Master's Thesis in Graduate School of
Library, Information and Media Studies

Saliency-based Trimap Generation for Image Matting

March 2021

201921639

Taniguchi Masaki

Saliency-based Trimap Generation for Image Matting
イメージマッティングのための顕著性マップに基づくtrimap生成手法

Student No.: 201921639
氏名: 谷口　正樹
Name: Taniguchi Masaki

Alpha matting is the task of splitting an image into the foreground and background on a very fine scale. In many of the existing implementations, an intermediate representation called a trimap is constructed by user inputs. Although trimaps are created on a much coarser scale than alpha mattes, the process of constructing them is still costly. This work proposes a generic neural network for a trimap generation that utilizes saliency map detection. Our model multi-modally learns a saliency map and a trimap, enabling it to focus on generating a more accurate trimap in the area with higher salience. Experiments showed that our model could generate trimaps that are almost identical to manually generated ones. The method can also be easily combined with existing alpha matting algorithms.

Principal Academic Advisor: Kei WAKABAYASHI
Secondary Academic Advisor: Masahiko MIKAWA

Saliency-based Trimap Generation for Image Matting

Taniguchi Masaki

Graduate School of Library, Information and Media Studies

University of Tsukuba

March 2021

Contents

List of Figures

Chapter 1

Introduction

In recent years, the task of separating the foreground and the background from a photograph has become increasingly important, since video production and graphic creation has become widely practiced. This task is called *alpha matting*, which the aim is to assign a probability to each pixel representing how likely it is to be a part of the foreground. It is closely related to semantic segmentation tasks but needs to be performed at a much more sufficient scale. Alpha matting is usually carried out in two steps: identifying the area of the main object and separating the object from the background.

In many implementations, the area of the main object is represented using a *trimap*, as illustrated in the second column in Figure 1.1. Trimaps have a data format where each pixel is classified into three classes: foreground, background, and ambiguous (or unknown). In a typical alpha matting system, trimaps must be created manually. It is much easier than manual alpha matting since whenever the boundary between the foreground and the background is too intricate, the user can label the whole area as ambiguous.

Trimap generation somewhat resembles semantic image segmentation, but two tasks are different. Areas labeled as ambiguous in trimap generation do not correspond to physical entities. In fact, ambiguous areas are not extracted as image segments in the regular semantic image segmentation task. Many algorithms of semantic image segmentation are optimized for extracting physical objects from an image, and they are unsuitable for trimap generation. The second step in a common system for alpha matting is to produce a grayscale image that separates the main object and the background on the basis of the original image and the trimap generated from it. The output of this step is called an alpha matte. In an alpha matte, each pixel takes a probability value between 0 and 1, representing how likely it is to be part of the foreground of the image. The following equation expresses image composition using an alpha matte.

$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i \quad \alpha_i \in [0, 1], \tag{1.1}$$

where i is an index representing a pixel. For pixel i, α_i is the value of the alpha matte, I_i is the color vector of the final image, F_i is the color vector of the first image, and B_i is the color vector of the second image. With this transformation, the foreground of the first image is placed on top of the second image.

There are many semi-automatic trimap generation algorithms, but most of them require some user intervention. In most cases, these methods ask the user to indicate where the

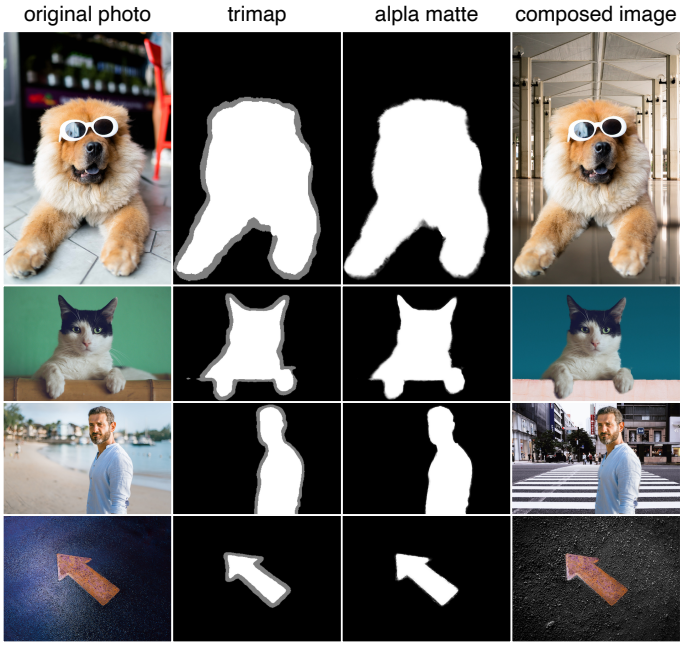

Figure 1.1: Trimaps, alpha mattes, and composed images generated by the proposed method. It can accurately detect the contours of people and non-human animals, even for non-living objects.

main object is located in the image. With that information, the system can automatically find the extent of the object and its ambiguous surrounding area. However, such intervention becomes too cumbersome when processing numerous images contained in a video. Full-automatic trimap generation is a relatively unexplored topic, and it is of significant importance in the field of image processing.

In this paper, we propose to automate the whole process by finding the most attention-attracting parts of images using a recently developed saliency map detection algorithm. The algorithm produces a binary image indicating the areas in an original image that are likely to attract human attention. The accuracy of saliency map detection has drastically improved in recent years due to the development of generative model using deep neural networks. To our knowledge, this paper is the first attempt to introduce saliency map detection using deep learning to the task of trimap generation. Our method enables to conduct alpha matting on videos fully-automatically. This can benefit a number of applications such as movie production and desktop video editing. Figure 1.1 shows trimaps generated by our method, alpha mattes generated from them using deep image matting (DIM) [1], and final overlaid images. As can be seen from this figure, our method can automatically crop the most prominent objects in images irrespective of types of objects.

