Instance Segmentation based Semantic Matting for Compositing Applications

Guanqing Hu
Centre for Intelligent Machines, McGill University
Montreal, Canada
bonniehu@cim.mcgill.ca

James J. Clark
Centre for Intelligent Machines, McGill University
Montreal, Canada
clark@cim.mcgill.ca

| Original image | Instance segmentation | Trimap for instance 1 | Trimap for instance 2 | Alpha matte 1 | Alpha matte 2 |

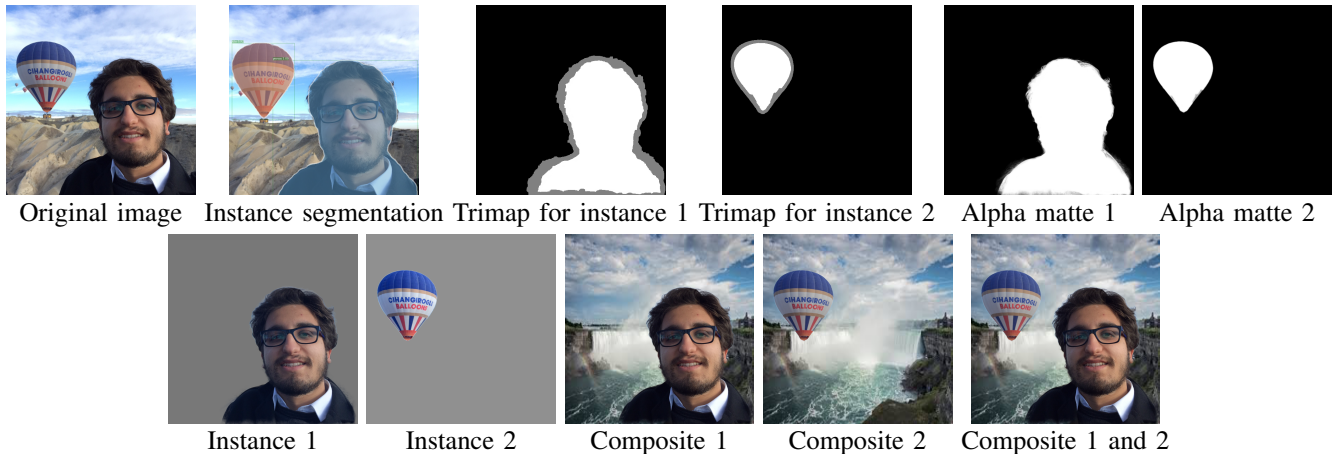| Instance 1 | Instance 2 | Composite 1 | Composite 2 | Composite 1 and 2 |

Figure 1: Our approach first generates coarse instance masks that are used to create trimaps. Then, the trimaps and the original image are used by the image matting network to produce an alpha matte for each instance. Finally, the alpha mattes are used for image compositing.

Abstract—**Image compositing is a key step in film making and image editing that aims to segment a foreground object and combine it with a new background. Automatic image compositing can be done easily in a studio using chroma-keying when the background is pure blue or green. However, image compositing in natural scenes with complex backgrounds remains a tedious task, requiring experienced artists to hand-segment. In order to achieve automatic compositing in natural scenes, we propose a fully automated method that integrates instance segmentation and image matting processes to generate high-quality semantic mattes that can be used for image editing task.**

Our approach can be seen both as a refinement of existing instance segmentation algorithms and as a fully automated semantic image matting method. It extends automatic image compositing techniques such as chroma-keying to scenes with complex natural backgrounds without the need for any kind of user interaction. The output of our approach can be considered as both refined instance segmentations and alpha mattes with semantic meanings. We provide experimental results which show improved performance results as compared to existing approaches.

Keywords-**image matting; instance segmentation;**

I. INTRODUCTION

Image compositing is a key technique in movie production and image editing. It combines visual elements from different sources into one image. The process of extracting visual elements from a source typically involves precise extraction of foreground objects from the background using either manual rotoscopy, where an artist traces the object to be extracted, or by an automatic chroma-keying when the object is in front of a uniformly colored background (e.g. green-screen). In this work, we present a method for automatic foreground object extraction that can work even with complex backgrounds. Our method combines instance segmentation and image matting processes, which allows for multiple foreground objects to be classified and segmented and extracted from the background. The semantic labels provided by the instance segmentation process provides a way to automatically extract objects of different types. For example, we can extract all people in the scene from the background, leaving other types of foreground objects behind. Or, we can obtain multiple mattes, one for each instance, and these can be used as desired in later compositing operations.

