博士論文

Image-to-Image Translation for Content Creation Support (コンテンツ生成支援のための画像変換技術)

48-187402

井上　直人

指導教員　山﨑　俊彦　准教授

東京大学大学院　情報理工学系研究科　電子情報学専攻

This dissertation is submitted for the degree of
Doctor of Philosophy

December 2020

Acknowledgements

I am incredibly grateful to my supervisor, Prof. Toshihiko Yamasaki, for his continuous supports, invaluable advice, and patience during my doctoral study. His profound knowledge and professional experience always have encouraged me in my academic research. I would like to express sincere appreciation to Prof. Kiyoharu Aizawa for countless hours of discussion and mentoring at Aizawa-Yamasaki-Matsui Lab. His questions based on a deep understanding on broad research areas sometimes shed light on valuable viewpoints that I would not come up with. I also thank all the current and previous Aizawa-Yamasaki-Matsui Lab members for fruitful discussion and having various and funny conversations. Especially, I thank Prof. Yusuke Matsui and Dr. Ryosuke Furuta, who were doctoral students when I first started to do research as an undergraduate student. They have been really passionate about their research topics, and I learned a lot from them.

I would like to gratefully acknowledge the committee members, Prof. Shin'ichi Satoh, Prof. Naoki Yoshinaga, and Prof. Yoichi Sato, for their insightful comments during my preparation for the dissertation. I appreciate my greatest collaborators, Daichi Ito, Dr. Ning Xu, Dr. Jimei Yang, Dr. Long Mai, Dr. Yannick Hold-Geoffroy, and Dr. Brian Price, with a wonderful industrial research experience at Adobe Research. It was exciting to work with the researchers who have published many excellent papers and simultaneously shipped them to real products used worldwide. I also thank Dr. Edgar Simo-Serra at Waseda University for his suggestions and recommendations for internships. I gratefully acknowledge the generous financial support for my Ph.D. research by The University of Tokyo - NEC AI scholarship for three years.

Finally, I would like to express the deepest gratitude to my parents for supporting me all the time.

Abstract

The emergence of digital tools has opened up an opportunity for ordinary people to create visual content digitally, such as images and videos. However, many steps still have to be done manually and are time-consuming in the creative workflow. Many methods that go beyond classical low-level filters have been introduced, such as image segmentation, owing to the recent evolution of machine learning methods.

Image-to-image translation (I2IT) is a task to convert an image in one domain to a corresponding image in another domain. I2IT is essential since many tasks in visual content creation are I2IT. Recent works have established a unified solution for supervised I2IT using convolutional neural networks (CNN). All one has to do is to prepare many pixel-aligned input-target image pairs for supervised learning. However, sometimes it is impossible or significantly hard in terms of time amd cost to collect the paired data. In this thesis, we have tackled several I2IT tasks that suffer from the aforementioned issue by exploring underlying priors to form an image.

First, we address the task of tracing photographs to create line drawings. We propose a two-stage extract-and-refine model to immitate the tracing workflow and present a self-supervised loss to learn the refinement process. We confirm that our model produces much more clean and expressive line drawings and illustrate two practical applications.

Second, we propose a novel task of adding a non-directional shading effect on an image. Inspired by approaches for a real-time approximation of global illumination in rendering, we propose to generate ambient occlusion (AO) map from an input image as a proxy task. We show that our model trained on synthetic RGB-AO pairs dataset performs well on a broad range of images and illustrate image editing applications: image composition and geometry-aware contrast enhancement.

Third, we address the task of shadow removal and detection. We derive a pipeline to synthesize the dataset from a classical physically-grounded shadow illumination model. We demonstrate that shadow removal and detection models trained on the dataset from our pipeline generalizes well to a broad range of datasets, thanks to the synthesized data's fidelity and diversity.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Learning-based I2IT and Content Creation

Content creation is a way for conveying thoughts and express ideas through some medium such as images, sounds, texts, and a combination of any of them. Technology to support content creation has changed drastically by the emergence of digital tools, enabled by a substantial increase in computers' speed and capacity in the past half-century. In addition to professional designers and artists, millions of novice users can easily create digital content using off-the-shelf software on their laptops, such as Adobe Photoshop, Microsoft Powerpoint, and Autodesk Maya for visual content creation. In the last decade, many methods based on machine learning research using neural networks (NNs) have been introduced to the digital tools. Such methods do not generally aim to replace the whole creation process. Instead, they aim to automate manual tasks that still exist in digital creation workflow (e.g., image matting) and spare users' time and effort to concentrate on more creative works.

In this thesis, our focus is in *image-to-image translation* (I2IT). In I2IT, an image $\boldsymbol{x}$ is converted to another image $\boldsymbol{y}$ that meets the requirement (e.g., grayscale to RGB images in the case of image colorization). Isola *et al*. [12] named the conversion as image-to-image translation, which is analogous to language translation. This operation has been quite common in content creation workflow. For instance, many classical filters have already been deployed in commercial software for visual content creation. The advantage of learning-based approaches such as those based on convolutional neural network (CNN) is in mainly two ways:

- It offers a unified approach to fast approximation of existing various image filters, which sometimes are very slow [13].
- It enables us to create any new conversion task by just giving examples for the training [12, 14], without trying to represent the conversion using complex equations.

Owing to the versatility, I2IT has been applied to a broad range of problems. We describe some super-concepts and tasks related to them. The first is image enhancement, which is to improve the interpretability or perception of information in images, such as image denoising [15, 16], image debluring [17–19], image super-resolution [20, 21], image inpainting [22–25] and image colorization [26–29]. The second is guided image synthesis, which is to generate an image from less complex image-like data or user inputs, such as label-to-image [12, 14, 30, 31], sketch-to-image [32, 33], and pose / skeleton-to-image [34, 35]. The third is image manipulation, which is to change the style of an image at various levels, such as style transfer [2, 36–38], semantic manipulation [39–43], and attribute manipulation [44–46].

In this thesis, the aim is to infer the underlying geometric structure of images essential for a single image manipulation by I2IT. Photos we see every day are the projection of the 3D scene on the 2D plane. However, without accurate capture or very precise estimation of the 3D scene such as surface geometry and lighting, humans can perform many geometry relevant tasks from a single image. We are motivated by this and propose some I2IT tasks by a learning-based approach in two areas, **(i) shape extraction** and **(ii) shading manipulation** in content creation. The rest of the introduction chapter is organized as follows. In Sec. 1.2, we discuss challenges common to all learning-based approaches for I2IT, which we also should overcome. In Sec. 1.3, we present an overview of our approach for shape extraction. In Sec. 1.4, we present an overview of our approach for shading manipulation.

1.2 Research Challenges

We first explain one of the most fundamental approach for learning-based I2IT, Pix2pix [12]. Then we describe the research challenges common among many learning-based I2IT models. Pix2pix is based on generative adversarial network (GAN) [47]. GAN is a framework for estimating generative models by jointly optimizing two models: a generator G that captures data distribution and a discriminator D that judges whether a sample is coming from real distribution (i.e., training data)

rather than fake distribution (i.e., an output of G). Pix2pix not only learns the mapping but also learns a loss function to facilitate training by D simultaneously. For the paired supervision, $(\boldsymbol{x}, \boldsymbol{y})$ are sampled. Random variable $\boldsymbol{z}$ are sampled from a prior distribution such as standard normal distribution. The optimization is performed as follows:

$$G^*, D^* = \underset{G}{\operatorname{argmin}} \underset{D}{\operatorname{argmax}} \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G), \tag{1.1}$$

where λ is a hyper-parameter to balance the two objectives. The first term $\mathcal{L}_{cGAN}(G, D)$ is based on conditional GAN (cGAN) [48]. G takes both $\boldsymbol{x}$ and $\boldsymbol{z}$ as the inputs, and generate the image $G(\boldsymbol{x}, \boldsymbol{z})$. The objective is expressed as:

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}[logD(\boldsymbol{x}, \boldsymbol{y})] + \mathbb{E}_{\boldsymbol{x},\boldsymbol{z}}[log(1 - D(\boldsymbol{x}, G(\boldsymbol{x}, \boldsymbol{z})))]. \tag{1.2}$$

The second term $\mathcal{L}_{reg}(G)$ is for simply solving a regression task. As the metric, L1 distance is used to suppress blurry outputs:

$$\mathcal{L}_{reg}(G) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y}),\boldsymbol{z}}\left[\left\|\boldsymbol{y} - G(\boldsymbol{x}, \boldsymbol{z})\right\|_1\right]. \tag{1.3}$$

Despite the versatility, there are several drawbacks to Pix2pix. The first drawback is that the result is limited to low-resolution and not realistic images, especially when generating photorealistic images from sparse or simple inputs. Recent approaches have tackled this issue by introducing new NN building blocks, such as multi-scale discriminator / generator [30] and spatially-adaptive normalization (SPADE) [31]. The second drawback is that there is minor stochasticity in the outputs despite $\boldsymbol{z}$. This drawback has been addressed by imposing some cycle-consistency using encoder for $\boldsymbol{z}$ [49] or adding regularization loss [50] on $\boldsymbol{z}$, and using conditional generative flow [51].

The last and most significant drawback is that it is sometimes difficult or impossible to obtain the paired data $(\boldsymbol{x}, \boldsymbol{y})$. This issue is fatal when the objective is to modify semantic attributes such as facial expressions. Recent approaches have tried to learn the mapping from unpaired data $\{\boldsymbol{x}_1, \boldsymbol{x}_2, ...\}$ and $\{\boldsymbol{y}_1, \boldsymbol{y}_2, ...\}$ such as CycleGAN [14] and DiscoGAN [52]. These approaches are further extended to tackle harder tasks, such as many-to-many or example-guided mapping [40, 53] and translation between multiple domains [44, 46, 41, 42]. However, the performance of these approaches are still behind their supervised counterparts in fidelity and efficiency, and most of them are not deployed for real products compared to the

supervised I2IT models. In contrast to the task-agnostic approaches to improve I2IT, our primal focus is to enhance the performance of supervised I2IT by developing new networks and losses based on task-specific priors in this thesis.

1.3 Shape Extraction

In the first half of the thesis, we address I2IT for shape extraction. Specifically, we are interested in line drawings made by a human. These drawings can capture the semantics of a scene in different levels of abstraction and are one of the most fundamental components for artwork creation. We develop an approach to obtain line drawing images by I2IT.

In Chapter 2, we present a new computational model for automatically tracing high-resolution photographs to create expressive line drawings. We define expressive lines as those that convey important edges, shape contours, and large-scale texture lines necessary to accurately depict the overall structure of objects (similar to those found in technical drawings) while still being sparse and artistically pleasing. Given a photograph, our algorithm extracts expressive edges and creates a clean line drawing using CNN. We construct two new datasets for the face / body and manga background domain and employ end-to-end trainable fully-convolutional CNN to learn the model in a data-driven manner. The model consists of two networks to cope with two sub-tasks, extracting coarse lines and refining them to be more clean and expressive, inspired by the artistic workflow. We also propose an automatic data generation method to facilitate the training of the latter task. The experimental results qualitatively and quantitatively demonstrate the effectiveness of our model. We further illustrate two practical applications, manga background generation and coloring books.

1.4 Shading Manipulation

In the latter half of the thesis, we address I2IT for manipulating shading in a single image. Shading changes the image remarkably since it serves as an important visual cue for a human to perceive 3D information from a single image. However, shading is hard to infer from a single image since it results from a complex interaction between lights and the geometry of a scene in 3D space. We take inspiration from a global illumination model in computer graphics in Chapter 3 and an image formation

equation considering illumination and reflectance in Chapter 4 to develop novel approaches for shading manipulation.

In Chapter 3, we focus on ambient occlusion (AO), which offers adding a non-directional shading effect that darkens enclosed and sheltered areas. It is not physically accurate but widely used to approximate the global illumination effect for real-time computer graphics rendering. We present RGB2AO, a novel task to generate AO from a single RGB image input instead of screen space buffers such as depth and normal. RGB2AO aims to enhance two 2D image editing applications: image composition and geometry-aware contrast enhancement. We first collect a synthetic dataset consisting of pairs of RGB images and AO maps. Subsequently, we propose a model for RGB2AO by supervised learning of CNN, considering the 3D geometry of the input image and data augmentation strategy specific to AO. Experimental results quantitatively and qualitatively demonstrate the effectiveness of our model.

In Chapter 4, we focus on adding a directional shading effect. Shadow removal is essential for pre-processing in computer vision since strong shadow degrades the performance of object recognition tasks such as object detection, as shown in [54]. We present SynShadow, a novel large-scale synthetic shadow/shadow-free/matte image triplets dataset, and a pipeline to synthesize it. We extend a classical physically-grounded shadow illumination model and synthesize a shadow image given an arbitrary combination of a shadow-free image, a matte image, and shadow attenuation parameters. Recent shadow removal approaches all train I2IT models on real paired shadow/shadow-free or shadow/shadow-free/mask image datasets. However, obtaining a large-scale, diverse, and accurate dataset has been a big challenge, and it limits the performance of the learned models on shadow images with unseen shapes/intensities. Owing to the diversity, quantity, and quality of SynShadow, we demonstrate that shadow removal models trained on SynShadow perform well in removing shadows with diverse shapes and intensities on a challenging USR benchmark. Furthermore, we show that merely fine-tuning from a SynShadow-pre-trained model improves existing shadow detection and removal models.

Chapter 2

Learning to Trace: Expressive Line Drawing Generation from Photographs

2.1 Introduction

Tracing is a fundamental step in the creation of artworks. When artists trace a photograph, this process is similar to extracting expressive lines while suppressing trivial ones. Tracing a photograph does not mean following visually salient edges; sometimes, a wavy line is simplified to a straight line, disconnected lines are joined into a single line, or some texture-like lines are ignored, to make the traced line drawing more expressive, clean, and editable. It has a diverse range of applications. For example, manga (i.e., Japanese comic) artists trace photographs of indoor or outdoor scenes to produce line drawings that are subsequently processed to create background scenes for manga.

Despite the importance of tracing a photograph, it is still challenging for both humans and machines. Even for professional artists, it may be very tedious and time-consuming. It can take several hours for manga artists to process a single photograph since they trace a lot of expressive lines carefully. For novice users, it may be beyond their capability. For machines, it is not always clear which texture lines are important to keep and which we should remove because tracing a photograph requires the semantic understanding of objects and scenes. Furthermore, subtle gradients are sometimes actually important lines. For example, the edges around the nose are sometimes very subtle, while artists do not miss to trace them. Although one

Figure 2.1: Examples produced by our model. Given an input photograph on the left, our model automatically generates a line drawing image on the right. The photograph of the woman is from svklimkin (Public Domain). (best viewed with zoom and in color.)

may try to compose some low-level image processing filters to replace the process, they cannot incorporate the semantic understanding thus the results contain small textures and other spurious lines, giving a look of filtered photographs but not line drawings, as we will later show in Sec. 2.6.1.

In this chapter, we aim to convert a photograph to a line drawing **automatically** in a raster format, as shown in Fig. 2.1. The produced line drawing should be expressive in that it conveys important edges, shape contours, and large-scale texture lines that are necessary to accurately depict the overall structure of an object, which is similar in spirit in Suggestive Contour [55] for rendering 2D images from 3D shapes. The produced line drawing should also be clean so that it is still artistically pleasing and fulfill the following requirements as many as possible for further editing:

- The lines are smooth but not fragmented or interrupted.
- There are no small texture-like line and noise.
- The width and intensity of the lines are consistent.

We propose to learn such tracing rules by training a convolutional neural network (CNN). It is challenging in two points. The first point is in a lack of data. It should consist of photograph and line drawing pairs, but collecting them costs a lot because they should be from the works of professional workers. Even if they are collected, the number of data is still not enough to learn all the rules sufficiently. The second point is in the density and complexity of lines that the learned model must

produce. Applying existing machine-learning-based 2D edge detection methods for photographs such as [56], which are originally designed for detecting a sparse set of edges, fails to fulfill the requirements.

We describe how we solve this task. First, we first curate datasets for two domains: *face / body* and manga background (manga BG) by collecting paired photographs and line drawings from artists and carefully-designed 3D renderings. Second, for the generation model, we propose an end-to-end trainable CNN that consists of two modules: the *generator* and the *restorer*. The *generator* extracts coarse lines, while the *restorer* refines the lines to be clean and expressive. We augment the dataset with synthetically-deteriorated line drawings to strengthen the *restorer*'s ability of refinement. Results show that our model can produce cleaner and more expressive line drawings than compared approaches regarding subjective evaluation. We further demonstrate potential applications that use the line drawing results generated from our model to render a manga BG image or a coloring book.

In summary, our contributions are as follows:

- The first reported approach to automatically produce expressive line drawings from photographs.
- A novel CNN architecture that can generate very clean and expressive line drawings. Our model is fast and can produce high-resolution results (we can process a 4096×3072 image in 1.57 seconds).
- A new dataset for learning to trace from photographs.
- Manga and coloring book applications of the proposed photograph tracing method.

2.2 Related Work

2.2.1 2D Edge Detection

Method

Edge detection methods from a single 2D image are usually designed so that the results can be further used for mid-level to high-level vision tasks, such as optical flow estimation, image segmentation, and object proposal generation. The history of edge detection is very long. We only refer to representative works in this chapter. Early approaches for edge detection find boundaries by convolving

the input grayscale image with pre-defined local derivative filters [57]. More recent approaches combine local color and texture cues and employ learning-based techniques [58, 3, 59]. However, the performance of these methods is still far below that of human perception.

Recently, methods based on deep CNN have achieved remarkable progress in this field [60, 61]. HED [60] can detect multi-scale edges simultaneously by fusing hierarchical representations extracted from fully-convolutional CNN. However, the lines extracted by HED are very blurry and thick. LPCB [56] proposes a new loss function to enhance the crispness. However, we show that LPCB still performs poorly in our setting to produce crisp and expressive line drawing. We propose a two-stage model by augmenting priors for artistic plausibility inspired by the recent line drawing generation methods.

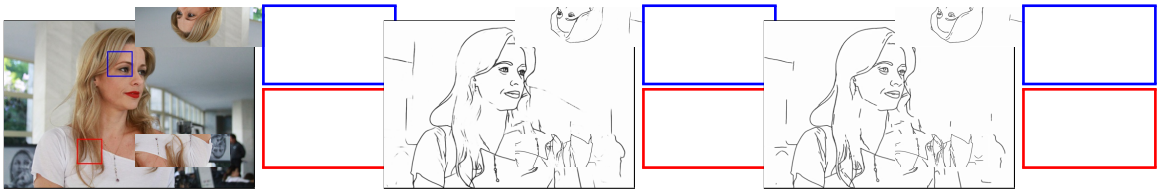
Dataset

The BSDS500 dataset [3], which mainly contains object boundaries, has been used for both training and evaluation of edge detection algorithms for a long time. [4] introduce Contour Drawing, which is a new dataset of contour drawings. It not only contains object boundaries, but also visually salient inner edges such as occluding contours, and visually salient background edges.

There are two points that our datasets for *face / body* and *manga BG* differ from the existing datasets. First, the number of lines to be extracted in images in our tracing datasets are much larger than those in the existing datasets to capture fine details, which is necessary for artworks. Second, the lines drawn by artists follow some expressive but sometimes visually subtle edges and do not necessarily follow the visually salient edges, which we show examples in Sec. 2.4. We collect our tracing datasets to learn such rules in a data-driven fashion.

2.2.2 3D Non-Photorealistic Rendering

It is possible to generate line drawings by non-photorealistic rendering (NPR) algorithms from 3D geometry nowadays [62]. A recent study [63] on how artists draw a single 3D object provides artists with a 3D model rendered from a given viewpoint and illumination and asks them to draw it to create a line drawing. They showed that the average matching rate of line drawings created by artists with those by existing NPR-based algorithms was around 80%. Although the number and variety of 3D objects are increasing, it is still tricky to compose a whole target



(a) Input $\boldsymbol{x}$ (b) Baseline model, $G(\boldsymbol{x})$ (c) Our model, $R(G(\boldsymbol{x}))$

Figure 2.2: Comparison with the results of the baseline model G and our model for *face / body*. The lines extracted by G are very blurry and choppy. The lines are with inconsistent intensity and thickness, and fail to preserve adequate details for the eye of the person. On the other hand, our model can produce much less blurry and choppy lines with consistent intensity and thickness owing to the restorer R. For more details of these models, please refer to Sec. 3.4. Photograph of Assembleia Legislativa do Parana (Public Domain). (best viewed with zoom and in color.)

scene or object consisting of a lot of 3D models manually. In contrast, our model uses a 2D photograph as the input to produce line drawings. We argue that it is complementary to the NPR-based approach. It only requires a 2D reference image as the input. The process of the proposed line drawing generation is fully automatic, and the runtime is independent of image complexity, i.e., a number of lines.

2.2.3 Line Drawing Generation

Converting a grayscale rough sketch image to a simplified clean line drawing image [64, 5, 65–67] has been studied. The most successful approach [67] is based on a fully-convolutional encoder-decoder CNN. The key technique in [67] is line normalization, which makes it possible to normalize the stroke width of raster line drawing images. By combining line normalization and weighted L1 loss, the network could produce clean and crisp line drawing images based on supervised learning on rough sketch and line drawing pairs. However, generating a line drawing from a photograph is a much more difficult problem than that from a rough sketch. A number of the edges in a photograph is significantly larger than that in a rough sketch. The edges are sometimes ambiguous due to noises and blur. The thickness of the edge immensely varies because some edges are far away while some edges are very close to the camera. The extracted lines by supervised learning are not very clean and expressive as we will later show in Fig. 2.2 even when employing [67]. To overcome this, we further use the restoration network to refine the lines.

Another related work is line drawing restoration from an old sketch image such as blueprints drawn in old papers [1]. There are two sources of datasets in their

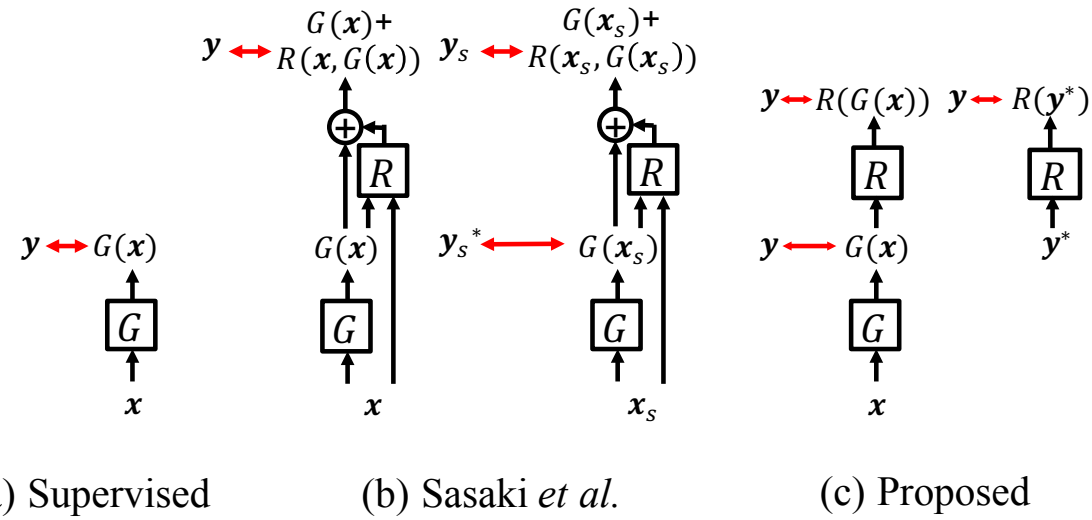


Figure 2.3: Overview of our model in the training phase, compared with different approaches. A red arrow indicates the loss computation for optimization. (a) Supervised learning encourages $G(\boldsymbol{x})$ to match $\boldsymbol{y}$. (b) Sasaki *et al.* [1] use both $(\boldsymbol{x}, \boldsymbol{y})$ from a real dataset and $(\boldsymbol{x}_s, \boldsymbol{y}_s, \boldsymbol{y}_s^*)$ from a synthetic dataset. (c) Our model only has access to $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{y}^*)$ from a real dataset, where $\boldsymbol{y}^*$ is automatically generated from $\boldsymbol{y}$ as described in Sec. 2.3.3.

approach: (i) pairs of an input old sketch image and a manually annotated clean sketch image and (ii) triplets of a clean sketch image which is composed of simple primitive shapes, a grayscale automatically deteriorated sketch image by simulating many types of distortions, and an RGB sketch image deteriorated automatically by augmenting textures of old papers. Using these synthetic triplets in addition to the real pairs, they trained a CNN to do restoration of deteriorated line drawing images. Although our model is similar in spirit to [1] in employing two sub-networks for the line extraction and refinement, their work does not apply to our learning-to-trace task. This is because we are not able to create the synthetic triplets because there is no reasonable way to synthesize a complex photograph from the corresponding line drawing image directly. Thus, we re-design the objective functions and the models to learn from the real pairs only.

Pencil-sketch generation is another related area [68]. The pencil drawing images consist of lines with various intensity and thickness, and sometimes a lot of regions are filled with textures and shadows. Accurately reproducing such effect has been investigated a lot [69, 38]. In contrast, the generated line drawing images of our model are clean and smooth, have lines consistent in width and intensity, and should not have texture-like lines and noise, as we discussed in Sec. 2.1.

2.3 Proposed Method

We propose an end-to-end trainable model for learning to trace. The model produces a line drawing that (i) only focuses on expressive lines and (ii) produces clean results (e.g., no noise and broken edges). To achieve this, we use two sub-networks, the generator G and the restorer R, each of which focuses on one of these tasks. G is responsible for extracting coarse lines by modeling long-range interactions between pixels and suppressing small and useless lines given $\boldsymbol{x}$. R is responsible for producing clean and expressive lines by considering nearby pixels given $G(\boldsymbol{x})$. At the testing phase, given an input RGB photograph $\boldsymbol{x} \in \mathbb{R}^{3\times W\times H}$, we extract the coarse lines as a intermediate grayscale image $G(\boldsymbol{x}) \in \mathbb{R}^{1\times W\times H}$, and subsequently obtain refined lines as a final grayscale image $R(G(\boldsymbol{x})) \in \mathbb{R}^{1\times W\times H}$. There is no need for any post-processing since $R(G(\boldsymbol{x}))$ produced by our model is already clean and expressive while preserving adequate details.

At the training phase, we have access to a dataset D which consists of pairs of $\boldsymbol{x}$ and a ground truth line drawing $\boldsymbol{y} \in \mathbb{R}^{1\times W\times H}$. The most straightforward approach that one may come up with is to train G, which we call the baseline model, by supervised learning $\boldsymbol{y} \approx G(\boldsymbol{x})$ on D. However, the lines produced by G are not clean and expressive, as shown in Fig. 2.2 due to the small number of D. We employ R on top of G to perform characteristics of restoration. This enables us to use an additional source of data for training. Given a clean ground truth line drawing $\boldsymbol{y}$, we randomly generate a deteriorated line drawing $\boldsymbol{y}^* \in \mathbb{R}^{1\times W\times H}$ using some low-level manipulations such as blurring and adding noises. The point is that we can incorporate a prior of ground truth line drawings for generating clean and expressive results by learning $\boldsymbol{y} \approx R(\boldsymbol{y}^*)$.

In Sec. 2.3.1, we show architectural details of G and R. In Sec. 2.3.2, we explain the objective functions for optimizing G and R jointly. In Sec. 2.3.3, we illustrate how to generate synthetically deteriorated line drawing images $\boldsymbol{y}^*$.

2.3.1 Model Architecture

Architectural Details for Two Sub-networks

G and R are both fully-convolutional encoder-decoder CNN. In the encoder, G and R decrease the resolution in three stages down to 1/8 of the original size. In the decoder, G and R increase the resolution in three stages to the original size by nearest neighbor upsampling followed by convolutional layers. All the values in the input and the

output of G and R are within the range of $[0.0, 1.0]$. For the detailed configuration of G and R, please refer to the supplementary material.

2.3.2 Optimization

We employ the approach of [67] for supervised learning of a baseline model. Let a predicted image $\boldsymbol{a}$ and ground truth image $\boldsymbol{b}$, the weighted L1 loss $\mathcal{L}_{WL1}$ used in [67] is computed as follows:

$$\mathcal{L}_{WL1}(\boldsymbol{a}, \boldsymbol{b}) = |(\boldsymbol{a} - \boldsymbol{b}) \odot (\mathbf{1} + \gamma(\mathbf{1} - \boldsymbol{b}))|, \tag{2.1}$$

where $\odot$ denotes element-wise multiplication and γ is a weighting hyper-parameter to. If γ equals to zero, the loss is similar to L1 loss (i.e., mean absolute error (MAE)). If the γ is larger than one, the loss put more weight on the line predictions being correct rather than the white backgrounds. Using L_{WL1} as the loss function L_{base}, the baseline model is optimized as follows:

$$G^* = \underset{G}{\operatorname{argmin}} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim D} \ \mathcal{L}_{base}(G(\boldsymbol{x}), \boldsymbol{y}). \tag{2.2}$$

We employ multi-task learning to improve the performance of the extraction and the refinement process jointly:

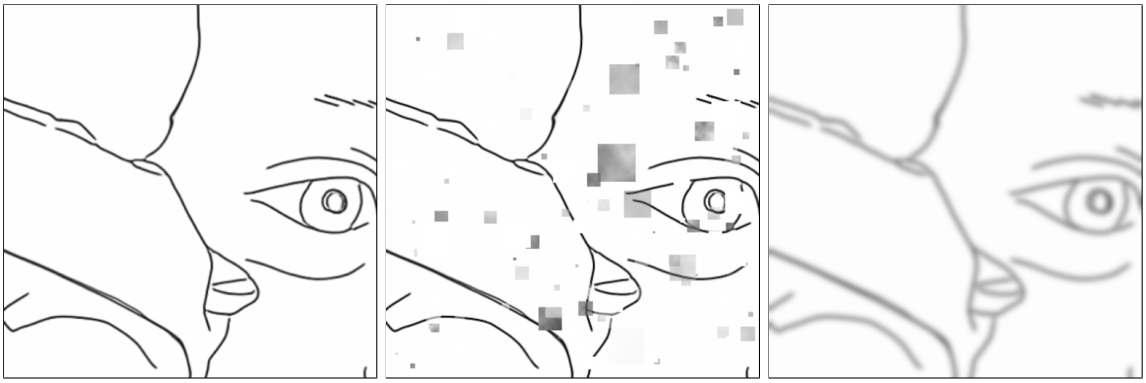
$$\mathcal{L}_{joint} = \mathcal{L}_{base}(G(\boldsymbol{x}), \boldsymbol{y}) + \alpha\mathcal{L}_{aux}(R(G(\boldsymbol{x})), \boldsymbol{y}) + \beta\mathcal{L}_{res}(R(\boldsymbol{y}^*), \boldsymbol{y}), \tag{2.3}$$

$$G^*, R^* = \underset{G,R}{\operatorname{argmin}} \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y},\boldsymbol{y}^*)\sim D} \ \mathcal{L}_{joint}, \tag{2.4}$$

where α and β are hyperparameters. $\mathcal{L}_{res}$ is for learning the restoration process using the pair of the clean line drawing $\boldsymbol{y}$ and the deteriorated line drawing $\boldsymbol{y}^*$. $\mathcal{L}_{aux}$ is an auxiliary loss function to stabilize the training. Without $\mathcal{L}_{aux}$, our model is not end-to-end trained. As the error metrics, L1 loss and the weighted L1 loss in Eq. (2.1) are used for $\mathcal{L}_{res}$ and $\mathcal{L}_{aux}$, respectively. The difference among the baseline model, our model, and Sasaki *et al*. [1] in the training phase is shown in Fig. 2.3.