There are at least two merits in generating a trimap first and then converting it into an alpha matte, instead of directly generating an alpha matte. One is that trimaps are much easier to edit than alpha mattes because pixels in trimaps take only three values, and also the user does not need to edit it on a fine scale. When the user is not satisfied with the final alpha matte, the trimap can be edited and the change is reflected in the final alpha matte. In other words, a trimap provides an intermediate representation for handling the alpha matte. It is much more difficult to edit alpha mattes since the user must carefully assign an appropriate grayscale value to each pixel at a fine granularity. Another benefit

of generating a trimap is that there are already numerous applications that can convert it to an alpha matte. Any of these applications can be used to obtain an alpha matte from a trimap, and the user can choose the one that is most suited for his/her purpose. The user will have more options than directly generating alpha mattes.

In this work, we make the following contributions:

- We propose a neural network model and an accompanying loss function that detect salient areas from an image and generate a trimap corresponding to the main object.
- We trained the proposed model using publicly-available datasets and achieved high performance in trimap generation tasks.
- The method is fully automatic and fast enough to be used for alpha matting in the video editing scene.

The rest of the paper is organized as follows. Chapter 2 discusses related work. Chapter 3 describes the proposed method, and Chapter 4 gives the results of our experiments. Chapter 5 concludes the paper.

Chapter 2

Related work

In this section, we describe existing work on alpha matting, trimap generation, and saliency map detection. We also discuss semantic image segmentation in the trimap generation section.

2.1 Alpha matting

The sampling-based approaches [2, 3, 4] and propagation approaches [5] are most widely used for alpha matting, but many methods using deep learning have recently been proposed. Shen et al. proposed a method targeting portrait photographs [6]. They focused on images where the person's upper body has similar postures. From such an image, their method creates a shadow-like outline of the person. An alpha matte is created using the extracted outline instead of using a trimap. Cho et al. proposed a method in which rough matting results are obtained by a non-neural network process, which are refined using a neural network [7]. However, this network can not directly learn the alpha matte from an image and trimap. Xu et al. proposed a convolutional neural network (CNN)-based encoder-decoder network that predicts the alpha matte from the image and trimap input by end-to-end training in DIM [1]. They achieved state-of-the-art results with higher accuracy compared with a manually constructed alpha matte. Their model consists of two parts. The first part is an encoder-decoder network that predicts the alpha matte from the original image and trimap. The second part is a simple network that converts the rough alpha matte generated in the previous step into a more accurate one. They also provided a dataset consisting of original images, trimaps, and ground-truth alpha mattes. Many other methods which train by end-to-end training that use trimap as input have been published in recent years [8, 9, 10, 11]. Several methods have also been proposed to separate the foreground and background directly from the photo without using trimap as input [12, 13]. However, these methods have a fundamental problem in that they do not allow the user to make corrections when the model chooses a wrong object.

2.2 Trimap generation

In many existing methods, the main object is recognized and its outline is then expanded to create trimap. Hsieh et al. used images that are roughly separated into foreground and background. The trimap is generated by enlarging the outline of the object on the basis

of the texture of the image [14]. Al-Kabbany et al. used the Gestalt laws of perceptual organization to identify objects that are likely to be recognized by humans [15]. Their method can generate a trimap from images without requiring any other input from the user. The method developed by Cho et al. generates a binary segmentation image by using depth information [16]. An accurate trimap was created using the Kullback-Leibler divergence between each background and foreground image. Gupta et al. used a superpixel image generated by simple linear image clustering (SLIC) instead of a binary segmentation image as used in Cho et al. [17, 18]. Semantic Human Matting (SHM) by Chen et al. [19] uses a fusion module that combines two encoder-decoder networks, a trimap generation stage, and an alpha matting stage, to enable end-to-end training.

Semantic Human Matting and our work incorporate semantic image segmentation techniques into the model for generating trimaps. This is because the goal of these two tasks is the same: to predict the class of each pixel. The following is a list of previous studies on semantic image segmentation. Semantic image segmentation is a popular research topic, and many methods and datasets are provided for this task [20, 21, 22, 23, 24]. Common practice had been to learn features by using random forests or Bayesian models, but more recently, it has become increasingly popular to use deep neural networks [25, 26, 27, 28]. An example of semantic image segmentation that relies only on color images is the pyramid scene parsing network developed by Zhao et al. [29]. They proposed a pyramid pooling module to capture features with different resolutions and achieved state-of-the-art performance on multiple benchmarks. We used their PSP-Net as the baseline method in our comparative experiment.

2.3 Saliency map detection

Saliency map detection aims to find important parts of a natural image that humans pay attention to. This task is used as a pre-processing stage for many other image recognition applications. Examples include semantic image segmentation and image retrieval. In classical approaches of saliency map detection, hand-crafted features have been used extensively. These methods make use of color contrast [30, 31] or background prior [32] and can produce results that better capture local features. However, it was difficult to obtain high-level and semantic information with these classical approaches, and now neural network-based methods have become increasingly popular. In addition, several large datasets are being developed to go along with this trend. [33, 32, 34, 35]

Currently, various CNN-based networks have been proposed, but many of them have modules for looking at multi-scale features. Liu et al. [36] and Li and Yu [37, 38] proposed models with pixel-wise and super-pixel-wise receptive fields to capture both local and global features. A number of methods have predicted saliency maps hierarchically, from global views to finer local views, using a U-Net-based encoder-decoder network [39, 40, 41]. The pyramid feature attention network (PFAN) for saliency detection of Zhao et al. [42] is an efficient neural network-based saliency detection method. They introduced a new feature extraction module and two attention modules to get multi-scale features. We also developed a method to generate trimaps on the basis of this PFAN saliency map detection method.

Chapter 3

Method

In our proposed method, two types of images, namely a trimap and a saliency map, are trained multi-modally during the training phase. The following subsections describe its components.