Object segmentation is considered as one of the most important and complex tasks in Computer Vision. There have been many recent advances in segmentation algorithms based on learning-based methods, in large part due to a number of publicly available datasets that provide human-

drawn segmentation masks as ground truths. However, most of these masks are coarse and unrefined, which makes the segmentation algorithms also provide coarse and unrefined object boundaries. These image segmentation techniques lack sufficient refinement to be used for high quality image composition tasks.

On the other hand, image matting is another fundamental problem in Computer Vision that has been studied since the 1950s. Similar to segmentation, which involves generating a coarse binary mask for each object, image matting extracts an interesting object from a static image or a set of video frames by estimating an alpha matte α containing the opacity value for each pixel in the image. Typically, pixels in the foreground have alpha values equal to 1 while those in the background are given alpha values equaling 0. However, because of the extended size of pixels as well as motion blur in videos, pixels on the boundary between foreground and background objects have contributions from both the foreground and background objects and hence are given alpha values between 0 and 1. This relation is interpreted in Equation 1, where i, FG, BG represent pixel position, foreground image, and background image respectively.

$$I_i = \alpha_i \times FG_i + (1 - \alpha_i) \times BG_i \qquad (1)$$

In equation 1, the only known value is the image input I, while the variables $FG, alpha, BG$ are unknown and need to be estimated. To simplify the estimation process, most image matting algorithms require a manual intervention in the form of user-labeled inputs. One type of user-labeled input is called the *trimap* (Figure2b), which densely labels the opacity of the known (foreground and background) and unknown (boundary) regions. In the trimap, the foreground and the background are labeled $\alpha = 1$ and 0, respectively, while the remaining regions are initially given a label of $\alpha_i = 0.5$. Another common type of user-labeled input are *strokes* (Figure2c), which labels the region of foreground and background coarsely using scribbled strokes. Stroke-based algorithms are faster, requiring less user input, but produces lower quality results. Finally, there exist semi-automated methods for generating matting features directly from RGB input (e.g. Levin's spectral matting approach [11]) as shown in Figure 2d. These techniques are considered semi-automated since they still require a small amount of user guidance to select the corresponding foreground features. The requirement of user interaction not only causes major delays and expense in the image editing workflow, but also severely limits the applications that image matting can be used on.

In this paper, we propose a fully automated image segmentation-matting approach in which accurate segmentation can be achieved on general natural image datasets such as COCO [5]. The proposed method takes an RGB image without any additional user-labeled input and generates an alpha matte for all the detected object instances in the scene.

(a) RGB input   (b) Trimap   (c) Strokes   (d) Spectral features   (e) Alpha matte
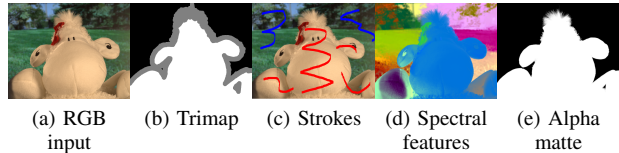
Figure 2: Image matting input and output indication. Spectral features in (d) requires user guidance to select foreground features.

The proposed deep learning-based approach consists of two stages. First, the input image is fed into a Mask R-CNN network [2] where the object bounding box and instance mask are generated. Then, using these two results, a trimap is estimated for each detected object in the image. Next, the trimap and the original RGB image are used together as inputs to a Deep Image Matting network [3] to generate the final alpha matte. The final output of the pipeline is not only the refined foreground object mask but also an alpha matte with an accompanying semantic label that can be used to extract specific types of objects in a complex scene.

We evaluate our technique qualitatively by comparing results with other approaches tackling automatic matting or automatic soft segmentation problems. Experimental results demonstrate comparable and, in some cases, superior, performance in extracting alpha mattes as compared with existing approaches. Our method is also capable of compositing with the motion blur that is often encountered in movies and videos so that the new background pixels show through on the blurred region. Our approach allows non-experts to perform natural image compositing easily by just selecting the object categories to keep and the auto-generated semantic alpha mattes will do the rest. It will also accelerate video compositing tasks that originally required artists to laboriously trace objects frame by frame. An example of using our approach in image compositing can be seen in Figure 1, including the intermediate trimaps and instance segmentations.

