Alleviate the Domain-shift Problem of Vision Tasks

by

Dong Huo

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computing Science
University of Alberta

Abstract

Computer vision tasks have seen breakthroughs in recent years thanks to the emergence of deep learning (DL). However, there exists different types of domain-shift problems that may impact the performance of DL-based methods. In low-level vision tasks, *e.g.*, image restoration, the degradation of the training data can be different from that of the testing data. In high-level vision tasks such as image segmentation, models trained on the common object datasets cannot be applied to some specific objects. In this dissertation, I present novel learning strategies to alleviate the domain-shift problems of both low-level and high-level vision tasks.

I start with low-level vision tasks, specifically, dynamic scene deblurring/deconvolution (2D spatial degradation). The degradation matrices of the training data are rarely seen during testing. Thus, I propose two approaches to solve this problem from different angles. In the first approach, the model trained on paired datasets can adaptively adjust to different magnitudes and directions of the motion blur, which is capable of solving unseen degradations. Although the generalization of the model is improved, it still depends on synthetic data for training because the number of real-world paired images available for training are limited. Hence, the performance is lowered on real blurred data. In the second approach, I adopt the deep image prior (DIP) to bypass supervised training and utilize only a single degraded image to update the neural network parameters, which is more flexible to variant blurs without the impact of the training data.

In 3D, the degradation is related to spatial resolution degradation in the context of novel view synthesis. Along this direction, I propose a novel view synthesis task,

which can reconstruct novel views of 3D objects. In this dissertation, I simplify the problem into texture generation for untextured 3D meshes. Novel views are synthesized using a depth conditioned image generation model with the source view used as the guidance, and the depth rendered from a given mesh. Similar to the 2D spatial degradation problem, most of the texture generation models are trained on synthetic data due to the limitation of real training data, and cannot generate photo-realistic textures. Recently, Stable Diffusion (SD) trained on large real-world image datasets for image generation has been applied to many down-stream tasks, *e.g.*, geometry generation, novel-view synthesis. I adopt the pretrained SD without fine-tuning to generate photo-realistic textures for 3D objects conditioned on an extra textual prompt.

Going from the spatial to the spectral domain, I address the problem of spectral degradation with the objective of recovering the spectral reflectance from a single RGB image. Due to the difficulty of obtaining paired training data, most of the methods are trained and tested on synthetic data. Similar to other degradation problems, the synthetic data are not the same as the real data so the trained models using these datasets fail when they are applied to real data. To address this issue, I propose to adopt meta-auxiliary learning to solve this problem by training the model on synthetic data but adapting it to the real data at test time with only several steps of gradient updates.

For high-level vision tasks, *e.g.*, semantic segmentation, I solve the glass surface segmentation problem where the semantic segmentation methods trained on common objects fail to detect transparent glass surfaces. Considering the different transmission of the glass with regard to the visible light and infrared (thermal) light, an extra thermal camera is exploited for better detection. In particular, I collected an extensive paired RGB-thermal image dataset with manually labeled masks for model training, and aggregated the trained model with existing semantic segmentation methods to generalize semantic segmentation to glass scenes.

Preface

All methods presented in this thesis are published or under review at the top venues of computer vision. I am accountable for conceptualizing and executing algorithm designs, experiments, and the majority of paper writing for all projects.

Chapter 3: Dong Huo, Abbas Masoumzadeh, and Yee-Hong Yang. Blind non-uniform motion deblurring using atrous spatial pyramid deformable convolution and deblurring-reblurring consistency. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022.

Chapter 4: Dong Huo, Abbas Masoumzadeh, Rafsanjany Kushol, and Yee-Hong Yang. Blind Image Deconvolution Using Variational Deep Image Prior. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2023.

Chapter 5: Dong Huo, Xinxin Zuo, Zixin Guo, Zhihao Shi, Juwei Lu, Peng Dai, Li cheng, Yee Hong Yang. Text-Guided Texture Generation for 3D Objects with Progressive Sampling. Under review, 2024.

Chapter 6: Dong Huo, Jian Wang, Yiming Qian, and Yee-Hong Yang. Learning to Recover Spectral Reflectance from RGB Images. IEEE Transactions on Image Processing, 2024.