2.3.3 Synthetic Line Drawing Deterioration

We explain how to process a ground truth line drawing $\boldsymbol{y}$ to a deteriorated line drawing $\boldsymbol{y}^*$ in detail. Examples of the deteriorated line drawing images are shown in



Input Output (fill) output (blur)

Figure 2.4: Example of the synthetic line deterioration for *face/body*. For visualization, we crop the images with the size of 384×384.

Fig. 2.4. With the chance of 75%, we applied the following deterioration algorithms independently with a randomly sampled value $\sigma \in [0.0, 1.0]$:

- Fading: $\boldsymbol{y}^* = \frac{\boldsymbol{y}+\sigma}{1+\sigma}$. This operation does not change the originally white pixels while fading originally black pixels.
- Blur: A two-dimensional Gaussian filter with the standard deviation σ is applied.
- Fill: For each $\boldsymbol{y}$, we randomly generate 10 to 50 holes whose sizes are between 16×16 pixels to 40×40 pixels. Further, we randomly generate 100 to 500 holes whose sizes are between 2×2 pixels to 16×16 pixels. For each hole, we perform one of the following three operations randomly with the chance of $\frac{1}{3}$; (i) Fading (described above). (ii) Turning pixels white. (iii) $\boldsymbol{y}^* = \sigma\boldsymbol{y} + (1-\sigma)\boldsymbol{t}$, where $\boldsymbol{t}$ is a texture patch randomly cropped from a cloud-like texture generated by the cloud filter in Adobe Photoshop [70]. This operation is for learning to remove texture-like dense edges.

We can synthesize diverse and unlimited number of $\boldsymbol{y}^*$ by choosing the low-level manipulations randomly during the training.

2.4 Dataset

Since there is no dataset that can meet our demand, we have collected new datasets of photograph and line drawing pairs in two domains: *face/body* and *manga BG*. We briefly describe each dataset.

2.4.1 Face / Body

A total of 139 photograph and line drawing pairs are collected. The photographs are collected from Helen dataset [71], Adobe Stock, and our personal collection to cover a diverse set of photographs. The resolution of the images ranges from 602×602 to $1,946 \times 2,432$. We asked professional artists to draw line drawings from these photographs. A few examples were collected from one artist at first. Subsequently, the other artists were asked to follow the way of drawing to collect the rest of the dataset. Only the silhouette is drawn for the hair region because it might be more subjective to get the line drawings on hair with consistent styles from different artists. Lines in the background are ignored unless they are very salient. Examples of the dataset are shown in Fig. 2.5. The width of the lines is not consistent among samples and strokes.

2.4.2 Manga BG

A total of 174 photograph and line drawing pairs are collected. Some of the examples are shown in Fig. 2.6. It is hard to ask the artists to draw the lines from scratch because the number of lines is enormous. Therefore, we use two existing resources: (i) 92 pairs are from synthetically rendered examples from 3D models of New York City from NONECG [72]. We use Mental Ray for Maya and render the images by Contour Shader. (ii) 82 pairs are collected from stock photos for manga artists. The resolution of the images ranges from $3,750 \times 5,000$ to $17,008 \times 11,304$. When collecting the pairs from existing resources, there are many mismatches between a photograph and corresponding line drawing because artists sometimes do not faithfully follow lines in a photograph, ignore some parts, or add some parts. These mismatches are not negligible. To overcome this problem, we manually add mask annotations $\boldsymbol{m} \in \mathbb{R}^{1 \times W \times H}$, where each value in $\boldsymbol{m}$ is either one (i.e., valid) or zero (i.e., invalid). We extend the weighted L1 in Eq. (2.1) to take $\boldsymbol{m}$ as the third argument and compute the loss function as follows:

$$\mathcal{L}_{WL1}(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{m}) = |\boldsymbol{m} \odot (\boldsymbol{a} - \boldsymbol{b}) \odot (\boldsymbol{1} + \gamma(\boldsymbol{1} - \boldsymbol{b}))|. \tag{2.5}$$

Figure 2.5: Example of photograph and line drawing pairs that we have collected for *face / body*. The photographs on the right are the cropped patches of those on the left. There may not be strongly visible contour along the nose, but the artists draw one in because it is artistically important. The photograph on the left and middle is from yurolaitsalbert - stock.adobe.com and Jean - stock.adobe.com, respectively. (best viewed with zoom and in color.)

2.5 Training

We train two models for *face / body* and *manga BG*, separately. For faster and stable training, we first pre-trained G following Eq. (2.2) and use this G as the starting point for our model and all the compared approaches. All the methods in this chapter were trained using Adam [73] optimizer for a fair comparison. We applied line normalization by nimble model [67] so that the line width is two pixels in the ground truth line drawing image $\boldsymbol{y}$. The images whose longest side is over 2,000 pixels were resized so that the longest side is 2,000 pixels to reduce computational cost during training.

In Sec. 2.5.1, we explain the detailed process of pre-training. In Sec. 2.5.2, we refer to the detailed process of our joint learning of G and R. In Sec. 2.5.3, we briefly describe data augmentation operations used for both pre-training and joint training.

Figure 2.6: Example of photograph and line drawing pairs that we have collected for *manga BG*. Only the cropped patches are shown due to the limited space. Only the silhouettes of plants are drawn since we want to manually draw the interior regions later, although there are visually very salient edges inside them. We can also see that even the rendered images on the right has the mismatches. (best viewed with zoom and in color.)

2.5.1 Pre-training

We can improve the ability of our network to learn with a small training set by pre-training some part of G on another task. For *manga BG* generation, we adopted the encoder of ResNet50 pre-trained on ImageNet [74] classification task. For *face / body* generation, we adopted a network that is similar to G stated in Sec. 2.3.1 and is pre-trained on a portrait dataset [75] for facial part segmentation. Subsequently, G was trained for 30,000 iterations with a learning rate of 1.0×10^{-3} and a batch size of 8, following Eq. (2.2).

2.5.2 Joint Learning

G and R were trained jointly for 10,000 iterations with a learning rate of 1.0×10^{-4} and a batch size of 4, following Eq. (2.4). For the weighting parameter γ in the weighted L1 loss in Eq. (2.1), we empirically set $\gamma = 2$ and $\gamma = 0$ for *face / body* and *manga BG*, respectively. For the joint optimization of our model, we set $(\alpha, \beta) = (0.1, 10.0)$ in

Eq. (2.3). With a probability of 10%, we just copied $\boldsymbol{y}$ to create $\boldsymbol{y}^*$ instead of the deterioration because if the lines extracted by G are already clean, there is no need for refinement.

2.5.3 Data Augmentation

Since the number of photograph and line drawing pairs is limited, we performed a large amount of data augmentation to enhance the generalization capability of our model. We applied random scaling between the range of 0.5 to 1.5 with 1.0 being the original scale, random flipping with the probability of 0.5, random rotation between the range of -0.25 to 0.25 with 0.0 being the original angle, and random cropping to 384×384 for both the input and output image pairs. Besides, we randomly changed the contrast, saturation, hue, and brightness of the input image.

2.6 Experiments

We evaluated our model on various high-resolution photographs. The photographs are down-sampled if the length of the longer edge is larger than 2000 pixels. Following the existing works for line drawing generation [65–67], we perform qualitative evaluation and user study. For two reasons, we did not employ OIS/ODS, which are used for the quantitative evaluation of coarse edge detection. First, mismatches between a photograph and corresponding line drawing are not negligible in our dataset for quantitavie comparison, as we have already explained in Sec. 2.4. Second, matching the predicted pixels to the ground truth pixels in calculating OIS/ODS is computationally very expensive and very slow in our case. We found it almost impossible in our setting since the number of ground truth pixels in our *face / body* and *manga BG* are enormously larger than those of BSDS500 and Contour Drawing.

2.6.1 Comparison with Existing Methods

To emphasize the importance of developing an appropriate model for learning to trace, we compared our model against the following comparable approaches:

- **Canny**: As a classical hand-crafted method for edge detection, we tested the Canny edge detector [57].

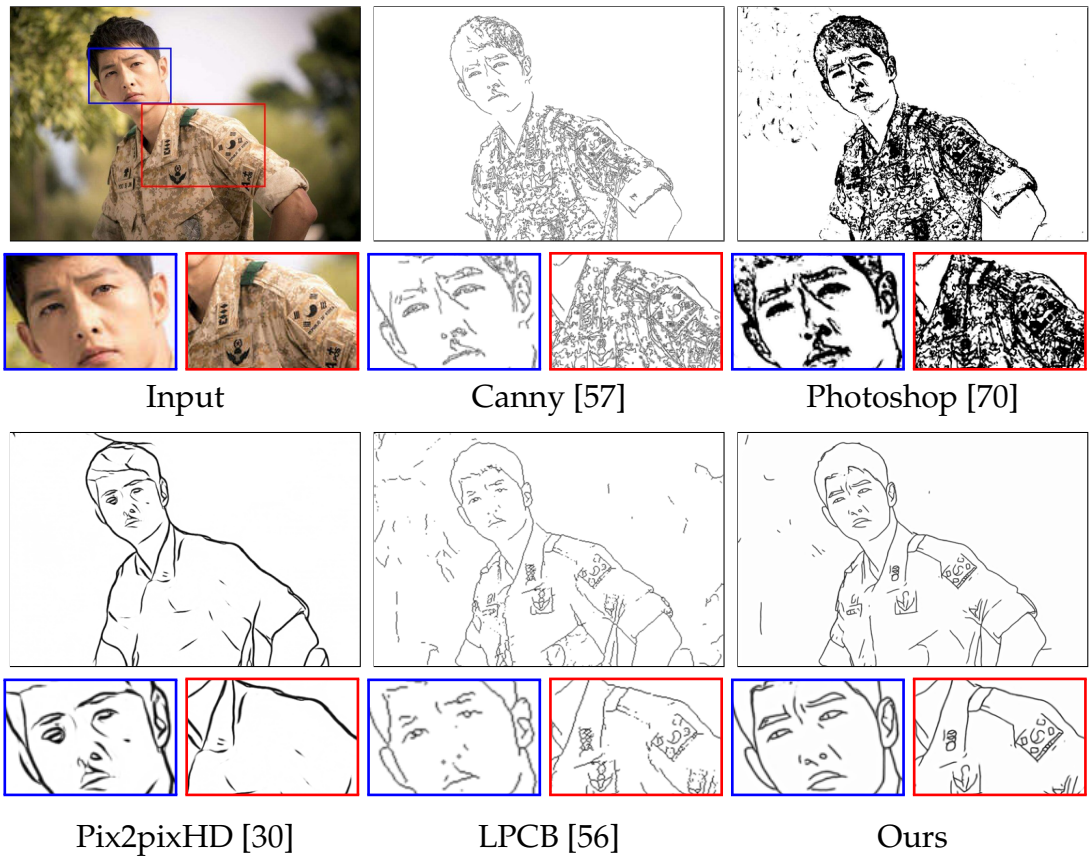


Figure 2.7: Comparison with the previous approaches for *face / body*. The first and third rows show the whole input image, while the second and fourth rows show the areas highlighted in red and blue in the input image. Note that no pre-processing and post-processing is applied. We can see that our approach clearly outperforms the other approaches regarding cleanness and expressiveness. The photograph is from Brenda Rochelle (Public Domain). (best viewed with zoom and in color.)

- **PhotoShop**: As an existing tool, we tested Photoshop [70]. We made a single filter composed of some existing filters built in PhotoShop such as 'Find edges' and 'Dust & Scratches' to make the extracted image clean and expressive as much as possible by adjusting hyper-parameters. The filter is used for all the images while keeping the hyper-parameters fixed.

- **Pix2pixHD**: Line drawing generation from photographs can be regarded as I2IT. In Pix2pixHD [30], the authors proposed novel adversarial loss, and multi-scale discriminator/generator to enable better high-resolution image generation than pix2pix [12]. We tested an off-the-shelf implementation of Pix2pixHD.

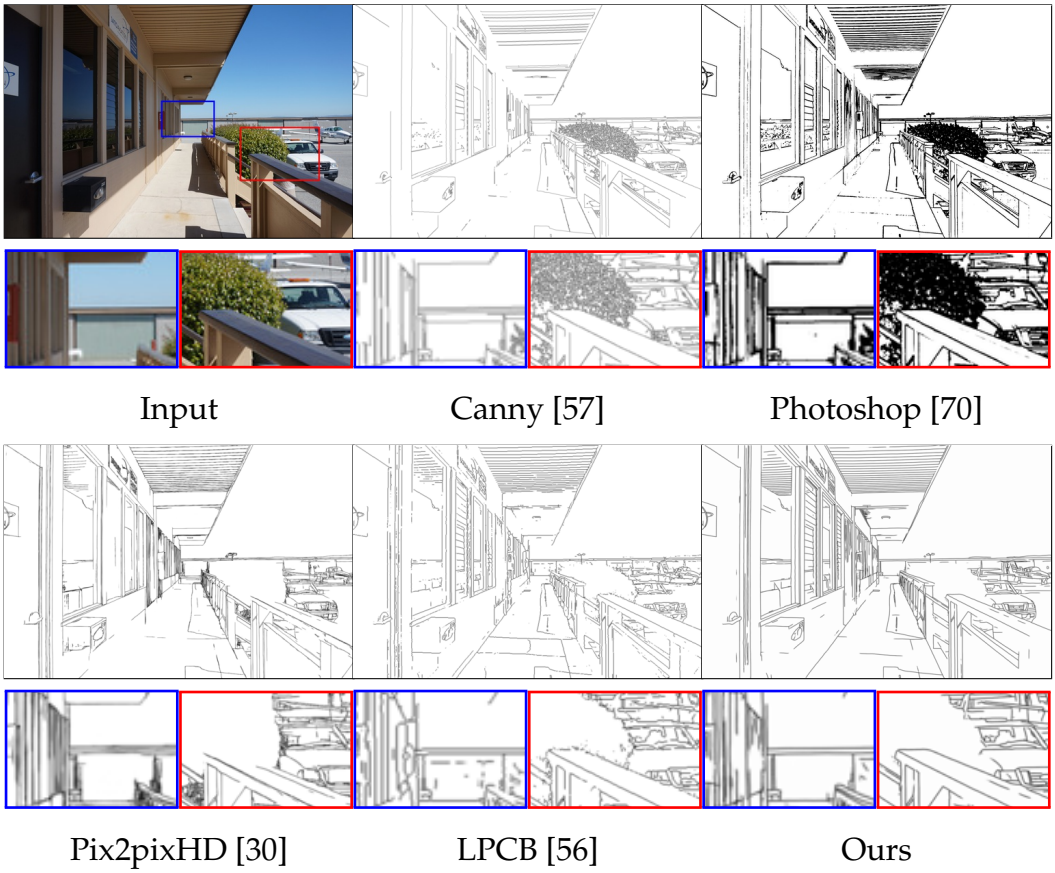


Figure 2.8: Comparison with the previous approaches for *manga BG*. The first and third rows show the whole input image, while the second and fourth rows show the areas highlighted in red and blue in the input image. Note that no pre-processing and post-processing is applied. We can see that our approach outperforms the other approaches regarding cleanness and expressiveness. (best viewed with zoom and in color.)

- **LPCB**: As a learnable CNN for edge detection, we tested LPCB [56]. Since the result of LPCB is still very blurry, we post-processed the result by binarization and morphological line thinning. Please refer to the supplementary material for the results without post-processing.

Qualitative Evaluation

The results of all the approaches for *face / body* are shown in Fig. 2.7. Canny and Photoshop produce a lot of artifacts and too short edges, and the results are far from line drawing images. Pix2pixHD captures rough outlines but fails to preserve details in the face and clothes. LPCB captures both rough and fine features, but the lines are

Table 2.1: Results of the user study. Each number indicates the percentage where our result is preferred over a compared method. Chance rate is at 50%.

	face / body	*manga BG*
Canny [57]	91.6%	77.2%
Photoshop [70]	70.0%	82.4%
Pix2pixHD [30]	68.4%	56.8%
LPCB [56]	82.0%	93.2%

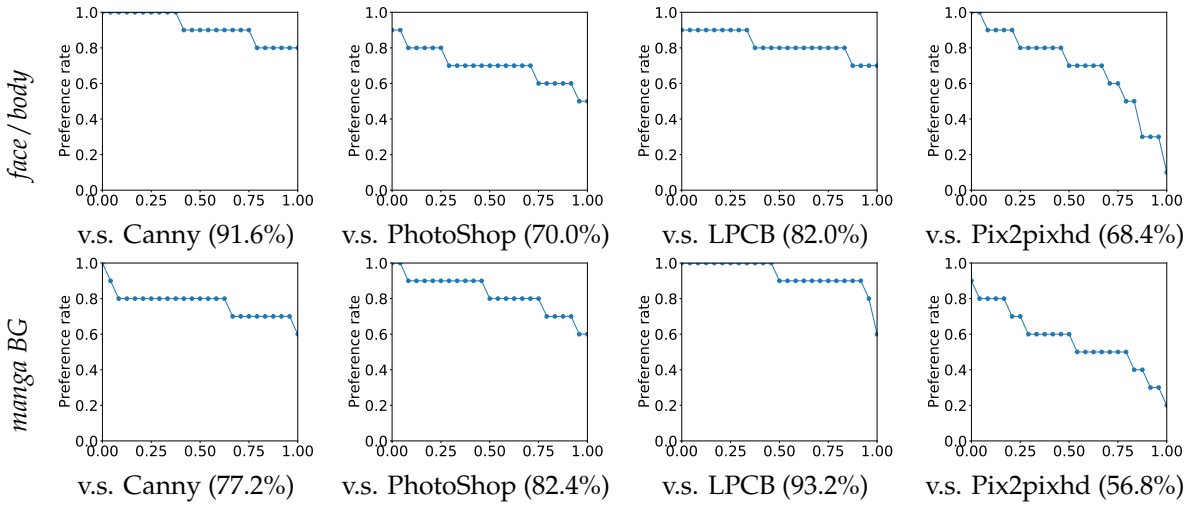


Figure 2.9: Visualization of the preference rate of our model over the compared model for each image in the user study. The percentage inside the parentheses indicates the average preference rate.

fragmented and interrupted. On the other hand, our model produces the cleanest and most expressive line drawing images without any post-processing.

The results of all the approaches for *manga BG* are shown in Fig. 2.8. We can see that Canny and Photoshop produce texture-like cluttered lines in the region of plants. The result of LPCB is wavy and sometimes makes no sense. Pix2pixHD seems to capture both coarse and fine details, but there are often texture-like edges and cluttered lines at a closer look. On the other hand, our model produces the cleanest and most expressive line drawing images. For more results, please refer to the supplementary material.

User Study

Since the qualitative evaluation discussed above is highly subjective, we performed a perceptual user study among comparable methods by cloud workers on Amazon

Table 2.2: Results of the user study in the ablation study. Each number indicates the percentage where our result is preferred over a compared method. Chance rate is at 50%.

	face / body	*manga BG*
G only	58.4%	75.2%
G + adv.	61.6%	61.2%

Mechanical Turk. We used 25 high-resolution photographs for evaluating the model. We displayed results from two different models from the same photograph side-by-side on a webpage in random order. Each subject was asked to choose the best one. As a result, we collected 250 votes from 10 subjects for evaluating the model trained on each dataset, and the results are shown in Table 2.1. We can see that our model is more preferred than the other methods. We additionally discuss how much the AMT workers votes are consistent with each other in the user study in Sec. 2.6.1. We sort the preference rate on each of the user study images and show the result in Fig. 2.9. As expected, judging the generated line drawing images' quality is relatively hard, but the voters are usually consistent with each other.

2.6.2 Ablation Study

We performed an ablation study regarding the choice of networks, training objectives and hyper-parameters.

G **only**: Training only the generator G by Eq. (2.2) is a strong baseline method. However, it produces lines with inconsistent intensity and thickness, as shown in Fig. 2.2. Our model is superior also in the user study, as shown in Table 2.2.

G **and adv.**: One may argue that optimizing G using line normalization and adversarial learning used in [12] can achieve our goal. However, we found it inefficient and prone to produce small noise-like edges, as shown in Fig. 2.10. Our model is superior also in the user study, as shown in Table 2.2.

Architecture choices: One may argue that our model does not fully understand the contextual information. We found that initializing the generator with the pre-trained model as described in Sec. 2.5.1 is enough. Although we tried to combine the result of the semantic segmentation trained on ADE20K [76] for *manga BG* or a portrait dataset [75] for *face / body* in addition to the input photograph, we did not observe remarkable improvement. Employing skip-connections between an encoder and a decoder is popular in image translation, but we found it unnecessary for our

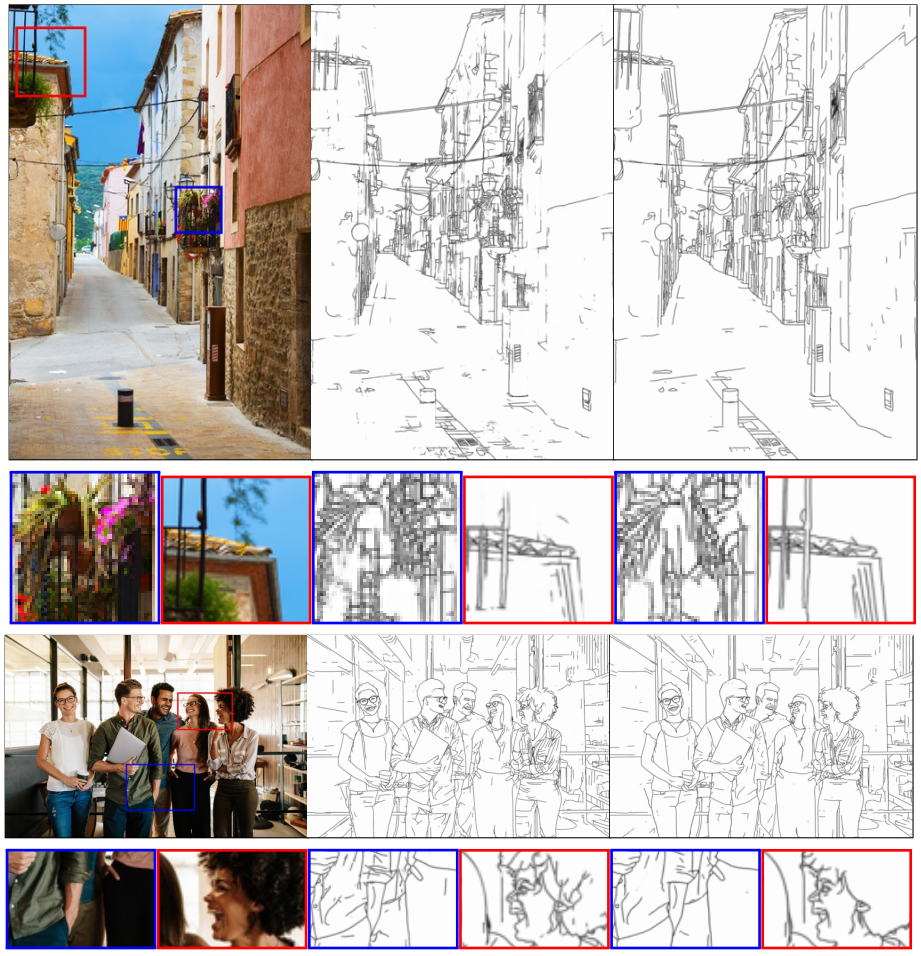
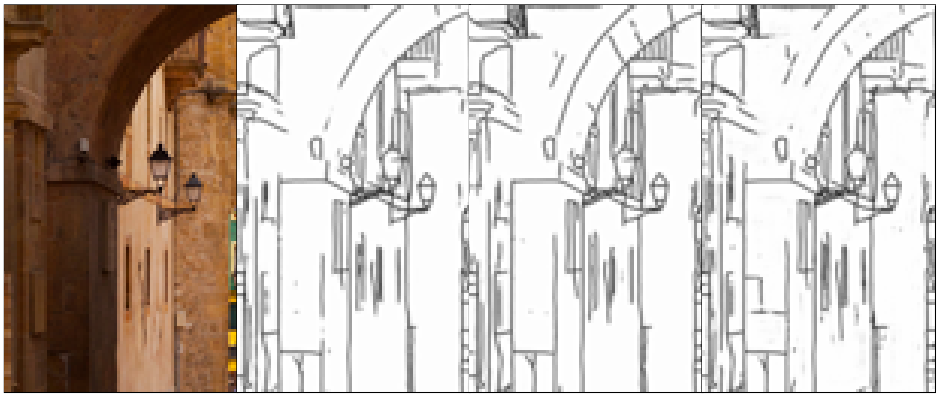


Figure 2.10: Comparison with our model and 'G and adv'. Our model generates less blurry and choppy results. (best viewed with zoom and in color.)

setting. A self-attention mechanism such as the one employed in [77] would be another option. However, we could not apply it due to its high memory usage for HD image generation.

Hyper-parameters: We optimize Eq. (2.4), which have two hyper-parameters α and β. Setting α too high or low causes degradation in the produced line drawing, as shown in Fig. 2.11. We also observed that setting too small β causes the same degradation as setting α too high.



Input $\alpha = 0.0$ $\alpha = 0.1$ $\alpha = 1.0$

Figure 2.11: Ablation study by changing α in Eq. (2.4). Setting too high α results in some artifacts and blurry lines ($\alpha = 1.0$) while setting too low α makes the restorer remove some important outlines ($\alpha = 0.0$). For visualization, a 300×500 patch from a photograph is used as the input photograph.

2.6.3 Comparison with Sasaki *et al.*

One may argue that Sasaki *et al.* [1] can be trained with only real pairs of photograph and line-drawing $(\boldsymbol{x}, \boldsymbol{y})$. Since we cannot generate the synthetic triplets dataset as we discuss in Sec. 2.2.3 and Fig. 2.3, the best we can do is to train G and R so that $G(\boldsymbol{x}) + R(\boldsymbol{x}, G(\boldsymbol{x}))$ matches $\boldsymbol{y}$ by weighted L1 loss Eq. (2.1). For fair comparison, we use G which is pre-trained on Eq. (2.2). We did not observe clear visual improvement over the baseline model optimized by Eq. (2.1). Our results obtained 80.0% and 82.8% preference rate over the results of the model described above for *manga BG* and *face / body*, respectively.

2.6.4 Comparison with Style Transfer

We also compared our model with style transfer. In addition to the input image, the style transfer algorithm takes a single target image as the reference for the stylization. The original algorithm of style transfer [78] with iterative optimization with multiple steps of forward and backward inference using a CNN is very slow. Therefore, we tested [2], which requires only one step of forwarding inference at the test time. We picked one of the images from the dataset we used to train our model as the style target image. The result is shown in Fig. 2.12. We can see that this approach cannot produce clean and expressive line drawing images.

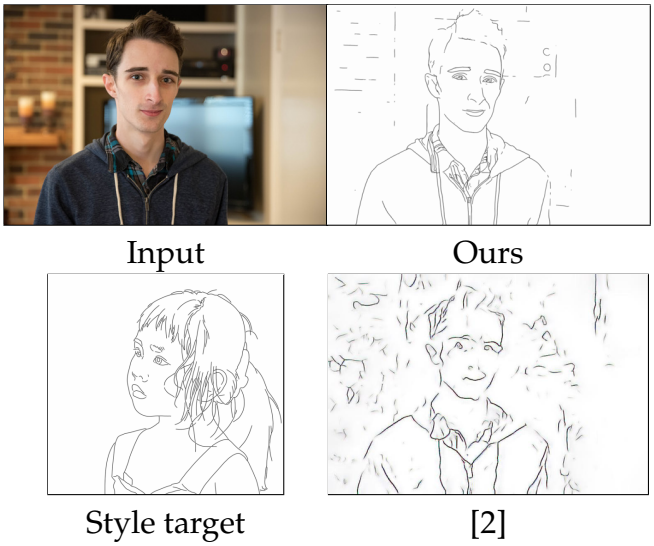


Figure 2.12: Comparison with the approach of style transfer [2] and our model for *face / body*. The style image is chosen from the training dataset for *face / body* and pre-processed by the line normalization. The result obtained by the style transfer contained a lot of blurry gray regions and failed to capture artistically salient parts such as the nose and the eyes. Photograph of Paradox Wolf (Public Domain). (best viewed with zoom and in color.)

2.6.5 Comparison with Existing Datasets

To emphasize the importance of collecting a dataset that is suitable for each domain, we trained our model on BSDS500 [3] and Contour Drawing [4], and show the results in Fig. 2.13. We observe a significant difference. In *face / body*, it can be seen that the model trained on BSDS500 and Contour Drawing only captured a rough outline of the person such as the boundary between body and background and ignored expressive parts such as eyes and mouths. This trend was also remarkable in *manga BG*.

2.6.6 Effect of Input Scale

While our model is automatic, the users can control the simplicity of the output by downscaling or upsampling the input image based on their intent, as shown in Fig. 2.14. The output of our model is simplified and loses some fine details as the input image is getting smaller. Although we perform a large amount of the data augmentation, our model is sometimes sensitive to the changes in the scale of the input image. The users also can easily search the scale of the input image to obtain the best result for their own use case.