3.1 Network structure

Figure 3.1 shows our proposed network. The main contribution of this work is to propose a network having multiple outputs, namely the saliency map and the trimap. It is illustrated in the upper half of the figure. The losses from both outputs contribute to training. They enable the network to learn to focus on regions with more salience and to generate an appropriate trimap in the focused region. The network is supplied with images, each represented as a tensor having the shape of (w, h, c), where h is the image height, w is the image width, and c is the number of color channels. Our proposed network has two outputs. The first is a trimap represented by one-hot vectors. For each pixel in the image, there is a one-hot vector. For a pixel at location (i, j), $I_{ij} = [1, 0, 0]$ represents that the pixel is in the background. $I_{ij} = [0, 1, 0]$ represents that it is an ambiguous pixel. Finally, $I_{ij} = [0, 0, 1]$ represents that it is in the foreground. We call this representation a *one-hot trimap* and it has the shape of $(w, h, 3)$. The second output is a saliency map. Here, each pixel takes a value between 0 and 1, similar to the format of a grayscale image and has the shape of (w, h).

The goal of the trimap generator is to classify each pixel into three classes, namely foreground, background, and ambiguous. This task is similar to semantic image segmentation. As in saliency map detection, we wanted to capture the subject for a wide range of domain photos, so we used the PFAN [42]. for this part of the network. It is shown in the bottom half of Figure 3.1. PFAN consists of three modules. Context-aware pyramid feature extraction is a module that extracts invariant features in terms of scale, shape, and location using an atrous convolution. Outputs from three high-level feature layers from the feature extractor are input to multiple convolution layers having various dilations and padding, and then concatenated. The two attention modules, channel-wise attention and spatial attention, are designed to reduce redundant information and focus on critical information. We applied channel-wise attention to the high-level features and spatial attention to the high- and low-level features, respectively, in the same way as in the PFAN network.

In the original PFAN paper, the pre-trained VGG16[43] was used for the feature extractor,

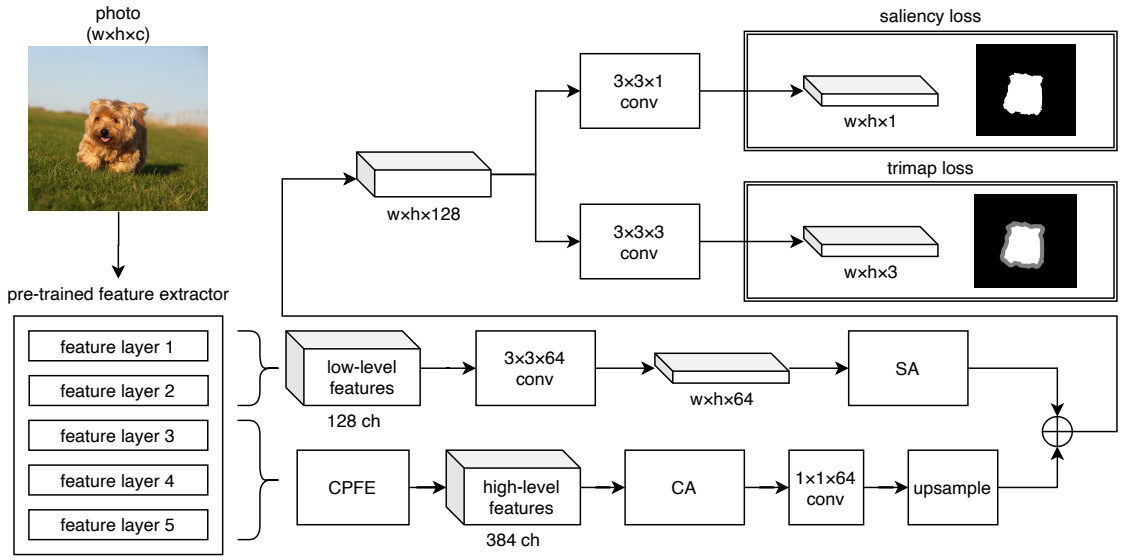

Figure 3.1: Network structure of the proposed model. The input image is put through feature extractors. Low- and high-level features are processed through different paths and then merged. The path is split to produce two output images and sent to two terms (the saliency loss and trimap loss) in the loss function.

but we have extended this module to be replaceable with several other models.

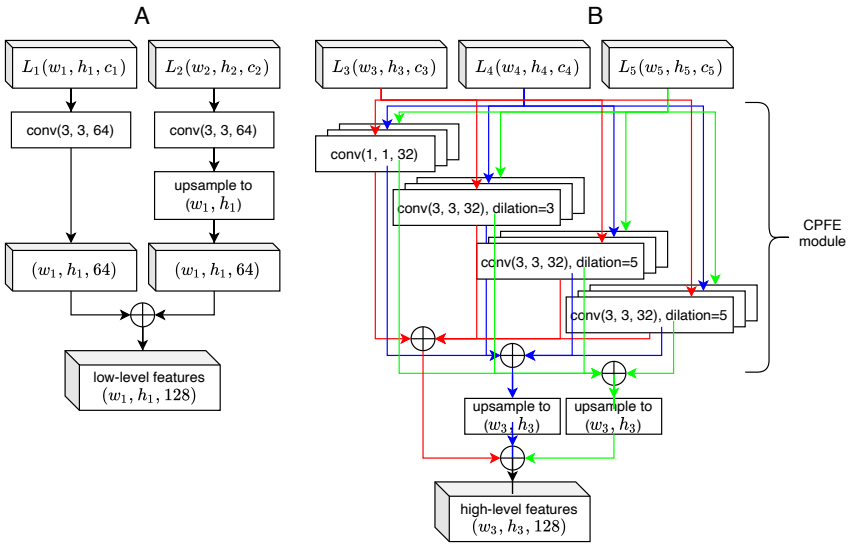

Figure 3.2: Processing of low- and high-level features in the proposed network.