II. RELATED WORK

A. Natural Image Matting

Image matting estimates an alpha matte for an interesting object by generating an opacity value for each pixel in the image. A trimap is a typical input to matting algorithms which indicates uncertain or unknown opacity (boundary) regions that the algorithms should work on, as well as known foreground and background regions. Matting methods can be categorized into: Non-learning-based and Learning-based methods. There are many non-learning image matting methods such as sampling-based [7], Bayesian-based [8], affinity-based methods like Closed-form Matting [10], and Poisson Matting [9]. Learning-based methods were only recently developed because of the difficulty in generating large scale
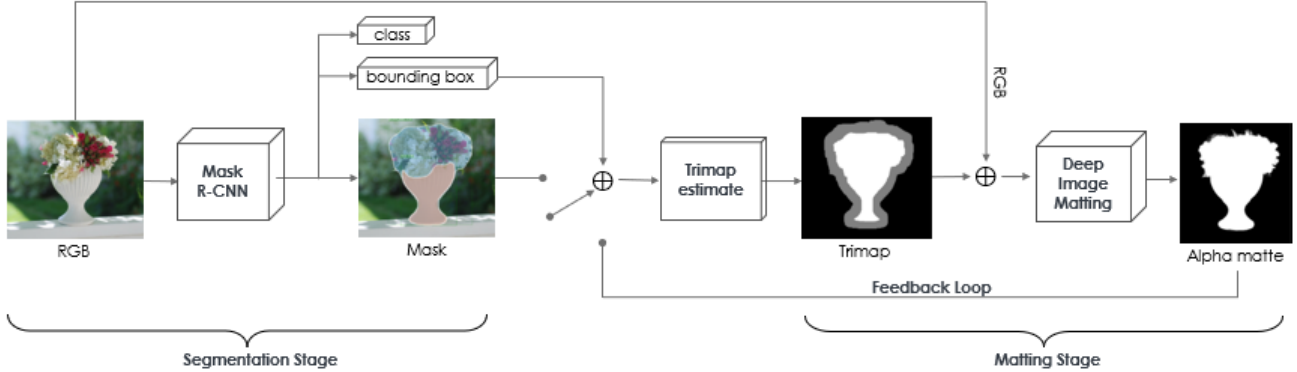
Figure 3: Proposed method as a pipeline

ground truth data. Xu et al. used deep learning in solving the matting problem by creating a new dataset with real world images and expanded the dataset scale by compositing it with different background images. They use a two-stage deep learning method consisting of a deep convolutional encoder-decoder network and a fully convolutional network for refinement [3]. Using the same dataset, [12] replaces the refinement stage in [3] with adversarial model, achieving state-of-the-art performance on benchmark.

B. Instance Segmentation

Instance Segmentation is a combination of object detection and semantic segmentation. Object detection classifies each object into a category and localize each object by predicting a bounding box. Semantic segmentation classifies each pixel in the image into an object category without distinguishing different instances. Mask R-CNN [2] is a state-of-the-art instance segmentation method. It takes an RGB image as input and generates three types of annotations for each detected object. The common dataset to train Mask R-CNN is COCO dataset [5] containing 81 object categories. For each recognized object in the input image, Mask R-CNN returns an object label, a bounding box indicating object location and scale, and a segmentation mask specifying the object in pixel level.

C. Soft Segmentation

Soft segmentation, in contrast to traditional segmentation that focuses on generating coarse object masks, estimates more precise object boundary transactions. Semantic Soft Segmentation [6] is a state-of-the-art work achieving good performance on automatic soft segmentation. It uses high-level semantic features extracted from the semantic segmentation model DeepLab [14] to categorize and combine low-level texture and color features generated from spectral decomposition.

III. METHODOLOGY AND EXPERIMENTAL SET-UP

This section introduces the pipeline of the proposed method (Figure 3) and how each key part works individually and together. The pipeline starts with taking an input image into a Mask R-CNN network through which the object bounding box and instance mask are generated. Using these intermediate outcomes, a coarse trimap can be estimated for each detected object. To estimate the final alpha matte, this trimap and the input image are fed to the image matting network. Due to the coarseness of the instance mask leading to low-quality trimaps, the alpha matte generated in the first pass is also poor. Estimating a new trimap from the generated alpha matte and passing through the matting stage again generally improves the segmentation and matting results. This forms a feedback loop in the pipeline where the trimap and alpha matte improve each other.

A. Segmentation stage

The first stage of the pipeline could use any instance segmentation network that produces instance masks and bounding boxes. We use the Mask R-CNN [2] algorithm in our approach. As the first stage of the pipeline, the inference error of Mask R-CNN influences the accuracy of the latter steps. Therefore, we use *Mask R-CNN with bells and whistles* that achieves higher accuracy in both bounding box and mask inference. The model has a deeper backbone architecture ResNeXt-152-32x8d-FPN trained on ImageNet-5k as opposed to the usual ImageNet-1k [13]. We use the pre-trained Mask R-CNN model from Detectron Pytorch version [13].

