Foreground-aware Image Inpainting

Wei Xiong[1*] Jiahui Yu[2] Zhe Lin[3] Jimei Yang[3] Xin Lu[3] Connelly Barnes[3] Jiebo Luo[1]
[1]University of Rochester [2]University of Illinois at Urbana-Champaign [3]Adobe Research
[1]{wxiong5,jluo}@cs.rochester.edu [2]jyu79@illinois.edu
[3]{zlin, jimyang, xinl, cobarnes}@adobe.com

Abstract

Existing image inpainting methods typically fill holes by borrowing information from surrounding pixels. They often produce unsatisfactory results when the holes overlap with or touch foreground objects due to lack of information about the actual extent of foreground and background regions within the holes. These scenarios, however, are very important in practice, especially for applications such as the removal of distracting objects. To address the problem, we propose a foreground-aware image inpainting system that explicitly disentangles structure inference and content completion. Specifically, our model learns to predict the foreground contour first, and then inpaints the missing region using the predicted contour as guidance. We show that by such disentanglement, the contour completion model predicts reasonable contours of objects, and further substantially improves the performance of image inpainting. Experiments show that our method significantly outperforms existing methods and achieves superior inpainting results on challenging cases with complex compositions.

1. Introduction

Image inpainting is an important problem in computer vision, and has many applications including image editing, restoration and composition. We focus on hole filling tasks encountered commonly when removing unwanted regions or objects from photos. Filling holes in images with complicated foreground and background composition is one of the most significant and challenging scenarios.

Conventional inpainting methods [8, 6, 5, 26] typically fill missing pixels by matching and pasting patches based on low level features such as mean square difference of RGB values or SIFT descriptors [19]. These methods can synthesize plausible stationary textures but often produce critical failures in images with complex structures. To alleviate the problem, different structures of images have been exploited [11, 12, 24]. For example, Huang *et al.* [11] explicitly utilize planar structures as guidance to rectify perspectively-distorted textures. However, these methods still rely on existing patches and low-level features, and thus are unable to handle challenging cases where holes overlap with or are close to foreground objects. In such cases, a higher understanding of image content is required.

Recently, deep learning based methods [13, 17, 28, 29, 7] have emerged as a promising alternative avenue by treating the problem as learning an end-to-end mapping from masked input to completed output. These learning-based methods are able to hallucinate novel contents by training on large scale datasets [15, 30]. To produce visually realistic results, generative adversarial networks (GANs) [9] are employed to train the inpainting networks. However, by default all these methods assume that a generative network can learn to predict or understand the structure in the image implicitly, without explicit modeling of structures or foreground/background layers in the learning process.

However, this has not been an easy task even for state-of-the-art models, such as PartialConv [17] and GatedConv [28]. For example, Fig. 1 shows two common failure cases. On the top case, both GatedConv [28] and PartialConv [17] fail to infer a reasonable contour in the missing region, and incorrectly predict a gold medal with an obvious notch. In addition, on the bottom case, both generate obvious artifacts around the neck of the dog. We conjecture that these failures may come from several limitations of current learning-based inpainting systems: (1) learning-based inpainting models are usually trained to fill randomly generated masks which are often completely located in the background or inside a foreground object. This is inconsistent with real-world cases where the holes might be close to or only have a small overlap with the foreground (*e.g.*, cases of distracting region removal); (2) without explicitly modeling background and foreground layer boundaries, current deep neural network-based methods may not be able to predict the structure accurately inside the holes by simply training to fill random masks.

To this end, we propose a foreground-aware image in-

[*]Work was primarily done while Wei Xiong was an Intern at Adobe.

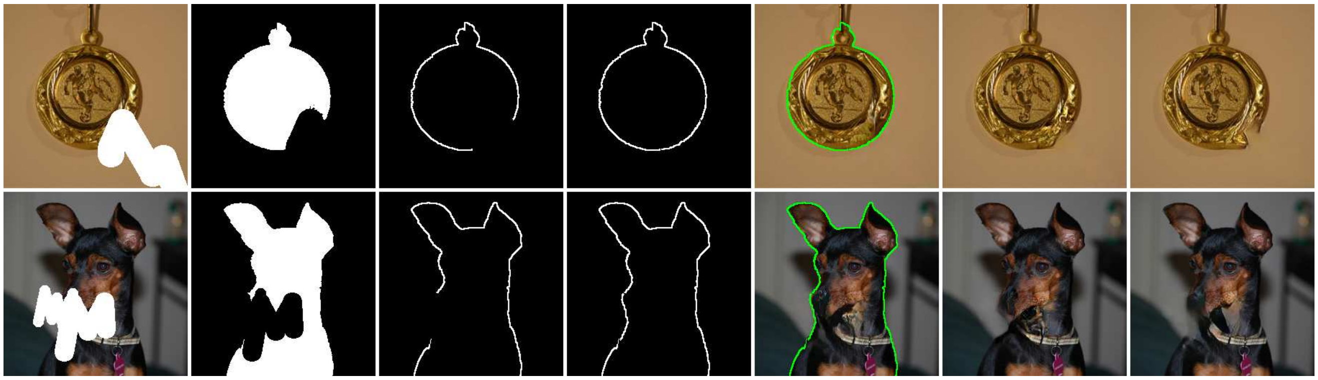


Figure 1. Our results compared with PartialConv [17] and GatedConv [28]. From left to the right are: image with holes, saliency map, incomplete contour and completed contour generated by our model, inpainting result of our model with completed contour (the green curve) on it, result of PartialConv [17], result of GatedConv [28], respectively.