Chapter 7: Dong Huo, Jian Wang, Yiming Qian, and Yee-Hong Yang. Glass segmentation with RGB-thermal image pairs. IEEE Transactions on Image Processing, 2023.

This thesis is a concatenation of the above five papers.

Acknowledgements

First and foremost, I extend my heartfelt gratitude to my PhD supervisor, Prof. Herbert Yang. His guidance, support, and friendship have been instrumental throughout my journey. I am deeply grateful for his patience and encouragement, especially during the challenging times when I faced self-doubt. Without his mentorship, this thesis would not have come to fruition.

I am also indebted to my esteemed PhD committee members: Prof. Pierre Boulanger, Prof. Nilanjan Ray, Prof. Lili Mou, and Prof. Li Cheng. Their valuable insights and constructive feedback have significantly enriched this work.

I extend my sincere appreciation to the Faculty of Graduate Studies and Research, as well as the Department of Computing Science, for their support over the past five years. I am thankful to all the staff members who have assisted me along the way. Additionally, I am grateful to my labmates in Prof. Herbert Yang's group, including Rafsanjany Kushol, Abbas Masoumzadeh, G.M. Mashrur E Elahi, Sangwon Lim, Ruiqin Pi, Shibai Yin, and Bernard Llanos, for their invaluable contributions to this thesis.

Special thanks go to my friends: Yiming Qian, Xinxin Zuo, Jian Wang, and Steve Sutphen, for their steadfast support and academic collaboration. They have significantly enhanced my PhD journey, making it both enjoyable and highly productive.

I would like to acknowledge the funding sources that have supported my PhD work, including the Natural Sciences and Engineering Research Council of Canada, Alberta Innovates, and the University of Alberta.

Lastly, I express my deepest gratitude to my family members for their uncon-

ditional love and support. I am especially grateful to my wife for her sacrifice and companionship throughout this journey, and to my parents for their financial assistance during my time in Canada.

Table of Contents

List of Tables

List of Figures

Chapter 1

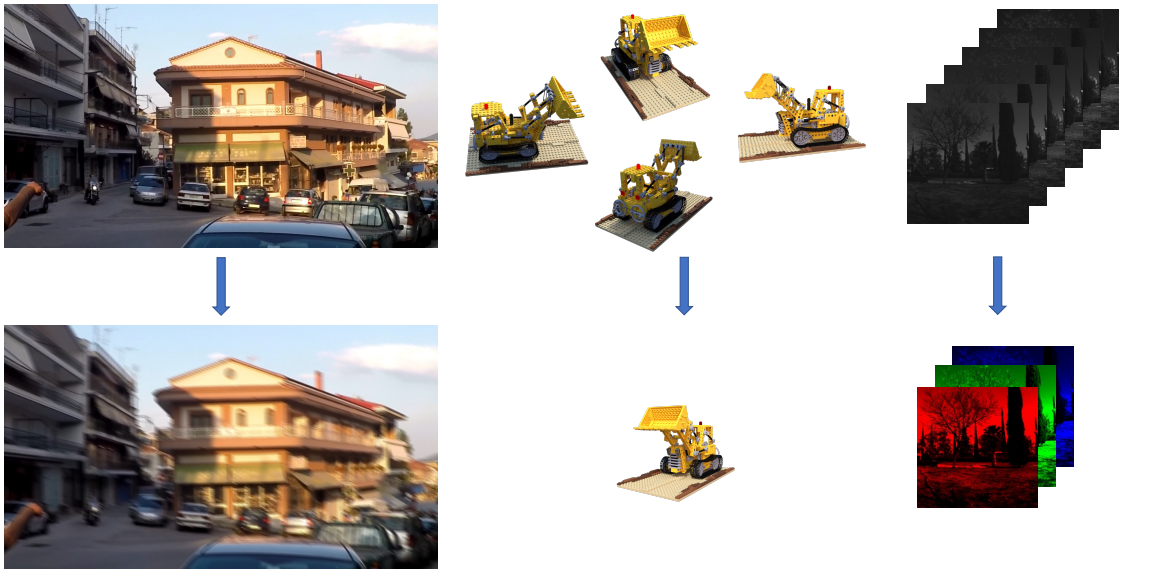
Introduction

1.1 Motivation

Ever since the emergence of deep learning (DL), computer vision has seen many breakthroughs in many tasks, *e.g.*, image classification [1–3], object detection/segmentation [4–8], image restoration [9–12], image generation [13–15], *etc.* However, when the distribution of data with which the model is trained is different from that for testing, the performance of the model drops significantly. Such an issue is commonly known as the domain-shift problem.