B. Trimap estimation stage

Trimap estimation makes use of the intermediate outputs from Mask R-CNN to produce a trimap for each detected object instance. We make an assumption that the region near the mask boundary is the region that requires the most further estimation, i.e. the unknown region where the image matting algorithm should focus on. Therefore, a certain

region dilated from the object mask is defined to be the unknown area in the trimap with $\alpha_i = 0.5$. The region further inside the mask defines the foreground area with $\alpha_i = 1$, while the region further outside the mask is assigned as the background with $\alpha_i = 0$.

The amount of the dilation is determined by the object size. The approximate width and height of an object can be estimated using its bounding box coordinates as $width = bbox[2] - bbox[0]$ and $height = bbox[3] - bbox[1]$. We choose the dilation rate to be a fixed percentage of the width-and-height average. There is a trade-off in choosing a higher or lower dilation rate. A precise trimap is favored by the matting network as it imposes stronger constraints. When the mask boundary is close to the object true boundary, a small rate of dilation is enough to cover the region that needs to be refined. But there are also cases when larger dilation rate is preferable to recover errors of poor masks, resulting in coarse trimaps that could degrade the matting stage.

C. Matting stage

For image matting, we use the learning-based method *Deep Image Matting* by Lin et al. [3]. It is a VGG16-based encoder-decoder network followed by a fully-connected refinement stage. Taking advantage of the semantic meaning extracted by VGG16, Deep image matting has an outstanding performance on natural scenes where the foreground color and background color are sometimes very similar. Its performance is also less dependent on the quality of the trimap than non-learning methods. This data-driven model uses the dataset Lin et al. created [3]. The dataset contains 431 unique objects with associated alpha mattes but the object variety is still limited compared to common object datasets (with no mattes) used for training object classifiers.

1) Network training: We only train the network without the last refinement layers due to hardware limitations. We expect our results to be even better with the refinement layers. The implementation uses the dataset from Lin et al. [3]. Training data of the network is pre-processed as 320×320 image patches centering at an unknown pixel (i.e. $\alpha_i = 0.5$) randomly cropped from training images.

2) Inference: During test time, we pre-process the test images similarly as in training in order to achieve better performance. When dealing high resolution test images we could simply resize them (in our case to 320×320). However, this may cause unavoidable issues in terms of network performance and coarse alpha mattes. Indeed, downsampling methods causes detail loss especially when the ratio of the size of test images to the desired size is very high. To avoid this, we implemented a *patch-based pre-processing method* on the input test images. This approach consists in cropping a test image into patches of 320×320. Each patch of that single image is separately fed to the network giving multiple alpha matte results. Then, these results are blended together. A limitation in using this technique is

the fact that we must make sure that the cropped input images are centered at an unknown pixel in the trimap. If this is not the case the network may perform poorly. In addition to that, the network performance is compromised when the content of the cropped patch is very different from the training data used during the training phase of the network. To address these issues, the high resolution test image is first downsampled to 640×640. Then, patches are randomly cropped with their centers moving along the gray region of the trimap until the whole unknown region gets covered. For each patch, an alpha matte patch is generated. All these alpha matte patches are pasted back to their original locations and averaged with the existing overlaps of patches. With this method of averaging we found some issues related to discontinuities especially along the borders as well as susceptibility to outliers. Hence, we developed a *multiple Sampling method* which consists in running the test algorithm on each image for $K = 10$ times and take the median of each set of alpha values at a pixel to exclude any possible outliers and proceed in smoothing the final alpha matte.

D. Feedback loop

Since the trimap generated directly from the instance mask is generally coarse, the alpha matte from the first pass of the pipeline is normally undesirable with low boundary accuracy. So, a feedback connection was inserted between the alpha matte output and the image matting network input. From this, a new trimap is created by dilation from the previous pass's alpha values. The latter is fed back to the image matting network for further refinement. The motivation behind this implementation is that the quality of trimap and alpha matte increases at each pass and leads to a simultaneous improvement. With this increase in quality, the dilation rate of the trimap is decreased. Generally, four feedback loops leads to accurate results as demonstrated in Section IV.

E. Handling multiple objects

When faced with an input image that contains multiple objects, we proceed in dealing with one object at a time (i.e. in a case of two objects in one image we consider the first object as foreground and the second as background image, then vice-versa). This method copes with superposing regions of the trimap after performing dilation of the boundaries. Moreover, since the matting network used in this work is one designed for single-object matting, this method of distinction of object within a single image is more effective.

IV. EXPERIMENTAL ANALYSIS

Natural image matting results can be evaluated numerically if the ground truth alpha matte is available. However, the purpose of our method is to generalize natural image matting to common datasets (e.g COCO) so that matting

techniques can adapt to more general applications. In these datasets, ground truth alpha matte is not available. Therefore, this section mainly focuses on analysing qualitative results. The first subsection analyses the effect of the pipeline structure that successfully refines coarse instance mask to alpha matte that can be used for image editing task. The second subsection compares results with existing fully automate methods including Semantic Soft Segmentation and Spectral Matting. The third subsection presents some limitations and failure cases.