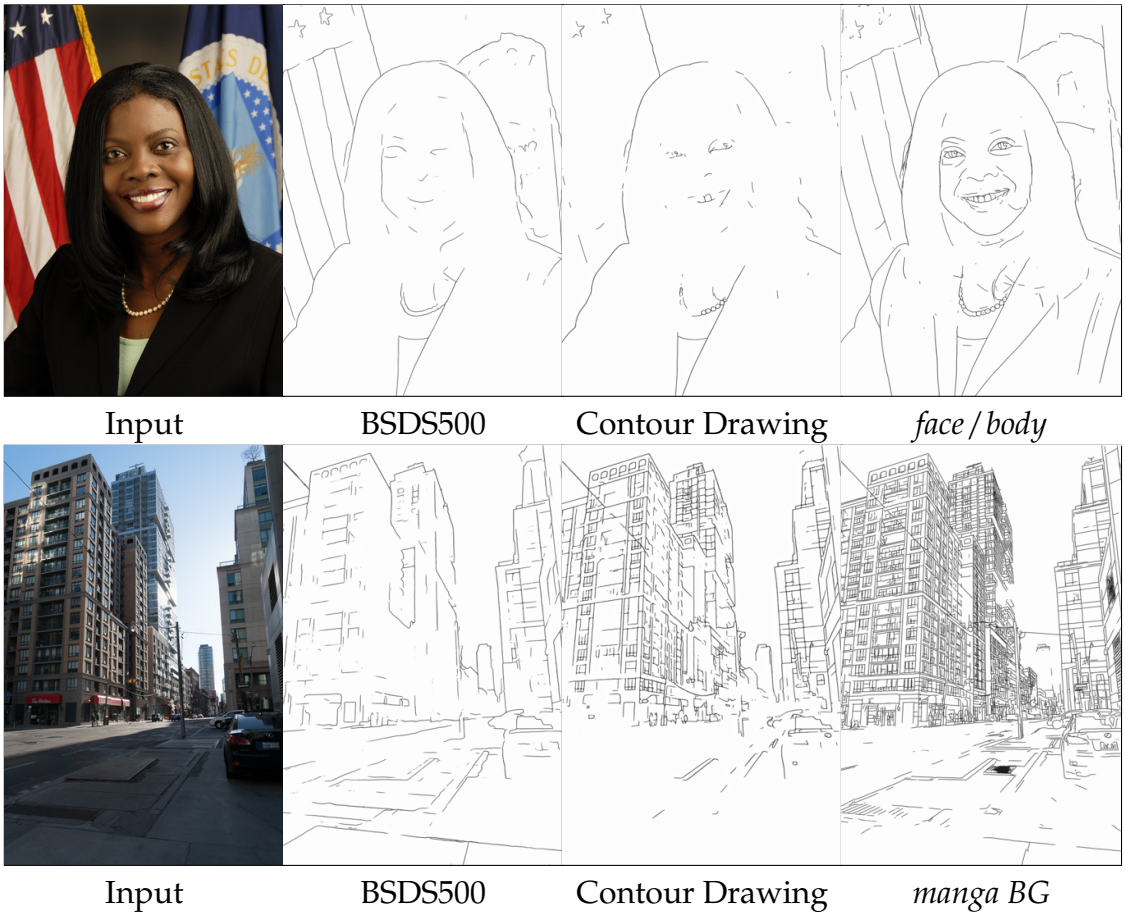
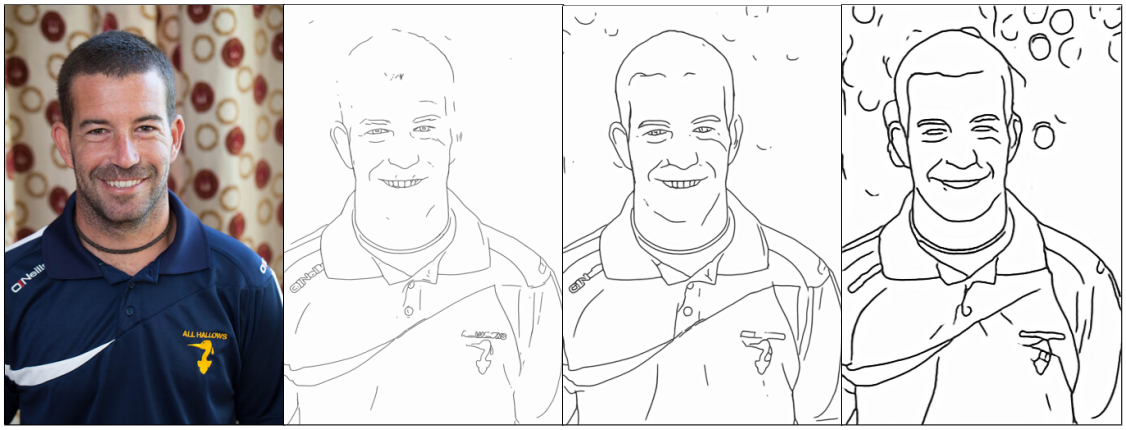


Figure 2.13: Comparison with the model trained on different datasets. The model trained on BSDS500 [3] and Contour Drawing [4] only captured rough outlines. The photograph in the first row is from U.S. Department of Agriculture (Public Domain). (best viewed with zoom and in color.)

2.6.7 Computation Time

Our model can take images with arbitrary sizes as the input. We tested images with various input sizes initialized randomly and show the average time on 100 runs on a GPU in Table 2.3 using our model for *manga BG*. For an image taken by iPhone5 (typically 4096×3072), our model could process it in about 1.6 seconds. If we have a much larger image, we can break it into small patches and process each of them individually. The result shows that our model is practical and can be applied in real-world scenarios.



Input $s = 1$ $s = 0.5$ $s = 0.25$

Figure 2.14: Result of the line drawing generation by our model for *face / body* on scaled versions of the input image (1365×2048). The scaling done with respect to the input image is denoted as s, where $s = 1$ denotes maintaining the original resolution. $s = 1$ maintained the clean background and captured the eyes and the mouth sufficiently while losing some important outlines around the hair. As the input resolution becomes smaller, our model only captures simplified outlines. Note that we train our model to produce lines whose width is two pixels by the line normalization regardless of the scale of the input image. Photograph of Owen Lucas (Public Domain). (best viewed with zoom and in color.)

2.6.8 Vectorization as Post-processing

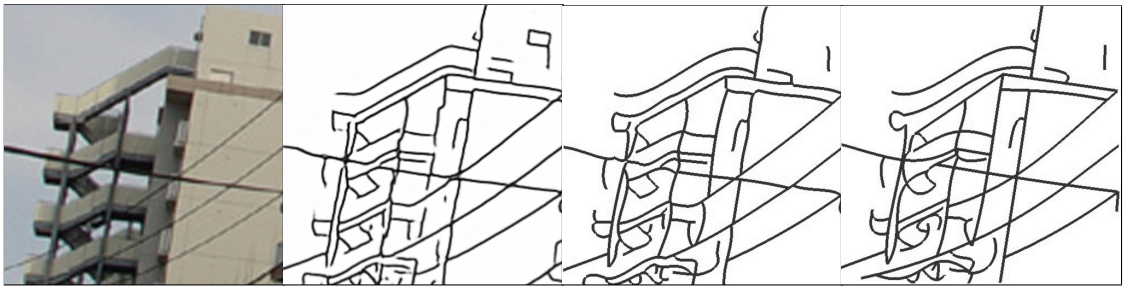
Although our model can extract clean line drawings that can be used for further editing, converting the output image into a vector format is also possible. A vector format is popular because of the precision, compactness, and editability. For example, a user can change width for some lines to control the attention of the viewer. We show the example processed by the automatic vectorization of [5] in Fig. 2.15. We can see that some errors at junctions are fixed and some wavy lines are converted to very straight lines, while some parts are missed by the vectorization.

2.7 Applications

Using our model, we demonstrate two possible downstream applications: *manga BG* generation and coloring books.

Table 2.3: Computation time for our model. NVIDIA TITAN Xp GPU is used for the benchmarking.

Image size	Pixels	Time [s]
512×512	262,144	0.05
1024×1024	1,048,576	0.14
2048×2048	4,194,304	0.52
4096×3072	12,582,912	1.57

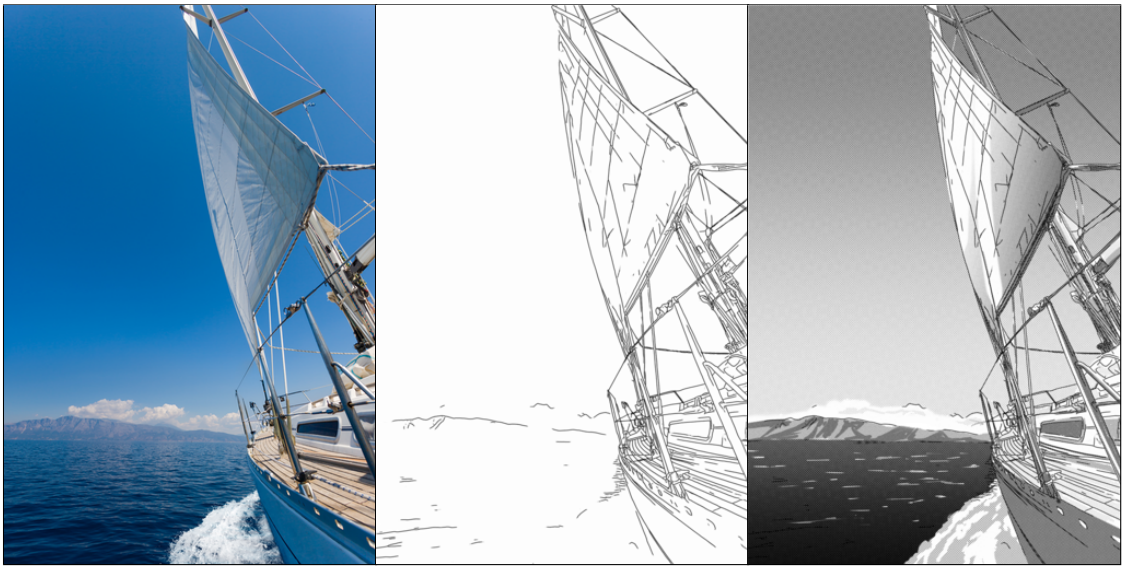


Input | Output | Vector output (low simplicity) | Vector output (high simplicity)

Figure 2.15: Example of vectorization using [5] as the post-processing. Since the output of our model is already clean, we can directly vectorize the line drawing without any pre-processing. For visualization, we use a 256×256 patch. We show two vectorized outputs by changing the hyper-parameter to control the simplicity in [5]. (best viewed with zoom and in color.)

2.7.1 Manga BG Generation

Manga is composed of two layers: line drawing and *halftone screen*. Halftone screen is a pre-defined pattern that expresses visual effects such as textures and shadows [79]. Manga artists first draw a clean line drawing and subsequently add a halftone screen layer to express fine details. When they draw complex scenes (e.g., streets, buildings, and landmarks), they often trace a reference photograph to create a line drawing, which is a very tedious and time-consuming process. We argue that our model can help this process. Given the reference photograph, our model creates a line drawing automatically. They can add some lines to depict more details, remove some unnecessary parts, or just use it without any modification. An example of the collaboration with our model and the artist is shown in Fig. 2.16. An artist simply added some halftone screens digitally on top of the extracted line drawing layer to make a scene for manga BG.



Input | Output | Output with halftone screens

Figure 2.16: From left to right: an input photograph, the line drawing image produced by our model for *manga BG*, the comic image produced by the artist based on the generated line drawing. Note that the artist only added halftone screens and did not modify the line drawing itself. Photograph of Netfalls - stock.adobe.com. (best viewed with zoom and in color.)

Richness-Preserving Manga Screening [80] was developed for automatic generation of the halftone screen layer. The authors even show that they can generate manga BG images in a fully-automatic way by employing an existing edge extractor [81] for the line drawing layer. We argue that our model can also be used for the line drawing layer in their process.

2.7.2 Coloring Books

A coloring book is a book with line drawing images to which users are intended to add colors by some artistic media such as pencils, crayons, and paints. We argue that our line drawing generation model is a new tool to create the coloring book. An example of the collaboration with our model and the artist is shown in Fig. 2.17. Since our line drawing generation model captures expressive lines adequately, the artist was able to create the artwork by simply adding some colors, without modifying the lines.

Figure 2.17: From top to bottom: an input photograph, the line drawing image produced by our model for *face / body*, the artwork produced by the artist based on the generated line drawing. Photograph of REDPIXEL - stock.adobe.com. (best viewed with zoom and in color.)

2.8 Limitations and Discussion

One limitation of our model is that the produced lines have some faults both locally and globally, and one may realize that the lines are not produced by a human but by some computational algorithm at a closer look. Although we show that raster image vectorization algorithms such as [5] are effective to some extent in Sec. 2.6.8, this process is still computationally very slow and misses some lines. The future direction of this chapter should be joint learning of line extraction and vectorization. Interactive line drawing generation assisted by neural networks [67] would also be helpful to fix the faults by user interaction.

Another limitation of our work is that our datasets for training are limited in size, and thus we struggle on objects or scenes that are not included in the dataset for the training. Some examples of the failure cases of our model are shown in Fig. 3.10. We

Figure 2.18: Examples of the failure cases of our model. The output on the left failed to capture boundary between the fluffy hair and the background because the boundary is ambiguous. The output on the right contained texture-like edges ahead of the handrail because there is no annotation for grass in *manga BG*. The photograph on the left is from Javier Ortiz (Public domain). (best viewed with zoom and in color.)

believe we can address this limitation by collecting the more high-quality annotation for images with more diverse scenes by professional workers.

Changing the thickness of the lines based on contents would be another future work. This is because the lines are often drawn with different width by human based on some factors such as the importance of the edge and distance from the camera in a photograph. However, predicting the exact width is probably difficult in current learning-based approach, since such criterion may differ based on the subject or even on the context.

Chapter 3

RGB2AO: Ambient Occlusion Generation from RGB Images

3.1 Introduction

Ambient occlusion (AO) [82, 83] is an important rendering technique in 3D computer graphics that significantly improves the visual quality of renders. Ambient occlusion works by darkening the regions in the image where the local scene geometry is concave, where neighboring surfaces shadow or occlude part of the ambient lighting. AO-based rendering is highly effective in adding realistic shading to renders that otherwise often look flat due to the lack of physically accurate global illumination effects. This technique has thus received much attention in the graphics community since its inception. It has benefited from multiple improvements to its efficiency, leading to its widespread use in real-time applications such as video games [84, 85].

Interestingly, the idea of applying ambient occlusion has been appreciated in other areas beyond 3D computer graphics. Professional artists working with 2D content have also developed creative techniques leveraging AO-like information to enhance the realism and impression of their artworks, such as photographs, painting, and illustrations. In particular, it has been demonstrated that applying AO-like shading effect in RGB images in the wild would enable multiple practical applications [86, 87], such as the ones shown in Fig. 3.1:

- 2D image composition: simply pasting an object (e.g., cars, boxes, and bottles) onto a background image (e.g., road and table) would look like the object is floating mid-air in the image. Adding AO around the inserted object would make the composite visually pleasing.

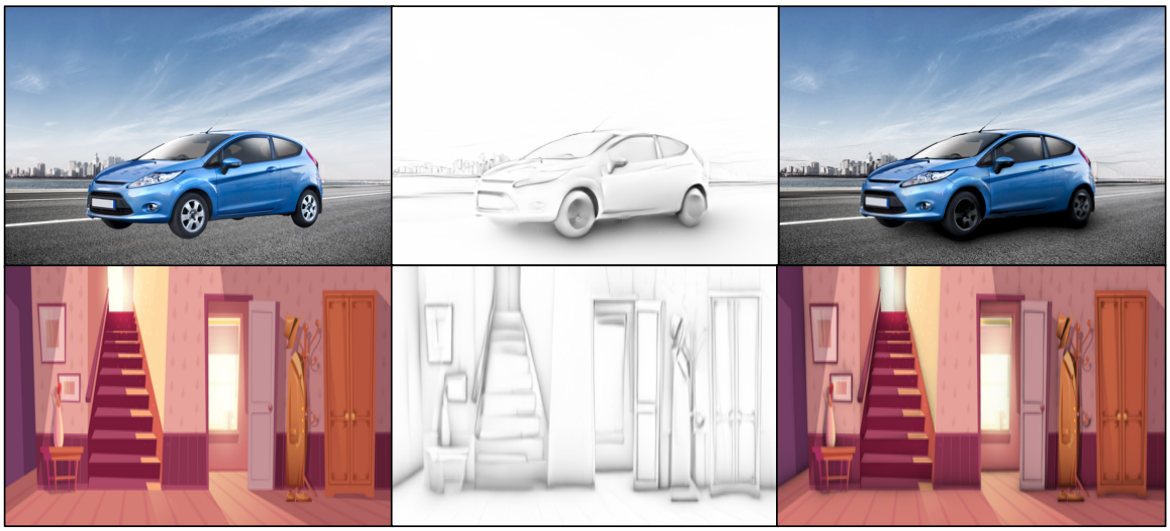


Figure 3.1: Ambient Occlusion (AO) generation from a single image with our RGB2AO framework. Given an RGB image (left), our model automatically generates an ambient occlusion map (center). We show two applications of applying the generated AO map effect: image composition (in the first row) and geometry-aware contrast enhancement (in the second row) on the right. The input images are from Adobe Stock. (best viewed with zoom and in color.)

- Geometry-aware contrast enhancement: it can be used to (de)emphasize or exaggerate the geometry in a photograph, providing a user control on perceivable depth.

Conventional techniques for computing AO, however, require 3D information about the scene such as depth buffers, surface normals, or even the whole scene geometry [85, 88, 89]. As such, these AO techniques are limited to the rendering of 3D scenes and cannot generate AO from RGB images. To achieve the aforementioned effects for 2D image composition and contrast enhancement, artists often need to manually construct the AO information themselves through a tedious AO map painting process [87, 90, 91].

This chapter introduces the novel task of AO generation from a single RGB image, so that it applies to 2D image editing applications that we have described above. One possible approach to this task would be to compute depth information from the input image with monocular depth estimation methods [92–94] and apply conventional screen-space AO generation from it to obtain the AO map. In our experiments, however, we found this simple approach fails to generate sound AO maps. We observe that while state-of-the-art depth prediction performance has rapidly improved recently, the depth prediction results are still coarse and not

accurate enough to be used with existing screen-based approaches that typically assume ground-truth depth values.

In this chapter, we present RGB2AO, a learning-based AO generation framework that learns to generate the AO map directly from the input image. Users can then use the generated AO map for editing tasks such as compositing or contrast enhancement. We develop our RGB2AO framework with an I2IT model based on a convolutional neural network. In addition, our model extends a recent I2IT model to account for the 3D geometry of scenes. We also explore data augmentation strategy that is specific and beneficial to AO generation. Our contributions encourage the network to learn to predict an accurate AO map from an RGB input alone.

A key challenge for single image AO generation is the lack of data with ground-truth AO maps for training. To address this challenge, we contribute a large-scale synthetic dataset with thousands of RGB-AO pairs. We construct our dataset from a large number of high-quality 3D scenes that we render realistically. Experimental results show that our model can produce more favorable results compared to existing methods quantitatively.

In summary, our contributions are as follows:

- We introduce the novel task of image-based ambient occlusion generation. To this end, we develop a CNN-based AO generation model considering the 3D geometry of the scenes. To the best of our knowledge, we provide the first approach to infer AO from a single RGB image automatically.

- We contribute a large-scale dataset dedicated to AO generation. This dataset consists of a large number of images with associated accurate ground-truth AO maps.

- We demonstrate the effectiveness of our RGB2AO framework in the application of our AO generation as a filter for 2D image composition and geometry-aware contrast enhancement.

3.2 Related Work

3.2.1 Ambient Occlusion

Ambient occlusion (AO) [82, 83] is a fast global illumination model that approximates the amount of light reaching a point on a surface considering occlusion by objects

and surfaces around them. With reduced computational cost compared to ray-tracing, AO can produce realistic lighting effects such as soft shadows around objects. Real-time AO computation usually requires 2D depth and normal buffer input with respect to the camera viewpoint. AO generation algorithms usually randomly sample nearby pixels and infer AO for each pixel independently. Many algorithms have been proposed to achieve a good tradeoff between accuracy and speed, such as SSAO [95, 96], HBAO [89], and ASSAO [97]. AO was generalized to directional occlusion by Ritschel et al. [98], adding directional shadows and color to the original AO darkening effect. Other similar perceptual effects using only the depth buffer were proposed, such as the unsharp masking operator [99].

Recently, data-driven methods using neural networks have been proposed for AO generation, e.g., NNAO [88] and Deep Shading [100]. They perform better than classical AO computation methods in the same runtime. In NNAO, the authors first collected a large number of paired depth/normal buffers and AO maps and trained a multi-layer perceptron (MLP). Deep Shading [100] confirms that CNN is better than MLP or classical methods for AO generation, in that it allows larger receptive fields through a stack of convolution and down-sampling layers. Our approach and those approaches are similar in spirit in employing neural networks for predicting the screen-space AO effect. However, those methods assume access to accurate screen-space buffers such as depth and normal. In contrast, our RGB2AO directly generates screen-space AO without an accurate estimation of normal and depth.

3.2.2 Intrinsic Image Decomposition

Intrinsic image decomposition [101] separates an image into a reflectance layer and a shading layer. Recent methods such as [102, 103] show promising results on real-world scenes. However, there is no clear way to extract AO from their shading estimation. Although shading and AO have an almost similar look under spatially uniform and non-directional lighting, our focus is on AO generation in real-world scenes that contain a diverse set of materials and objects lit by complex illumination. [104] detects AO from multiple images captured with varying illumination. [105] detects normal, reflectance, and illumination from shading. In comparison, our proposed method focuses on generating the AO of a whole scene from a single image.

Most related to our work, Innammorati *et al*. [106] decomposes a single RGB image into diffuse albedo, diffuse illumination, and specular shading and ambient occlusion. Their method aims to *estimate* AO that is already present in an image. In

contrast, we *generate* reasonable AO that is not present by inferring geometry and semantics. We emphasize that estimation and generation are entirely different tasks. This difference becomes clear in applications such as image composition, where there is no AO present between the foreground and background in the image, as shown in Sec. 3.6.1.

3.2.3 Image Editing

Image Composition Image composition is one of the most common tasks in image editing. In image composition, a foreground region of one image is extracted and pasted to a background region of another image. Generating realistic composited images requires a plausible match for both contexts and appearances. Given a composited image and a mask to identify the foreground, image harmonization methods try to match the appearance of the foreground to that of the background (or vice versa) using global statistics [107–110], gradient domain [111, 112], or supervised learning [113, 6, 114]. However, these approaches only modify inside the foreground region and do not consider the effect of placement, such as occlusion by the placed foreground region itself. For example, those methods cannot produce the soft shadow underneath a car on a sunny day. To the best of our knowledge, our RGB2AO is the first attempt to produce such an effect in image composition.

Image Relighting Lighting estimation from a single image has long been studied [115]. There has been much progress in this field thanks to data-driven approaches for both outdoor scenes [116, 117] and indoor scenes [118, 119]. Estimated lighting condition is used to photo-realistically render 3D objects into background images with many lighting conditions. However, our RGB2AO aims for inserting 2D objects, and these approaches cannot be applied.

3.2.4 Depth Estimation

Monocular depth estimation from a single RGB image is a fundamental task and has long been studied [120–123]. Recent methods try to encourage smoother gradient changes and sharp depth discontinuities [93] or obtain a model that generalizes well on datasets unseen during training [94]. However, estimating depth that is accurate enough to generate AO on top of it is very hard, as we will later show in Sec. 3.5.3.

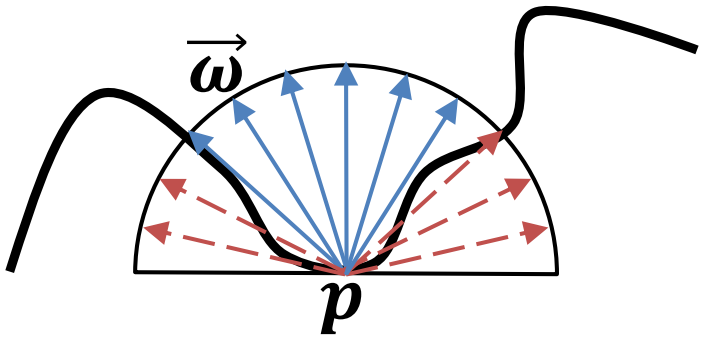


Figure 3.2: Overview of how AO is computed. A hemisphere of rays $\vec{\omega}$ is constructed for a given point $\vec{p}$ on a surface. Red (blue) arrows are rays that are (not) occluded by surrounding surfaces.

3.3 AO Formation Model

We briefly summarize how AO is typically computed and used to approximate global illumination in computer graphics. Given a surface point $\boldsymbol{p}$ shown in Fig. 3.2, ambient occlusion illumination $AO(\boldsymbol{p},\vec{n})$ is defined as follows [96]:

$$AO(\boldsymbol{p},\vec{n}) = \frac{1}{\pi}\int_{\Omega} V(\vec{\boldsymbol{\omega}},\boldsymbol{p})\max\left(0,\vec{\boldsymbol{\omega}}\cdot\vec{n}\right)d\vec{\boldsymbol{\omega}}\,, \tag{3.1}$$

where $\vec{n}$ is the surface normal at point $\boldsymbol{p}$, and $V(\vec{\boldsymbol{\omega}},\boldsymbol{p}) \in \{0,1\}$ is the visibility function over the normal-oriented hemisphere Ω, and $V(\vec{\boldsymbol{\omega}},\boldsymbol{p})$ is one if a ray starting from $\boldsymbol{p}$ intersects an occluder within some fixed distance from $\boldsymbol{p}$ and otherwise zero. The range of $AO(\boldsymbol{p})$ is $0 \leq AO(\boldsymbol{p}) \leq 1$, where $\boldsymbol{p}$ is zero when $\boldsymbol{p}$ is fully visible, and $\boldsymbol{p}$ is one when the whole hemisphere at $\boldsymbol{p}$ is occluded.

Computing integrals in Eq. (3.1) for each point of a 3D scene is requires excessive computational cost for real-time rendering. To generate a plausible AO efficiently for a specific camera viewpoint, most approaches use information from the screen-space buffers such as depth and normal of neighboring pixels to speed up the computation [95, 96, 89].

Let $\boldsymbol{x} \in \mathbb{R}^{3\times H\times W}$ be an RGB image, where H and W represent height and width of the image. Applying Eq. (3.1) on each pixel of $\boldsymbol{x}$, we obtain its corresponding grayscale AO $\boldsymbol{y} \in \mathbb{R}^{H\times W}$. An example of the generated $\boldsymbol{y}$ is shown in Fig. 3.3. Here, we instead plot the values of $1-\boldsymbol{y}$ for the purpose of intuitive visualization. We call $1-\boldsymbol{y}$ the AO map. When applying the AO effect to create a new image $\boldsymbol{x'} \in \mathbb{R}^{3\times H\times W}$, each pixel in $\boldsymbol{x'}$ is computed by multiplying its color value by the corresponding

AO map | RGB image (AO off) | RGB image (AO on)

Figure 3.3: Ambient occlusion effect applied to an RGB image. Ambient occlusion works by darkening the regions in the image according to the local scene geometry. This effect is achieved by multiplying the AO map (left) to the RGB image (middle) to obtain the enhanced render (right).

pixel in the AO map:

$$x'_{ijk} = \left(1 - a \cdot y_{jk}\right) \cdot x_{ijk}\,, \tag{3.2}$$

where i is the index for the channel dimensions, and j, k are the indices for the pixel dimensions, $0 \leq a \leq 1$ is an arbitrary chosen scaling factor. When $a = 0$, the image is not modified ($x'_{ijk} = x_{ijk}$). When $a = 1$, Eq. (3.2) is reduced to $x'_{ijk} = \left(1 - y_{jk}\right) \cdot x_{ijk}$.

In the following sections, we describe how we approximate Eq. (3.1) by using only RGB information.

3.4 Proposed Method

We propose an end-to-end trainable neural network model for AO generation from a single image. First, we describe the baseline approach following recent conditional GAN methods for I2IT in Sec. 3.4.1. We subsequently describe our AO generation model in Sec. 3.4.2. We propose two extensions: (i) multi-task learning of AO and depth prediction (in Sec. 3.4.2) and (ii) data augmentation that is specific to AO generation (in Sec. 3.4.2).

3.4.1 Baseline Model

Here we briefly introduce a recent I2IT model for our baseline. Given a set of images $\{..., (\boldsymbol{x}_i, \boldsymbol{y}_i)\}$, the objective is to obtain a generator G that converts $\boldsymbol{x}$ to $\boldsymbol{y}$. To obtain a better G, conditional GAN methods for I2IT such as Pix2pix [12] introduce a discriminator D that aims to distinguish real images from generated images.

Conditional GANs model the conditional distribution of real images by the following minimax game:

$$\min_G \max_D \mathcal{L}_{GAN}(G,D), \tag{3.3}$$

where the objective function $\mathcal{L}_{GAN}(G,D)$ is defined as

$$\mathcal{L}_{GAN}(G,D) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}[\log D(\boldsymbol{x},\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{x}}[\log(1 - D(\boldsymbol{x}, G(\boldsymbol{x})))], \tag{3.4}$$

where we use $\mathbb{E}_{\boldsymbol{x}} \triangleq \mathbb{E}_{\boldsymbol{x}\sim p_{data}(x)}$ and $\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})} \triangleq \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim p_{data}(x,y)}$.

To have larger and better receptive field, Pix2pixHD [30] uses multi-scale discriminators $D_1, ..., D_N$. An input to D_n is down-sampled by a factor of 2^{n-1}. Then, Eq. (3.3) becomes as follows:

$$\min_G \max_{D_1,...,D_N} \sum_{k=1}^{N} \mathcal{L}_{GAN}(G, D_k). \tag{3.5}$$

Pix2pixHD [30] also introduces a feature matching loss that matches an intermediate representation of a discriminator from the real and the synthesized image. Given the i-th layer feature extractor of discriminator D_k as $D_k^{(i)}$, the feature matching loss $\mathcal{L}_{FM}(G, D_k)$ is as follows:

$$\mathcal{L}_{FM}(G, D_k) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})} \sum_{i=1}^{T} \frac{1}{N_i} \left[\| D_k^{(i)}(\boldsymbol{x},\boldsymbol{y}) - D_k^{(i)}(\boldsymbol{x}, G(\boldsymbol{x})) \|_1 \right], \tag{3.6}$$

where N_i indicates the total number of elements in each layer and T is the total number of layers. Thus, the full objective is as follows;

$$\min_G \left(\left(\max_{D_1,...,D_N} \sum_{k=1}^{N} \mathcal{L}_{GAN}(G, D_k) \right) + \alpha \sum_{k=1}^{N} \mathcal{L}_{FM}(G, D_k) \right), \tag{3.7}$$

where α is a hyper-parameter to balance the two terms.