painting system that explicitly incorporates the foreground object knowledge into the training process. Our system disentangles structure inference and image completion, and leverages accurate contour prediction to guide image completion. Specifically, our model first detects a foreground contour of the corrupted image, and then completes the missing contours of the foreground objects with a contour completion module. The completed contour along with the input image are then fed to the image completion module as guidance to predict contents in holes.

The disentanglement of structure inference and image completion is conceptually simple and highly effective. Fig. 1 shows that our model benefits greatly from the inferred contours. Our contour completion module is able to infer a reasonable structure in the missing region. Further, the image completion module takes predicted contours as guidance and generates cleaner contents around the borders of the objects.

To summarize, our contributions are as follows: (1) We propose to explicitly disentangle structure inference and image completion to address challenging scenarios in image inpainting where holes overlap with or touch foreground objects. To the best of our knowledge, our work is among the first of a few studies that inpaint images with explicit contour guidance. (2) To infer the structure of images, we propose a contour completion module trained explicitly to guide image completion. (3) To effectively integrate all the modules, we propose to adopt curriculum training on both the contour and image completion modules. (4) Our experiments demonstrate that the system produces higher-quality inpainting results compared to existing methods.

2. Related Work

Image inpainting approaches can be roughly divided into two categories: traditional methods based on pixel propagation or patch matching, and recent methods based on deep neural network training. Traditional methods such as [3, 4] fill in holes by propagating the neighborhood appearance based on techniques like isophote direction field. These methods are quite effective for small or narrow holes, but when the holes are large or the textures vary heavily, they often generate significant visual artifacts. Patch-based methods predict missing regions by searching for the most similar and relevant patches from the uncorrupted regions of the image. These methods work in an iterative way and can generate smooth and photo-realistic results, but at the cost of high computation cost and memory usage. To reduce the runtime and improve memory efficiency, tree-structure based search [21] and randomized methods [5] are proposed. PatchMatch [5] is a typical patch based method that greatly speeds up the conventional algorithms and achieves high-quality inpainting results. A major drawback of PatchMatch lies in the fact that it searches for relevant patches from the whole image, without using any high-level information to guide the search. These methods work reasonably well for pure background inpainting tasks where holes are only surrounded by background textures, but could easily fail if holes overlap with an object or are close to an object.

Recently, learning based inpainting methods [17, 29, 7] have significantly improved inpainting results by learning semantics from large scale dataset. These methods typically train a convolutional neural network as a mapping function from a corrupted image to a completed one end-to-end. A significant advantage of these methods over the non-learning ones is the ability to learn and understand semantics of images for inpainting, which is especially important in cases of complex scenes, faces, objects and many others. Among these methods, Context Encoders is one of the first attempts [22] that use a deep convolutional neural network to fill in the holes. It maps an image with a square hole to a complete image, and trains the model with L2 loss in the pixel space and an adversarial loss to generate sharper results. Similarly, Iizuka et al. [13] use two discriminators to enforce that both the global appearance (whole image) and the local appearance (content in hole) of the generated result are visually plausible. The method, however, still relies heavily on the post-processing of the completed image that blends both results from neural networks and traditional patch-matching methods. Yu et al. [29] propose contextual attention to model long-range dependencies in images and a refinement network to eliminate post-processing, thus the whole system can be trained and tested end-to-end. However, these deep learning based inpainting methods typically infer the missing pixels conditioned on both valid pixels and the substitute values in the masked holes, which may lead to artifacts. Liu et al. [17] address this problem by masking the convolution operation and updating the mask in each layer, so that the prediction of the missing pixels is only conditioned on the valid pixels in the original image. Yu et al. [28] further propose to learn the mask automatically with gated convolutions, and achieve better inpainting qualities. Additionally, Song et al. [27] apply a pretrained image segmentation network to obtain the foreground mask of the corrupted image, then fill the segmentation mask and use it to guide the completion of the image. However, these methods do not explicitly model the foreground and background boundaries. Therefore, they could fail in images where the masked region covers both foreground and background.

3. Approach

Given an incomplete image, our goal is to output a complete image with a visually pleasing appearance. The overall framework of our inpainting system is shown in Fig. 2. It is a cascade of three modules: incomplete contour detection module, contour completion module and image completion module. We automatically detect the contour of the incomplete image using the contour detection module. Then we use the contour completion module to predict the missing parts of the contour. Finally, we input both the incomplete image and the completed contour to the image completion module to predict the final inpainted image. To train our foreground-aware model, we need to prepare specific training samples and holes. In the following sections, we first introduce how we collect data and generate specific hole masks tailored to our task. Then we introduce the detailed implementation of our inpainting system.