The shapes of the low- and high-level features in our network are different for each feature extractor and are also slightly different from the original implementation of PFAN. If the five feature layers from the feature extractor are $L_n (n \in [1..5])$ in order of shallowness, and the shape of each feature layer is (w_n, h_n, c_n), then the low-level and high-level features (created through the CPFE module) are created as shown in Figure 3.2-A and B.

3.2 Loss function

We use three loss functions for training in the trimap stage: cross-entropy loss for the trimap, cross-entropy loss for the saliency map, and edge-hold loss. The cross-entropy loss is commonly used in saliency map detection and is defined as the cross-entropy between the final predicted saliency map and the ground-truth saliency map, namely

$$L_S = -\sum_{i=0}^{size(Y)} (\alpha_s Y_i \log(P_i) + (1-\alpha_s)(1-Y_i)\log(1-P_i))$$

where Y is the ground-truth saliency map and P is the saliency map predicted from the network. α_s is a balance parameter that is calculated from the ground truth of the training set; we set α_s to 0.528. The cross-entropy loss mainly considers the difference of positions between objects, so the difference of intricate parts of the object boundaries does not contribute much. To solve this problem, PFAN uses the edge-hold loss, which uses the Laplace operator to calculate the cross-entropy loss after extracting edges from both saliency maps. This enables the network to focus more on correcting errors around boundaries. Convolution with the Laplace operator is defined as

$$\Delta\widetilde{f} = \mathrm{abs}(\tanh(\mathrm{conv}(f, K_{\mathrm{Lap}})))$$

where K_{Lap} is the Laplace kernel. We used a 3×3 matrix,
$((-1,-1,-1),(-1,8,-1),(-1,-1,-1))$, for K_{Lap}.

The edge-hold loss function is defined as

$$L_B = -\sum_{i=0}^{\mathrm{size}(Y)} (\Delta Y_i \log(\Delta P_i) + (1-\Delta Y_i)\log(1-\Delta P_i))$$

The overall loss that we used for training is

$$L_{\mathrm{overall}} = \beta(\alpha L_S + (1-\alpha)L_B) + (1-\beta)L_T$$

where L_T is the cross-entropy loss of trimap prediction. We set α to 0.5 and β to 0.4 in the experiment.

Chapter 4

Experiment

To evaluate the proposed method, we conducted experiments using benchmark datasets commonly used for alpha matting and saliency map detection. In the following subsections, we describe the datasets used for training, the details of implementation, and the result of experiments.

4.1 Dataset and prepossessing

We used the DUTS image dataset (DUTS) [34] to train the proposed network. This dataset is a commonly-used benchmark dataset for saliency map detection tasks. All images come from the ImageNet DET training/evaluation/test datasets and also the SUN dataset. In addition, salient areas are annotated for each images manually. We used 10,553 training images and 1,000 test images from this dataset. The data contains natural images and their saliency maps, but does not contain ground-truth trimaps nor ground-truth alpha mattes. Therefore, we generated pseudo-ground-truth trimaps from the saliency maps.

The ground-truth saliency map was expanded and contracted, and the pixels that did not match between the two images were considered as ambiguous pixels. Kernel size used in this erosion / dilation process was randomly selected for each iteration between 1 and 10. We overlaid the ambiguous part on the original binary mask and used it as a trimap, as indicated in Figure 4.1. In the training phase, trimaps were converted into the one-hot-trimap format.

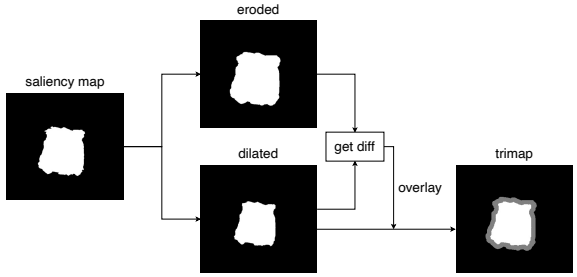

Figure 4.1: Steps in creating a pseudo-ground-truth trimap from a saliency map.

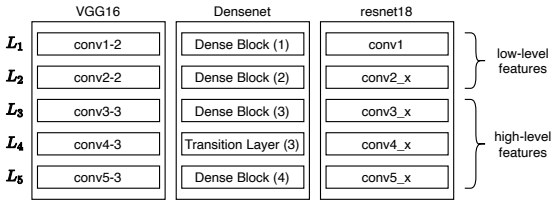

Figure 4.2: Backend feature extractors and layers used as low- and high-level features.

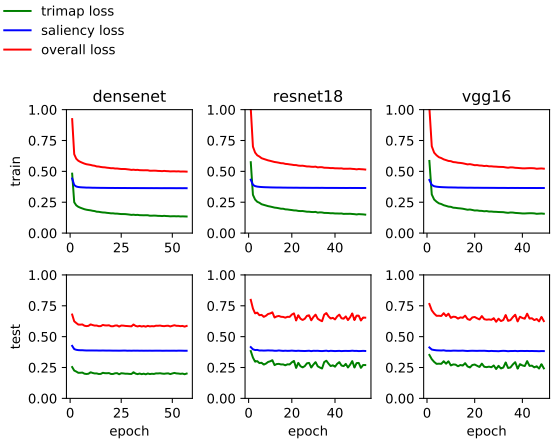

Figure 4.3: Learning curves for different network structures.

4.2 Hyperparameters for training

We implemented our proposed method using Python programming language and PyTorch [44] framework. We prepared three models for the experiment, each having a unique pre-trained feature extractor as a backend, namely Resnet18, VGG16, and Densenet. All feature extractors are pre-trained by the ImageNet classification task. Figure 4.2 illustrates feature layers used in the model. The names of each layer in the figure conform to implementations in torchvision for VGG16 [45], Densenet [46], and Resnet [47].

We used Adam as the optimizer, setting the learning rate to 10^{-5}. We also used early stopping in test loss in all models to determine when to terminate the training. Figure 4.3 illustrates the learning curves for different network structures.