(a) RGB input   (b) Mask RCNN   (c) Trimap, no feedback

(d) Alpha, no feedback   (e) Trimap after 2nd pass   (f) Alpha after 2nd pass

(g) Trimap after 4th pass   (h) Alpha after 4th pass   (i) Compositing result
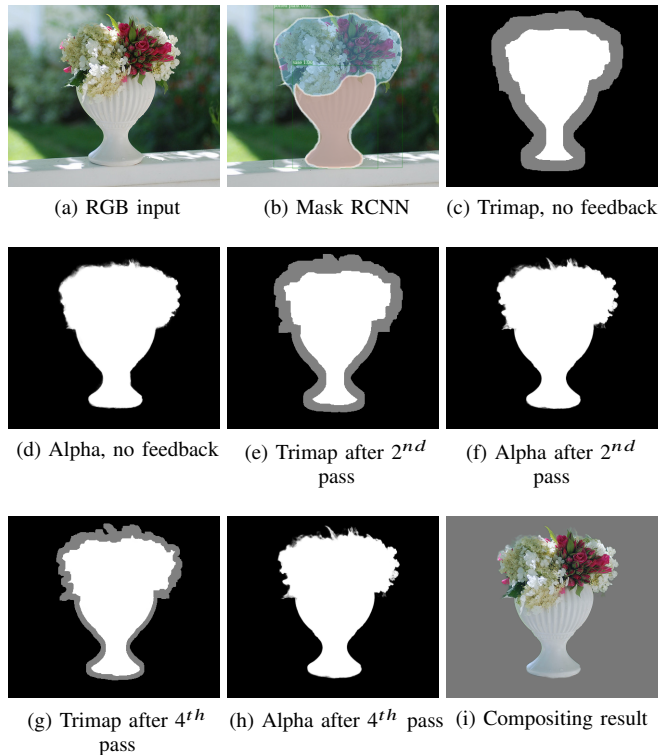
Figure 4: Matting results of potted plant image

Figure 4 presents result details of an example image from the COCO dataset [5]. Mask RCNN detects the pot and the potted plant as two separate objects. Since the boundary structure of the pot is fairly simple, we only show the matting results of the potted plant. Figure 4c is the initial trimap generated directly from the instance segmentation mask with the widest and most coarse uncertain region. The alpha matte based on such a trimap, as shown in Figure 4d, lacks detail, but is slightly better than the original instance mask. A second trimap can be created by dilating this alpha matte, resulting in better trimap which then leads to better alpha matte. Figures 4e-4h demonstrate the gradual improvement of trimaps and alpha mattes, where the number indicates at which feedback loop the corresponding result is generated from. A composite is created with the extracted plant region using the alpha matte generated after four

feedback passes (Figure 4h).

The number of feedback passes is a hyper-parameter that need to be adjusted based on the application requirements. We found that four feedback passes are generally enough to achieve high-quality performance.

As a comparison, segmentation results of the bouquet are also generated using Semantic Soft Segmentation [6] and Spectral matting [11] as shown in Figure 5. Both approaches suffer at the regions where the foreground object color is similar to the background as outlined in red. Semantic Soft Segmentation takes advantage of high-level semantic meanings and groups low-level features into three layers. However, there are still important details missing when the color of the foreground is too close to the background (e.g. around the most left leaves of the bouquet).

(a) Semantic Soft Segmentation results [6]
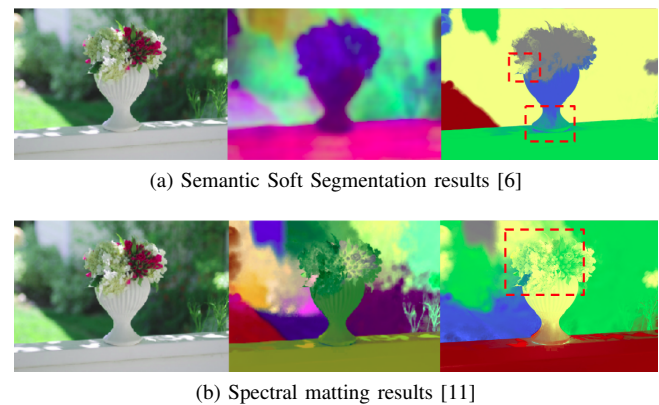

(b) Spectral matting results [11]

Figure 5: Results of Semantic Soft Segmentation and Spectral matting. From left to right: original image, extracted features, final result.