3.1. Data Acquisition and Hole Generation

Image Acquisition and Processing. Existing datasets for image inpainting such as Places2 [30], Paris [23], or CelebFace [18] do not require any annotations, and training data pairs (image with hole and the ground-truth image) are typically constructed by generating random masks on the original images and by setting the original pixel values under the masks as the ground truth. Our proposed framework for foreground-aware image inpainting requires us to train a contour completion module and infer the contour automatically, so we need a training dataset with labeled contours. One possibility is to directly use contour detection datasets, e.g. BSD500 [2]. However, such datasets are quite small in size and thus are not adequate to train an image inpainting model. Instead, we use salient object segmentation datasets as an alternative. We collect over 15,762 natural images that contain one or several salient objects, from a variety of public datasets, including MSRA-10K [10], manually annotated Flickr natural image dataset, and so on. Each image in this saliency dataset is annotated with an accurate segmentation mask. The dataset is quite diverse in content, containing a large variety of objects, including animals, plants, persons, faces, buildings, streets and so on. The relative size of objects in each image has a large variance, making the dataset quite challenging. We split all the samples into 12,609 training images and 3,153 testing images.

We then apply the Sobel edge operator on the segmentation mask to obtain the contours of the salient objects. Specifically, we first obtain the filtered mask C_f by applying the Sobel operator: $C_f = |G_x| + |G_y|$, where G_x and G_y are the vertical and horizontal derivative approximations of the image, respectively. Then we binarize the filtered mask with a simple threshold and obtain the final binary contour C_{gt} as the ground-truth contour of the original image.

Hole Mask Sampling. In real-world inpainting applications, the distractors that users want to remove are usually arbitrarily-shaped, and usually not square-shaped. In order to simulate the real world inputs and learn a practical model, we draw holes on each image with arbitrary shapes randomly with a brush, based on the sampling method in [28]. We generate two types of holes: 1). arbitrarily-shaped holes that can appear in any region of the input image. Under this setting, holes have a probability of overlapping with the foreground objects. This scenario is designed to handle the situations where unwanted objects are inside the foreground objects or partially occlude the salient objects; 2). arbitrarily-shaped holes that are restricted so that they have no overlap with the foreground objects. This type of holes are generated to simulate the situation where the unwanted regions or distracting objects are behind the salient objects. To deal with the second situation, we first randomly generate arbitrarily-shaped holes, then we remove the parts of holes that have overlap with the saliency objects.

3.2. Contour Detection

During the inference stage, we do not have a contour mask of the input image. We therefore use DeepCut [1] to detect the saliency objects in the image automatically.