4.3 Results

Two steps are required for image matting by the proposed method: generating trimaps by our proposed network and generating alpha mattes using the predicted trimap. We used DIM [1] and its PyTorch implementation [48] to implement the latter part.

4.3.1 Performance comparison

We used PSPNet [29] as the basic structure of the baseline models. It is a network that achieved high accuracy in the field of semantic segmentation. We prepared several models having different backend feature extractors. We trained the baseline models using the same loss as the proposed method. Since there is no ground-truth alpha mattes in the saliency

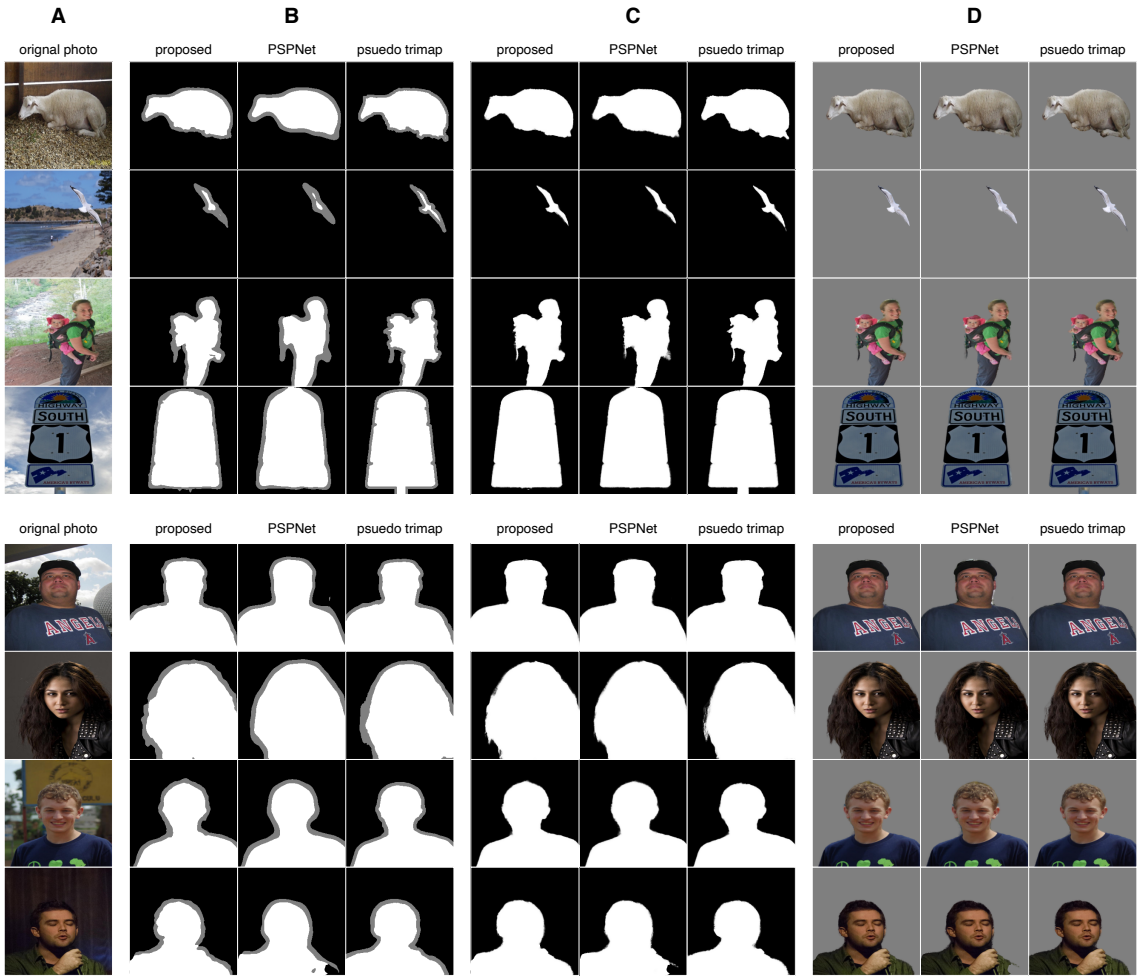

Figure 4.4: Visual comparison of original and generated images. A: Original photographs, B: Trimaps generated by each method. C: Alpha mattes generated by each trimap and DIM. D: Composition results by using each alpha matte.

map detection datasets, we generated pseudo-ground-truth data. First, we created pseudo-trimaps from the ECSSD [33] dataset using the same approach as in the training stage. Then we applied DIM to create pseudo-ground-truth alpha mattes. The sum of absolute differences (SAD), mean square error (MSE), gradient error, and connectivity error were measured with respect to the pseudo-ground-truth alpha matte. These four metrics are commonly used in the field of image matting; SAD and MSE directly correlate with the distance from ground-truth data in the image matting data; gradient error and connectivity error are metrics proposed by Rhemann et al. [49] to reflect the visual quality of the alpha matte when annotated by a human. When calculating these metrics, we normalized both the predicted alpha matte and ground-truth from 0 to 1, and computed metrics for pixels labeled as "ambiguous" in the corresponding trimap.

Table 4.1 summarizes the results. The pseudo-ground-truth alpha mattes were generated by mechanically expanding trimap boundaries, so their details may differ from real alpha mattes. Table 4.1 compares the performance of extracting main objects from images. It shows that for the ECSSD dataset, the proposed method performed better than other methods in each of the metrics except for the connectivity error when using Resnet18.

Method	SAD	MSE ($\times 10^{-2}$)	Gradient ($\times 10^{3}$)	Connectivity
VGG16				
PSPNet	17.560	9.811	3.608	1.356
proposed	**13.680**	**7.490**	**2.799**	**1.178**
Resnet18				
PSPNet	16.572	9.326	3.489	**1.146**
proposed	**12.681**	**6.825**	**2.543**	1.228
Densenet				
PSPNet	15.043	8.475	3.172	1.101
proposed	**12.580**	**6.844**	**2.550**	**1.151**

Table 4.1: Performance results for the ECSSD dataset.