3.4.2 AO Generation Model

In this chapter, we have a set of triplets $\{..., (\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{d}_i)\}$, where $\boldsymbol{x}_i \in \mathbb{R}^{3\times H\times W}$, $\boldsymbol{y}_i \in \mathbb{R}^{1\times H\times W}$, $\boldsymbol{d}_i \in \mathbb{R}^{1\times H\times W}$ indicate an RGB image, an AO map, and a depth map, respectively. Our objective is to obtain a best generator G that converts $\boldsymbol{x}$ to $\boldsymbol{y}$. We extend the baseline model in Sec. 3.4.1 in two points that are specific to AO generation: (i)

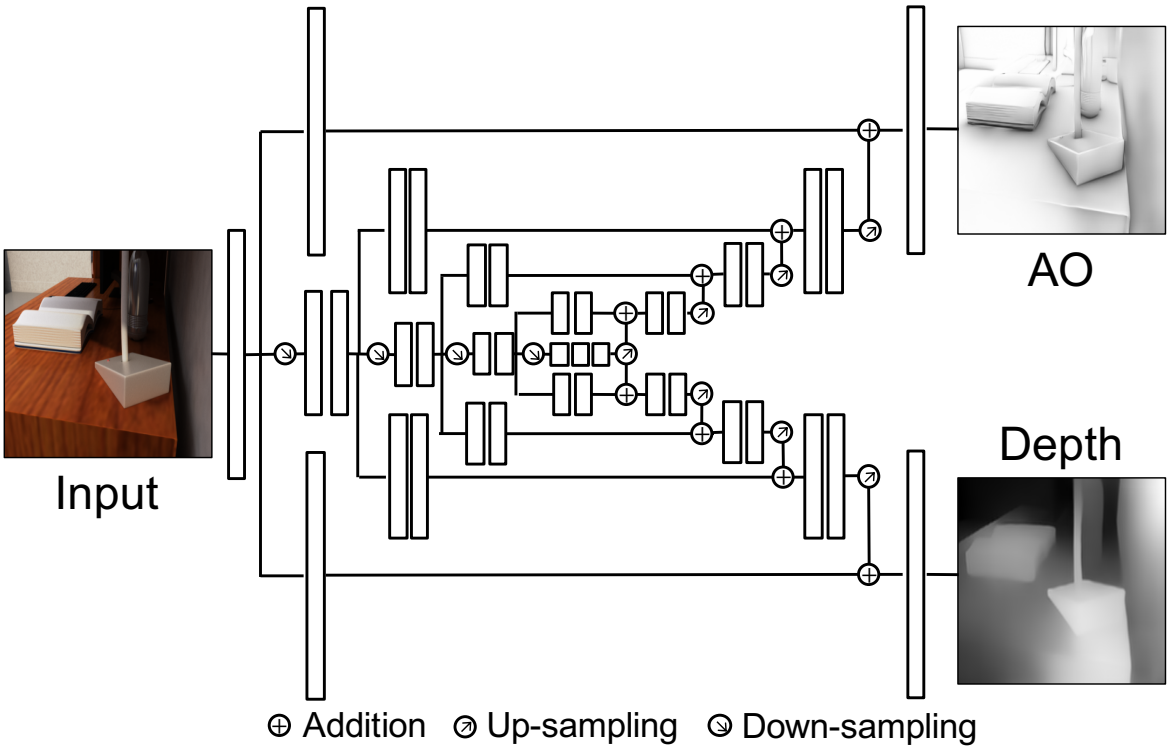


Figure 3.4: RGB2AO model overview. We develop a fully convolutional network for AO map generation from an input RGB image. We extend a variant of the Hourglass network to enable multi-task learning, so as to encourage the learned feature to capture geometry-aware information relevant to the AO generation task.

multi-task learning of AO and depth and (ii) data augmentation by randomizing AO distribution.

Multi-task Learning of AO and Depth

Simultaneously inferring both AO and depth is beneficial because ambient occlusion is closely related to 3D geometry of a scene. As shown in Fig. 3.4, we introduce a depth estimation model F that takes $\boldsymbol{x}$ and generates the depth $F(\boldsymbol{x}) \in \mathbb{R}^{1\times W\times H}$. For F, we use a similar CNN with G and share the encoder part for multi-task learning. For loss functions, we adopt loss functions used in MegaDepth [93].

$$\mathcal{L}_D(F) = \mathcal{L}_{data}(F) + \gamma \mathcal{L}_{grad}(F), \tag{3.8}$$

where γ is a hyper-parameter and $\mathcal{L}_{data}(F)$ and $\mathcal{L}_{grad}(F)$ are called scale-invariant data term and multi-scale scale-invariant gradient matching term, respectively.

Scale-Invariant Data Term Let r_i be the residual of values between predicted and ground truth log-depth at pixel position i, $\mathcal{L}_{data}(F)$ is defined as follows:

$$\mathcal{L}_{data}(F) = \frac{1}{n}\sum_{i=1}^{n}(r_i)^2 - \frac{1}{n^2}\left(\sum_{i=1}^{n} r_i\right)^2, \tag{3.9}$$

where n is the number of valid depth values in the ground truth depth maps.

Multi-Scale Scale-Invariant Gradient Matching Term $\mathcal{L}_{grad}(F)$ is defined as follows:

$$\mathcal{L}_{grad}(F) = \frac{1}{n}\sum_{k}\sum_{i}(|\nabla_x r_i^k| + |\nabla_y r_i^k|), \tag{3.10}$$

where r_i^k is the log-depth residual at position i and scale k.

Therefore, the full objective is as follows:

$$\min_{G,F}\left(\left(\max_{D_1,\ldots,D_N}\sum_{k=1}^{N}\mathcal{L}_{GAN}(G,D_k)\right) + \alpha\sum_{k=1}^{N}\mathcal{L}_{FM}(G,D_k) + \beta\mathcal{L}_D(F)\right). \tag{3.11}$$

Here, β is a hyper-parameter to balance the multi-task learning.

AO augmentation

Our synthetic dataset used for training contains RGB images devoid of AO-like effects. However, those effects are already present to some unknown extent on real-world images. In order for our method to generalize to real captured images, we propose to augment the input RGB images by adding some AO darkening during training. Formally, we use Eq. (3.2) to generate a new image, with the scaling factor a taken from the uniform distribution $\mathcal{U}(a_{min}, a_{max})$. We empirically set $(a_{min}, a_{max}) = (0.0, 0.5)$. This AO augmentation is applied on each image with probability $p = 0.75$, leaving 25% of the images without AO effects during training.

3.4.3 Dataset

To train our data-driven RGB2AO model, triplets of RGB-AO-depth data are required. We have collected a synthetic dataset since there is no dataset available. The dataset consists of 8590 triplets of RGB-AO-depth data in a resolution of 1920×1120. The dataset is rendered from 3D scenes using Maya [124] with Arnold renderer for ray-tracing. Each rendered data has its unique view, and we manually sample the

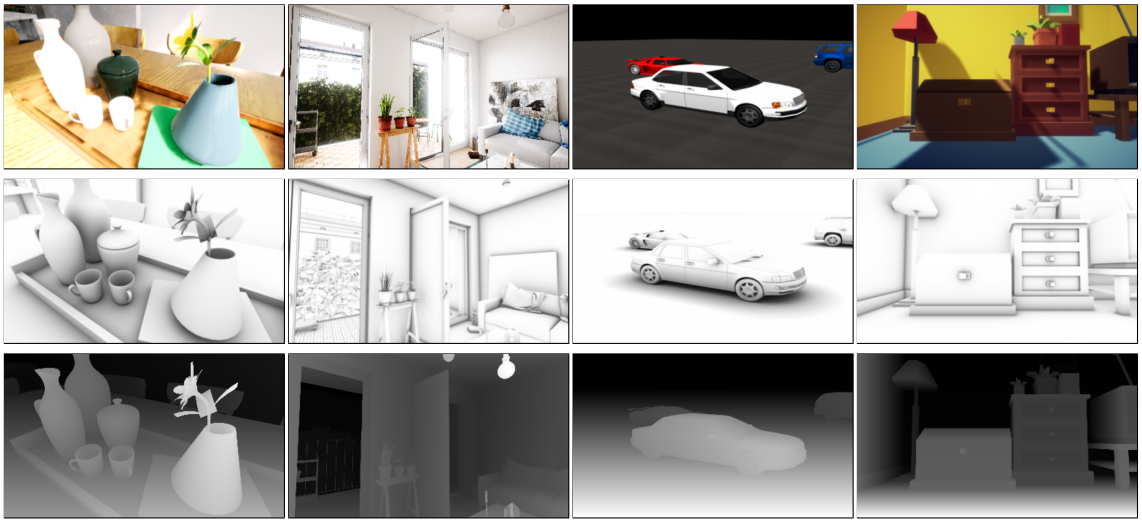


Figure 3.5: Dataset for image-based AO generation. We show some samples from our large-scale synthetic data for AO generation. From top to bottom: RGB images, AO maps, and depth maps. The data is created from high-quality 3D scenes covering a wide range of scene content. The images as well as the corresponding AO maps and depth maps are rendered using Maya with the Arnold ray-tracing based renderer.

Table 3.1: Dataset statistics.

Subset	# 3D scenes	# images
Train	21	7188
Validation	3	397
Test	4	1005

view to cover a broad range of situations, as shown in Fig. 3.5. Most of the scenes come from typical indoor scenes such as kitchen, living room, and bedroom while we include some outdoor scenes and non-photorealistic scenes. We also manually compose some scenes that contain objects on the floor or cars on the synthetic ground to cover possible situations.

For rendering, we use a perspective camera with the focal length of 18 to make the view wide enough for indoor situations. We set the "falloff" parameter for the AO in Arnold to 0.2. We found that using larger values caused the AO to spread very far from the objects making it harder for the model to infer. In addition to RGB and AO, we also rendered a screen-space depth buffer to see whether simultaneously estimating 3D geometry information benefits the AO generation.

3.5 Experiments

3.5.1 Network Architecture

For the generator G, we used a variant of a hourglass network used in monocular depth estimation [92]. The hourglass network consists of multiple convolutions by a variant of inception [125] and down-sampling, followed by multiple convolutions and up-sampling, interleaved with skip connections. We duplicate skip-connection parts and decoder parts for multi-task learning, as shown in Fig. 3.4. Since it is a fully convolutional network, it can be applied to images with arbitrary sizes during the test phase. For the discriminator D, we used a discriminator which is identical to the one used in Pix2pixHD [30].

3.5.2 Training

G and D were trained for 100 epochs starting with a learning rate of 2.0×10^{-4} and a batch size of 6 with Adam optimizer [73]. We kept the learning rate for the first 50 epochs and then linearly decay the rate to zero over the next 50 epochs. It took about a day for the training to finish. For the hyper-parameters we set $(\alpha, \beta, \gamma) = (1.0, 1.0, 0.5)$ in all the experiments. Since the number of samples in the dataset is limited, we also performed massive standard data augmentation to enhance the generalization capability of our model. We changed contrast, brightness, saturation, and hue of each image. All the images were resized to 384×224. During training, the images were randomly flipped and cropped to 352×192. During testing, images were center-cropped to 352×192 for the quantitative evaluation.

3.5.3 AO Generation Performance

We performed the quantitative evaluation on our synthetic dataset to prove the validity of our model for AO generation. We split the images in our dataset into train, validation, and test subsets so that each does not share the same 3D scenes, as shown in Table 3.1. The quantitative evaluation was performed on the images with no AO applied in the test subset.

Evaluation Metrics

We evaluated the performance of AO generation methods by comparing the generated AO maps with the corresponding ground-truth AO maps over the whole testing set.

In addition to mean absolute error (MAE) and mean squared error (MSE), we used the following evaluation metrics:

- SSIM: Structural similarity (SSIM) index [126] is widely used in quantifying the perceptual similarity between two images. Higher is better.
- LPIPS: Learned Perceptual Image Patch Similarity (LPIPS) metric [127] is a recently developed measure for perceptual similarity assessment. LPIPS uses features extracted from CNN trained on a dataset of human perceptual similarity judgments on image pairs. Lower is better.

Compared Methods

To the best of our knowledge, there is no existing image-based AO generation method in the literature. In this experiment, we evaluated the AO generation performance of our method and compared it with the following methods: (i) screen-space AO methods on top of monocular depth estimation results or (ii) Innamorati *et al*. [106]'s model originally for AO estimation.

Depth Estimation + Screen-Space AO We first trained a monocular depth estimation network, which we call RGB2D for short, using the same dataset. We further improved the quality of the generated depth map by applying bilateral filter [128] to smooth the predicted depth values. For training the RGB2D model, we used the same network with F to extract depth. Second, we applied different screen-space AO generation methods on top of the resulting depth estimation results to obtain the final AO map. We experimented with the following three variants.

- **RGB2D+SSAO**: We used a traditional AO generation method, SSAO [96].
- **RGB2D+CNN**: We used a CNN that is almost similar to G for a fair comparison. Only the difference is changing the input from a three-channel RGB image to a one-channel depth map.
- **RGB2D (fixed) + CNN**: One may argue that monocular depth estimation methods trained on a mixture of various datasets can be used to extract fine depth without training on our dataset. We tested a pre-trained monocular depth estimation method TMRDE [94], which generalizes well to an unseen dataset, without training.

Innamorati *et al*. We compare three methods derived from Innamorati *et al*. [106]'s model.

Table 3.2: Experimental results of AO generation on our synthetic dataset. ↓ and ↑ indicate that lower and higher is better, respectively.

	MAE ↓	MSE ↓	SSIM ↑	LPIPS ↓
RGB2D+SSAO	0.0848	0.0193	0.611	0.524
RGB2D+CNN	0.0785	0.0181	0.687	0.484
RGB2D(fixed)+CNN	0.0797	0.0186	0.689	0.549
Innamorati-est	0.0952	0.0215	0.671	0.423
Innamorati-ft-est	0.0655	0.0120	0.764	0.311
Innamorati-ft-gen	0.0668	0.0107	0.763	0.329
Ours	**0.0589**	**0.0103**	**0.767**	**0.235**

- **Innamorati-est**: We directly run publicly available Innamorati *et al.*'s model which is trained on their own dataset.

- **Innamorati-ft-est**: We finetuned Innamorati *et al.*'s model on our dataset for AO *estimation*. Because the task is AO estimation, we used pairs of RGB with AO and AO as the input and the target of the model, respectively.

- **Innamorati-ft-gen**: For a fairer comparison, we finetuned the model on our dataset for AO *generation*. We applied the L2 loss to the occlusion output only. We ignored the other outputs and losses because they are irrelevant or detrimental. For example, the reconstruction loss forces the network to *detect* the AO present and thus unfairly hurt the network's ability to *generate* missing AO.

Results

Accuracy Quantitative results are shown in Table 3.2. Our model outperforms both types of approaches in each metric by a significant margin. We show generated AO on an excerpt of the test set in Fig. 3.6. The Screen-Space AO approaches fail to capture many of the details on real images, because they rely heavily on monocular depth estimation results, which does not capture the high-frequency details for accurate SSAO computation. This demonstrates the importance of learning a direct RGB-to-AO mapping.

Innamorati *et al.*-based approaches produce very blurry outputs even if the model is finetuned for AO generation. This demonstrates the importance of the choice of the networks and loss functions specialized for AO generation.

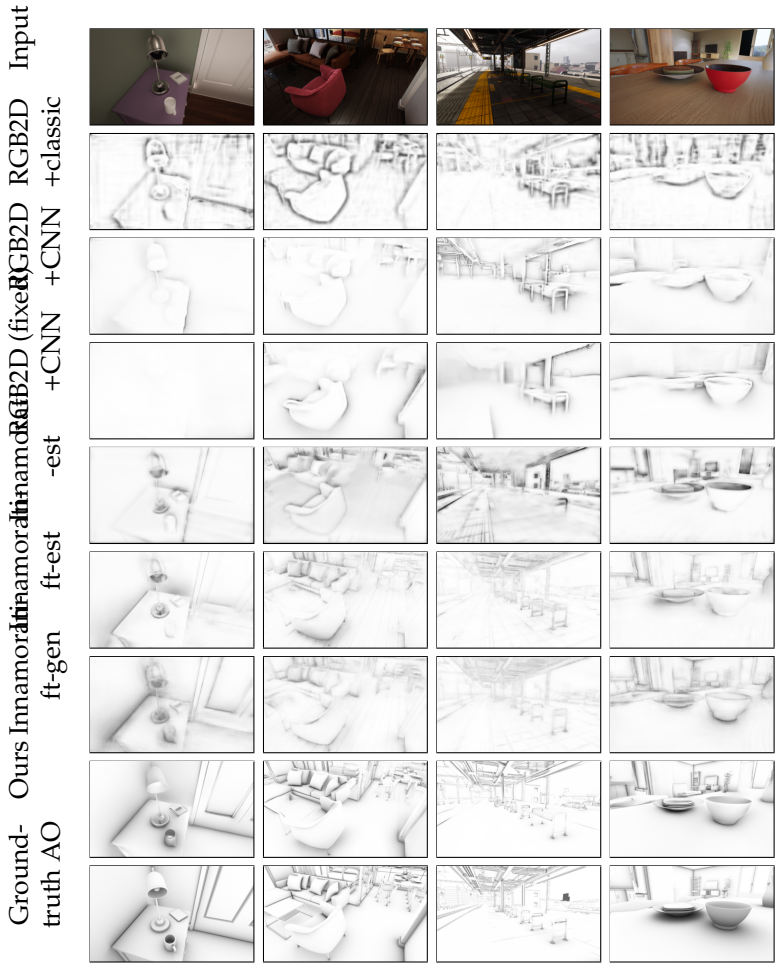


Figure 3.6: Comparison of AO generation methods on the test subset of our collected synthetic dataset with no AO present in the input image. (best viewed with zoom and in color.)

Speed We benchmarked our model on several image resolutions. Since our model is a fully convolutional network, it can take images with arbitrary aspect ratios as input. We tested images initialized randomly and show the average time on 100 runs on a TITAN V GPU. Our model runs at about 15 FPS and 8 FPS for 384×224 and 768×448 inputs, respectively.

3.5.4 Ablation Study

Contribution of Proposed Components

We performed quantitative evaluation by changing components of our model and the result is shown in Table 3.3. Both of the two components are essential to achieve the

Table 3.3: An ablation study on our proposed components. (i) and (ii) indicate AO augmentation (Sec. 3.4.2) and multi-task learning of AO and depth prediction (Sec. 3.4.2), respectively. ↓ and ↑ indicate that lower and higher is better, respectively.

(i)	(ii)	MAE ↓	MSE ↓	SSIM ↑	LPIPS ↓
		0.0622	0.0110	0.748	0.243
✓		0.0600	**0.0103**	0.760	**0.235**
	✓	0.0607	0.0107	0.755	0.241
✓	✓	**0.0589**	**0.0103**	**0.767**	**0.235**

Table 3.4: An ablation study on changing a_{max} in our AO augmentation (Sec. 3.4.2) during training.

a_{max}	0.0	0.25	0.5	0.75	1.0
SSIM ↑	0.755	0.762	**0.767**	0.764	0.762

best result. Surprisingly, the model without both AO augmentation and multi-task learning already clearly surpasses all the compared methods in Table 3.2. This is due to our choice of networks and loss functions specific to AO generation. We performed an ablation study about these choices in the supplementary material.

AO Augmentation

AO augmentation that we propose in Sec. 3.4.2 has one hyper-parameter a_{max}, where $0.0 \leq a_{max} \leq 1.0$, to control the strength of the AO effect that we apply during training. Bigger a_{max} leads to include images with much AO effect during training. We tested different a_{max} and the result is shown in Table 3.4. As we increase a_{max} from 0.0 (no AO effect applied) to increase the AO effect, the performance was steadily improved until $a_{max} = 0.5$. However, it decreased when $a_{max} > 0.5$. This indicates that including images with excessive AO effect is harmful to improve the performance.

3.6 Applications

AO generation is useful for many image editing and enhancement tasks. We demonstrate its ability to improve 2D image composition and for increasing contrast in a geometry-aware manner. We edit the input RGB image using the generated AO by the simple element-wise multiplication following Eq. (3.2). We set the scaling

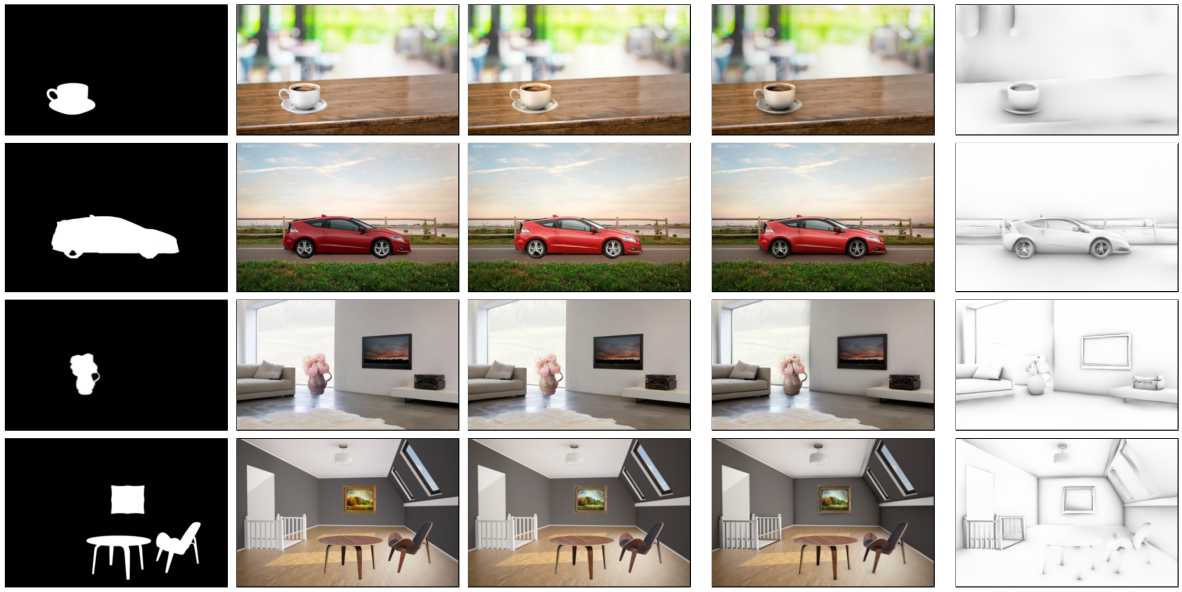


Figure 3.7: Example results on 2D image composition. Our method adds a plausible shadow-like effect on both the foreground and background region using the generated AO. Our method is complementary to DIH [6], which only changes the appearance inside the foreground region. Images in the second and third row are from DIH. Images in the first and fourth row from Adobe Stock. (best viewed with zoom and in color.)

factor $a = 1.0$ in Eq. (3.2) to show all the results in this section. The experiments are performed on images of size 384×224. Larger input images can be handled by (i) resizing the image to a lower resolution, (ii) estimating AO, (iii) resizing it back to the original resolution, and (iv) multiplying the generated AO with the input. We show some examples of processing high-resolution images in the supplementary material.

3.6.1 2D Image Composition

For image composites to look realistic, the lighting of the inserted object must match the background scene. Prior harmonization methods such as [6] have been developed to address this. Additionally, the object will cast shadows or otherwise affect the lighting of the scene around it. While lighting estimation methods have been used to insert 3D objects, inserting 2D objects is more complicated. With our AO generation, we can address this. Given a composited image, we can apply the generated AO

to make it look more realistic as the AO will darken the contact regions around the composited object to simulate the shadows and help ground the object in the scene.

User Study To evaluate the effectiveness of our image composition results on real images quantitatively, we performed a user study. We created a set of 26 images that covers a wide variety of real examples, where some objects are composited (e.g., cups, cars, and chairs). Some of the images were taken from the evaluation set of Deep Image Harmonization (DIH) [6]. Some example results are shown in Fig. 3.7. To show that our method is complementary to existing image harmonization methods, we used the harmonized image by DIH as an input to our RGB2AO model. We showed the two images to the subject and asked which one looks more realistic. As a result, a total of 9 subjects participated in this study, with a total of 324 votes. Our method obtained 80% of the votes (260 votes) over the compared method, showing a clear preference for our method on this composition task.

Discussion One limitation of RGB2AO applied to 2D image composition is that our composition sometimes changes some regions of the background image, where compositing the foreground region should not affect. For example, in the third and fourth row of Fig. 3.7, boundaries between walls and floors were exaggerated apparently, but the composition of objects should not affect those regions. A practical solution is to let users choose the region where the generated AO is multiplied. Extending the current AO generation to automatically propose regions is a promising research direction for future work.

3.6.2 Geometry-Aware Image Contrast

We can apply the generated AO map as an image filter for RGB images in order to improve image contrast in such a way as to emphasize the depth in the image. We enumerate some usages:

Avoiding Flat Look Images without sufficient contrast appear flat and dull. Applying pixel-level contrast enhancement methods can improve the image appeal, but might break its overall coherence. For example, shadows may become either over- or under-exposed. Instead, our method allows applying contrast in a geometry-aware manner by using AO to darken the right areas based on the scene geometry. Some example results are shown in Fig. 3.8. For comparison, we also tested an auto-contrast enhancement method, Auto Contrast in Adobe Photoshop. We can see that our geometry-aware image contrast behaves entirely differently from Auto Contrast from the second and third columns in Fig. 3.8.

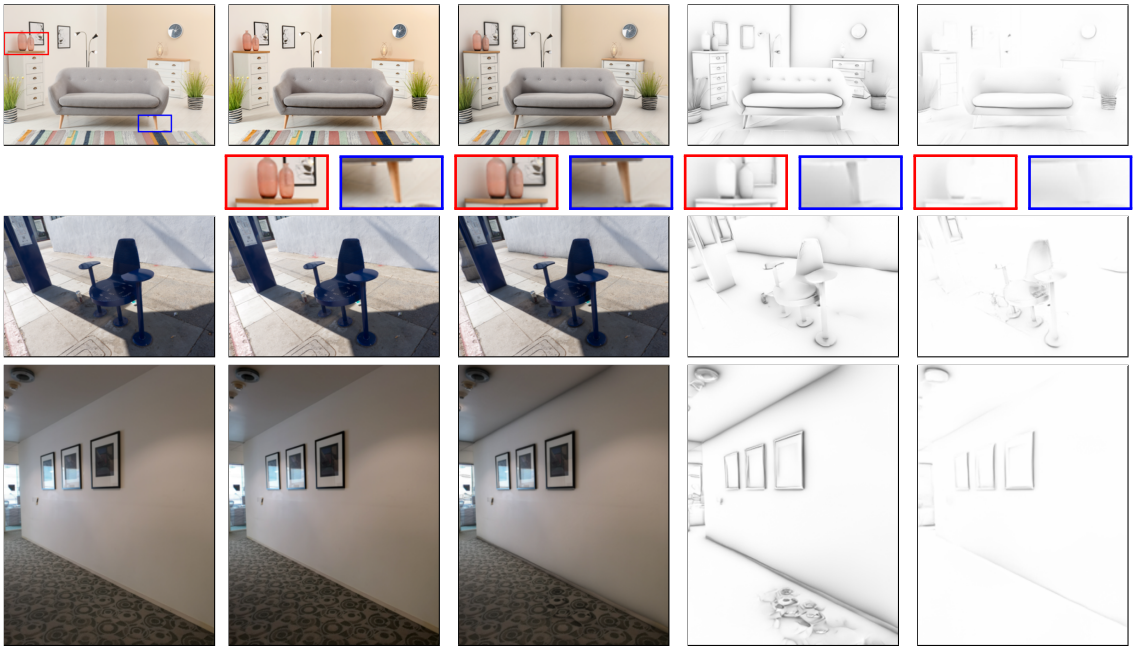


Figure 3.8: Comparison between our AO-based contrast enhancement and Auto Contrast in Adobe Photoshop. Note how Auto Contrast changes the global appearance of the image while our technique focuses on darkening object boundaries such as the bottles (red) and under the sofa (blue) in the first row. AO estimation struggles to detect AO under real photos with complex illumination, such as the boundary between the wall and the ceiling in the third row. Images in the first and second rows from Adobe Stock. (best viewed with zoom and in color.)

One may argue that AO estimation can perform similarly on real photos where no AO is missing. We drop AO augmentation part from our model and train it for AO estimation using pairs of RGB with AO on and AO map. We show the result of AO generation and estimation in the fourth and fifth columns in Fig. 3.8, respectively. Real photos can contain full AO if the scenes are lit only by spatially uniform and non-directional lighting. However, real scenes are usually lit by complex illumination, making the visible AO reduced and making the AO estimation difficult, as we can see. Thus, we believe that AO generation is different from AO estimation even in real photos, and it is suitable for our downstream application, geometry-aware contrast enhancement.

Manipulating Non-Photorealistic Images The usage of our RGB2AO model is not limited to photorealistic RGB images. Some artists have tried to depict AO-like soft shadows on illustrations manually by brush or some other tools. In contrast,

Input | Generated AO | Geometry-Aware Image Contrast

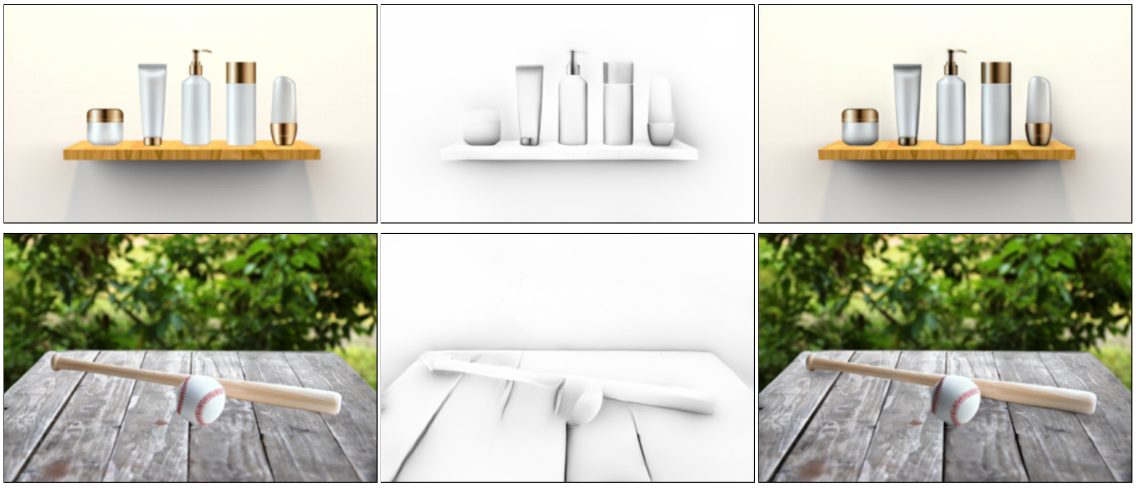
Figure 3.9: Results of our RGB2AO on non-photorealistic images. Images from Adobe Stock.

our RGB2AO model can do it in a fully-automatic way. Results of manipulated non-photorealistic illustration images are shown in Fig. 3.9. We can see that our model can generate plausible AO on everyday objects (e.g., chairs, plants, and pots) and scenes (e.g., room).

3.7 Limitations and Discussion

One limitation of our RGB2AO model is that the model struggles on families of objects and scenes that are not under-represented in the dataset for training. Some examples of the failure cases are shown in Fig. 3.10. Collecting larger datasets with more diverse scenes would alleviate this type of errors.

Another limitation is that our RGB2AO can only handle non-directional shadow-like effects. Augmenting our AO generation framework with lighting prediction in addition to geometry understanding can potentially handle such directional shadow or inter-reflection. We hope our proposed method paves the way for future work in this direction.



Input Generated AO AO Magnification

Figure 3.10: Failure case examples. Our model failed to obtain AO on the bottom of bottles and a ball in the first and second row, due to the lack of similar content in the training data. Images from Adobe Stock. (best viewed with zoom and in color.)

Chapter 4

Learning from Synthetic Shadows for Shadow Detection and Removal

4.1 Introduction

Shadows are prevalent in nature. Detecting and manipulating shadows is essential in many computer vision and computer graphics downstream tasks [129–131]. Research on shadow removal has advanced drastically by learning-based methods [132–134], especially by convolutional neural networks (CNN) [131, 9, 135, 136]. These methods are usually trained on pairs of shadow and shadow-free images of identical scenes in a supervised manner. In order to make these pairs, we must first take a photo of a scene with shadows and then take another photo after removing the occluder. Additionally, we should manually or automatically annotate a binary mask indicating the presence of shadows if the model requires mask information. This process dramatically limits the number and variety of collectible scenes and can cause noisy/biased supervision. Shadow removal models learned from such data do not generalize well to broad real-world shadow images, as discussed in [10]. Besides, there is no guarantee that shadow-free regions are unchanged under the sunlight conditions, making the paired training images unreliable.