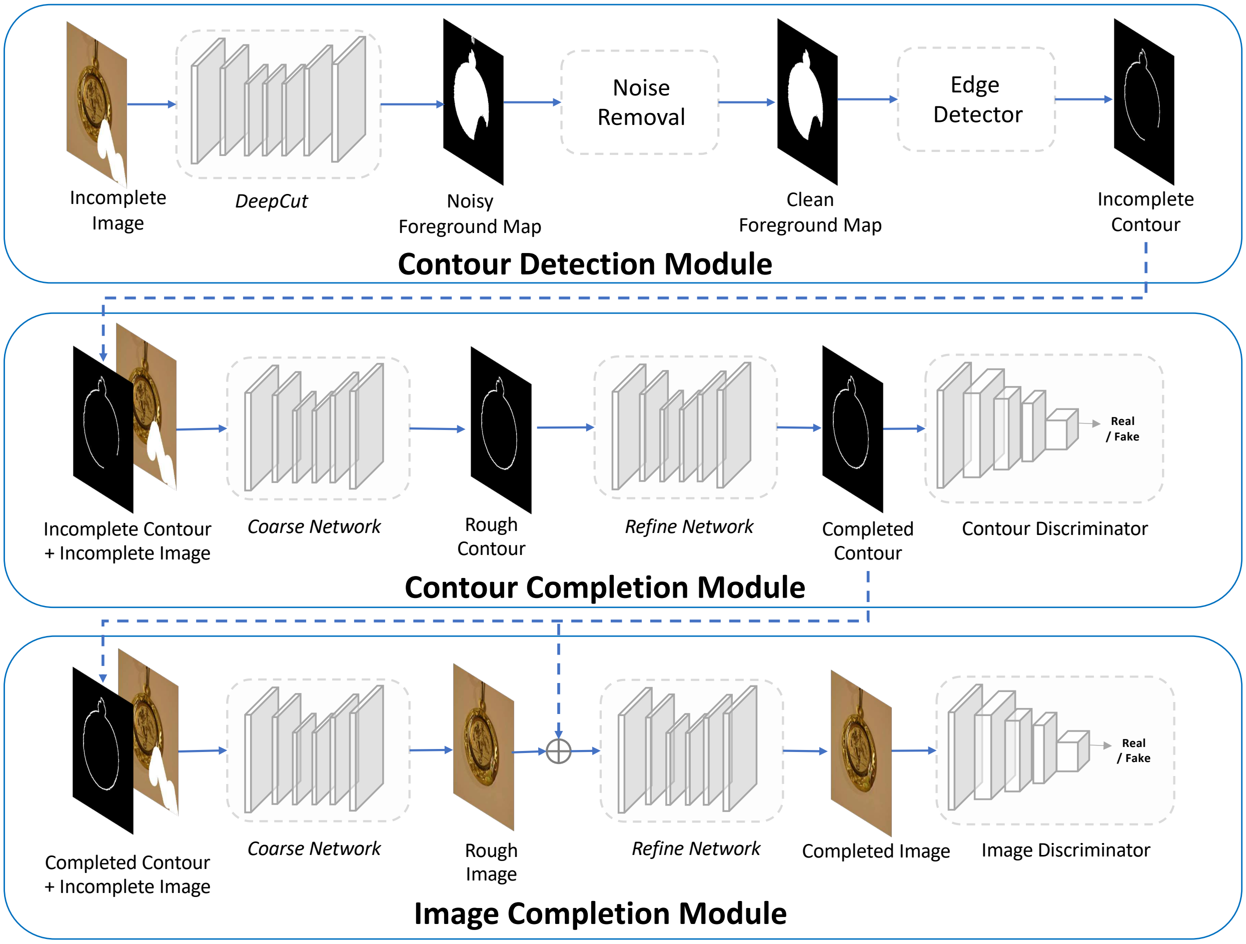


Figure 2. The overall architecture of our inpainting model.

DeepCut uses a CNN-based architecture that extracts and combines high-level and low-level features to predict a salient object mask with accurate boundaries. Since the input image is corrupted with holes, the resulting segmentation map contains noise. In some situations, holes can even be treated as salient objects. To address this issue, we use the binary hole mask to remove the regions in the segmentation map that may be mistaken as salient objects. Then we apply connected component analysis [25] to further remove some of the small clusters in the map to obtain the foreground mask. Then we adopt the Sobel Operator to detect the incomplete contour of the object from the segmentation map. The incomplete contour is then fed to the contour completion module to predict the missing contours.

3.3. Contour Completion Module

The goal of our contour completion module is to complete the missing contours of the input image that are inside the hole regions. Given the incomplete image I_{in}, incomplete contour C_{in} and the hole mask H indicating the locations of the missing pixels, we aim to predict the complete contour C_c for the corrupted foreground objects. C_c is a binary map with the same shape as the input image, with 1 indicating the boundary of the foreground objects and 0 for other pixels in the image.

3.3.1 Architecture

The contour completion module is composed of a generator and a discriminator. The generator is a cascade of a coarse network and a refinement network. For training, instead of using predicted contours, we extract a clean incomplete contour C_{in} of the foreground objects directly from the ground-truth contour C_{gt} with the hole mask H, i.e., $C_{in} = H * C_{gt}$. Then we input the incomplete image, the incomplete contour image, and the hole mask into our coarse network, which outputs a coarse complete contour C_c^{cos}. The coarse network is an encoder-decoder network with several convolutional and dilated convolutional layers. The coarse contour map is a rough estimate of the missing contours. The predicted contours around the holes can be blurry and cannot be used as an effective guidance for the image completion module.

To infer a more accurate contour, we adopt the refinement network which takes the coarse contour as input, and output a cleaner and more precise contour C_c^{ref}. The refinement network has a similar architecture as the coarse network, except that we use a contextual attention layer [29], to explicitly attend on global feature patches while inferring the missing values. Note that the pixel value of the predicted contour C_c^{ref} ranges from 0 to 1, indicating the probability that the pixel to be on the actual contour.

The refined contour is then fed to the contour discriminator for adversarial training. The contour discriminator is a fully convolutional PatchGAN discriminator [14] that outputs a score map instead of a single score, so as to tell the realism of different local regions of the generated contour mask. Unlike discriminators for images, we discover that if we only input the contour mask (generated or ground-truth) to the discriminator, the adversarial loss is hard to optimize and the training tends to fail. This may be due to the sparse nature of the contour data. Unlike the natural images which have an understandable distribution on every local region, the pixels in the contour mask is sparsely distributed and contain less information for the discriminator to judge whether the generated distribution is close to the ground-truth distribution or not.

To address this issue, we propose to adopt the ground-truth image as an additional condition, and use the image and contour pair as inputs to the contour discriminator. With this setup, the generated contour is not only required to be similar to the ground-truth contour, but also required to align with the contour of the image. The discriminator then obtains adequate knowledge to tell the difference between the generated distribution and the real distribution, and the training becomes stable.