Method	SAD	MSE ($\times 10^{-2}$)	Gradient ($\times 10^{3}$)	Connectivity
VGG16				
PSPNet	23.179	13.792	5.046	0.538
proposed	**11.654**	**6.769**	**2.446**	**0.215**
Resnet18				
PSPNet	21.252	12.686	4.651	0.304
proposed	**15.870**	**9.424**	**3.440**	**0.190**
Densenet				
PSPNet	16.528	9.848	3.611	**0.138**
proposed	**15.176**	**9.011**	**3.271**	0.152

Table 4.2: Performance results for the Matting Human Datasets.

We also conducted trimap generation using the Matting Human Dataset [50]. Note that the ground-truth alpha mattes in the Matting Human Datasets are manually created, unlike alpha mattes in ECSSD that are mechanically generated from pseudo-trimaps. The former is therefore more suitable to compare how different methods generate fine details of alpha mattes. However, one drawback of the Matting Human Datasets is that it contains human portraits only.

Table 4.2 summarizes the result for the Matting Human Dataset. As indicated in the table, the model adopting the PFAN network scored better than the PSP model on each of the metrics except for the connectivity error when using Densenet. This result, together with that of ECSSD, indicates that our method performs better than baseline methods in terms of both recognizing main objects and matting in details.

4.3.2 Visual comparison

Figure 4.4 compares trimaps generated by the proposed method, those generated by PSPNet using ResNet18, and the pseudo-ground-truth masks. From each trimap, an alpha matte was generated using DIM. Foregrounds extracted by alpha mattes are presented also. Images in the upper rows are from the ECSSD dataset, and those in the lower rows are from the Matting Human Datasets. The result shows that our method can generate trimaps nearly equal to pseudo-trimaps generated using correct saliency maps. The first row shows that both the proposed method and PSPNet can extract the overall image of the object with reasonable quality. However, the trimaps generated by PSPNet have smoothed outlines and do not reflect the detailed shapes of the objects. The difference is also perceivable after

Method	SAD	MSE ($\times 10^{-2}$)	Gradient ($\times 10^{3}$)	Connectivity
ECSSD				
no SA	13.313	7.159	2.682	1.296
no CA	15.293	8.472	3.173	1.161
no L_T	14.398	8.000	2.971	**1.140**
proposed	**12.681**	**6.825**	**2.543**	1.228
Human Matting Dataset				
no SA	14.010	8.211	3.013	0.256
no CA	17.935	10.678	3.906	0.226
no L_T	20.947	12.568	4.574	0.242
proposed	**15.870**	**9.424**	**3.440**	**0.190**

Table 4.3: Contributions from each component. We used ECSSD and Human Matting Dataset as a dataset and Resnet18 as a feature extractor.

matting them with DIM. The second and third rows show that the trimaps generated by the proposed method have finer edges and tend to capture local features more. The fourth row shows that the proposed method can generate accurate trimaps even for non-living objects. The results from the Matting Human Dataset show a smaller difference between the proposed method and PSPNet than the results from the ECSSD, indicating that both methods can capture the contours of portrait images successfully. However, the proposed method tends to be more capable of tracing local features. In addition, a number of the contours are missing in the results generated by PSPNet.

4.3.3 Contributions from each component

We trained the model by omitting each component and evaluated it in the same way as mentioned earlier for ECSSD to see the extent to which the components of the proposed method have an impact on the final prediction results. The results are listed in Table 4.3. We used the Resnet18 backend for this experiment.

When compared with the models having components removed, the proposed method scored better for all criteria except for the connectivity error. It indicates the effectiveness of multi-modal training and also the use of PFAN as the network structure.

4.3.4 Processing of high-resolution images

In the experiments, we trained the neural network model using images consisting of 400×400 pixels. In practice, the user would want to process images of various sizes. The network itself can accept images of different sizes without downsampling, but we expected the quality would differ if the sizes of the training images and test images were different. We conducted experiments to investigate how the size of the image affects the quality of the generated trimap and the computation time.

Figure 4.5 shows the results of generating trimaps of different resolutions using a network trained by images of 400×400 pixels. It shows that there are more errors when generating higher resolution images such as the one having 600×600 pixels. As shown in Figure 4.7, the result suggests that when processing high-resolution images, it would be better to downsample the original image to the size of the training images, generate a trimap using the proposed method, upsample the trimap, and perform image matting. Since image matting

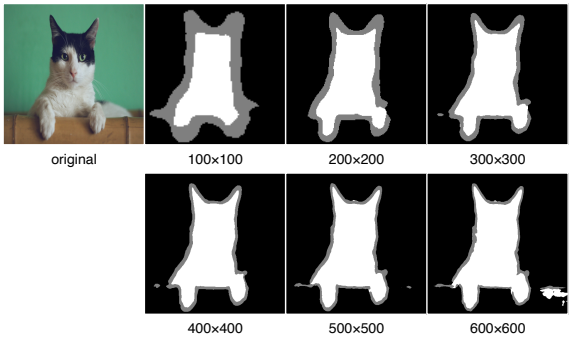

Figure 4.5: Generated trimaps of different resolutions from a network trained by images of 400×400 pixels.