Some approaches have been proposed to overcome this challenge. The first approach [10] learns the model on unpaired shadow and shadow-free images. It is prevalent in many image-to-image translation tasks (e.g., CycleGAN [14]), but the optimization is under-constrained, and shadow models learned from unpaired data often perform poorly. The second approach [8] augments the existing paired triplets datasets, but the variety of images produced by this approach is limited because it

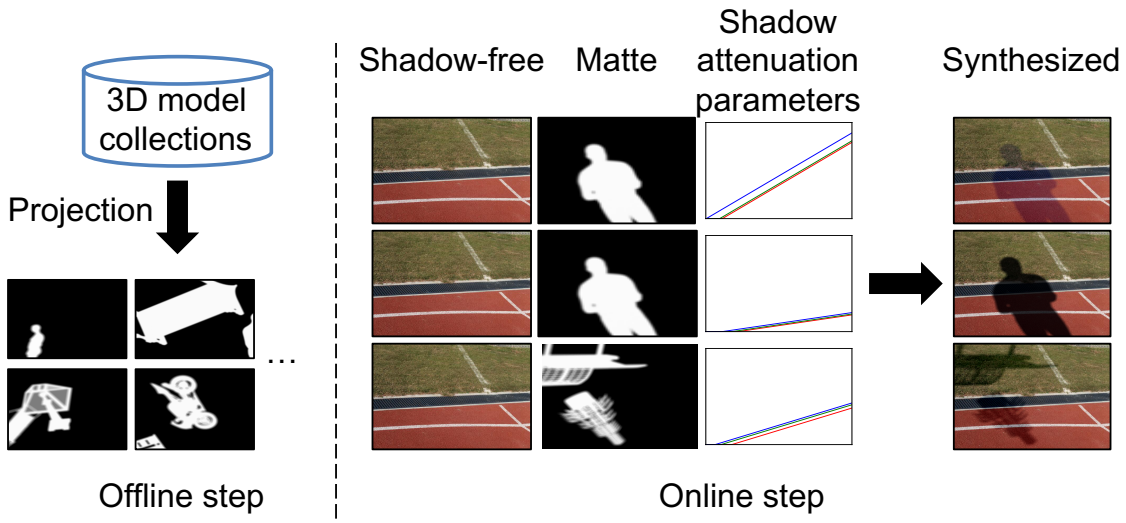


Figure 4.1: Overview of our shadow synthesis pipeline. It can efficiently synthesize diverse and realistic shadow/shadow-free/matte image triplets. The triplet can be obtained from arbitrary combination of a background image, shadow shape, and shadow attenuation property. Note that matte is **not** binary.

just slightly darken or enlighten shadows given a triplet. The third approach [7] obtains a new shadow image given a combination of shadow-free and mask image pairs. However, the variety of generated shadows is limited, especially in terms of shadow intensity. This is because the mapping between them is learned fully in a data-driven way from the existing shadow datasets, which is a severe bottleneck, as we have mentioned above.

In this paper, we tackle the mentioned challenge in shadow removal by generating a large-scale, diverse, yet realistic shadow/shadow-free/matte triplets dataset for supervised learning. As shown in Fig. 4.1, in our pipeline, a shadow image is composed of three components; (i) a shadow-free image as a background, (ii) a grayscale matte image indicating where the shadows are observed, and (iii) parameters indicating the shadow attenuation property. We find that regarding these components to be independent of each other is essential compared to the previous composition approach [7]. For (iii), we extend a physically-grounded shadow illumination model [137]. We transform the model so that simply randomly sampling a set of parameters of it enables us to obtain diverse but realistic shadow attenuation.

Similar to the condition of existing shadow removal datasets, we assume all the occluder objects are outside the camera view in practice. We separate the generation

process into two independent parts, shadow matte generation (offline) and shadow composition using a shadow illumination model extending [137] (online), as shown in Fig. 4.1. In the offline step, we render shadow matte images from publicly available 3D models. In the online step, we randomly sample the three components and then synthesize a shadow image from them. The online step is computationally very cheap, and we can obtain new and diverse triplets on-the-fly during the training of neural network models.

Based on the pipeline, we propose a new dataset called SynShadow, containing shadow/shadow-free/matte image triplets synthesized from rendered 10,000 matte images and about 1,800 background images. We train a variety of shadow detection and removal models on it to demonstrate its usefulness. We demonstrate that simply fine-tuning from a SynShadow-pre-trained model improves existing shadow detection and removal models. We demonstrate robust shadow removal results on challenging USR [10] dataset, outperforming both supervised and unsupervised learning models trained on existing datasets in a user study. We provide empirical and detailed analysis of the synthesis pipeline's design and demonstrate its superiority over existing shadow synthesis methods.

In summary, our contributions are as follows:

- We present SynShadow, a large-scale dataset of shadow/shadow-free/matte image triplets, and the pipeline to synthesize the diverse and realistic triplets.
- We demonstrate the usefulness of SynShadow in improving various existing CNN models for shadow detection/removal by fine-tuning and achieving robust shadow removal on more challenging inputs.

4.2 Related Work

4.2.1 Shadow Removal Methods and Datasets

Shadow removal is essential mainly in two aspects; (i) it is useful as a pre-processing step for downstream applications that are not robust to shadow (e.g., document recognition [138–141]) and (ii) it will give a perceptually better photo to users (e.g., in portraits [142]). Classical approaches remove shadows via user interaction [11] or hand-crafted features [132, 143–145]. The emergence of CNN enables shadow removal to learn from shadow/shadow-free image pairs, or shadow/shadow-free/mask triplets [9, 135, 136] by regressing all the pixel values directly. In this approach,

researchers have been trying to efficiently fuse both global and local contexts such as a multi-stream network [9], a two-stage approach [135], a direction-aware attention mechanism [136], or a recurrent module [146]. In contrast to these works, SP+M [8] proposes a shadow image decomposition-based approach. First, given an input shadow image, SP+M predicts parameters of a physical shadow illumination model [137] to generate a relit image. Second, it predicts a shadow matte layer to blend the input shadow image and the relit image to generate the final de-shadowed image. We show that our SynShadow can improve the performance of both approaches.

The datasets for shadow removal that we have discussed about are constructed by taking a photo with shadows and then taking another photo of the scene by removing the occluder. This approach suffers from several limitations, as discussed by [10]. First, it is a very time-consuming process, resulting in a limited number of unique scenes in the exiting shadow removal datasets. Second, it limits the diversity of the collected shadow shapes since some occluder such as buildings and trees cannot be removed. Third, the training pairs may have color/luminosity inconsistency or a slight shift in the camera view since the camera exposure, pose, and environmental lighting may change during the pair collection process.

Mask-ShadowGAN (MSGAN) [10] proposes to learn shadow removal from unpaired and diverse shadow/shadow-free images by extending CycleGAN [14]. CycleGAN was modified to learn a mapping from a set of a shadow-free image and a binary shadow mask to a single shadow image and vice versa. However, learning from unpaired data is quite unstable, and even shadow-free regions are often retouched, resulting in unnatural/undesired color/texture shifts. In contrast, we convert diverse shadow-free images into the paired triplets for supervised learning to cope with this limitation.

4.2.2 Shadow Synthesis

Existing approaches for synthesizing diverse and realistic shadow images are divided into three.

The first approach is to render shadow/shadow-free image pairs directly using a 3D renderer. Sidorov *et al*. [147] renders these pairs in urban landscape from a computer game. However, these images are significantly different from popular benchmarks for shadow removal, SRD [9] and ISTD [135], because most of the occluder in these images is inside the camera view. Gryka *et al*. 's work [133] for user-guided shadow removal first loads it to a 3D renderer, places automatically-

segmented silhouettes of real objects, and renders shadows on the background by ray-tracing. However, their rendering results may be physically incorrect because the renderer has no access to the background image's material information. We composite shadows on background images with the help of shadow illumination model [137] that directly models the relationship between shadow and shadow-free images.

The second approach is to augment existing real datasets, which we call *shadow augmentation*. Le *et al*. [8] use the same shadow illumination model with our approach. Specifically, given a shadow/shadow-free/binary mask triplet, they estimate the parameters of the shadow illumination model and only slightly perturb the shadow attenuation slope (by a scaling factor ranging from 0.8 to 1.2) to create slightly different shadow images. However, there are two drawbacks. First, the variety of the generated shadow images is not diverse since they only augment existing datasets and cannot generate the images from a new scene. Second, they regard these parameters as being dependent of physical conditions in the scene of the given triplet. The parameters obtained from one triplet cannot be transferred to another triplet. On the other hand, using our proposed parameter randomization of the shadow illumination model, we can synthesize shadow/shadow-free/matte image triplets given arbitrary combinations of shadow-free background images and shapes of occluder objects.

The third approach is composing shadows given an arbitrary shadow-free image as a background, which we call *shadow composition*. Cun *et al*. [7] propose Shadow Matting GAN (SMGAN), which composes realistic shadows given a shadow-free image and a randomly sampled shadow mask using CNN. However, there are two drawbacks. First, SMGAN itself is learned from existing real shadow/shadow-free/mask image triplets, which also limits the variety of shadows that SMGAN can generate in terms of shadow intensity. Second, SMGAN assumes a one-to-one correspondence between shadow-free/mask pair collections and shadow image collections for learning the mapping between them. This is unrealistic because an unlimited number of shadow images with varying shadow intensity can be possible given a shadow-free/mask image pair. In contrast, we generate hundreds of different shadow images of a scene with varying shadow intensity given the same pair.

There are some recent papers on shadow generation [148–150]. However, they are not applicable to our setting. ARShadowGAN [148] and RGB2AO [149] are for casting shadows caused by the occluder inside the camera view. Wang *et al*. 's work [150] is limited to a scene where a video from a single viewpoint is available.

4.2.3 Shadow Detection

Traditional approaches for shadow detection develop a physical model based on illumination invariant assumption [143, 151–153] or employ various hand crafted features [129, 132, 134, 154–157]. Recent approaches use CNN [131, 158, 159] and try to effectively capture global and local context [135, 136, 160, 161]. DSDNet [162] proposes to mine hard-positive/negative regions in the existing dataset and develop losses and modules to attend these regions explicitly. Being orthogonal to DSDNet, we synthesize a large-scale dataset including various hard-positive/negative regions and let the detection models learn from them implicitly.

Collecting datasets for shadow detection is essential since prevalent approaches are data-hungry. Existing datasets are classified into three: (i) when the occluder is outside the camera view [9, 135], (ii) when the occluder is inside the camera view [159, 163], and (iii) mixture of both [54]. We show that our synthetic dataset is useful to improve the performance of shadow detection models for the case (i).

4.3 SynShadow Pipeline

Our goal is to synthesize a large number of triplets $(\boldsymbol{x}^s, \boldsymbol{x}^{ns}, and \boldsymbol{m})$. $\boldsymbol{x}^s \in \mathbb{R}^{H\times W\times 3}$, $\boldsymbol{x}^{ns} \in \mathbb{R}^{H\times W\times 3}$, and $\boldsymbol{m} \in \mathbb{R}^{H\times W}$ indicate a shadow image, a shadow-free image, and a shadow matte, respectively. H and W indicate the height and width of the images, respectively. All the values in $\boldsymbol{x}^s$, $\boldsymbol{x}^{ns}$, $\boldsymbol{m}$ are between zero and one. Our pipeline consists of two steps, as shown in Fig. 4.1: shadow matte generation and shadow composition. We assume that we can compose any shadow matte on random background images only when occluders are outside the image.

In Sec. 4.3.1, we briefly summarize the shadow illumination model used in [8, 137] formulating the relation between shadow and shadow-free pixels. In Sec. 4.3.2, we explain how to synthesize shadow given an arbitrary combination of ($\boldsymbol{x}^{ns}$, $\boldsymbol{m}$) using the illumination model. In Sec. 4.3.3, we explain how to obtain diverse shadow matte images.

4.3.1 Preliminary: Shadow Illumination Model

In [137], the model is derived from the image formation equation proposed in [164]:

$$I(p,\lambda) = L(p,\lambda)R(p,\lambda), \tag{4.1}$$

where $I(p, \lambda)$ is a scalar value indicating the intensity of the light reflected from the point p at the wavelength λ. L and R indicate the luminance and the reflectance, respectively. The model assumes a single primary light source and an ambient light as the sources of illumination. It is further assumed that the shadow is cast solely by the primary light source (e.g., the sun in outdoor scenes). If p is shadow-free (i.e., lit), L at p can be expressed as a sum of two terms:

$$L(p, \lambda) = L^d(p, \lambda) + L^a(p, \lambda), \tag{4.2}$$

where L^d is the direct illumination and L^a is the ambient illumination. Therefore, the intensity $I^{lit}(p, \lambda)$ we see at p is expressed as

$$I^{lit}(p, \lambda) = L^d(p, \lambda)R(p, \lambda) + L^a(p, \lambda)R(p, \lambda). \tag{4.3}$$

In the case where some objects occlude the primary light source from the point p and shadows are casted, the reflected intensity I^{dark} is

$$I^{dark}(p, \lambda) = a(p)L^a(p, \lambda)R(p, \lambda), \tag{4.4}$$

where $a(p)$ is a factor indicating the attenuation of the ambient illumination by the occluder. $a(p)$ is assumed to have roughly the same spectral distribution from all incident directions. It is thus assumed to be independent of λ. By combining Eq. (4.3) and Eq. (4.4), the relation between I^{lit} and I^{dark} is formulated as follows:

$$I^{lit}(p, \lambda) = L^d(p, \lambda)R(p, \lambda) + \frac{1}{a(p)} I^{dark}(p, \lambda). \tag{4.5}$$

When a photo is taken, the actual color at a pixel in the photo corresponding to the 3D point p in the scene is obtained by integrating the both sides of Eq. (4.5) with the camera's spectral response functions. This operation is assumed not to change the affine nature of the relationship between the shadowed and illuminated intensities. Thus, the relation between I^{lit} and I^{dark} at any pixel for the k-th color channel (k = 0, 1, and 2 for red, green, and blue) are expressed as follows:

$$I_k^{lit} = \alpha_k + \gamma\, I_k^{dark}, \tag{4.6}$$

where both α_k and γ are scalar values. Thus, the shadow attenuation property of a 3D scene is represented by α_k and γ. These parameters depend on camera and scene properties, such as the material of surfaces and lighting conditions. Shor *et al.* [137]

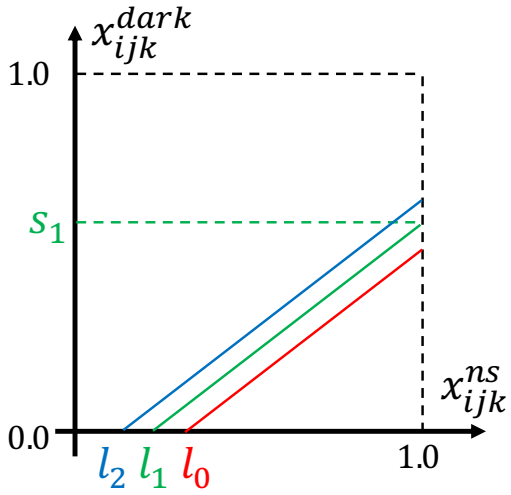


Figure 4.2: Parameters (l_0, l_1, l_2, s_1) we introduce for analysis.

demonstrated that it applies well to actual photos, even though it is not perfect due to the presence of noise and the variations in the reflectance of surfaces.

4.3.2 Shadow Synthesis

Here we describe the proposed shadow synthesis pipeline. Let a shadow free image be $\boldsymbol{x}^{ns}$. We first obtain an image $\boldsymbol{x}^{dark} \in \mathbb{R}^{H \times W \times 3}$, where all the pixels are sheltered and have the same attenuation property. We use the affine model in Eq. (4.6) to compute $\boldsymbol{x}^{dark}$ from $\boldsymbol{x}^{ns}$:

$$x_{ijk}^{ns} = \alpha_k + \gamma \, x_{ijk}^{dark} \tag{4.7}$$

$$\Longleftrightarrow x_{ijk}^{dark} = \frac{1}{\gamma} \, x_{ijk}^{ns} - \frac{\alpha_k}{\gamma}, \tag{4.8}$$

where i and j indicate indices for row and column axes, respectively, for all the images.

The final image with shadows in some regions, $\boldsymbol{x}^s$, is obtained by composing $\boldsymbol{x}^{ns}$ and $\boldsymbol{x}^{dark}$ by alpha composition using the shadow matte $\boldsymbol{m}$ as the alpha factor;

$$x_{ijk}^{s} = (1 - m_{ij}) x_{ijk}^{ns} + m_{ij} x_{ijk}^{dark}. \tag{4.9}$$

We next discuss how to sample α_k and γ to produce both plausible and diverse shadows. This is unknown and non-trivial. We provide observation for understanding the relation of α_k and γ. To clearly explain it, we convert α_k and

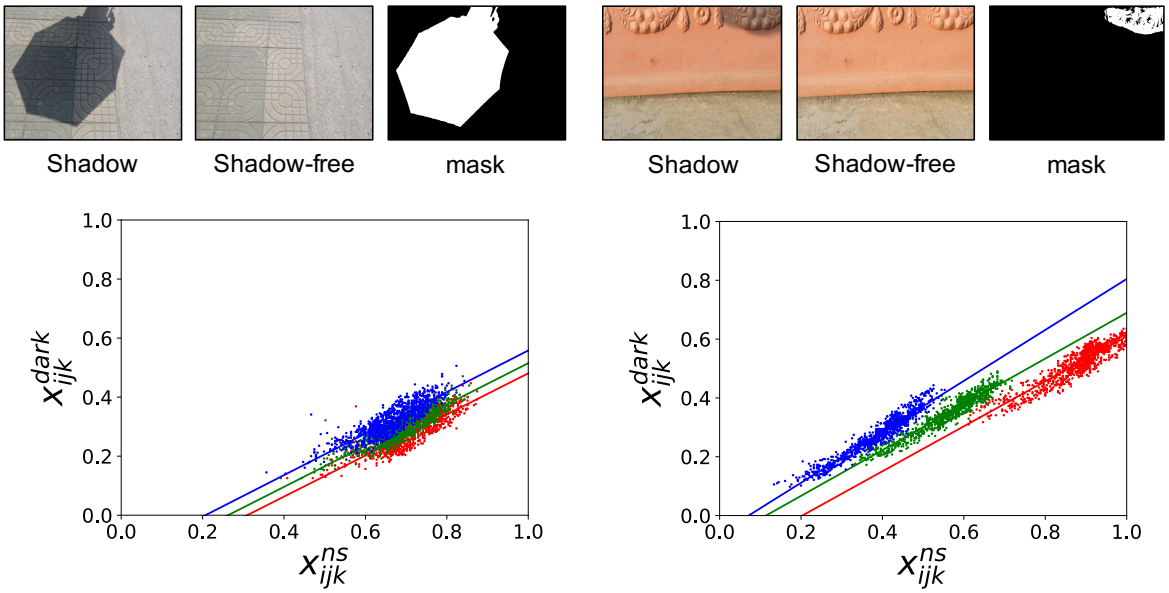


Figure 4.3: Visualization of the shadow attenuation for each RGB channel in ISTD+ and SRD+ training set. The left and right side are the visualization of an example triplet from ISTD+ and SRD+ training set, respectively. We fit linear functions regressing x^{dark}_{ijk} from x^{ns}_{ijk} for each channel and show the estimated functions as the lines.

γ to four parameters (l_0, l_1, l_2, s_1) as shown in Fig. 4.2. Formally, this is written as: $s_1 = \frac{1-\alpha_1}{\gamma}, l_k = \alpha_k$. Given a shadow/shadow-free/mask image triplet in the ISTD [8] and SRD [9] datasets, we visualize the relationship between the intensity of pixels in the shadowed region in Fig. 4.3. We confirm that the shadow illumination model in [137] holds. Besides, we observe that (l_0, l_1, l_2) correlate to each other, and often the relation is $l_0 > l_1 > l_2$. We conjecture that this is because the ambient light is from the blueish sky in outdoor scenes, which is also suggested in [156].

Based on this observation, we parametrize the shadow illumination model that covers a plausible but diverse range of shadows. Inspired by domain randomization [165], we obtain a set of parameters, where each of them is easy to sample from an interpretable yet straightforward prior distribution and independent of each other. Since l_k's are dependent of each other based on the observation, we further introduce $\Delta l_0 = l_0 - l_1$ and $\Delta l_2 = l_2 - l_1$ and sample $(l_1, s_1, \Delta l_0, \Delta l_2)$. We employ a uniform distribution $\mathcal{U}(a,b)$ for both l_1 and s_1. $(a,b) = (0.0, 0.25)$ and $(a,b) = (0.1, 0.9)$ are employed for l_1 and s_1, respectively. We employ normal distribution $\mathcal{N}(\mu,\sigma)$ for both Δl_0 and Δl_2. $(\mu,\sigma) = (0.05, 0.025)$ and $(\mu,\sigma) = (-0.05, 0.025)$ are employed for Δl_0 and Δl_2 for R and B channel, respectively.

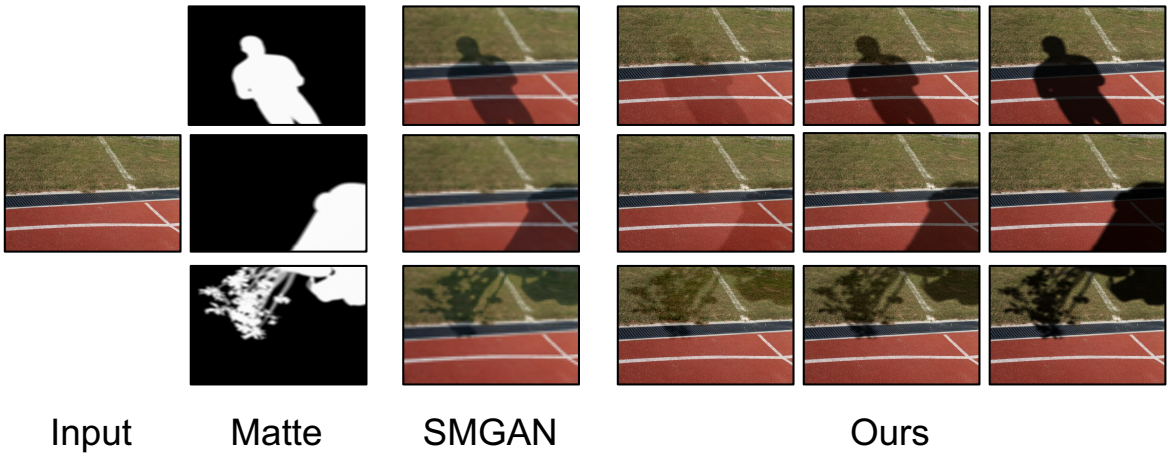


Figure 4.4: Difference between Shadow Matting GAN (SMGAN) [7] and our shadow composition model. Note that SMGAN takes a binary mask input.

Finally, $\boldsymbol{x}^{dark}$ is computed as follows:

$$x_{ijk}^{dark} = \begin{cases} \frac{s_1}{1.0-l_1}(x_{ijk}^{ns} - l_k) & \text{if } x_{ijk}^{ns} - l_k \geq 0.0, \\ 0 & \text{if } x_{ijk}^{ns} - l_k < 0.0. \end{cases} \tag{4.10}$$

If the slope of the shadow attenuation $\frac{s_1}{1.0-l_1}$ is larger than one, it is unnatural. In that case we re-sample $(l_1, s_1, \Delta l_0, \Delta l_2)$ to stabilize the quality of the sampled shadow images.

We show the result of our shadow composition in Fig. 4.4. While SMGAN only generates a single shadow image given an input image and a matte, our composition model is able to create various shadow images. We show the comparison of the datasets for shadow removal in Fig. 4.5. Although the composited shadow may not fully match the background geometry, we can obtain realistic and diverse shadows.

4.3.3 Shadow Matte Generation

We obtain a shadow matte $\boldsymbol{m}$, where $m_{ij} = 1$ if the pixel is inside the umbra, $0 \leq m_{ij} \leq 1$ if the pixel is in the penumbra, and $m_{ij} = 0$ otherwise. We put a light, occluder objects, a camera, and a virtual plane in Blender[1] as shown in Fig. 4.6. We capture the virtual plane through the camera, obtain the amount of light reaching each point, and normalize the amount to obtain the shadow matte. Note that all the occluder objects are assumed to be outside the camera view, and thus they are not apparent

[1] https://www.blender.org/

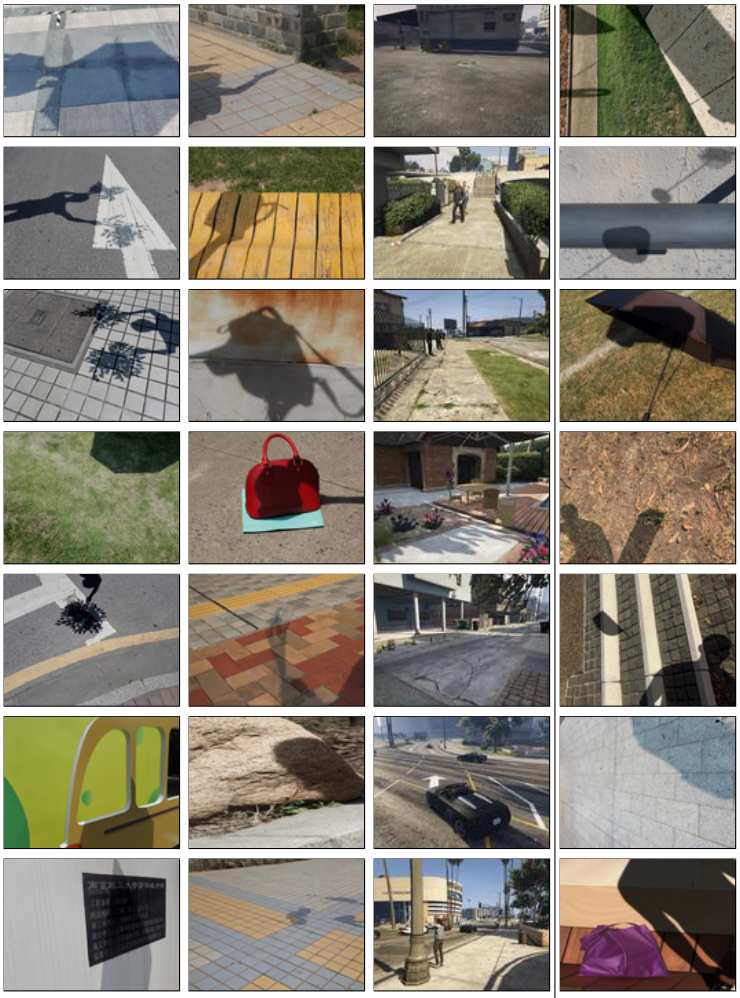


Figure 4.5: Comparison of the datasets for shadow removal. Shadows in GTAV are mostly caused by occluder objects inside the camera, while shadows in ISTD, SRD, and SynShadow are caused by those outside the camera.

in the final shadow/shadow-free/matte images. Therefore, we need the geometric relation of the light, the occluder, the camera, and the virtual plane. We randomly scale, translate, and rotate each component to generate a diverse shadow matte. We sample occluder and light as follows.

Occluder: We use AMASS [166] and ShapeNet [167], which are collections of publicly available 3D models as the occluder objects. We ignore the material or other information of the models and only use their geometric information. AMASS contains a large number of captured human 3D mesh sequences. We randomly sample 3D meshes from AMASS to obtain 3D human meshes. For common objects, we randomly sample 3D meshes from 26 object categories that are often seen in

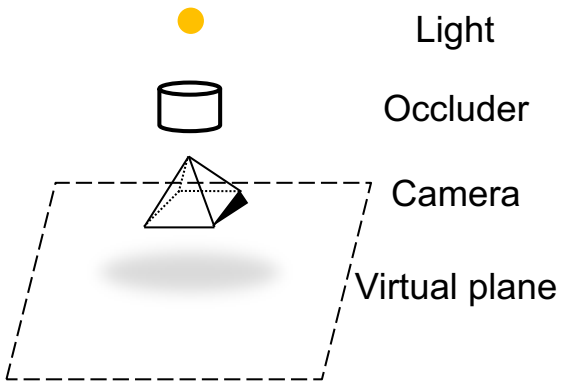


Figure 4.6: Overview of the occluder projection.

outdoor scenes from ShapeNet. In a single capture, we sample up to two models, where each model is either a human or an ordinary object.

Light: By changing the radius of the spherical light in Blender, a different penumbra can be obtained in the same geometrical configuration. We randomly sample various radii to obtain a diverse variety of penumbras for the same scene. Note that there is no need to randomize the brightness of the light. We will randomize this parameter separately later using the shadow illumination model.

4.4 Experiments on Shadow Removal

4.4.1 Datasets, Models, and Evaluation Metrics

We evaluate the potential of SynShadow comparing with other datasets by using several state-of-the-art algorithms. Note that we are not proposing novel techniques for shadow removal but a new dataset.

Datasets

We employed three datasets, ISTD+, SRD+, and USR, for evaluating shadow removal performance.

ISTD+: ISTD+ [8] consists of shadow/shadow-free/mask image triplets. Note that the mask is represented in a binary format. It has 1,330 and 540 triplets for training and testing, respectively. It is a color-adjusted version of ISTD [135] to cope with a color inconsistency issue between the shadow images and their corresponding shadow-free images caused by the triplet collection process. Although all the prior

works used the original ISTD, [8] showed that ISTD is not appropriate for evaluating shadow removal methods fairly.

SRD+: SRD+ has 2,675 and 406 shadow/shadow-free/mask image triplets for training and testing, respectively. SRD+ is based on SRD [9]. We found that the original train-test split of SRD is inappropriate since images coming from the identical background are both in the training and testing sets, as shown in Table B.1. Therefore, we re-split SRD so that there is no overlap of scenes in the two sets and removed near-duplicate images. SRD initially consists of shadow/shadow-free image pairs. Mask for shadow region is extracted following [7].

USR: USR [10] consists of unpaired 2,445 shadow images and 1,770 shadow-free images. It is more challenging than ISTD+/SRD+ because it contains diverse shadows. It consists of thousands of different diverse scenes, while SRD and ISTD are only comprised of hundreds of scenes. The shadow images are split into 1,956 and 489 images for training and testing, respectively.

We also describe the detailed configuration of SynShadow.

SynShadow: We generate 10,000 shadow matte images to generate SynShadow as described in Sec. 4.3. The shadow-free images are obtained from USR.

Models

We employed three approaches.