3.3.2 Loss Functions

To train the contour completion module, we will minimize the distance between the generated contour map C_c^{cos}, C_c^{ref} and the ground-truth contour map C_{gt}. A straightforward way is to minimize the L1 or L2 distance between the masks in raw pixel space. However, this is not very effective as the contours in the mask are sparse, leading to the data imbalance problem. Determining the proper weights of each pixel is difficult. To address this issue, we propose to make use of the inherent nature of the contour mask, i.e., each pixel in the mask can be interpreted as the probability that the pixel is a boundary pixel in the original image. Therefore we can take the contour map as samples of a distribution, and calculate the distance with the ground-truth contour by calculating their binary cross-entropy between each pixel. We then adopt a focal loss [16] to balance the importance of each pixel. Since our primary goal is to complete the missing contours, we pay more attention to the pixels in the holes by assigning them a larger weight. We formulate this loss as the content loss for contour completion L_{con}^C. The final loss function for the coarse contour is:

$$
\begin{aligned}
&\mathcal{L}_{con}^C\left(C_c^{cos}, C_{gt}\right) \\
&= \frac{\lambda}{N}\sum_p \left(H[p](C_c^{cos}[p] - C_{gt}[p])^2 \mathcal{L}_e(C_c^{cos}[p], C_{gt}[p])\right) \\
&+ \frac{1}{N}\sum_p \left((1 - H[p])(C_c^{cos}[p] - C_{gt}[p])^2 \mathcal{L}_e(C_c^{cos}[p], C_{gt}[p])\right) .
\end{aligned}
\tag{1}
$$

where [p] denotes to the pixel spatial location of the contour map, N is the number of pixels in the contour map, $\mathcal{L}_e(x, y)$ is the binary cross-entropy loss function, x and y are predicted probability score and the ground-truth probability, respectively.

Similarly, we use a content loss for the refined contour $\mathcal{L}_{con}^C\left(C_c^{ref}, C_{gt}\right)$. The final content loss function for contour completion is:

$$
\mathcal{L}_{con}^C = \mathcal{L}_{con}^C(C_c^{cos}, C_{gt}) + \mathcal{L}_{con}^C(C_c^{ref}, C_{gt}) . \tag{2}
$$

The focal loss helps to generate a clean contour. However, we observe that although we are able to reconstruct sharp edges in the uncorrupted regions, the contours in the corrupted regions are still blurry. To encourage the generator to produce sharp and clean contours, we use the contour discriminator D^C to perform adversarial learning. Specifically, we use the recent technique called Spectral Normalization [20] to stabilize the training of the GAN model. We use the hinge loss function to determine whether the input is real or fake. The adversarial loss for training the contour discriminator and the generator are as follows, respectively, where σ denotes the ReLU function.

$$
\mathcal{L}_{adv}^C = \mathbb{E}[\sigma(1 - D^C(C_{gt}))] + \mathbb{E}[\sigma(1 + D^C(C_c^{ref})] . \tag{3}
$$

$$
\mathcal{L}_{adv}^C = -\mathbb{E}[D^C(C_c^{ref})] . \tag{4}
$$

3.3.3 Curriculum Training

Completing the contours is a challenging task. Although we have adopted a focal loss to balance the sparse data, and a spectral normalization GAN to obtain sharper results, we observe that it is still difficult to train the whole contour completion module. The training tends to fail if both the content loss and the adversarial loss are applied simultaneously even though the weights between the two types of losses are carefully adjusted. To avoid the issue, we use curriculum learning to gradually train the model. In the first stage, the contour completion module is required only to output a rough contour, thus we only train the model with the content loss. Then in the second stage, we fine-tune the pre-trained network with our adversarial loss, but with a very small weight compared to the content loss, i.e., 0.01 : 1 to avoid training failure due to the instability of the GAN loss for contour prediction. In the third stage, we fine-tune the whole contour completion module with the weight of adversarial loss and the weight of content loss to be 1:1.

3.4. Image Completion Module

3.4.1 Architecture

Guided by the completed contours, our model gains the basic knowledge of where the foreground and background pixels are. This knowledge provides strong clues for the completion of the image. The image completion module takes the incomplete image I_{in}, the completed contour and the hole mask H as inputs, and outputs the completed image I_c. It shares a similar architecture as the contour completion module. The generator of our image completion module also contains a coarse network and a refinement network. The coarse network outputs a coarsely completed image, which can be blurry with missing details. Then the refinement network takes the coarse image as input, and generates a more accurate result.

By inputting both the incomplete image and completed contour to the coarse network, however, we observe that the final output of the generator tends to ignore the guidance of the completed contour. The shape of the generated image is not consistent with the input contour in the hole regions. This problem may be caused by the depth of the image completion networks. After layers of mapping, the knowledge provided by the completed contour can be forgotten or weakened, due to error accumulation. To tackle this problem, we input the completed contour to both the coarse network and the refinement network to enhance the effect of the condition. In this way, the effect of the contour condition can be stronger in the second stage of the image completion module.

The discriminator takes the generated image/ground-truth image along with the hole mask indicating the location of the missing pixels as inputs, and tells whether the input pair is real or fake. Similar to the contour completion module, we use a PatchGAN structure and a hinge adversarial loss to train the model.

