng and Chandra Kambhamettu. Learning based digital matting. In *2009 IEEE 12th international conference on computer vision*, pages 889–896. IEEE, 2009.