Supervised learning: Supervised learning models are trained on paired shadow/shadow-free images such as ISTD+, SRD+, and SynShadow. We mainly employed the best recent models, decomposition-based SP+M [8] and regression-based DHAN [7]. We also employed regression-based DSC. For a fair comparison, when we train models on SynShadow, training details such as learning schedule and hyper-parameters are similar to those used for training on ISTD+/SRD+. Note that the codes of ST-CGAN [9], DeshadowNet [135], and ARGAN [146] are not publicly available, and we could not evaluate their performance on ISTD+/SRD+ unfortunately.

Unsupervised learning: Unsupervised learning models are trained on unpaired shadow/shadow-free images. Following [10], we tested MSGAN [10] and CycleGAN [14]. We also trained these models on SRD+/ISTD+ in an unpaired manner for a more fair comparison with supervised learning models.

Traditional methods: We tried Guo *et al*. [132] and Gong *et al*. [132]. Note that Gong *et al*. [11] is an interactive method that requires the user's manual input.

Table 4.1: RMSE comparison with the state-of-the-art methods in LAB color space. Gong *et al*. [11]* is an interactive method. I/S is short for ISTD+ *or* SRD+, so that the dataset for training and evaluation is similar. SS is short for SynShadow.

Tested on		ISTD+			SRD+		
Metrics		S	NS	ALL	S	NS	ALL
Methods	Trained on						
(a) *Traditional*							
Guo *et al*. [132]	-	22.3	4.3	7.1	24.3	6.5	10.3
Gong *et al*. [11]*	-	14.4	**3.4**	5.1	18.5	3.9	7.0
(b) *Unsupervised*							
MSGAN [10]	USR	24.7	6.9	9.9	30.3	11.0	15.1
CycleGAN [14]	USR	25.6	7.4	10.2	29.9	11.5	15.4
MSGAN [10]	I/S	14.1	7.6	8.6	16.5	7.6	9.5
CycleGAN [14]	I/S	14.3	7.9	8.9	17.5	7.5	9.6
(c) *Supervised*							
SP+M [8]	SS	11.3	3.6	4.9	11.6	4.1	5.7
DHAN [7]	SS	9.7	4.0	4.9	13.8	5.1	6.9
SP+M [8]	I/S	8.5	3.6	4.4	12.2	**3.4**	5.3
DHAN [7]	I/S	7.4	4.8	5.2	12.7	5.9	7.4
DSC [136]	I/S	8.3	4.6	5.2	16.2	5.8	8.0
(d) *Supervised, pre-trained on SynShadow*							
SP+M [8]	I/S	**6.9**	**3.4**	**4.0**	**10.9**	3.6	**5.2**
DHAN [7]	I/S	6.6	4.2	4.6	10.6	5.6	6.6

Evaluation Metrics

We followed all the prior works on shadow removal in quantitative evaluation. We used root-mean-square error (RMSE) in LAB color space between the ground truth and predicted shadow-free images. RMSE is reported for all pixels (ALL). Additionally, RMSE is reported for only shadow pixels (S) and only non-shadow pixels (NS). Smaller RMSE indicates better performance.

4.4.2 Experiments on ISTD+/SRD+ Datasets

We demonstrate how we can use SynShadow to improve the supervised shadow removal models on the existing shadow removal datasets, ISTD+ and SRD+. We considered two scenarios for using SynShadow:

Figure 4.7: Qualitative comparison of shadow removal models. Results in top and bottom two rows are from models trained and evaluated on ISTD+ and SRD+, respectively. DHAN-ft and SP+M-ft indicates DHAN and SP+M pre-trained on SynShadow and later fine-tuned on each dataset.

Zero-shot: We trained the models on SynShadow and evaluated them on ISTD+/SRD+ test set.

Fine-tuning: We pre-trained the models on SynShadow, fine-tuned them on ISTD+/SRD+ train set, and evaluated them on ISTD+/SRD+ test set.

Quantitative Evaluation

The results are shown in Table 4.1. We discuss the results in each scenario.

Zero-shot: Models trained on SynShadow (the first two rows in group c) outperformed unsupervised learning (group b) and traditional methods (group a) that do not use the ISTD+/SRD+ dataset for training. Furthermore, the model trained on SynShadow is almost compatible with most of the supervised learning models (the last three rows in group c) trained and evaluated on a similar dataset.

Fine-tuning: When we fine-tuned SynShadow-pre-trained models (group d) for each specific dataset, we constantly obtained the best results.

Qualitative Evaluation

In Fig. 4.7, we show the visual comparison of some supervised learning models discussed in Table 4.1. The fine-tuned model's improvement is attributed to not

Table 4.2: Comparison by changing the training/fine-tuning dataset in shadow removal. Top two results in each setting are highlighted in **red** and blue, respectively. SS is short for SynShadow.

Tested on	ISTD+							SRD+						
Model	SP+M			DHAN				SP+M			DHAN			
Metrics	S	NS	ALL	S	NS	ALL		S	NS	ALL	S	NS	ALL	
Training														
GTAV	26.9	4.5	8.0	28.1	10.9	13.6	GTAV	27.8	4.1	9.1	32.5	7.2	12.6	
SRD+	14.9	3.8	5.6	12.2	6.3	7.2	SRD+	12.2	**3.4**	**5.3**	**12.7**	5.9	7.4	
SS	11.3	**3.6**	4.9	9.7	**4.0**	**4.9**	SS	**11.6**	4.1	5.7	13.8	**5.1**	**6.9**	
ISTD+	**8.5**	**3.6**	**4.4**	**7.4**	4.8	5.2	ISTD+	14.9	4.5	6.7	16.6	8.1	9.9	
Fine-tuning														
GTAV→ISTD+	7.8	3.6	4.2	7.2	**4.2**	4.7	GTAV→SRD+	11.9	3.6	5.4	13.0	5.6	7.2	
SRD+→ISTD+	7.6	3.5	4.1	7.2	4.7	5.1	ISTD+→SRD+	11.9	**3.4**	**5.2**	11.8	**5.5**	6.8	
SS→ISTD+	**6.9**	**3.4**	**4.0**	**6.6**	**4.2**	**4.6**	SS→SRD+	**10.9**	3.6	**5.2**	**10.6**	5.6	**6.6**	

modifying non-shadow areas (the 1st and 2nd rows) and improved relit estimation (the 3rd and 4th rows).

Transferability of Shadow Removal Datasets

We performed more detailed analysis on transferability of many shadow removal datasets listed in Fig. 4.5. We again consider both zero-shot and fine-tuning setting in Table 4.2.

Zero-shot: The result is shown in the upper half of Table 4.2. When the domain of datasets for training and evaluation is different, the models trained on SynShadow perform best. Surprisingly, the DHAN model trained on SynShadow even performs better than those trained on similar domain of datasets for training and evaluation. We conjecture that regression-based models may overfit to a small dataset.

Fine-tuning: The result is shown in the lower half of Table 4.2. We can clearly see the advantage of pre-training on SynShadow, compared to the other datasets.

Ablation Study

We perform ablation study on how to generate shadow images.

Table 4.3: Ablation study on design of randomizing parameters in the shadow illumination models. SP+M [8] is trained on each variant for quantitative evaluation. Top two results in each setting are highlighted in **red** and blue, respectively.

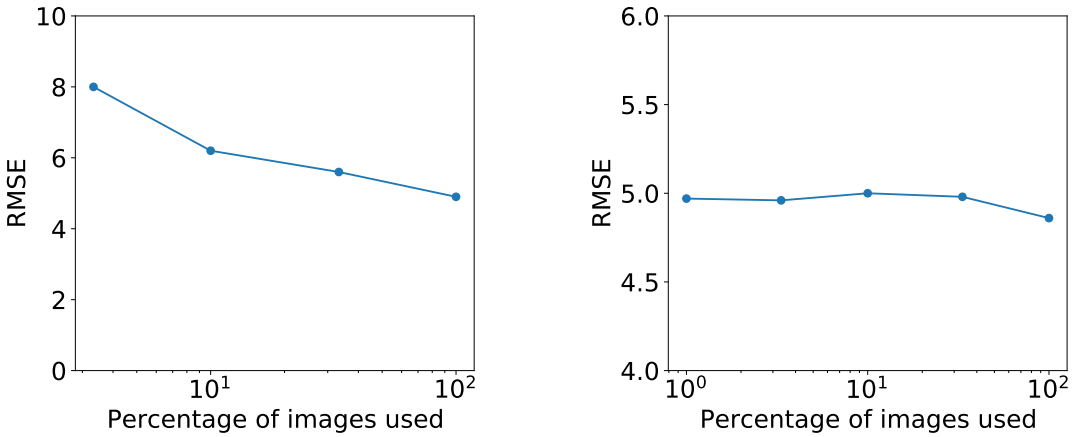
Tested on	ISTD+			SRD+		
Metrics	S	NS	ALL	S	NS	ALL
Color jitter	39.3	4.6	10.1	44.0	4.6	13.0
Color jitter (dark)	**10.8**	4.7	5.6	**11.4**	4.7	6.1
Gamma correction	23.3	3.9	6.9	33.6	4.1	10.3
Independent	13.4	4.0	5.4	12.6	4.2	6.0
Zero intercepts	14.9	3.7	5.4	13.2	4.2	6.1
Similar intercepts	23.3	**3.5**	6.7	18.9	**4.0**	7.1
Non-biased intercepts	12.1	5.5	6.6	12.8	5.9	7.3
Proposed	11.3	3.6	**4.9**	11.6	4.1	**5.7**

Shadow Model We tested various settings on how to generate the affine shadow illumination model parameters (l_0, l_1, l_2, s_1). Five variants including the proposed method are tested:

- Independent params: (l_0, l_1, l_2, s_1), are randomized independently. Note that the range of each parameter is similar to our setting.
- Zero intercepts: l_1 is fixed at 0.0. Note that l_0 and l_2 are still randomized with respect to l_1.
- Similar intercepts: The relation between (l_0, l_1, l_2) is $l_0 = l_1 = l_2$. In this case, the shadow attenuation is assumed to be similar in RGB channels.
- Non-biased intercepts: We set $\mu = 0.0$ in randomizing parameters.
- Proposed: This is based on our proposal, and we did not try any of the above modifications.

We also tested some possible approaches for generating shadow images directly by simple low-level image filtering operations. In all the cases, we used the same shadow-free and matte images as those used for obtaining SynShadow.

- Color jitter: We tried color jitter, which is often used for image augmentation, for each RGB channel to make the shadow images. In this case, attenuation in each channel is assumed to be independent of each other. We followed a paper for portrait shadow manipulation [142], where color jitter is formulated as a 3×3 color correction matrix.



(a) Shadow-free images (b) Matte images

Figure 4.8: Ablation study on changing the number of images used for shadow-free and matte images. The result of the SP+M model trained on ISTD+ is reported.

- Color jitter (dark): We found that the original implementation of [142] sometimes makes the shadowed region brighter. We limited the range of color jitter so that the shaded region is always darker for a fair comparison.

- Gamma correction: We tried gamma correction to make the shadow images. We compute $x_{ijk}^{dark} = (x_{ijk}^{ns})^{\gamma}$, where $1.5 \leq \gamma \leq 3.0$ is randomly sampled to produce diverse shadows.

Comparison among all the variants is shown in Table 4.3. The proposed methods performed much better than the other variants in both datasets. This result stresses the importance of carefully setting the range in our proposed randomization of parameters and selecting the proper shadow model.

Number of shadow-free and matte images We perform additional ablation study on changing the number of shadow-free images and matte images in Fig. 4.8. The number of shadow-free and matte images used in SynShadow is 1,770 and 10,000, respectively. We randomly sample only a fraction of the datasets and trained the SP+M model on the ISTD+ dataset. We observe that changing the number of shadow-free images affects the performance mostly while changing those of matte images affect RMSE slightly. Collecting more shadow-free images would further enhance the utility of SynShadow.

Table 4.4: Comparison of models trained on different datasets. User study results on the USR testing set is reported.

(a) Model: SP+M [8]

Trained on	Rating
ISTD+	2.00 ± 1.02
SRD+	1.52 ± 0.83
SynShadow	**3.00** ± 1.40

(b) Model: DHAN [7]

Trained on	Rating
ISTD+	2.52 ± 1.18
SM-ISTD+	1.68 ± 0.96
SRD+	2.67 ± 1.17
SM-SRD+	2.43 ± 1.16
SynShadow	**3.07** ± 1.19

Table 4.5: Comparison with unsupervised learning and traditional approaches. User study results on the USR testing set is reported. Gong *et al*. [11]* is an interactive method.

Method	Rating
Guo *et al*. [132]	1.71 ± 1.21
Gong *et al*. [11]*	2.61 ± 1.24
MSGAN [10]	2.37 ± 1.37
DHAN (SynShadow)	**3.25** ± 1.2
SP+M (SynShadow)	3.00 ± 1.45

4.4.3 Experiments on USR

User Study

We performed a user study since there is no paired supervision for quantitative evaluation in the USR dataset. The models trained on SynShadow are compared with the best approaches from supervised/unsupervised/traditional methods. 30 shadow images are randomly chosen from the testing set. We displayed results from a single input on a webpage in random order at the same time. Unlike the user study in the MSGAN paper [10], we showed the original input as a reference so that the evaluators can notice the difference between the results and the input. We asked Amazon Mechanical Turk workers to rate each result on a scale from 1 (bad) to 5 (excellent). 300 votes are obtained from 10 users, and we report the mean and standard deviation of the ratings in Table 4.4. Since comparing results from an arbitrary combination of methods and datasets at once is very difficult, we performed several experiments.

Table 4.6: Comparison with shadow augmentation approach proposed in [8]. User study results on the USR testing set is reported.

Trained on	Rating
ISTD+	1.94 ± 1.05
Augmented ISTD+	2.12 ± 1.05
SynShadow (BG-ISTD+)	**2.81** ± 1.29

Fixed supervised learning model, different datasets: We fixed the model and changed the dataset for supervised learning to demonstrate the significance of SynShadow. First, we show the result using the SP+M model in Table 4.4a. The result of the model trained on SynShadow obtained much better ratings compared to those trained on the other datasets. Second, we show the result using the DHAN model in Table 4.4b. For comparison with an existing shadow composition method, SMGAN [7], we report the results using additional datasets. Following [7], we combined images generated by SMGAN with the ISTD+ and SRD+ datasets, and obtained SM-ISTD+ and SM-SRD+, respectively. Although SM-ISTD+, SM-SRD+, and SynShadow internally use USR shadow-free images, the model trained on SynShadow performed the best.

Comparison with unsupervised learning or traditional approaches: SP+M and DHAN models trained on SynShadow are compared with traditional approaches and unsupervised learning approaches in Table 4.5. Both methods obtained much better ratings than the compared approaches. Fig. 4.9 shows the visual comparisons. The previous best approach for the USR dataset, MSGAN [10], tends to fail to focus only on the shadow region, resulting in drastic unwanted changes in textures in the shadow-free areas. Therefore, supervised learning on the diverse data, even though they are synthetic, is essential for robust shadow removal.

Comparison with shadow augmentation: In Table 4.6, we compare our approach with the shadow augmentation approach [8] using the SP+M model. Specifically, we consider the following settings in addition to the original ISTD+ dataset:

- Augmented ISTD+: Following [8], we augmented existing shadow/shadow-free/mask triplets in the original ISTD+ dataset. Please refer to the third paragraph in Sec. 4.2.2 for details.

- SynShadow (BG-ISTD+): For unbiased evaluation, we used shadow-free images in the original ISTD+ dataset as the background for obtaining SynShadow.

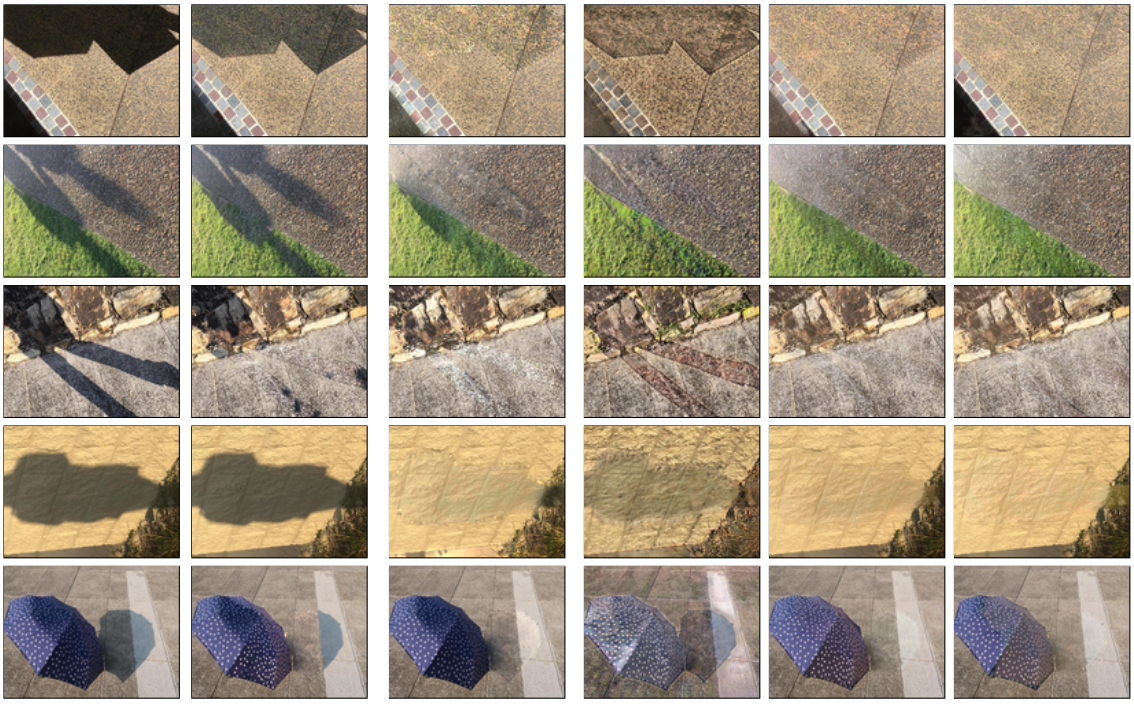


Figure 4.9: Qualitative comparison of shadow removal methods on USR test set. SS stands for SynShadow.

Although [8]'s augmentation contributed to the improved rating, the model based on SynShadow obtained a much better rating, even when the shadow-free images are from the ISTD+ dataset. This result further highlights the importance of adequately randomizing the shadow illumination model as we propose, compared to [8].

Ablation Study on Components of Shadow Synthesis Framework

For a more detailed analysis of how each randomized component of our shadow synthesis framework contributes to the improved removal result in the USR dataset, we compared the removal results by changing two factors of our pipeline:

- Shadow shape: We tested shadow masks from the ISTD+ and SRD+ datasets, and shadow masks/matte from rendered 3D models.
- Composition: We tested the shadow illumination model and SMGAN [7] for comparison.

As shown in Fig. 4.10, choosing the shadow illumination model instead of SMGAN significantly boosts the shadow removal results. This result also suggests the importance of the proposed illumination randomization.

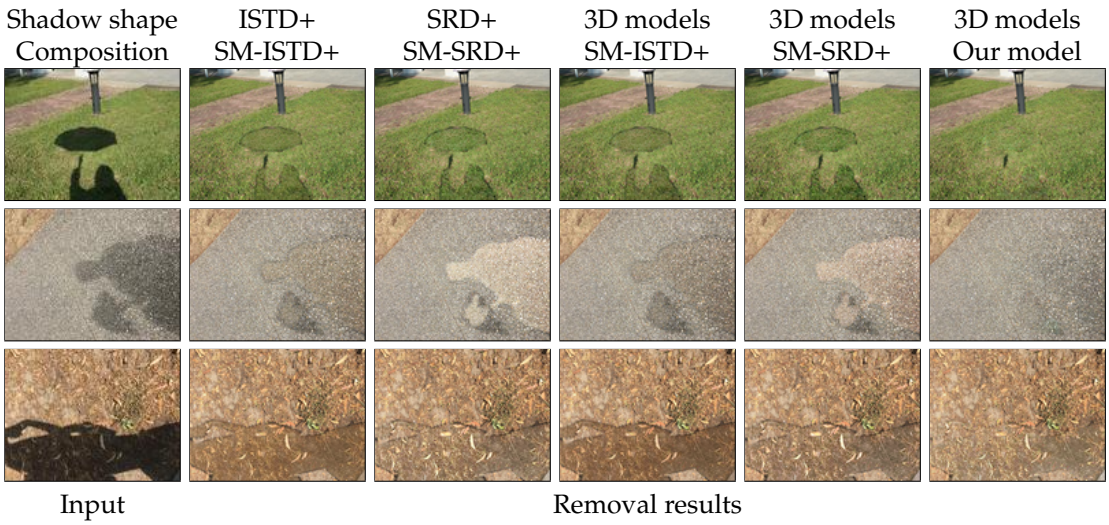


Figure 4.10: Ablation study on the choice of shadow composition model and source of shadow shape evaluated on USR test set. As a shadow removal model, we used SP+M [8]. SM-ISTD+ indicates SMGAN [7] trained on ISTD+ dataset.

4.5 Experiments on Shadow Detection

4.5.1 Evaluation Metrics

Following the prior works, we used the balance error rate (BER) [134] for quantitative evaluation.

$$BER = 1 - 0.5 \times (\frac{N_{tp}}{N_p} + \frac{N_{tn}}{N_n}), \tag{4.11}$$

where N_{tp}, N_{tn}, N_p, and N_n indicate the numbers of true positives, true negatives, shadow pixels, and non-shadow pixels, respectively. A lower score indicates better performance. BER is reported for all pixels (ALL). Additionally, BER is reported for only shadow pixels (S) and only non-shadow pixels (NS).

4.5.2 Models and Datasets

For models, we used two models, DSDNet++ and BDRAR [161]. We modified DSDNet [162], and only use weighted binary cross-entropy loss for training, and did not use Distraction-aware Shadow (DS) loss proposed in the original DSDNet. This is because we observed that it harms BER in our fine-tuning setting. We call this variant DSDNet++.

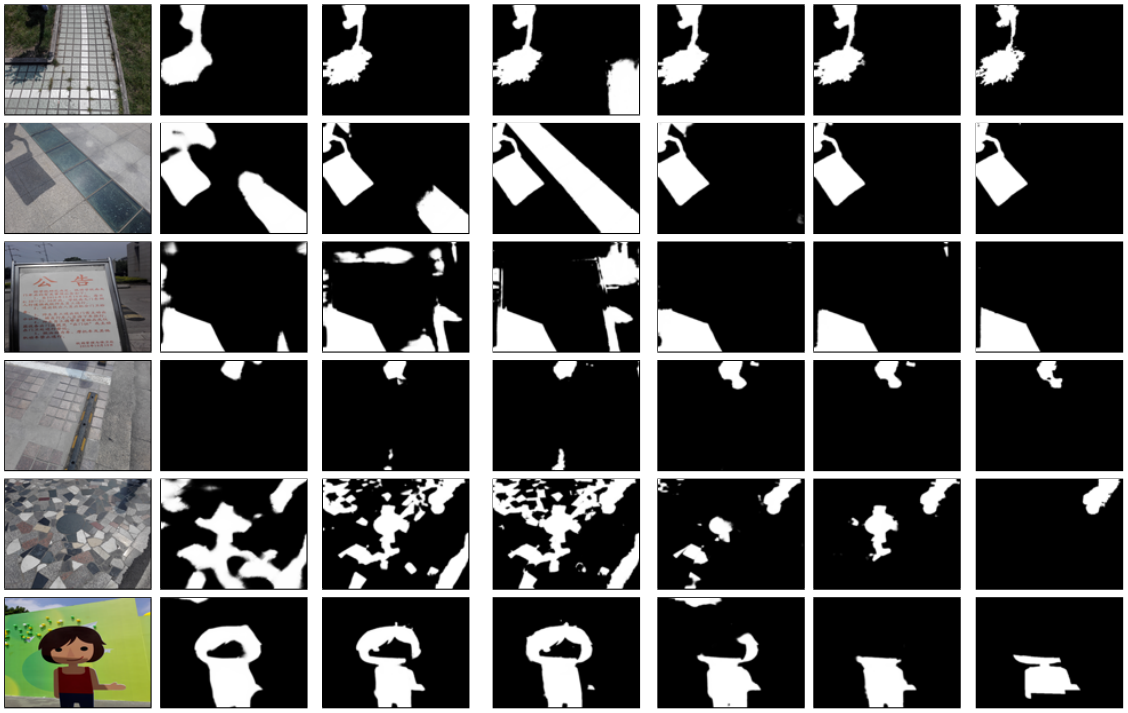


Figure 4.11: Comparison of shadow detection models evaluated on ISTD test set. BDRAR-ft and DSDNet†-ft denotes BDRAR and DSDNet† trained on SynShadow and fine-tuned on ISTD train set, respectively.

For datasets, we use the ISTD testing set for the evaluation. For the training/fine-tuning, we additionally consider the SBU dataset [159] consisting of 4085 and 638 images for training and evaluation, respectively.

4.5.3 Experiments on ISTD Dataset

We first trained shadow detection models on SynShadow and then fine-tuned them on the ISTD dataset. We show some example shadow detection results in Fig. 4.11. We observe that models trained on SynShadow and fine-tuned on the ISTD dataset can better distinguish correct shadow regions and challenging fake shadow regions such as tiles and dark areas by only employing pre-training and fine-tuning strategies. The quantitative evaluation results are shown in Table 4.7. The fine-tuned results clearly outperform the compared results, having almost 50% lower BER scores than the current best model, DSDNet. Most of the improvement is attributed to non-shadow regions.

Table 4.7: Quantitative shadow detection results evaluated on ISTD test set. Top two results in each setting are highlighted in **red** and blue, respectively.

Method	S	NS	ALL
Supervised			
Stacked-CNN [159]	7.96	9.23	8.60
scGAN [160]	3.22	6.18	4.70
ST-CGAN [135]	2.14	5.55	3.85
DSC [136]	3.85	3.00	3.42
BDRAR [161]	**0.50**	4.87	2.69
DSDNet [162]	1.36	2.98	2.17
Supervised, pre-trained on SynShadow			
BDRAR	0.62	1.57	1.10
DSDNet†	1.13	**1.04**	**1.09**

For more detailed analysis, we performed a comparison by changing the dataset for training and fine-tuning in Table 4.8. The model trained on SynShadow (2.74 BER in the ISTD dataset) is already comparable to most of the state-of-the-art models except the DSDNet model in Table 4.7. We observe that SynShadow provides a better pre-trained model than GTAV [147], SRD+, and SBU for fine-tuning on ISTD by comparing the fourth and fifth rows in each table. This result supports our argument that SynShadow provides complementary information compared to the existing datasets.

We took the same procedure for the SBU dataset. However, there was no performance improvement in BER. We conjecture that this is due to the nature of the datasets. Most of the shadows in the SBU dataset are caused by occluder objects visible in the camera view. In contrast, many shadows in the ISTD dataset are caused by occluder objects invisible from the camera, which we also assume.

4.6 Limitations and Discussion

We discuss some of the limitations in SynShadow below.

Objects inside the view: It cannot handle the shadow cast by objects inside the view. However, we would like to argue that there are many applications where we want to remove shadows cast by objects outside the view, such as document and portrait shadow removal, as we discussed in Sec. 4.2.1.

Table 4.8: Comparison by changing the dataset for training and fine-tuning of shadow detection models. Evaluation is performed on ISTD test set. Top two results in each setting are highlighted in **red** and blue, respectively. SS is short for SynShadow.

Model	DSDNet†			BDRAR		
Metrics	S	NS	ALL	S	NS	ALL
Training						
GTA	15.83	19.37	17.60	12.83	46.67	29.75
SRD+	10.68	2.36	6.52	4.63	8.64	6.64
SBU	7.19	**2.14**	4.66	7.75	**2.10**	4.92
SS	1.37	4.10	2.74	2.32	3.17	2.74
ISTD	**1.07**	3.01	**2.04**	**0.50**	4.87	**2.69**
Fine-tuning						
GTA→ISTD	1.24	1.98	1.61	0.74	2.89	1.81
SRD+→ISTD	**0.74**	2.89	1.81	0.41	3.85	2.13
SBU→ISTD	1.02	2.07	1.55	**0.64**	2.55	1.59
SS→ISTD	1.13	**1.04**	**1.09**	0.62	**1.57**	**1.10**

Uneven surface: It assumes flat surfaces thus cannot explicitly cast shadows on uneven surfaces. Viewers can sometimes find that the synthesized images look unnatural. However, we can see some improved shadow removal results on uneven surfaces in Fig. 4.7. Thus, shadow removal models can benefit from the proposed pipeline due to diverse shadows.

Multiple lights: It is not straightforward to extend the shadow illumination model to work in environments where there are multiple primary lights, which can occur only in complex indoor scenes. However, we believe that shadow removal is often required for avoiding strong shadow effects, which is usually caused by a single primary light like the sun or the brightest light in indoor scenes.

Chapter 5

Conclusions

To cope with the data scarcity problem in supervised I2IT, we had a closer look on the image formation process in this thesis. We proposed physically or artistically motivated NN modules and data synthesis pipeline for more efficient and robust optimization of I2IT models. We tackled three topics in I2IT: from photo to line-drawing, from photo to ambient occlusion, from shadow to shadow-free images.

In Chapter 2, we presented the new approach for automatic photograph tracing using neural networks. Given a photograph with an arbitrary size, our model produces the clean and expressive line drawing images automatically in a raster format. Our model which consists of two encoder-decoder based CNN is end-to-end trainable, able to learn from noisy annotations, and very fast (e.g., 1.57s for a 4096×3072 RGB image) on a GPU. We showed that our model can produce more favorable results than comparable approaches. We also demonstrated the potential applications of our model in collaboration with artists to make manga BG images and coloring books.

In Chapter 3, we presented RGB2AO, a novel task that generates AO from a single RGB image with arbitrary size. Our model for RGB2AO is a fully convolutional CNN specially designed for this task by extending an I2IT model in two points: data augmentation specific for AO and joint inference of 3D-geometry information by multi-task learning. We showed that our model can generate AO adequately. We also applied the generated AO to two image modification tasks, contrast enhancement and 2D image composition, and showed remarkable results that would not be possible without our AO-based modification.

In Chapter 4, we presented SynShadow, a large-scale synthetic dataset of shadow/shadow-free/matte image triplets, by integrating the shadow illumination model, 3D models, and the shadow-free image collections. This was enabled by

introducing some assumptions such as occluder objects being outside the camera view and a flat surface for the shadow projection. We showed that SynShadow is very useful. SynShadow-trained shadow removal models outperformed existing approaches by achieving the best rating in a user study on the challenging USR dataset. Fine-tuning SynShadow-pre-trained models achieved up to about 50% BER reduction on the ISTD dataset in shadow detection, and up to about 10% RMSE reductions on the ISTD+ and SRD+ datasets in shadow removal, compared to the best-performing approaches. We hope our proposed shadow synthesis pipeline paves the way for future work to detect and remove diverse and challenging shadows, benefitting from the growing number of 3D models and shadow-free image collections.

Future Directions

In visual content creation, there are many interesting topics for future research. We focus on some trends, list advantages and disadvantages, and discuss some directions.