3.4.2 Loss Functions

The loss function for the image completion module also consists of a content loss $\mathcal{L}^I_{con}$ and an adversarial loss $\mathcal{L}^I_{adv}$. The adversarial loss has a very similar form as the loss for contour completion, except that we apply the loss to the images instead of the contours. Note that the adversarial loss is only applied to the result of the refinement network. We do not apply the loss to the result of the coarse network. For the content loss, we use L1 loss to minimize the distance between the generated image and the ground-truth image. The image content loss is:

$$\mathcal{L}^I_{con} = \frac{1}{N}\sum_p \left(|I_c^{cos}[p] - I_{gt}[p]| + |I_c^{ref}[p] - I_{gt}[p]|\right) . \tag{5}$$

where I_c^{cos}, I_c^{ref} and I_{gt} are the output of the coarse network, the refinement network, and the ground-truth, respectively. [p] denotes the pixel spatial location of the image, N is the number of pixels in the image.

3.4.3 Training

Our image completion module is first pre-trained on the large-scale Places2 dataset without the extra channel for the contour map, then fine-tuned on the saliency dataset with the guidance from the output of the contour completion module. Since the network we will fine-tune on the saliency dataset takes different inputs (takes additional contour as input) compared to the network we pretrain on the Places2 dataset, when fine-tuning our network, we keep the parameters of all the layers in the pretrained network except the first layer, and randomly initialize the first layers of our image completion module. To stabilize the training, we use a similar curriculum training strategy as the training of the contour completion module.

There are two variations in our training process. The first one is to fix the parameters of the contour completion module, and only fine-tune the image completion module. The second way is to jointly fine-tune both modules. In our experiments, we observe that there are minor differences between these two so we fix our method as the second setting.