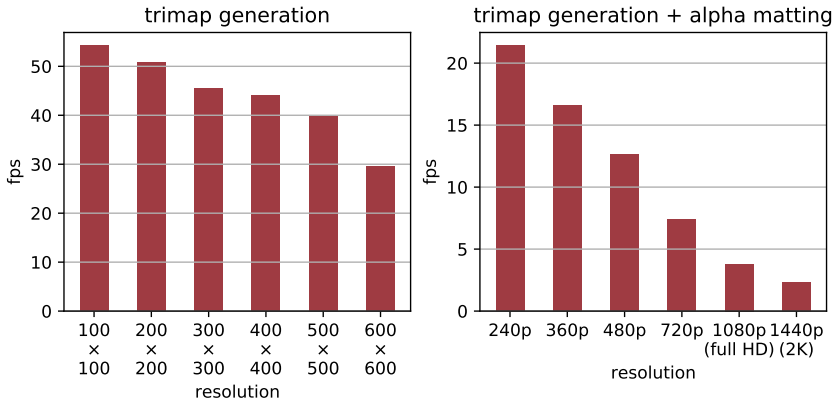

Figure 4.6: Computation time of the proposed method. The upper graph shows the the FPS for generating trimaps only. The lower graph shows the FPS for generating trimaps having 400×400 pixels and then image matting using DIM. The graphs show the average FPS for $1,000$ images.

methods such as DIM do not require the input trimap to have fine details, upsampling of the trimap would not reduce the quality. Examples of using the proposed method to produce image matting for high-resolution images are indicated in Figure 1.1.

We used an NVIDIA Quadro RTX 8000 as the GPU to measure the computation time. The left graph of Figure 4.6 shows the frames per second (FPS) for generating trimaps of different sizes. The right graph compares the FPS for the whole process of image matting when the resolution of the trimap generation was set to 400×400. The results show that it is difficult to use the system in real time, but it works very efficiently in offline video editing situations, for example.

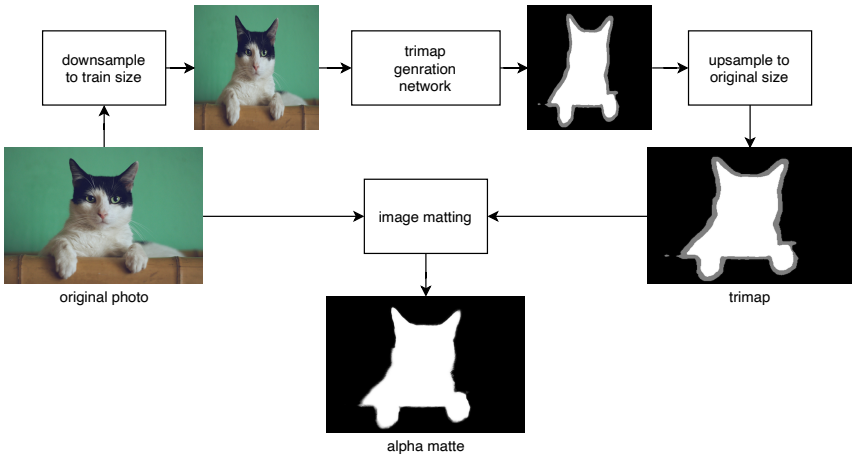

Figure 4.7: Proposed image matting process for a high-resolution image. 1. The image is downsampled to match the size of the training images. 2. A trimap is generated from the downsampled image by a trained network. 3. The trimap is upsampled to match the size of the original image. 4. The trimap and the original image is processed by an image matting algorithm to produce an alpha matte.

Chapter 5

Conclusion

We proposed a method for converting a natural image into a trimap that can then be used to generate an alpha matte, the standard representation used for foreground extraction. Our method generates trimaps and saliency maps simultaneously and computes the loss function for both images. We verified the effectiveness of the proposed method by measuring its performance on generating alpha mattes for both the DUTS test dataset and the Matting Human Dataset [50].

Future work includes training the model using larger datasets. The Matting Human Dataset that we used in the evaluation is limited to portrait photographs. A larger subject-matting dataset for general objects will be needed to improve the accuracy of this task.

Also, the generative adversarial network (GAN) is now widely used for generating new images that resemble existing images in a training dataset [51]. It trains two networks, namely a generator that generates images from random number sequences and a discriminator that tries to judge whether it is a "real" image contained in the training dataset or a "fake" image generated by the generator. After training, the generator can generate images that are not easily distinguishable from the training images. Isola et al. proposed an extension of this method called Pix2Pix [52], which converts an image to another style. These methods are known to produce outputs with low noise [53, 54]. The critical point of these methods is that by using a discriminator, we can tackle problems where it is difficult to define an optimal loss function for each task. Incorporating them into trimap generation may result in images that have even more natural segmentation.

Drawing a trimap is a tedious task that requires a significant amount of time. Our method will reduce that time significantly. It will make alpha matting accessible to more users, enabling them to create more interesting contents through integrating multiple images and videos.

Acknowledgement

I would like to express my deepest gratitude to Associate Professor Taro Tezuka, who gave me for his continuous support and thoughtful guidance throughout this thesis. A sincere gratitude I give to Associate Professor Kei Wakabayashi for his useful advice during the joint seminars and camps. I would also like to express my special gratitude to Associate Professor Masahiko Mikawa for his support and willingness to serve as my secondary advisor. Finally, I would like to thank Tezuka-Wakabayashi Laboratory members who gave me various advice through my regular research activities.

References

[1] Ning Xu, Brian Price, Scott Cohen, and Thomas Huang. Deep image matting. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

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

[3] Kaiming He, Christoph Rhemann, Carsten Rother, Xiaoou Tang, and Jian Sun. A global sampling method for alpha matting. In *CVPR 2011*, pp. 2049–2056. IEEE, 2011.

[4] Ehsan Shahrian, Deepu Rajan, Brian Price, and Scott Cohen. Improving image matting using comprehensive sampling sets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 636–643, 2013.

[5] Anat Levin, Dani Lischinski, and Yair Weiss. A closed-form solution to natural image matting. *IEEE transactions on pattern analysis and machine intelligence*, Vol. 30, No. 2, pp. 228–242, 2007.

[6] Xiaoyong Shen, Xin Tao, Hongyun Gao, Chao Zhou, and Jiaya Jia. Deep automatic portrait matting. In *European Conference on Computer Vision*, pp. 92–107. Springer, 2016.