Many techniques have been proposed to alleviate the domain-shift problem. The most straightforward solution is to train the model on a large dataset (*e.g.*, ImageNet [16], LAION-5B [17]) that is able to cover the testing data distribution as much as possible. However, a large dataset is difficult to obtain due to the expensive manual collection and labeling process. Thus, models trained on large datasets are usually exploited as the pretrained backbones for downstream tasks [7, 18, 19].

Let us first consider examples of low-level vision tasks, in particular, the problem of dynamic scene deblurring/deconvolution (2D spatial degradation). Since the motion is arbitrary, the degradation matrices of the testing data might be dissimilar to those of the training data. Thus, adopting the same pretrained model on all testing images is sub-optimal [10, 20–22]. Besides, capturing real-world blurred images with the corresponding ground-truth sharp images requires complicated devices [23],



(a) 2D Spatial degradation (a) 3D Spatial degradation (c) Spectral degradation

Figure 1.1: (a) An example of 2D spatial degradation. In particular, the motion blur. (b) An example of 3D spatial degradation, where only a single view of the object has been recorded. (c) An example of spectral degradation that integrate hyperspectral spectrum into three RGB channels.

which is hard to obtain. Hence, training datasets are often generated synthetically by averaging several consecutive frames of a high FPS (frames per second) video to create blurred video frames [24, 25]. Although the dataset is easy to obtain, the performance of models trained using this method is reduced on real data. The limitation of available data for training in 3D spatial degradation is even worse. Typically, models trained on synthetic data are unable to correctly synthesize novel views of real objects [26, 27]. Similar problems also occur in spectral reconstruction which reconstructs the spectral reflectance from RGB images [28–30].

In addition to low-level vision tasks, high-level vision tasks also suffer from a similar domain-shift problem. In this dissertation, I focus on the problem of glass surface segmentation. Most of the datasets for training semantic/instance segmentation models contain only common opaque objects [31, 32] and hence, the trained models fail to detect transparent glass surfaces and mis-segment out reflections or objects behind glass surface.

1.2 Background and Contributions

1.2.1 Inverse Problems

Images/videos captured in different environments by different devices may suffer from 2D spatial degradations such as spatial downsampling [33] and blurring [34], which significantly impact the performance of subsequent high-level computer vision tasks, *e.g.*, semantic segmentation [35], object tracking [36]. Besides, the captured RGB image represents the projection of a 3D scene onto the 2D screen, which suffers the issue of 3D spatial degradation where the spatial information of the 3D scene from other viewpoints is lost. The projection matrix can be regarded as a special case of the degradation matrix. In addition to spectral degradation, RGB cameras also suffer from spectral degradation compared with multispectral/hypersepctral cameras that produce images with more than three channels, making them difficult to distinguish metameric colors. Some examples are shown in Fig. 1.1. All of the above mentioned problems can be formulated as

$$Y = X \circledast D + N, \tag{1.1}$$

where X is the original high-quality RGB image or the 3D scene, Y is the degraded image, D is the degradation matrix, $\circledast$ represents matrix multiplication, and N denotes an additive noise. The problems that satisfy Eqn. 1.1 are called inverse problems. An inverse problem is ill-posed when D and N are unknown or the spatial/spectral dimension of X is larger than that of Y.

In the past few decades, many conventional optimization-based methods have been proposed to solve the above inverse problems, *e.g.*, super resolution [37, 38], deblurring [39, 40], novel view synthesis [41, 42] and hypersepctral reconstruction [43, 44], but the results are sub-optimal due to the inflexibility of hand-crafted priors or regularizers. Recently, deep-learning (DL) methods improve the performance of restoration by a large margin [11, 28, 29, 45–47]. Indeed we no longer need to consider the convexity of the optimization target or the selection of appropriate hand-crafted

Figure 1.2: Example of spatially variant blur. The blue block denotes the region with large blur, and the yellow block denote the region with tiny blur.

priors, and the whole process is done end-to-end with gradient descent with the image priors implicitly learned within the network.