Results of more images from COCO dataset are presented in Figure 7 and 8 including comparisons with Semantic Soft Segmentation and Spectral matting. The index on the left of each line in Figure 7 and 8 will be used to refer to each presented image. Image 1-5 are the ones shown in the Semantic Soft Segmentation paper [6]. The resolution of these images are slightly different than the originals in COCO. However, we expect the method to be robust to small image scale changes. For image 1-4 and 8, our approach achieves as good performance as Semantic Soft Segmentation. In image 4, our approach is the only one that correctly labels the black region on the top of the person's right shoulder as background. Although in image 8, the person's left hand is not detected, the composited result is still visually correct.

Image 5 demonstrates our approach's capability of capturing motion blur like the person's right arm, while the other two methods cannot. Image 6, 7, and 9 demonstrate the advantage of using instance-level segmentation as guideline because when the scene contains more than one object that belongs to the same semantic category, only using

semantic segmentation features tend to merge multiple different instances into one. Our approach is more flexible in choosing which instance to keep and this selection can also be automated by using the class label and bounding box position as instance identity.

A. Limitations

One limitation of our approach is that the trimap generation is limited at object borders, hence for objects with large area of transparency or other type of complex opacity, the proposed method cannot generate trimaps covering such a large region, which results in incorrect alpha matte. The image matting network also suffers from the limited amount of object categories it is trained on. When applied to unfamiliar objects, the network tends to generate fuzzy and transparent alpha even though the truth object boundary is solid. As can be seen in Figure 6b, our approach confuses background grass as part of the fuzzy boundaries of the fire hydrant. The Semantic Soft Segmentation algorithm does not have the same issue, but has problems at the bottom of the hydrant.

V. CONCLUSION

In this work, we propose a fully automated image segmentation-matting approach in which accurate segmentation can be achieved on general image datasets such as COCO [5]. The proposed method does not require any additional user-labeled input and generates individual alpha matte for all the detected objects in the image. Our approach can be seen as both a segmentation enhancing approach and a fully automated image matting approach that works on common dataset. The performance is evaluated and compared with related approaches. Our approach performs as well as these approaches with advantage in instance-level selections. Our approach could help non-expertise in image compositing tasks and accelerate the image editing process.

ACKNOWLEDGMENTS

(a) RGB input   (b) Our result   (c) Our result, zoomed in   (d) Semantic Soft Segmentation
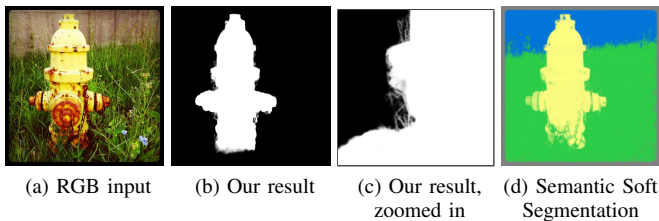
Figure 6: Example of a failure case.

REFERENCES

[1] Q. Zhu, P. A. Heng, L. Shao, and X. Li, "What's the role of image matting in image segmentation?," 2013 IEEE Int. Conf. Robot. Biomimetics, ROBIO 2013, no. December, pp. 1695-1698, 2013.

[2] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," Proc. IEEE Int. Conf. Comput. Vis., vol. 2017-Octob, pp. 2980-2988, Mar. 2017.

[3] N. Xu, B. Price, S. Cohen, and T. Huang, "Deep image matting," Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, vol. 2017-Janua, pp. 311-320, Mar. 2017.

[4] C. Rhemann, C. Rother, J. Wang, M. Gelautz, P. Kohli, and P. Rott, "A Perceptually Motivated Online Benchmark for Image Matting," in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009, pp. 1826-1833.

[5] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollât'ar, and C. L. Zitnick. Microsoft COCO: Common objects in context. In ECCV, 2014. 2, 5

[6] Y. Aksoy, T.-H. Oh, S. Paris, M. Pollefeys, and W. Matusik, "Semantic soft segmentation," ACM Trans. Graph., vol. 37, no. 4, pp. 1-13, 2018.

[7] M. A. Ruzon and C. Tomasi, "Alpha estimation in natural images," in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2000, pp. 18-25.

[8] Y. Chuang, B. Curless, D. H. Salesin, and R. Szeliski, "A Bayesian Approach to Digital Matting," in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2001, pp. 264-271.

[9] E. S. L. Gastal and M. M. Oliveira, "Shared sampling for real-time alpha matting," Computer Graphics Forum, vol. 29, pp. 575-584, 2010.

[10] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patch- match: A randomized correspondence algorithm for structural image editing," ACM Trans. Graph. (TOG), vol.28, no.3, 2009.