Unconditional Generative Models Owing to the recent development of GAN, photorealistic, diverse, and high-quality images can now be generated from randomly sampled latent codes such as StyleGAN variants [168, 169]. Recent works use these models as fixed pre-trained models, and utilize them for image editing. These works find ways to embed images into the latent codes of unconditional GAN models [170, 171], and find editing vector in the latent codes space to alter specific attributes [172, 173]. The advantage of these works is versatility; modifying any attribute can be achieved in principle, unlike conditional GANs trained to change the only specific type of attributes [44, 45]. However, the edited results often contain unnecessary changes due to the entanglement of multiple attributes in latent codes, and large-scale datasets are required. In addition to disentanglement, how we can benefit from these approaches (i) for data-scarce domains such as comics and (ii) for various types of user interaction beyond simple labels such as scribble inputs would be an interesting research question.

Generation in More Structured Format Although 2D raster image generation has been well studied, generation in other formats is less explored. Vector graphics are widely used in digital graphics since its scalability and editability. Little effort

has been paid for the generation of vector graphics using NNs, since they usually have complex structures. Recently some works have been emerging. For example, logo images in well-known SVG format are generated by sequence generation using transformers [174] in DeepSVG [175]. Some works introduce a differentiable renderer for vector graphics to use raster-based loss functions and iterative optimization procedures [176, 177]. However, current approaches cannot generate complex topology from scratch, or adding/removing elements during optimization.

Efficient Models for Deployment Enormous number of research papers have been published recently and the results are undoubtedly promissing. However, some of them are still far from deployment for real products due to its high computational costs. A model for a single task can be compressed using various techniques used for classification such as quantization and distillation as shown in [178, 179]. Still, packing dozens of tasks into a single unified model is required to deploy many tasks efficiently.

References

[1] K. Sasaki, S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Learning to Restore Deteriorated Line Drawing," *The Visual Computer*, vol. 34, no. 6-8, pp. 1077–1085, 2018.

[2] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M.-H. Yang, "Universal style transfer via feature transforms," in *Proc. NeurIPS*, 2017, pp. 386–396.

[3] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE TPAMI*, vol. 33, no. 5, pp. 898–916, 2011.

[4] M. Li, Z. Lin, R. Mech, E. Yumer, and D. Ramanan, "Photo-sketching: Inferring contour drawings from images," in *Proc. WACV*, 2019.

[5] J.-D. Favreau, F. Lafarge, and A. Bousseau, "Fidelity vs. simplicity: a global approach to line drawing vectorization," *ACM TOG*, vol. 35, no. 4, p. 120, 2016.

[6] Y.-H. Tsai, X. Shen, Z. Lin, K. Sunkavalli, X. Lu, and M.-H. Yang, "Deep image harmonization," in *Proc. CVPR*, 2017.

[7] C. Xiaodong, P. Chi-Man, and S. Cheng, "Towards ghost-free shadow removal via dual hierarchical aggregation network and shadow matting gan," in *Proc. AAAI*, 2020.

[8] H. Le and D. Samaras, "Shadow removal via shadow image decomposition," in *Proc. ICCV*, 2019.

[9] L. Qu, J. Tian, S. He, Y. Tang, and R. W. Lau, "Deshadownet: A multi-context embedding deep network for shadow removal," in *Proc. CVPR*, 2017.

[10] X. Hu, Y. Jiang, C.-W. Fu, and P.-A. Heng, "Mask-shadowgan: Learning to remove shadows from unpaired data," in *Proc. ICCV*, 2019.

[11] H. Gong and D. Cosker, "Interactive shadow removal and ground truth for variable scene categories," in *Proc. BMVC*, 2014.

[12] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. CVPR*, 2017.

[13] Q. Chen, J. Xu, and V. Koltun, "Fast image processing with fully-convolutional networks," in *Proc. ICCV*, 2017.

[14] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. ICCV*, 2017.

[15] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep cnn denoiser prior for image restoration," in *Proc. CVPR*, 2017.

[16] S. Lefkimmiatis, "Universal denoising networks: a novel cnn architecture for image denoising," in *Proc. CVPR*, 2018.

[17] L. Xu, J. S. Ren, C. Liu, and J. Jia, "Deep convolutional neural network for image deconvolution," *Proc. NeurIPS*, 2014.

[18] J. Sun, W. Cao, Z. Xu, and J. Ponce, "Learning a convolutional neural network for non-uniform motion blur removal," in *Proc. CVPR*, 2015.

[19] S. Nah, T. Hyun Kim, and K. Mu Lee, "Deep multi-scale convolutional neural network for dynamic scene deblurring," in *Proc. CVPR*, 2017.

[20] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE TPAMI*, vol. 38, no. 2, pp. 295–307, 2015.

[21] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. CVPR*, 2017.

[22] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proc. CVPR*, 2016.

[23] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Globally and locally consistent image completion," *ACM TOG*, vol. 36, no. 4, pp. 1–14, 2017.

[24] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpainting with contextual attention," in *Proc. CVPR*, 2018.

[25] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Free-form image inpainting with gated convolution," in *Proc. ICCV*, 2019.

[26] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Let there be color! joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification," *ACM TOG*, vol. 35, no. 4, pp. 1–11, 2016.

[27] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Proc. ECCV*, 2016.

[28] R. Zhang, J.-Y. Zhu, P. Isola, X. Geng, A. S. Lin, T. Yu, and A. A. Efros, "Real-time user-guided image colorization with learned deep priors," *ACM TOG*, vol. 36, no. 4, pp. 1–11, 2017.

[29] S. Iizuka and E. Simo-Serra, "Deepremaster: temporal source-reference attention networks for comprehensive video enhancement," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 6, pp. 1–13, 2019.

[30] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *Proc. CVPR*, 2018, pp. 8798–8807.

[31] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, "Semantic image synthesis with spatially-adaptive normalization," in *Proc. CVPR*, 2019.

[32] P. Sangkloy, J. Lu, C. Fang, F. Yu, and J. Hays, "Scribbler: Controlling deep image synthesis with sketch and color," in *Proc. CVPR*, 2017.

[33] W. Chen and J. Hays, "Sketchygan: Towards diverse and realistic sketch to image synthesis," in *Proc. CVPR*, 2018.

[34] L. Ma, X. Jia, Q. Sun, B. Schiele, T. Tuytelaars, and L. Van Gool, "Pose guided person image generation," in *Proc. NeurIPS*, 2017.

[35] L. Ma, Q. Sun, S. Georgoulis, L. Van Gool, B. Schiele, and M. Fritz, "Disentangled person image generation," in *Proc. CVPR*, 2018.

[36] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Proc. ECCV*, 2016, pp. 694–711.

[37] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proc. ICCV*, 2017.

[38] Y. Li, C. Fang, A. Hertzmann, E. Shechtman, and M.-H. Yang, "Im2pencil: Controllable pencil illustration from photographs," in *Proc. CVPR*, 2019.

[39] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Proc. NeurIPS*, 2017.

[40] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *Proc. ECCV*, 2018.

[41] M.-Y. Liu, X. Huang, A. Mallya, T. Karras, T. Aila, J. Lehtinen, and J. Kautz, "Few-shot unsupervised image-to-image translation," in *Proc. ICCV*, 2019.

[42] K. Saito, K. Saenko, and M.-Y. Liu, "Coco-funit: Few-shot unsupervised image translation with a content conditioned style encoder," 2020.

[43] T. Park, J.-Y. Zhu, O. Wang, J. Lu, E. Shechtman, A. A. Efros, and R. Zhang, "Swapping autoencoder for deep image manipulation," in *Proc. NeurIPS*, 2020.

[44] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proc. CVPR*, 2018.

[45] A. Pumarola, A. Agudo, A. M. Martinez, A. Sanfeliu, and F. Moreno-Noguer, "Ganimation: Anatomically-aware facial animation from a single image," in *Proc. ECCV*, 2018.

[46] Y. Choi, Y. Uh, J. Yoo, and J.-W. Ha, "Stargan v2: Diverse image synthesis for multiple domains," in *Proc. CVPR*, 2020.

[47] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. NeurIPS*, 2014, pp. 2672–2680.

[48] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[49] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman, "Toward multimodal image-to-image translation," in *Proc. NeurIPS*, 2017.

[50] Q. Mao, H.-Y. Lee, H.-Y. Tseng, S. Ma, and M.-H. Yang, "Mode seeking generative adversarial networks for diverse image synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1429–1437.

[51] A. Pumarola, S. Popov, F. Moreno-Noguer, and V. Ferrari, "C-flow: Conditional generative flow models for images and 3d point clouds," in *Proc. CVPR*, 2020.

[52] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, "Learning to discover cross-domain relations with generative adversarial networks," in *Proc. ICML*, 2017.

[53] H.-Y. Lee, H.-Y. Tseng, Q. Mao, J.-B. Huang, Y.-D. Lu, M. Singh, and M.-H. Yang, "Drit++: Diverse image-to-image translation via disentangled representations," *Springer IJCV*, pp. 1–16, 2020.

[54] X. Hu, T. Wang, C.-W. Fu, Y. Jiang, Q. Wang, and P.-A. Heng, "Revisiting shadow detection: A new benchmark dataset for complex world," *arXiv preprint arXiv:1911.06998*, 2019.

[55] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Santella, "Suggestive contours for conveying shape," in *ACM TOG*, vol. 22, no. 3, 2003, pp. 848–855.

[56] R. Deng, C. Shen, S. Liu, H. Wang, and X. Liu, "Learning to predict crisp boundaries," in *Proc. ECCV*, 2018.

[57] J. Canny, "A computational approach to edge detection," *IEEE TPAMI*, no. 6, pp. 679–698, 1986.

[58] D. R. Martin, C. C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues," *IEEE TPAMI*, vol. 26, no. 5, pp. 530–549, 2004.

[59] P. Dollár and C. L. Zitnick, "Structured forests for fast edge detection," in *Proc. ICCV*, 2013.

[60] S. Xie and Z. Tu, "Holistically-nested edge detection," in *Proc. ICCV*, 2015.

[61] J. Yang, B. Price, S. Cohen, H. Lee, and M.-H. Yang, "Object contour detection with a fully convolutional encoder-decoder network," in *Proc. CVPR*, 2016.

[62] P. Bénard and A. Hertzmann, "Line drawings from 3d models," *arXiv preprint arXiv:1810.01175*, 2018.

[63] F. Cole, A. Golovinskiy, A. Limpaecher, H. S. Barros, A. Finkelstein, T. Funkhouser, and S. Rusinkiewicz, "Where do people draw lines?" in *ACM TOG*, vol. 27, no. 3. ACM, 2008, p. 88.

[64] G. Noris, A. Hornung, R. W. Sumner, M. Simmons, and M. Gross, "Topology-driven vectorization of clean line drawings," *ACM TOG*, vol. 32, no. 1, p. 4, 2013.

[65] E. Simo-Serra, S. Iizuka, K. Sasaki, and H. Ishikawa, "Learning to simplify: fully convolutional networks for rough sketch cleanup," *ACM TOG*, vol. 35, no. 4, p. 121, 2016.

[66] E. Simo-Serra, S. Iizuka, and H. Ishikawa, "Mastering sketching: adversarial augmentation for structured prediction," *ACM TOG*, vol. 37, no. 1, p. 11, 2018.

[67] E. Simo-Serra, S. Iizuka, and H. Ishikawa, "Real-time data-driven interactive rough sketch inking," *ACM TOG*, vol. 37, no. 4, p. 98, 2018.

[68] Q. Liu, X. Tang, H. Jin, H. Lu, and S. Ma, "A nonlinear approach for face sketch synthesis and recognition," in *Proc. CVPR*, 2005.

[69] C. Lu, L. Xu, and J. Jia, "Combining sketch and tone for pencil drawing production," in *Proc. NPAR*, 2012.

[70] A. S. Inc, "Adobe photoshop cc," https://www.adobe.com/products/photoshop.html.

[71] V. Le, J. Brandt, Z. Lin, L. Bourdev, and T. S. Huang, "Interactive facial feature localization," in *Proc. ECCV*, 2012, pp. 679–692.

[72] NoneCG.com, "Nonecg," https://www.nonecg.com/, 2019, accessed: 2019-04-22.

[73] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2015.

[74] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. CVPR*, 2009, pp. 248–255.

[75] X. Shen, A. Hertzmann, J. Jia, S. Paris, B. Price, E. Shechtman, and I. Sachs, "Automatic portrait segmentation for image stylization," in *Computer Graphics Forum*, vol. 35, no. 2. Wiley Online Library, 2016, pp. 93–102.

[76] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, and A. Torralba, "Semantic understanding of scenes through the ade20k dataset," *Springer IJCV*, vol. 127, no. 3, pp. 302–321, 2019.

[77] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *Proc. ICML*, 2019.

[78] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Proc. CVPR*, 2016, pp. 2414–2423.

[79] K. Ito, Y. Matsui, T. Yamasaki, and K. Aizawa, "Separation of manga line drawings and screentones." in *Eurographics (Short Papers)*, 2015.

[80] Y. Qu, W.-M. Pang, T.-T. Wong, and P.-A. Heng, "Richness-preserving manga screening," *ACM TOG*, vol. 27, no. 5, p. 155, 2008.

[81] P. Meer and B. Georgescu, "Edge detection with embedded confidence," *IEEE TPAMI*, vol. 23, no. 12, pp. 1351–1365, 2001.

[82] R. L. Cook and K. E. Torrance, "A reflectance model for computer graphics," *ACM TOG*, vol. 1, no. 1, pp. 7–24, 1982.

[83] S. Zhukov, A. Iones, and G. Kronin, "An ambient light illumination model," in *Rendering Techniques' 98*, 1998, pp. 45–55.

[84] R. Fernando, *GPU Gems: Programming Techniques, Tips and Tricks for Real-Time Graphics*. Pearson Higher Education, 2004.

[85] T. Akenine-Moller, E. Haines, and N. Hoffman, *Real-Time Rendering*, 3rd ed. A. K. Peters, Ltd., 2008.

[86] L. Lanier, *Professional Digital Compositing: Essential Tools and Techniques*. Alameda, CA, USA: SYBEX Inc., 2009.

[87] N. Sam, "Sam nielson: Painting process," http://theartcenter.blogspot.com/2010/03/sam-nielson-painting-process.html, 2010.

[88] D. Holden, J. Saito, and T. Komura, "Neural network ambient occlusion." in *SIGGRAPH Asia Technical Briefs*, 2016, p. 9.

[89] L. Bavoil, M. Sainz, and R. Dimitrov, "Image-space horizon-based ambient occlusion," in *ACM SIGGRAPH 2008 talks*, 2008, p. 22.

[90] B. Marco, "Ambient occlusion (and ambient light) for painters," https://www.youtube.com/watch?v=7fLV5ezO64w&feature=youtu.be, 2018.

[91] I. Dorian, "How to make an ambient occlusion study," https://www.youtube.com/watch?v=WiC4tiOSn_M&feature=youtu.be, 2016.

[92] W. Chen, Z. Fu, D. Yang, and J. Deng, "Single-image depth perception in the wild," in *Proc. NeurIPS*, 2016.

[93] Z. Li and N. Snavely, "Megadepth: Learning single-view depth prediction from internet photos," in *Proc. CVPR*, 2018.

[94] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *IEEE TPAMI*, 2020.

[95] M. Mittring, "Finding next gen: Cryengine 2," in *ACM SIGGRAPH 2007 courses*, 2007, pp. 97–121.

[96] P. Shanmugam and O. Arikan, "Hardware accelerated ambient occlusion techniques on gpus," in *Proc. I3D*, 2007, pp. 73–80.

[97] M. McGuire, B. Osman, M. Bukowski, and P. Hennessy, "The alchemy screen-space ambient obscurance algorithm," in *Proceedings of the ACM SIGGRAPH Symposium on High Performance Graphics*, 2011, pp. 25–32.

[98] T. Ritschel, T. Grosch, and H.-P. Seidel, "Approximating dynamic global illumination in image space," in *Proc. I3D*, 2009.

[99] T. Luft, C. Colditz, and O. Deussen, "Image enhancement by unsharp masking the depth buffer," *ACM TOG*, vol. 25, no. 3, 2006.

[100] O. Nalbach, E. Arabadzhiyska, D. Mehta, H.-P. Seidel, and T. Ritschel, "Deep shading: convolutional neural networks for screen space shading," vol. 36, no. 4, pp. 65–78, 2017.

[101] E. H. Land and J. J. McCann, "Lightness and retinex theory," *Josa*, vol. 61, no. 1, pp. 1–11, 1971.

[102] S. Bell, K. Bala, and N. Snavely, "Intrinsic images in the wild," *ACM TOG*, vol. 33, no. 4, p. 159, 2014.

[103] H. Yue, J. Yang, X. Sun, F. Wu, and C. Hou, "Contrast enhancement based on intrinsic image decomposition," *IEEE TIP*, vol. 26, no. 8, pp. 3981–3994, 2017.

[104] D. Hauagge, S. Wehrwein, K. Bala, and N. Snavely, "Photometric ambient occlusion for intrinsic image decomposition," *IEEE TPAMI*, vol. 38, no. 4, pp. 639–651, 2015.

[105] J. T. Barron and J. Malik, "Shape, illumination, and reflectance from shading," *IEEE TPAMI*, vol. 37, no. 8, pp. 1670–1687, 2014.

[106] C. Innamorati, T. Ritschel, T. Weyrich, and N. J. Mitra, "Decomposing single images for layered photo retouching," *Computer Graphics Forum*, vol. 36, no. 4, pp. 15–25, 2017.

[107] E. Reinhard, M. Adhikhmin, B. Gooch, and P. Shirley, "Color transfer between images," *IEEE Computer graphics and applications*, vol. 21, no. 5, pp. 34–41, 2001.

[108] J.-F. Lalonde and A. A. Efros, "Using color compatibility for assessing image realism," in *Proc. ICCV*. IEEE, 2007.

[109] K. Sunkavalli, M. K. Johnson, W. Matusik, and H. Pfister, "Multi-scale image harmonization," *ACM TOG*, vol. 29, no. 4, p. 125, 2010.

[110] S. Xue, A. Agarwala, J. Dorsey, and H. Rushmeier, "Understanding and improving the realism of image composites," *ACM TOG*, vol. 31, no. 4, p. 84, 2012.

[111] P. Pérez, M. Gangnet, and A. Blake, "Poisson image editing," *ACM TOG*, vol. 22, no. 3, pp. 313–318, 2003.

[112] M. W. Tao, M. K. Johnson, and S. Paris, "Error-tolerant image compositing," in *Proc. ECCV*, 2010.

[113] J.-Y. Zhu, P. Krahenbuhl, E. Shechtman, and A. A. Efros, "Learning a discriminative model for the perception of realism in composite images," in *Proc. ICCV*, 2015.

[114] F. Zhan, H. Zhu, and S. Lu, "Spatial fusion gan for image synthesis," in *Proc. CVPR*, 2019.

[115] J.-F. Lalonde, A. A. Efros, and S. G. Narasimhan, "Estimating the natural illumination conditions from a single outdoor image," *Springer IJCV*, vol. 98, no. 2, pp. 123–145, 2012.

[116] Y. Hold-Geoffroy, K. Sunkavalli, S. Hadap, E. Gambaretto, and J.-F. Lalonde, "Deep outdoor illumination estimation," in *Proc. CVPR*, 2017.

[117] Y. Hold-Geoffroy, A. Athawale, and J.-F. Lalonde, "Deep sky modeling for single image outdoor lighting estimation," in *Proc. CVPR*, 2019.

[118] M.-A. Gardner, K. Sunkavalli, E. Yumer, X. Shen, E. Gambaretto, C. Gagné, and J.-F. Lalonde, "Learning to predict indoor illumination from a single image," *ACM TOG*, 2017.

[119] M. Garon, K. Sunkavalli, S. Hadap, N. Carr, and J.-F. Lalonde, "Fast spatially-varying indoor lighting estimation," in *Proc. CVPR*, 2019.

[120] A. Saxena, S. H. Chung, and A. Y. Ng, "Learning depth from single monocular images," in *Proc. NeurIPS*, 2005.

[121] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Proc. NeurIPS*, 2014.

[122] F. Liu, C. Shen, G. Lin, and I. Reid, "Learning depth from single monocular images using deep convolutional neural fields," *IEEE TPAMI*, vol. 38, no. 10, pp. 2024–2039, 2015.

[123] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *Proc. 3DV*, 2016.

[124] A. Inc, "Maya," https://www.autodesk.com/products/maya/overview, 2019, accessed: 2019-09-19.

[125] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. CVPR*, 2015.

[126] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli *et al.*, "Image quality assessment: from error visibility to structural similarity," *IEEE TIP*, vol. 13, no. 4, pp. 600–612, 2004.

[127] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proc. CVPR*, 2018.

[128] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. ICCV*, 1998.

[129] D. Xu, J. Liu, X. Li, Z. Liu, and X. Tang, "Insignificant shadow detection for video segmentation," vol. 15, no. 8, pp. 1058–1064, 2005.

[130] T. K. Shih, N. C. Tang, and J.-N. Hwang, "Exemplar-based video inpainting without ghost shadow artifacts by maintaining temporal continuity," vol. 19, no. 3, pp. 347–360, 2009.

[131] S. H. Khan, M. Bennamoun, F. Sohel, and R. Togneri, "Automatic shadow detection and removal from a single image," *IEEE TPAMI*, vol. 38, no. 3, 2015.

[132] R. Guo, Q. Dai, and D. Hoiem, "Paired regions for shadow detection and removal," *IEEE TPAMI*, vol. 35, no. 12, 2012.

[133] M. Gryka, M. Terry, and G. J. Brostow, "Learning to remove soft shadows," *ACM TOG*, vol. 34, no. 5, 2015.

[134] T. F. Y. Vicente, M. Hoai, and D. Samaras, "Leave-one-out kernel optimization for shadow detection and removal," *IEEE TPAMI*, vol. 40, no. 3, 2017.

[135] J. Wang, X. Li, and J. Yang, "Stacked conditional generative adversarial networks for jointly learning shadow detection and shadow removal," in *Proc. CVPR*, 2018.

[136] X. Hu, C.-W. Fu, L. Zhu, J. Qin, and P.-A. Heng, "Direction-aware spatial context features for shadow detection and removal," *IEEE TPAMI*, 2019.

[137] Y. Shor and D. Lischinski, "The shadow meets the mask: Pyramid-based shadow removal," in *Computer Graphics Forum*, vol. 27, no. 2, 2008.

[138] S. Bako, S. Darabi, E. Shechtman, J. Wang, K. Sunkavalli, and P. Sen, "Removing shadows from images of documents," in *Proc. ACCV*, 2016.

[139] N. Kligler, S. Katz, and A. Tal, "Document enhancement using visibility detection," in *Proc. CVPR*, 2018.

[140] S. Das, K. Ma, Z. Shu, D. Samaras, and R. Shilkrot, "Dewarpnet: Single-image document unwarping with stacked 3d and 2d regression networks," in *Proc. ICCV*, 2019.

[141] Y.-H. Lin, W.-C. Chen, and Y.-Y. Chuang, "Bedsr-net: A deep shadow removal network from a single document image," in *Proc. CVPR*, 2020.

[142] X. C. Zhang, J. T. Barron, Y.-T. Tsai, R. Pandey, X. Zhang, R. Ng, and D. E. Jacobs, "Portrait shadow manipulation," *ACM Trans. Graph.*, vol. 39, no. 4, Jul. 2020. [Online]. Available: https://doi.org/10.1145/3386569.3392390

[143] G. D. Finlayson, S. D. Hordley, C. Lu, and M. S. Drew, "On the removal of shadows from images," *IEEE TPAMI*, vol. 28, no. 1, 2005.

[144] Z. Liu, K. Huang, and T. Tan, "Cast shadow removal in a hierarchical manner using mrf," vol. 22, no. 1, pp. 56–66, 2011.

[145] Q. Yang, K.-H. Tan, and N. Ahuja, "Shadow removal using bilateral filtering," *IEEE TIP*, vol. 21, no. 10, 2012.

[146] B. Ding, C. Long, L. Zhang, and C. Xiao, "Argan: attentive recurrent generative adversarial network for shadow detection and removal," in *Proc. ICCV*, 2019.

[147] O. Sidorov, "Conditional gans for multi-illuminant color constancy: Revolution or yet another approach?" in *CVPRW*, 2019.

[148] D. Liu, C. Long, H. Zhang, H. Yu, X. Dong, and C. Xiao, "Arshadowgan: Shadow generative adversarial network for augmented reality in single light scenes," in *Proc. CVPR*, 2020.

[149] N. Inoue, D. Ito, Y. Hold-Geoffroy, L. Mai, B. Price, and T. Yamasaki, "RGB2AO: Ambient Occlusion Generation from RGB Images," *Computer Graphics Forum*, vol. 39, no. 2, pp. 451–462, 2020.

[150] Y. Wang, B. Curless, and S. Seitz, "People as scene probes," in *Proc. ECCV*, 2020.

[151] E. Salvador, A. Cavallaro, and T. Ebrahimi, "Cast shadow segmentation using invariant color features," vol. 95, no. 2, pp. 238–259, 2004.

[152] G. D. Finlayson, M. S. Drew, and C. Lu, "Entropy minimization for shadow removal," *Springer IJCV*, vol. 85, no. 1, 2009.

[153] M. Russell, J. J. Zou, G. Fang, and W. Cai, "Feature-based image patch classification for moving shadow detection," 2017.

[154] J.-F. Lalonde, A. A. Efros, and S. G. Narasimhan, "Detecting ground shadows in outdoor consumer photographs," in *Proc. ECCV*, 2010.

[155] J. Zhu, K. G. Samuel, S. Z. Masood, and M. F. Tappen, "Learning to recognize shadows in monochromatic natural images," in *Proc. CVPR*, 2010.

[156] X. Huang, G. Hua, J. Tumblin, and L. Williams, "What characterizes a shadow boundary under the sun and sky?" in *Proc. ICCV*, 2011.

[157] A. Panagopoulos, C. Wang, D. Samaras, and N. Paragios, "Simultaneous cast shadows, illumination and geometry inference using hypergraphs," *IEEE TPAMI*, vol. 35, no. 2, pp. 437–449, 2012.

[158] L. Shen, T. Wee Chua, and K. Leman, "Shadow optimization from structured deep edge detection," in *Proc. CVPR*, 2015.

[159] T. F. Y. Vicente, L. Hou, C.-P. Yu, M. Hoai, and D. Samaras, "Large-scale training of shadow detectors with noisily-annotated shadow examples," in *Proc. ECCV*, 2016.

[160] V. Nguyen, Y. Vicente, F. Tomas, M. Zhao, M. Hoai, and D. Samaras, "Shadow detection with conditional generative adversarial networks," in *Proc. ICCV*, 2017.

[161] L. Zhu, Z. Deng, X. Hu, C.-W. Fu, X. Xu, J. Qin, and P.-A. Heng, "Bidirectional feature pyramid network with recurrent attention residual modules for shadow detection," in *Proc. ECCV*, 2018.

[162] Q. Zheng, X. Qiao, Y. Cao, and R. W. Lau, "Distraction-aware shadow detection," in *Proc. CVPR*, 2019.

[163] T. Wang, X. Hu, Q. Wang, P.-A. Heng, and C.-W. Fu, "Instance shadow detection," in *Proc. CVPR*, 2020.

[164] H. Barrow, J. Tenenbaum, A. Hanson, and E. Riseman, "Recovering intrinsic scene characteristics," *Comput. Vis. Syst*, vol. 2, no. 3-26, 1978.

[165] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *Proc. IROS*, 2017.

[166] N. Mahmood, N. Ghorbani, N. F. Troje, G. Pons-Moll, and M. J. Black, "Amass: Archive of motion capture as surface shapes," in *Proc. ICCV*, 2019.

[167] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An Information-Rich 3D Model Repository," Tech. Rep. arXiv:1512.03012, 2015.

[168] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proc. CVPR*, 2019.

[169] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proc. CVPR*, 2020.

[170] R. Abdal, Y. Qin, and P. Wonka, "Image2stylegan: How to embed images into the stylegan latent space?" in *Proc. ICCV*, 2019.

[171] R. Abdal, Y. Qin, and P. Wonka, "Image2stylegan++: How to edit the embedded images?" in *Proc. CVPR*, 2020.

[172] E. Härkönen, A. Hertzmann, J. Lehtinen, and S. Paris, "Ganspace: Discovering interpretable gan controls," in *Proc. NeurIPS*, 2020.

[173] A. Tewari, M. Elgharib, G. Bharaj, F. Bernard, H.-P. Seidel, P. Pérez, M. Zollhofer, and C. Theobalt, "Stylerig: Rigging stylegan for 3d control over portrait images," in *Proc. CVPR*, 2020.

[174] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. NeurIPS*, 2017.

[175] A. Carlier, M. Danelljan, A. Alahi, and R. Timofte, "Deepsvg: A hierarchical generative network for vector graphics animation," in *Proc. NeurIPS*, 2020.

[176] T.-M. Li, M. Lukáč, G. Michaël, and J. Ragan-Kelley, "Differentiable vector graphics rasterization for editing and learning," *ACM TOG*, vol. 39, no. 6, pp. 193:1–193:15, 2020.

[177] P. Reddy, P. Guerrero, M. Fisher, W. Li, and M. J. Mitra, "Discovering pattern structure using differentiable compositing," *ACM TOG*, 2020.

[178] M. Li, J. Lin, Y. Ding, Z. Liu, J.-Y. Zhu, and S. Han, "Gan compression: Efficient architectures for interactive conditional gans," in *Proc. CVPR*, 2020.

[179] H. Wang, S. Gui, H. Yang, J. Liu, and Z. Wang, "Gan slimming: All-in-one gan compression by a unified optimization framework," in *Proc. ECCV*, 2020.

[180] Z. Li and N. Snavely, "Cgintrinsics: Better intrinsic image decomposition through physically-based rendering," in *Proc. ECCV*, 2018.

Publications

Publications related to the thesis

International Journal

[1] Naoto Inoue, Daichi Ito, Ning Xu, Jimei Yang, Brian Price, and Toshihiko Yamasaki, "Learning to Trace: Expressive Line Drawing Generation from Photographs," *CGF (also Proc. Pacific Graphics2019)*, vol. 38, no. 7, pp. 69-80, 2019.

[2] Naoto Inoue, Daichi Ito, Yannick Hold-Geoffroy, Long Mai, Brian Price, and Toshihiko Yamasaki, "RGB2AO: Ambient Occlusion Generation from RGB Images," *CGF (also Proc. EuroGraphics2020)*, vol. 39, no. 2, pp. 451-462, 2020.

[3] Naoto Inoue and Toshihiko Yamasaki, "Learning from Synthetic Shadows for Shadow Detection and Removal," in IEEE TCSVT 2020.

Publications non-related to the thesis

International Journal

[4] Ryosuke Furuta, Naoto Inoue, and Toshihiko Yamasaki, "Efficient and interactive spatial-semantic image retrieval," *MTAP*, vol. 78, no. 13, pp. 18713-18733, 2019.