Most of the DL-based methods for inverse problems are built upon several well-known backbones, *e.g.*, UNet, ResNet, Inception-ResNet, and apply the same convolutional kernels (including shapes, receptive fields, and values) on different image regions, resulting in the needs for more parameters to achieve better generalization. An example is shown in Fig. 1.2, regions with different types of blur should be treated separately. Moreover, they are trained on large datasets which are impractical to obtain, in particular, in areas such as medical imaging and 3D reconstruction. Even if the amount of data is sufficient for training, directly adopting the same trained model for all unseen testing images is sub-optimal [48, 49]. The trained model does not generalize well when the testing images contain unique features and different degradations, especially when the model is trained on synthesized images and tested on real images (as shown in Fig. 1.3). One solution is to train image-specific models using deep image prior (DIP) [50, 51], which regards the architecture of a neural

(a) Synthesized image (b) Real image

Figure 1.3: (a) An image synthesized by averaging several consecutive frames of a video, which shows discontinuous artifacts along the direction of the motion, whereas a real motion blurred image (b) is continuous and smooth.

network as a hand-crafted prior and optimizes the parameters with only the input degraded image. However, it is hard to find a proper network architecture because the relationship between images and their corresponding network architectures is unclear, and the inference time using the DIP is long (usually in hours) because the network is trained on each testing image separately. The method that I propose to address the above mentioned limitations are described in Chapter 3 to 6 of this thesis.

In Chapter 3, I propose a novel non-uniform motion deblurring method with Atrous Spatial Pyramid Deformable Convolution (ASPDC) modules, with different sizes of receptive fields that realize region-specific convolution, where the ASDPC module with different dilation rates extract information of different magnitudes of motion and separates the image into regions with the help of attention maps, which reduce the workload of each branch by focusing on regions with specific magnitudes of motion, instead of always considering the entire image.

In Chapter 4, I propose a novel variational deep image prior (VDIP) based method for single image blind deconvolution by integrating the deep image prior (DIP) and variational Bayes. A complete derivation of the final loss function and a mathematical analysis are provided to demonstrate that the proposed method can constrain the

optimization better than that of the DIP.

In Chapter 5, I propose a novel architecture motivated by a new mathematical derivation that integrates physical properties of the spectral reflectance into the network with an unknown camera spectral sensitivity (CSS). I also propose a unified framework for recovering spectral reflectance from RGB images captured under more than one illumination. In order to reduce the domain gap between the training data and the testing data, I present the first work that successfully adopt meta-auxilary learning (MAXL) to spectral reflectance recovery (SRR). To the best of my knowledge, it is the first attempt to explore the potential of MAXL in this task.

In Chapter 6, I present a method to solve the novel view synthesis task. To simplify the problem, I focus on the texture generation task for a given geometry where the front view is synthesized using a given mesh. In order to avoid the limitation of small dataset training, I leverage a 2D image generation model [52] which is pre-trained on a large image dataset and can generate high quality images. A novel texture sampling scheme is proposed to lift the generated 2D image to 3D. It is noteworthy that the proposed framework can naturally support text-driven texture editing as well.

1.2.2 Glass Segmentation

Human-made environments are full of architectural elements constructed from glass materials such as glass windows, glass doors, glass panels on railings, and glass walls. Accurately identifying and distinguishing these objects has numerous applications in robotics [53], manufacturing [54] and assistive care [55]. Compared to opaque materials, transparent glasses do not have their own colors and their appearances are acquired from the background, posing inconsistent visual features if the background or viewpoint is changed. Therefore, glass objects with background-dependent appearances often pose challenges for visual recognition methods that are tailored to opaque objects.

With the emergence of deep neural networks, recent data-driven methods are capa-

ble of segmenting glass regions from a single RGB image and have utilized contextual information [56], reflection detection [57] and boundary supervision [58]. While neural networks are powerful, they are based on unreliable RGB colors or directly adopt the learning frameworks for opaque materials, resulting in limited accuracy. Several methods seek to leverage alternative cues such as depth [59], light-field [60] or polarized light [61]. However, these methods are not robust enough for clear glass recognition, and produce noisy segmentation masks (holes, rough boundaries).