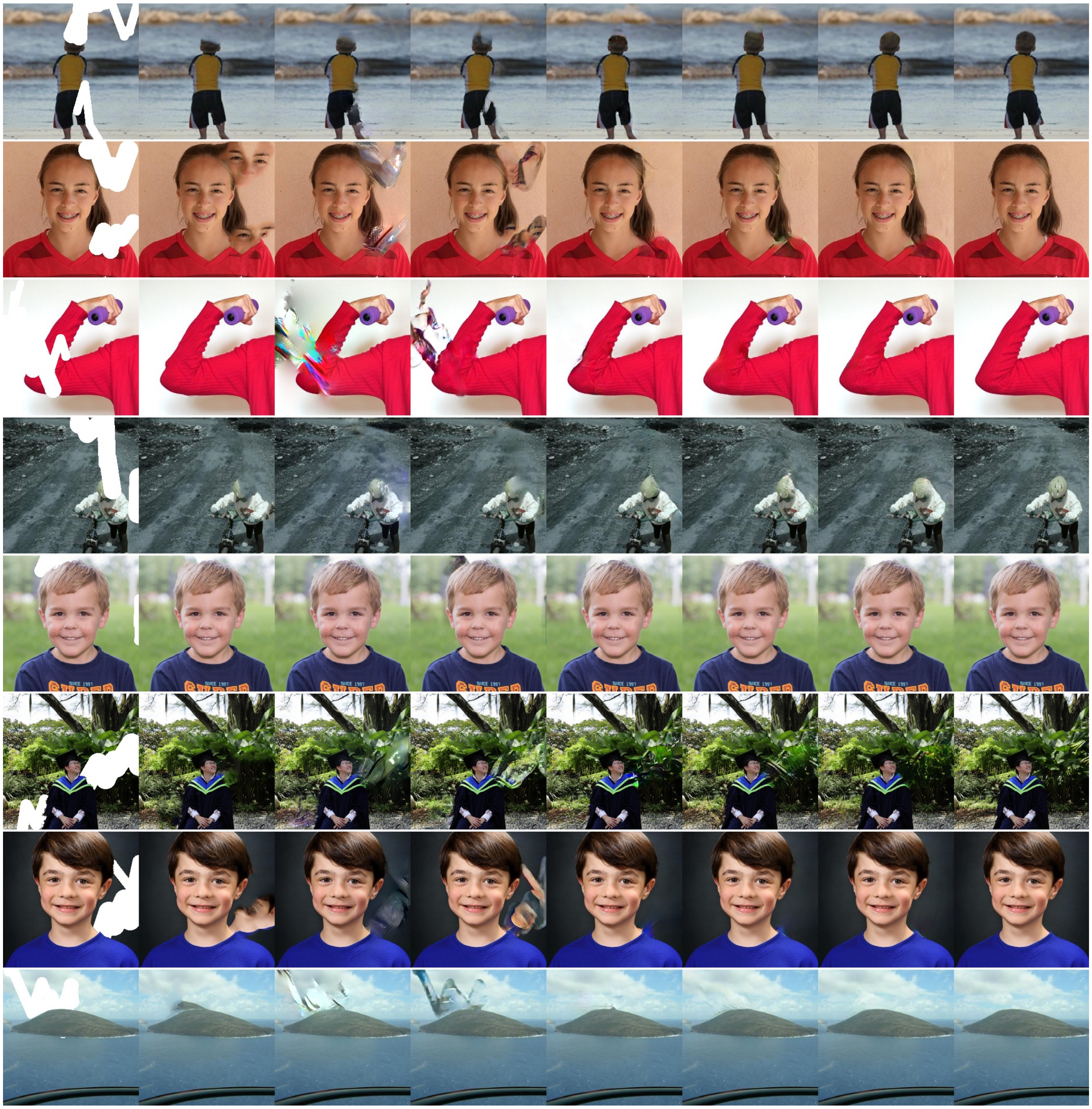
4. Experiments

4.1. Implementation Details

We obtain the incomplete contour of the foreground objects from our contour detection module and the pretrained DeepCut model [1], without any finetuning. Then we train our contour completion module only on the saliency dataset. On the third stage, we first train the image completion module on the Places2, then finetune it on our saliency dataset. We also finetune both the contour completion module and the image completion module end-to-end on our saliency dataset. We use Adam as the optimizer, with a learning rate of 0.0002 and batchsize of 64 for both the contour completion module and the image completion module. λ in Eq. 1 is set to 5 on training the contour completion module.

4.2. Comparison with state-of-the-arts methods

In this part, we compare our proposed model with the state-of-the-art image inpainting methods on the validation set of our saliency dataset. We compare our full method (denoted as "Ours Guided") with GatedConv [28], PartialConv [17], ContextAttention [29], Global&Local [13], and



Input PatchMatch Global&Local ContextAttention PartialConv GatedConv Ours Ground-Truth

Figure 3. Qualitative comparison between the state-of-the-art methods. Row 1-4 are samples with overlapped holes, while Row 5-8 are samples with non-overlapped holes. Please zoom in to see the details.

PatchMatch [5]. For a fair comparison, we also compare with GatedConv [28] fine-tuned on our saliency dataset, which can be regarded as the baseline - our model without contour prediction and guidance (denoted as "Ours No Guide").

4.2.1 Quantitative Evaluation

We randomly select 500 images from the testing saliency dataset and generate both overlap and non-overlap holes for each image. Then we run each method on the corrupted images to obtain the final results. We use common evaluation metrics, i.e., L1, L2, PSNR, and SSIM, calculated using the complete image and the ground-truth image in pixel space, to quantify the performance of the models. Table 1 shows the evaluation results. Among the deep learning-based methods, our models outperform all the other methods in all four metrics. The results can be explained by that existing methods only consider making the textures of the completed image realistic, but ignore the structures of the image. Furthermore, our model with contour guidance brings consistent improvements over the baseline without

Table 1. Quantitative results on the saliency dataset.

Method	L1 Loss	L2 Loss	PSNR	SSIM
PatchMatch [5]	0.01386	0.004278	26.94	0.9249
Global&Local [13]	0.02450	0.004445	25.55	0.9005
ContextAttention [29]	0.02116	0.007417	24.01	0.9035
PartialConv [17]	0.01085	0.002437	29.24	0.9333
GatedConv [28]	0.009966	0.002531	29.26	0.9353
Ours No Guided	0.010002	0.002597	29.35	0.9356
Ours Guided	**0.009327**	**0.002329**	**29.86**	**0.9383**

guidance, demonstrating the validity of our proposed idea of leveraging contour prediction.

4.2.2 Qualitative Evaluation

Fig. 3 shows visual comparisons of our method with existing methods. Seen from the figure, PatchMatch [5] generates quite smooth textures. However, since it lacks an understanding of the image semantics, the generated image is not visually realistic when the holes are near the boundary of the foreground objects. Although Global&Local [13] and ContextAttention [29] show the potential of handling holes with arbitrary shape (e.g., combining multiple small square holes to form an arbitrary shaped hole), since they are not specifically trained on arbitrary-shaped hole masks, they can generate artifacts which make the images unrealistic. PartialConv [17], GatedConv [28] and our model without contour guidance (denoted as "Ours No Guide") can generate smooth and plausible images, but artifacts still exist around the borders of the objects. In addition, the shapes of the generated objects are not as natural as the real-world objects. Our full contour guided model not only generates a completed image with less artifacts, but also well completes the missing parts of the objects so that they have a very natural boundary.

4.2.3 User Study

To make a more thorough evaluation of our method in terms of visual quality, we conduct a user study and show the result in Table 2. Specifically, we randomly select 50 images from our testing dataset, corrupt them with random holes and then obtain the inpainted results of each method. We show the results of each image to 22 users and ask them to select a single best result. Finally we collect 1,099 valid votes from all users. We count the number of times that each method is preferred by users. Table 2 shows the user preferences of each method. Our full model is preferred the most, outperforming all the other methods by a large margin. This demonstrates the superiority of our foreground-aware model in terms of visual quality.

4.3. Ablation Study

We also analyze how our contour completion module contributes to the final performance of image inpainting. We compare our full model to the model without contour

Table 2. User preference for the results of each method.