[7] Donghyeon Cho, Yu-Wing Tai, and Inso Kweon. Natural image matting using deep convolutional neural networks. In *European Conference on Computer Vision*, pp. 626–643. Springer, 2016.

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

[9] Jingwei Tang, Yağız Aksoy, Cengiz Öztireli, Markus Gross, and Tunç Ozan Aydın. Learning-based sampling for natural image matting. In *Proc. CVPR*, 2019.

[10] 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 International Conference on Computer Vision*, pp. 8819–8828, 2019.

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

[12] 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*, pp. 13676–13685, 2020.

[13] Soumyadip Sengupta, Vivek Jayaram, Brian Curless, Steve Seitz, and Ira Kemelmacher-Shlizerman. Background matting: The world is your green screen. In *Computer Vision and Pattern Regognition (CVPR)*, 2020.

[14] Chang-Lin Hsieh and Ming-Sui Lee. Automatic trimap generation for digital image matting. In *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2013 Asia-Pacific*, pp. 1–5. IEEE, 2013.

[15] Ahmad Al-Kabbany and Eric Dubois. A novel framework for automatic trimap generation using the gestalt laws of grouping. In *Visual Information Processing and Communication VI*, Vol. 9410, p. 94100G. International Society for Optics and Photonics, 2015.

[16] Donghyeon Cho, Sunyeong Kim, Yu-Wing Tai, and In So Kweon. Automatic trimap generation and consistent matting for light-field images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, No. 8, pp. 1504–1517, 2017.

[17] Vikas Gupta and Shanmuganathan Raman. Automatic trimap generation for image matting. *CoRR*, Vol. abs/1707.00333, , 2017.

[18] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, Sabine Süsstrunk, et al. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 34, No. 11, pp. 2274–2282, 2012.

[19] 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 (ACM Multimedia)*, pp. 618–626, 2018.

[20] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, Vol. 88, No. 2, pp. 303–338, 2010.

[21] Gabriel J Brostow, Julien Fauqueur, and Roberto Cipolla. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, Vol. 30, No. 2, pp. 88–97, 2009.

[22] 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*, pp. 740–755. Springer, 2014.

[23] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *Proceedings of The IEEE Conference on Computer Vision and Pattern Recognition*, pp. 567–576, 2015.

[24] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision*, pp. 746–760. Springer, 2012.

[25] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, Vol. 39, No. 12, pp. 2481–2495, 2017.

[26] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

[27] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[28] Yi Zhu, Karan Sapra, Fitsum A Reda, Kevin J Shih, Shawn Newsam, Andrew Tao, and Bryan Catanzaro. Improving semantic segmentation via video propagation and label relaxation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8856–8865, 2019.

[29] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2881–2890, 2017.

[30] Dominik A Klein and Simone Frintrop. Center-surround divergence of feature statistics for salient object detection. In *2011 International Conference on Computer Vision*, pp. 2214–2219. IEEE, 2011.

[31] Ming-Ming Cheng, Niloy J Mitra, Xiaolei Huang, Philip HS Torr, and Shi-Min Hu. Global contrast based salient region detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 37, No. 3, pp. 569–582, 2014.

[32] Chuan Yang, Lihe Zhang, Huchuan Lu, Xiang Ruan, and Ming-Hsuan Yang. Saliency detection via graph-based manifold ranking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3166–3173, 2013.

[33] Qiong Yan, Li Xu, Jianping Shi, and Jiaya Jia. Hierarchical saliency detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1155–1162, 2013.

[34] Lijun Wang, Huchuan Lu, Yifan Wang, Mengyang Feng, Dong Wang, Baocai Yin, and Xiang Ruan. Learning to detect salient objects with image-level supervision. In *CVPR*, 2017.

[35] Yin Li, Xiaodi Hou, Christof Koch, James M Rehg, and Alan L Yuille. The secrets of salient object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 280–287, 2014.

[36] Nian Liu, Junwei Han, Dingwen Zhang, Shifeng Wen, and Tianming Liu. Predicting eye fixations using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 362–370, 2015.

[37] Guanbin Li and Yizhou Yu. Visual saliency based on multiscale deep features. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5455–5463, 2015.

[38] Guanbin Li and Yizhou Yu. Deep contrast learning for salient object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 478–487, 2016.

[39] Nian Liu and Junwei Han. Dhsnet: Deep hierarchical saliency network for salient object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 678–686, 2016.

[40] Zhiming Luo, Akshaya Mishra, Andrew Achkar, Justin Eichel, Shaozi Li, and Pierre-Marc Jodoin. Non-local deep features for salient object detection. In *Proceedings of the IEEE Conference on computer vision and pattern recognition*, pp. 6609–6617, 2017.

[41] Pingping Zhang, Dong Wang, Huchuan Lu, Hongyu Wang, and Xiang Ruan. Amulet: Aggregating multi-level convolutional features for salient object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 202–211, 2017.

[42] Ting Zhao and Xiangqian Wu. Pyramid feature attention network for saliency detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3085–3094, 2019.

[43] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, Vol. abs/1409.1556, , 2015.

[44] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

[45] pytorch. pytorch vision. `https://github.com/pytorch/vision`.

[46] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

[47] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[48] foamliu. Deep image matting pytorch. `https://github.com/foamliu/Deep-Image-Matting-PyTorch`.

[49] Christoph Rhemann, Carsten Rother, Jue Wang, Margrit Gelautz, Pushmeet Kohli, and Pamela Rott. A perceptually motivated online benchmark for image matting. In

2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1826–1833. IEEE, 2009.

[50] AISegment.com. Matting human datasets. `https://github.com/aisegmentcn/matting_human_datasets`.

[51] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.

[52] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017.

[53] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in Neural Information Processing Systems*, pp. 1486–1494, 2015.

[54] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.