I take a step towards fusing RGB and thermal images (RGB-T) for glass segmentation, which is the core contribution. Compared to visible light, which has nearly 100% transmission, thermal radiation with wavelength in the range from 8 to 12μm cannot pass through a typical glass, *i.e.*, 0% transmission. Such a unique physical difference between visible light and thermal energy makes glass easy to be detected if an RGB image and a thermal image are jointly processed instead of using the RGB image only. As expected, our experimental results validate our intuition that the proposed RGB-T fusion method outperforms the RGB-only solution by a large margin.

In Chapter 7, I propose a neural network architecture that takes as input a pair of RGB and thermal images (in short RGB-T) and predicts a binary segmentation mask for the glass regions. Following the encoder-decoder framework, our architecture employs (1) two ResNet encoders for feature extraction of the two images, (2) a novel transformer-based fusion module that uses self-attention for correlating the two images at the feature level, and (3) a decoder that uses convolution-based spatial attention for adaptively selecting features for the final mask generation. I also collected a new dataset consisting of 5551 aligned RGB-T image pairs captured by an off-the-shelf RGB-T camera, where the ground-truth (GT) segmentation masks are created manually.

1.3 Organization

The rest of this dissertation is organized as follows. Chapter 2 reviews previous works on image deblurring, 3D content generation, spectral reconstruction from RGB images, and transparent object/glass recognition. The four methods proposed for solving different inverse problems discussed in Section 1.2.1 are introduced in Chapter 3, 4, 5, 6, respectively. The RGB-T glass segmentation mentioned in Section 1.2.2 is introduced in Chapter 7. For each problem, I first present the proposed approach, followed by the experimental results. Chapter 8 concludes the thesis and discusses future directions. Appendix A to E consist of detailed derivation, algorithms, hyper-parameters and more experimental results of corresponding chapters.

Chapter 2

Related Work

2.1 Blind Image Deconvolution/Deblurring

Some conventional single image blind deconvolution/deblurring methods focus on the distribution of image gradient for sparse high-frequency information. Fergus *et al.* [62] propose a heavy-tailed natural image prior, which is approximated by a mixture-of-Gaussian model. Shan *et al.* [63] demonstrate that the ringing effect on the deblurred image results from the estimation error of the blur kernel and noise. Cho and Lee [64] utilize the bilateral filter and the shock filter to remove noise and to enhance edges. Xu and Jia [65] find that edges smaller than the kernel size are harmful to kernel estimation and propose an r-map to measure the usefulness of edges. Krishnan *et al.* [66] adopt the ratio of the L1 norm and the L2 norm to avoid the scale variant prior, which is much closer to the L0 norm. Levin *et al.* [67] prove that the MAP with the sparse image prior favors a blurred solution so that they approximate the marginalization of the blur kernel, which has a closed-form solution when using the Gaussian image prior. Babacan *et al.* [68] exploit the concave conjugate of the super-Gaussian prior and directly estimate the posterior distribution using VB to avoid the issues of the sparse MAP. Dong *et al.* [69] adopt a piecewise function to mimic the L0 norm around zero and to smooth out significant outliers, which is similar to the work of Xu *et al.* [70]. Chen *et al.* [71] who enhance the sparse prior by combining the L0 and L1 norm. Yang *et al.* [72] introduce a restarting technique to further improve the

performance of VB-based methods.

Some other conventional methods utilize properties of images to form priors. Michaeli and Irani [73] find that blur significantly decreases between cross-scale patches. Thus, they constrain the output by minimizing the dissimilarity between nearest-neighbor patches cross scales, which does not perform well when the image contains repetitive patterns. Lai *et al.* [74] observe that each local patch should contain two primary colors, and the distance between them should be maximized by deconvolution. Pan *et al.* [39] apply the dark channel prior to handle blind deconvolution and achieve good results. Yan *et al.* [40] combine the bright and the dark channel priors to overcome the limitation on bright dominant images. Ren *et al.* [75] derive an enhanced low-rank prior to reduce the number of non-zero singular values of the image. Pan *et al.* [76] exploit the phase-only image of a blurred image to estimate the start and end points of the blur kernel, which is efficient for linear motion. Bai *et al.* [77] utilize the downsampled blurred image as the prior and recover the latent sharp image from coarse to fine. Chen *et al.* [78] calculate the bright channel of the gradient maps for deblurring images without enough number of dark and bright pixels.