[11] A. Levin, A. Rav-Acha, and D. Lischinski, "Spectral matting," IEEE Trans. Pattern Anal. Mach. Intell., vol. 30, no. 10, pp. 1699-1712, 2008.

[12] S. Lutz, K. Amplianitis, and A. Smolic, "AlphaGAN: Generative adversarial networks for natural image matting," BMVC. Jul. 2018.

[13] Roytseng-Tw, "roytseng-tw/Detectron.pytorch," GitHub, 15-Aug-2018. [Online]. Available: https://github.com/roytseng-tw/Detectron.pytorch. [Accessed: 07-Feb-2019].

[14] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy and A. L. Yuille, "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 4, pp. 834-848, 1 April 2018.
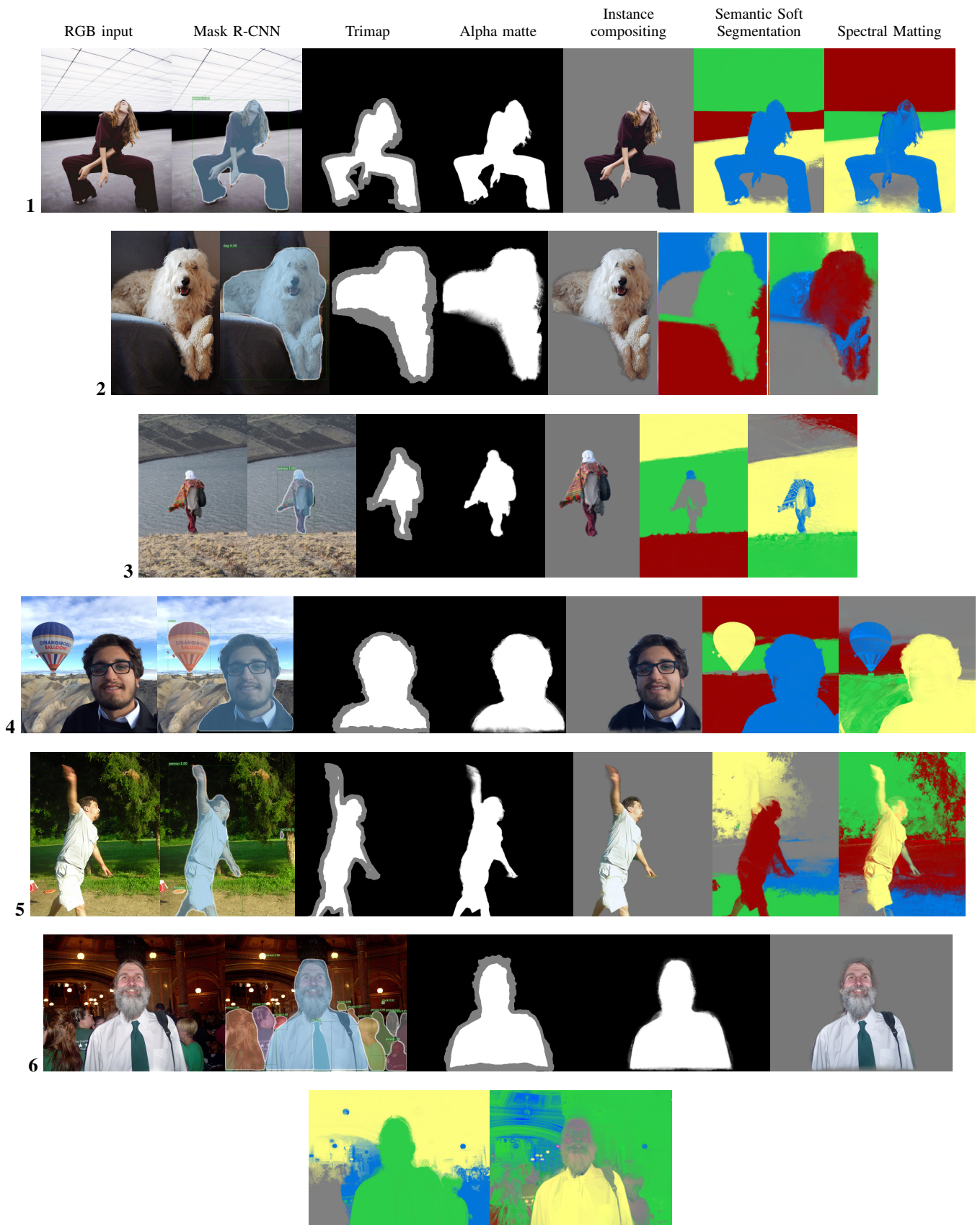