Method	Preference Counts
PatchMatch [5]	23
Global&Local [13]	5
ContextAttention [29]	4
PartialConv [17]	90
GatedConv [28]	100
Ours No Guide	146
Ours Guided	**731**

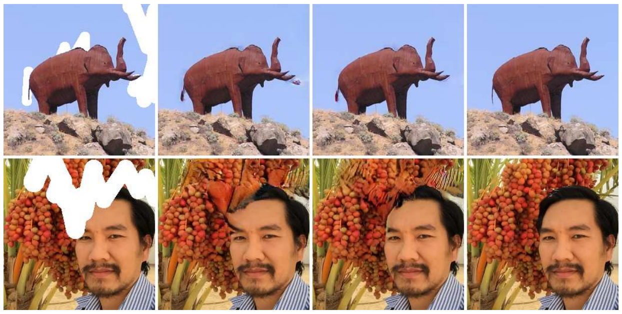


Figure 4. Comparison between our model with/without contour as guidance. From left to right: input image with holes, our model without contour guidance, our full model, the ground-truth.

as guidance, as is shown in Fig. 4. The top row shows the results where holes have no overlap with the foreground object, while the bottom shows the case where holes overlap with the object. In both cases, our model without contour guidance generates obvious artifacts around the border of the foreground object, while our model with contour guidance can infer object boundaries correctly and produce realistic inpainting results. The comparison indicates that the completed contours greatly improve the performance of the image inpainting model and that contour guidance is a crucial part to the success of our model.

5. Conclusion

In this paper, we propose the foreground-aware image inpainting model for challenging scenarios involving prediction of both foreground and background pixels. Our model first detects and completes the contours of the foreground objects in the image, then uses the completed contours as a guidance to inpaint the image. It is trained on a specifically collected saliency image dataset. Experiments show that our model can generate natural contours of objects, which are of great benefit for image completion. Our model significantly outperforms various state-of-the-art models both quantitatively and qualitatively. This shows that using structures to indicate the foregrounds and backgrounds of the input image, then explicitly guide the completion of the image is a promising direction for inpainting tasks.

6. Acknowledgements

This work is partially supported by NSF awards #1704309, #1722847 and #1813709.

References

[1] Anonymous. Deep y-net: A neural network architecture for accurate image segmentation. 2017.

[2] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916, May 2011.

[3] Michael Ashikhmin. Synthesizing natural textures. In *Proceedings of the 2001 symposium on Interactive 3D graphics*, pages 217–226. ACM, 2001.

[4] Coloma Ballester, Marcelo Bertalmio, Vicent Caselles, Guillermo Sapiro, and Joan Verdera. Filling-in by joint interpolation of vector fields and gray levels. *IEEE transactions on image processing*, 10(8):1200–1211, 2001.

[5] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (TOG) (Proceedings of SIGGRAPH 2009)*, 2009.

[6] Marcelo Bertalmio, Luminita Vese, Guillermo Sapiro, and Stanley Osher. Simultaneous structure and texture image inpainting. *IEEE transactions on image processing*, 12(8):882–889, 2003.

[7] Tali Dekel, Chuang Gan, Dilip Krishnan, Ce Liu, and William T Freeman. Sparse, smart contours to represent and edit images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3511–3520, 2018.

[8] Alexei A Efros and Thomas K Leung. Texture synthesis by non-parametric sampling. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 1033–1038. IEEE, 1999.

[9] 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*, pages 2672–2680, 2014.

[10] Qibin Hou, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, and Philip Torr. Deeply supervised salient object detection with short connections. In *IEEE CVPR*, pages 3203–3212, 2017.

[11] Jia-Bin Huang, Sing Bing Kang, Narendra Ahuja, and Johannes Kopf. Image completion using planar structure guidance. *ACM Transactions on graphics (TOG)*, 33(4):129, 2014.

[12] Jason C Hung, Chun-Hong Huang, Yi-Chun Liao, Nick C Tang, and Ta-Jen Chen. Exemplar-based image inpainting base on structure construction. *JSW*, 3(8):57–64, 2008.

[13] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *ACM Transactions on Graphics (TOG)*, 36(4):107, 2017.

[14] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint*, 2017.

[15] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[16] Tsung-Yi Lin, Priyal Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[17] Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. *arXiv preprint arXiv:1804.07723*, 2018.

[18] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[19] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[20] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[21] David M Mount and Sunil Arya. Ann: library for approximate nearest neighbour searching, 1998.

[22] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2536–2544, 2016.

[23] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[24] Eftychios A Pnevmatikakis and Petros Maragos. An inpainting system for automatic image structure-texture restoration with text removal. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pages 2616–2619. IEEE, 2008.

[25] Hanan Samet and Markku Tamminen. Efficient component labeling of images of arbitrary dimension represented by linear bintrees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(4):579–586, 1988.

[26] Denis Simakov, Yaron Caspi, Eli Shechtman, and Michal Irani. Summarizing visual data using bidirectional similarity. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[27] Yuhang Song, Chao Yang, Yeji Shen, Peng Wang, Qin Huang, and C-C Jay Kuo. Spg-net: Segmentation prediction and guidance network for image inpainting. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2018.

[28] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. *arXiv preprint arXiv:1806.03589*, 2018.

[29] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. *arXiv preprint*, 2018.

[30] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.