[5] Ryosuke Furuta, Naoto Inoue, and Toshihiko Yamasaki, "PixelRL: Fully Convolutional Network With Reinforcement Learning for Image Processing," *TMM*, vol. 22, no. 7, pp. 1704–1719, 2019.

International Conference

[6] Yuki Takada, Naoto Inoue, Toshihiko Yamasaki, and Kiyoharu Aizawa, "Similar Floor Plan Retrieval Featuring Multi-Task Learning of Layout Type Classification and Room Presence Prediction," in *Proc. ICCE*, 2018.

[7] Ryosuke Furuta, Naoto Inoue, and Toshihiko Yamasaki, "Efficient and Interactive Spatial-Semantic Image Retrieval," in *Proc. MMM*, 2018.

[8] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa, "Cross-Domain Weakly-Supervised Object Detection through Progressive Domain Adaptation," in *Proc. CVPR*, 2018.

[9] Ryosuke Furuta, Naoto Inoue, and Toshihiko Yamasaki, "Fully Convolutional Network with Multi-Step Reinforcement Learning for Image Processing," in *Proc. AAAI*, 2019.

[10] Naoto Inoue and Toshihiko Yamasaki, "Fast Instance Segmentation for Line Drawing Vectorization," in *Proc. BigMM (short)*, 2019.

[11] Takehiko Ohkawa, Naoto Inoue, Hirokatsu Kataoka, and Nakamasa Inoue, "Augmented Cyclic Consistency Regularization for Unpaired Image-to-Image Translation," in *Proc. ICPR*, 2020.

Domestic Conference

[12] 井上直人, 古田諒佑, 山崎俊彦, 相澤清晴, "スタイル転移を利用した物体検出におけるドメイン適応," IE研究会, 2018.

[13] 井上直人, 山崎俊彦, "Learning from Synthetic Shadows", PRMU研究会, 2021.

Invited Talk / Poster

[14] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa, "Cross-Domain Weakly-Supervised Object Detection through Progressive Domain Adaptation," in *2nd WebVision Workshop at CVPR*, 2018.

[15] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa, "Cross-Domain Weakly-Supervised Object Detection through Progressive Domain Adaptation," in *MIRU*, 2018.

[16] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa, "Cross-Domain Weakly-Supervised Object Detection through Progressive Domain Adaptation," in *FIT*, 2019.

Awards

[17] 井上直人, 2018年度 東京大学大学院情報理工学系研究科　優秀修論賞.

[18] 古田諒佑，井上直人, 山崎俊彦　2018年度 画像工学研究会 IE賞・IE特別賞.

Appendix A

Supplementary Results on Ambient Occlusion Generation

A.1 Ablation Study

We performed ablation study on our AO generation model from three viewpoints.

Loss Functions We show that selecting an appropriate loss functions is important. We tested L1, L2, and SSIM loss functions, which are used in Deep Shading [100] for AO generation from ground truth depth and normal buffer. The comparison of models optimized by different loss functions is shown in Table A.1. Our model performs better, especially in LPIPS and MAE. To see the difference, we show some results in Fig. A.1. Our results are significantly better than the others in that they have less blur and artifacts.

Models We show that selecting an appropriate CNN model is important. In Table A.2, we show comparison with different models used in previous works.

- Enc-Dec: encoder-decoder is widely used in image-to-image translation such as Pix2pix [12]. We used 'resnet-6blocks' model with two down/up-sampling operations and six residual blocks from Pix2pix.

- UNet: Variants of UNets are widely used in intrinsic image decomposition [180] and screen-space AO algorithm using CNN [100]. We used 'unet-128' model with seven down/up-sampling operations from Pix2pix.

- Innarmorati: Innarmorati *et al*. [106] designed another variant of UNet. We considered two variants. 'Innarmorati-sc' is the model trained from scratch.

Table A.1: Ablation study on loss functions for AO generation. For fair comparison, all the models are optimized with our AO augmentation and multi-task learning. ↓ and ↑ indicate that lower and higher is better, respectively.

	MAE ↓	MSE ↓	SSIM ↑	LPIPS ↓
L1	0.0599	0.0102	0.764	0.303
L2	0.0615	**0.0094**	0.758	0.287
SSIM	0.0605	0.0096	**0.786**	0.250
Ours	**0.0589**	0.0103	0.767	**0.235**

Table A.2: Ablation study on different models for AO generation. For fair comparison, all the models are optimized with AO augmentation and without the multi-task learning. ↓ and ↑ indicate that lower and higher is better, respectively.

	MAE ↓	MSE ↓	SSIM ↑	LPIPS ↓	#params
Enc-Dec	0.0659	0.0120	**0.766**	0.258	7.8M
UNet	0.0650	0.0119	0.745	0.277	41.8M
Innarmorati-sc	0.0686	0.0129	0.743	0.293	8.5M
Innarmorati-ft	0.0669	0.0126	0.741	0.291	8.5M
CAN	0.0828	0.0187	0.663	0.398	0.08M
Hourglass	**0.0600**	**0.0103**	0.760	**0.235**	5.4M

'Innarmorati-ft' is the model fine-tuned based on the weights obtained in their model for AO estimation.

- CAN: context aggregation networks [13] is used to approximate a wide variety of standard low-level image processing operators by CNN trained on input-output pairs. We used 'CAN32+AN' model.

We use Hourglass [92] network, and it performs significantly better in LPIPS despite of the smaller number of parameters of the CNN. One reason may be that it has a larger receptive field by inception module [125], which uses several convolutional layers with different kernel size in parallel. This is essential to capture interaction among large objects and stuff (e.g., ground, wall, ceiling) consistently for AO generation. Although fine-tuning Innarmorati *et al*. [106]'s model trained for AO estimation in their dataset for a single object helps to improve the result slightly compared to the model trained from scratch, we observe that selecting the appropriate network, hourglass, is much more important.

Resolution We show that our model performs better if it is trained and evaluated on larger resolution inputs. We trained and evaluated our model on

Figure A.1: Comparison of AO generation models optimized by different loss functions tested on our synthetic dataset. Our results have less blur and artifacts. (best viewed with zoom and in color.)

two times larger input resolution (448×768). The model trained on larger input resolution performs better in almost all the metrics, as shown in Table A.3. This trend is also clear in the results shown in Fig. A.2.

A.2 Results on High Resolution Inputs

We show that our model can process high resolution images (e.g., 1024×2048) easily and successfully. This is done as follows: (i) resizing the image to a lower resolution, (ii) estimating AO, (iii) resizing it back to the original resolution, and (iv) multiplying the generated AO with the input. Some results of AO generation on real high

Table A.3: Ablation study on the resolution that the model is trained and evaluated on for AO generation. For fair comparison, the images for evaluation are not center-cropped during testing. ↓ and ↑ indicate that lower and higher is better, respectively.

Input resolution	MAE ↓	MSE ↓	SSIM ↑	LPIPS ↓
224×384	0.0566	**0.0097**	0.778	0.234
448×768	**0.0561**	**0.0097**	**0.834**	**0.220**

Table A.4: Comparison of different models for AO estimation. ↓ and ↑ indicate that lower and higher is better, respectively.

Network	Loss	MAE ↓	MSE ↓	SSIM ↑	LPIPS ↓
w/o depth info.					
Innamorati	L2	0.0498	0.0064	0.836	0.220
Innamorati	Innamorati	0.0500	0.0064	0.836	0.217
Innamorati	ours	0.0491	0.0070	0.835	0.191
Hourglass	L2	0.0445	**0.0051**	0.847	0.204
Hourglass	Innamorati	0.0476	0.0055	0.833	0.225
Hourglass	ours	**0.0420**	0.0052	**0.856**	**0.163**
w/ depth info.					
Hourglass	ours	0.0414	0.0051	0.857	0.162

resolution images are shown in Fig. A.3, Note that AO generation is performed on the resized low-resolution image (384 pixels on the larger side).

A.3 Additional Experiments on AO Estimation

We show that our model also performs better than the existing AO estimation model [106] with minimal modification. We trained AO estimation model using different networks and loss functions from pairs of (i) multiplication of RGB and AO and (ii) AO map. We did not use our proposed AO augmentation in all the experiments, because it is specialized for AO generation. The result is shown in Table A.4. Similarly to the ablation study of AO generation in Sec. A.1, selecting proper network and loss functions are both important to achieve the best performance in AO estimation.

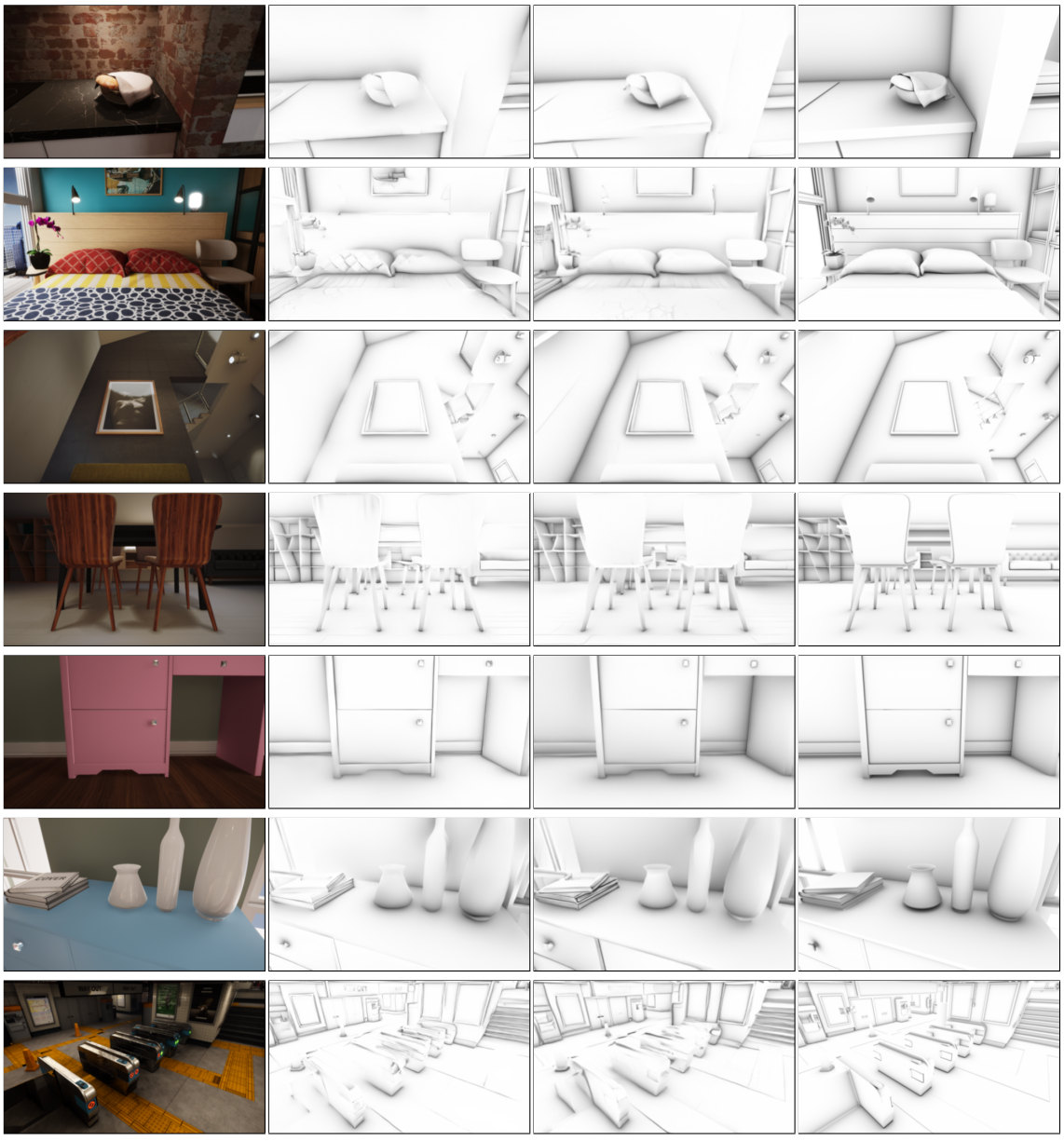


Figure A.2: Results from models trained on different input resolutions. The model trained on the lower resolution performs poorly. For example, generated AO is inconsistent along the boundary of planes or objects (from the first to third row), texture on flat surface is mistakenly detected as the source of AO (in the fourth row), unnatural and sharp AO change (in the fifth and sixth row), and missing AO around small objects (in the seventh row). (best viewed with zoom and in color.)

Input Generated AO AO magnification

Figure A.3: Results on real high resolution images (2048 pixels for the larger side). Due to file size limitation, the larger side is further resized to 512 pixels. Note that 'Generated AO' images are predicted on low resolution (384 pixels for the larger side) and then up-sampled. (best viewed with zoom and in color.)

Appendix B

Supplementary Results on Synthetic Shadow Generation

B.1 Further Analysis on SRD and SRD+

We find that the original train-test split of SRD [9] is inappropriate since images coming from the identical background are both in training and testing set. These images likely share the almost similar shadow attenuation property, because the scenes are captured with nearly the same camera and outdoor lighting condition, as shown in Fig. B.1. We argue that this makes the evaluation of shadow removal models using SRD unfair.

To prove the argument, we made SRD+ by re-splitting SRD so that there is no overlap of the background images in the two sets and removed duplicated images. We run various methods for shadow removal on both SRD and SRD+. As is done in all the experiments for shadow removal, we reported RMSE in 640×480 image resolution. The lower value indicates better performance. We tested both regression-based methods (e.g., MSGAN [10], CycleGAN [14], DHAN [7]) and decomposition-based methods (e.g., SP+M [8]).

First, we show that SRD+ is less challenging in Table B.1. We first trained a model on a different dataset other than SRD and SRD+, and then evaluated on both SRD+ and SRD. We observe the clear performance improvement on all the methods when we used SRD+ instead of SRD for the evaluation.

Second, the performance of regression-based methods clearly drops when we switch the dataset for training and evaluation from SRD to SRD+ as shown in Table B.2. This result supports our argument because regression-based methods can 'cheat'

Figure B.1: Examples of the scene overlap between the train-test split in SRD [9]. The images in the odd and even columns are from the training and testing set, respectively. The left pair seems to be near duplicates. The right pair share the exactly the same background.

Table B.1: Training/evaluation on a different dataset. SRD+ is less challenging compared to SRD in terms of RMSE.

	Evaluated on					
	SRD+			SRD		
	S	NS	ALL	S	NS	ALL
Trained on: USR						
CycleGAN	29.9	11.5	15.4	31.5	11.7	16.5
MSGAN	30.3	11.0	15.1	31.0	11.1	16.0
Trained on: ISTD+						
DHAN	16.6	8.1	9.9	19.9	9.2	11.8
SP+M	14.9	4.5	6.7	17.5	6.0	8.8

by just memorizing examples used for training if there is the background overlap between the training and test set. The only exception is the decomposition-based SP+M.

B.2 Additional Results on Shadow Removal and Shadow Detection

We show the more results of supervised-learning-based shadow removal models trained on different datasets. The result is shown in Fig. B.2 and Fig. B.5 for SP+M [8] and DHAN [7], respectively. We also show more results comparing the models trained on SynShadow with traditional approaches and unsupervised learning approaches in Fig. B.3. We show some failure cases that even the best shadow removal models trained on SynShadow cannot still handle in Fig. B.4. We hope to solve them by collecting the more diverse and large-scale data in the future work.

Table B.2: Training/evaluation on a same dataset. There is a remarkable performance drop in the regression-based methods when we used SRD+ instead of SRD, which suggests that the original split of SRD is inappropriate.

	Trained / evaluated on SRD+			SRD		
	S	NS	ALL	S	NS	ALL
CycleGAN [14]	17.5	7.5	9.6	17.2	6.0	8.7
MSGAN [10]	16.5	7.6	9.5	12.3	5.4	7.1
DHAN [7]	12.7	5.9	7.4	8.5	4.1	5.1
SP+M [8]	12.2	3.4	5.3	11.3	4.3	6.0

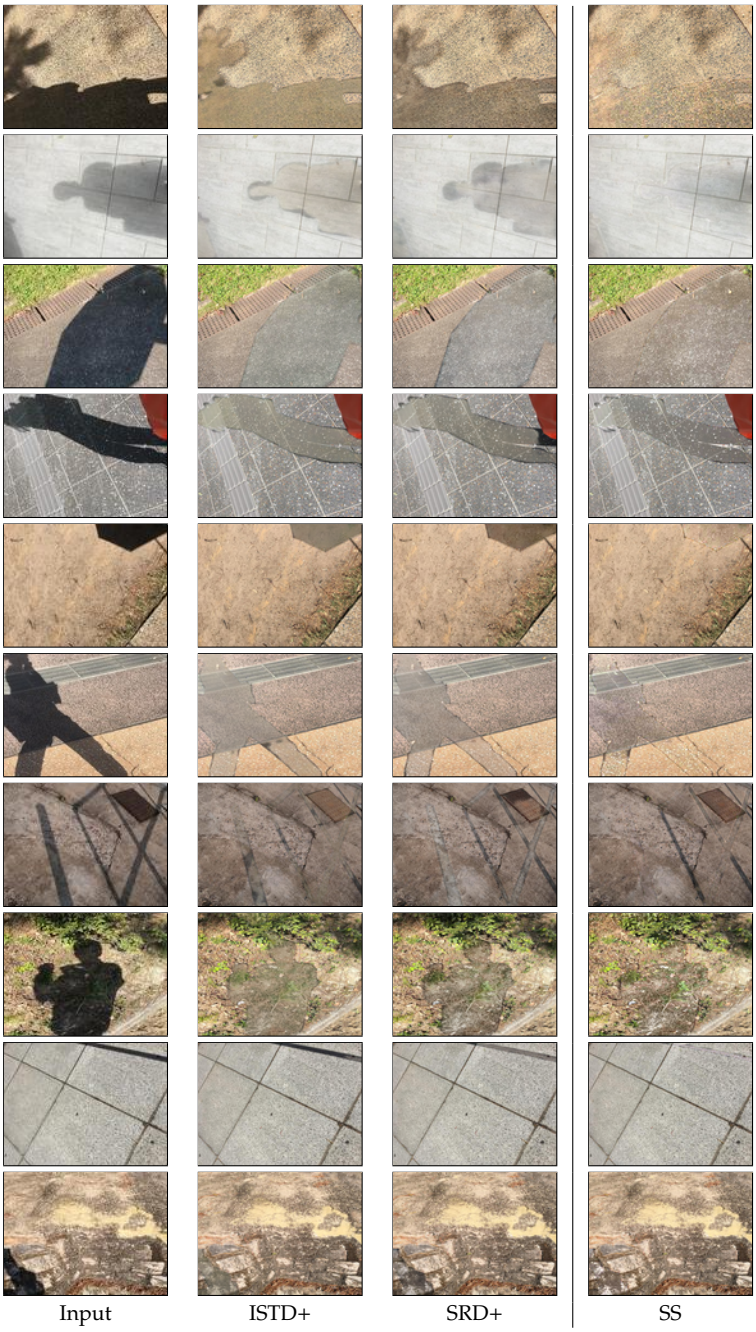


Figure B.2: Results of SP+M [8] trained on different datasets and tested on USR [10] test set. SS is short for SynShadow.

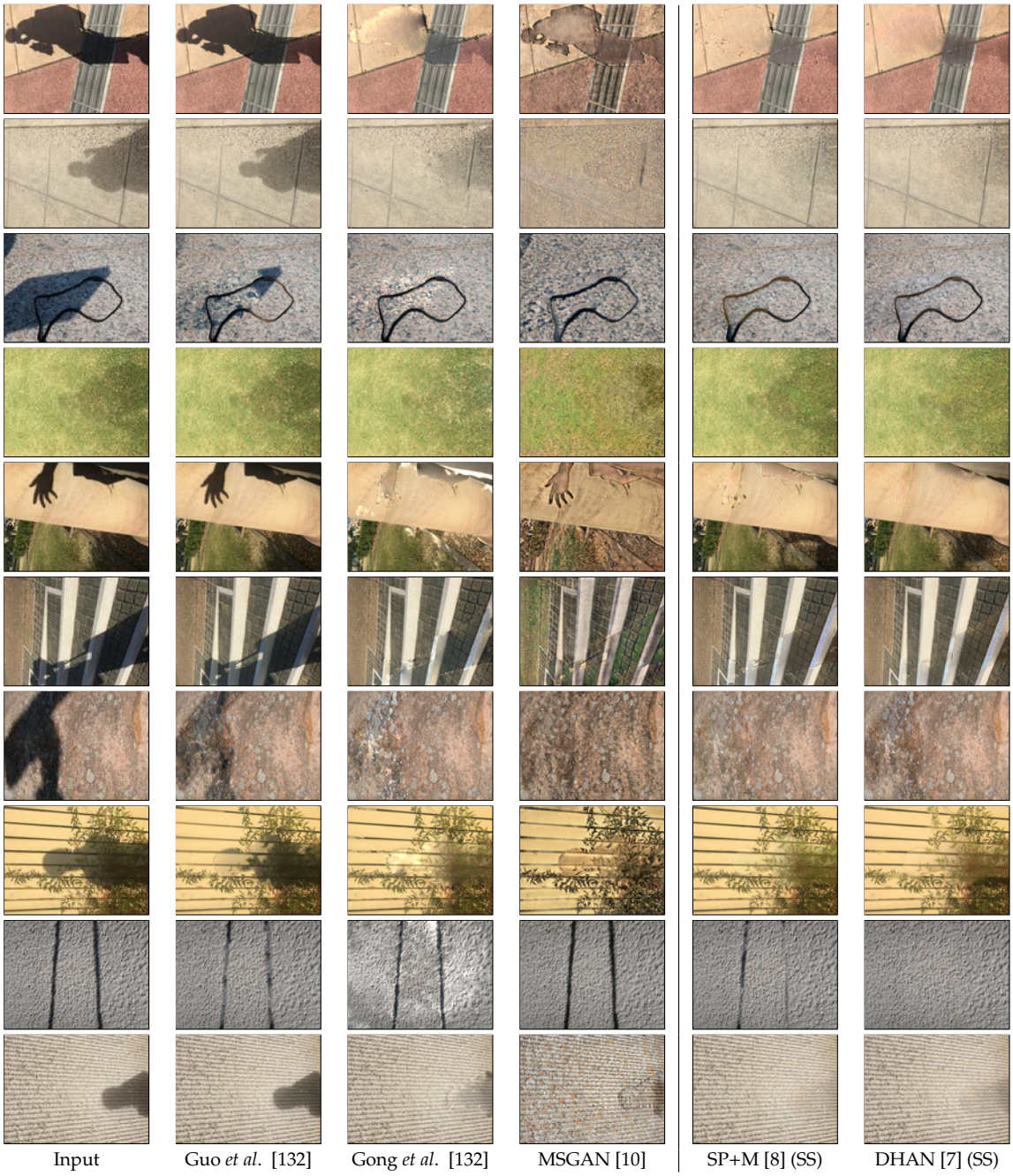


Figure B.3: Qualitative comparison of shadow removal methods on USR [10] test set. SS stands for SynShadow.

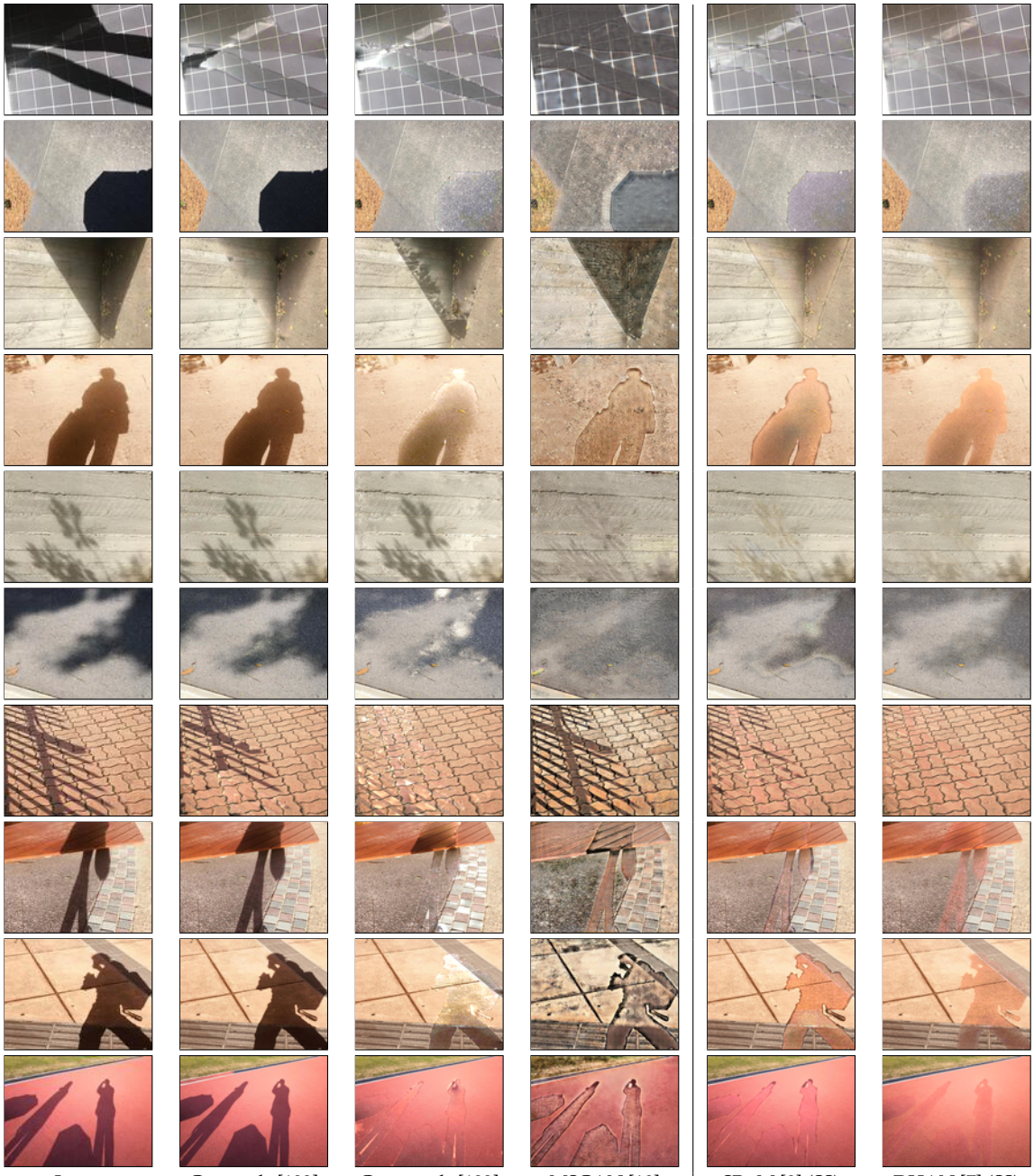


Figure B.4: Failure cases of shadow removal models. SS stands for SynShadow. Failure cases are due to (i) very strong shadows (the first and second row), (ii) non-uniform shadows (the third and fourth row), (iii) shadows with complex or unseen shape (from the fifth to seventh row), and (iv) very bright background (from the eighth to tenth row).

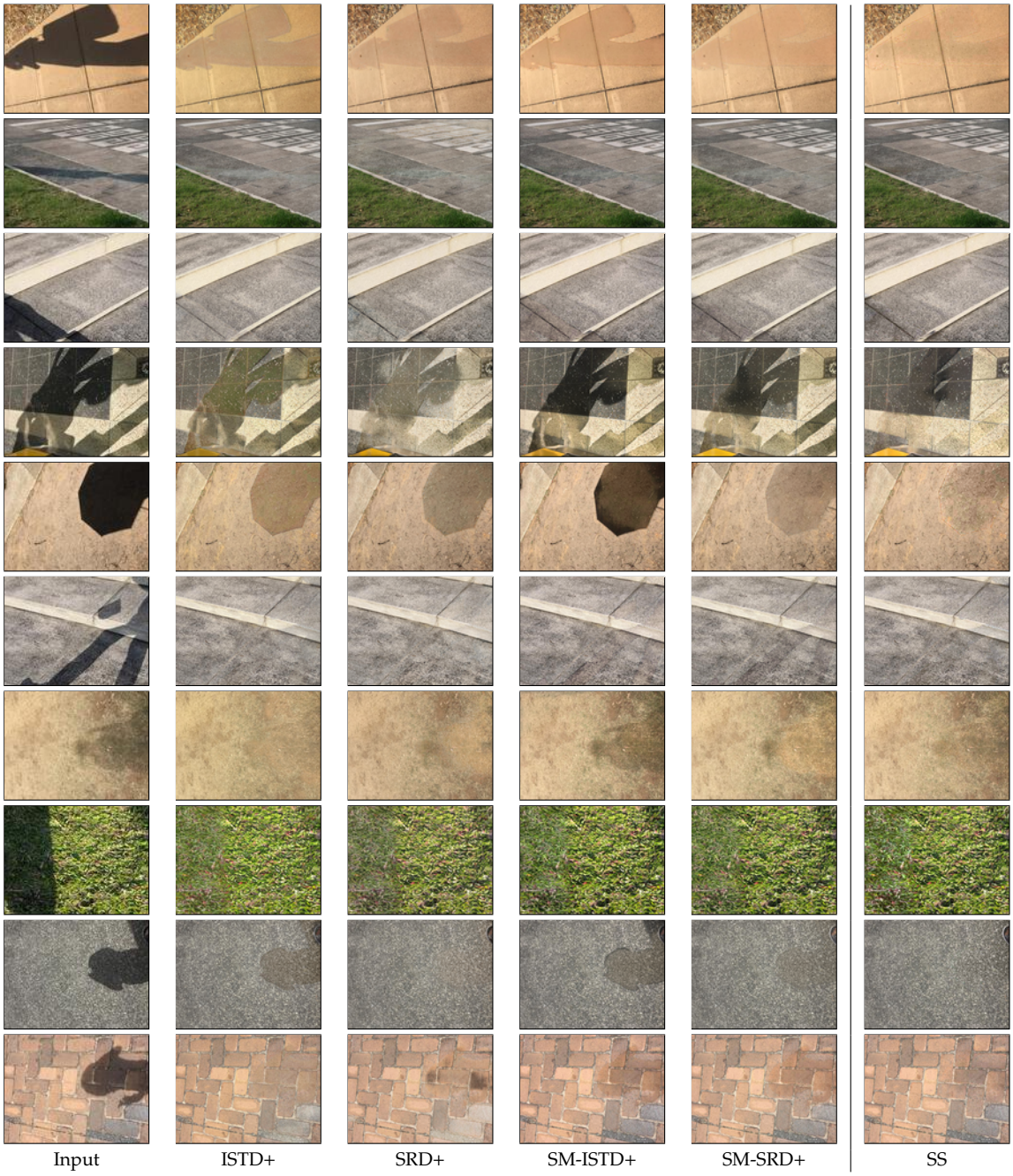


Figure B.5: Results of DHAN [7] trained on different datasets and tested on USR [10] test set. SS is short for SynShadow. SM-ISTD+ and SM-SRD+ indicates ISTD+ and SRD+ augmented by SMGAN generated images, respectively.