Figure 7: From left to right are RGB input image, Mask-RCNN, trimap at the 4th feedback loop, alpha matte at the 4th feedback loop, instance compositing, result of Semantic Soft Segmentation, and result of spectral matting.
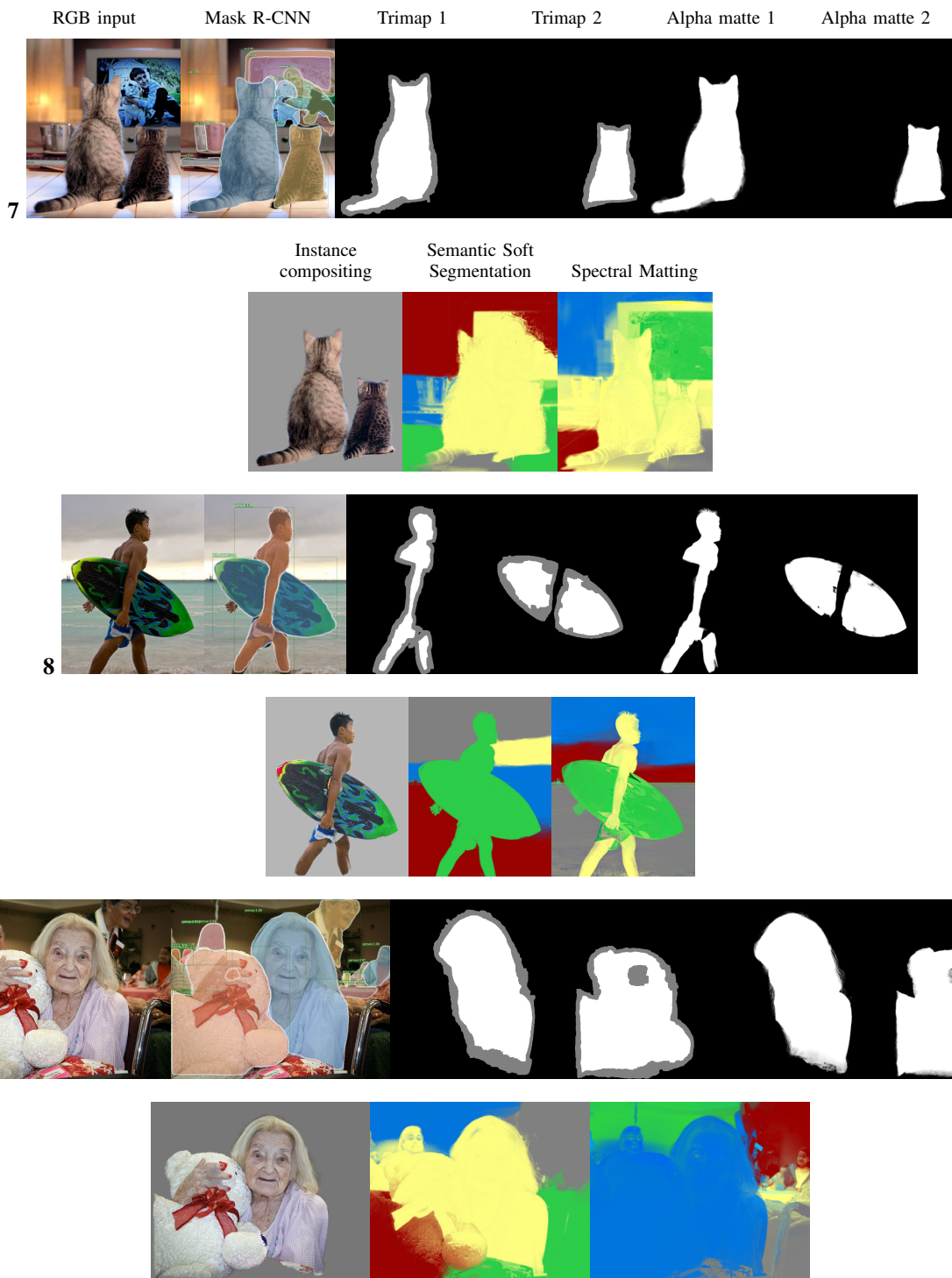
RGB input Mask R-CNN Trimap 1 Trimap 2 Alpha matte 1 Alpha matte 2

7

Instance compositing Semantic Soft Segmentation Spectral Matting

8

9

Figure 8: From left to right are RGB input image, Mask-RCNN, trimap of instance 1 at the 4th feedback loop, , trimap of instance 2 at the 4th feedback loop, alpha matte of instance 1 at the 4th feedback loop, alpha matte of instance 2 at the 4th feedback loop, instance compositing, result of Semantic Soft Segmentation, and result of spectral matting.