Deep-learning-based methods are also applied to the deblurring problem. Chakrabarti [79] trains a network to estimate the Fourier coefficients of blur kernels. Liu *et al.* [80] and Zhang *et al.* [81] exploit recursive filters to take advantage of context information. Generative adversarial networks (GANs) are also exploited to provide faster convergence and better visual quality of results [10, 20, 82]. Gong *et al.* [83] adopt a network to learn the motion flow. Xu *et al.* [84] develop a network to generate sharp gradient maps for kernel estimation. To enhance the network output, some utilize multi-stage strategies, e.g., multi-scale [21, 24, 85], multi-patch [11, 22, 45] and multi-temporal [86]. Asim *et al.* [87] adopt a well-trained sharp image generator to generate a sharp image closest to the blurred one. Tran *et al.* [88] develop a sharp image auto-encoder and a blur representation learning network, then two well-trained networks are fixed as a deep generative prior [87]. Li *et al.* [89] adopt a well-trained classifier,

which can distinguish between blurred and sharp images, as an extra constraint of the MAP framework, and optimize the model with the half-quadratic splitting method similar to that used in conventional methods.

Different from previous works [87–91] in which the priors need to be trained on external datasets, my proposed method is optimized with only one single blurred input image and the whole framework is optimized by gradient descent instead of conventional optimization-based methods [89]. Although Asim *et al.* [87] also provide a method optimized with a single image, the method degenerates to the DIP [51] with a sparse image prior and learnable inputs, which cannot avoid the problems of the sparse MAP. As well, none of the mentioned deep-learning-based methods consider the standard deviation of the image.

2.2 Deep Image Prior

Ulyanov *et al.* [50] introduce the concept of the deep image prior (DIP) that the structure of a randomly-initialized network can be used as an image prior for image restoration tasks. Ren *et al.* [51] adopt the DIP to implicitly learn the image prior and the kernel prior for blind image deconvolution. Early stopping with carefully chosen time, added random noise to the input and to the gradient with fixed noise level are applied to avoid the suboptimal solution of DIP [92]. Neural architecture search (NAS) can help to search for these hyper-parameters heuristically [93], but with the substantial increase in computational cost. Double-DIP [94] can handle the image separation problems, e.g., image segmentation, image dehazing, and transparency separation, but does not perform well for blind image deconvolution [51]. Some methods stabilize the optimization by adding extra priors to the loss function [95, 96]. However, this technique only works when the degradation kernel is known.

2.3 Variational Auto-encoder

Kingma *et al.* [97] introduce the concept of variational auto-encoder (VAE) for image generation. The goal is to learn a model that generates an image x given a sampled latent variable z, which can be formulated as $P(x|z) = P(x)P(z|x)/P(z)$, where $P(x)$ is constant. Since obtaining the true distribution of $P(z|x)$ is nontrivial, they utilize a Gaussian distribution $Q(z)$ to approximate $P(z|x)$ with a network to learn the expectation and the standard deviation. Thus, the target of VAE can be converted to minimizing the KL divergence between $Q(z)$ and $P(z|x)$. Vahdat *et al.* [98] further stabilize the training of VAE by partitioning the latent variables into groups. Similar to image generation, the target of image deconvolution is to learn a model to generate a blurred image I_b given a sampled latent sharp image I_s and a blur kernel k, and the distributions of $P(I_s|I_b)$ and $P(k|I_b)$ are learned by the network. Using predefined hand-crafted $P(I_s)$ and $P(k)$ can help to constrain the optimization.

2.4 Diffusion Models in 3D Domain

Inspired by the success of 2D image generation with diffusion models, researchers have attempted to utilize diffusion models to generate 3D objects in the form of various representations, such as point clouds [99–102], and neural fields [103, 104]. For example, Point·E [102] trains a diffusion model using a large synthetic 3D dataset to produce a 3D RGB point cloud conditioned on a synthesized single view from a text prompt. However, these works mainly focus on geometry generation and do not specifically tackle 3D texture synthesis. Yu *et al.* [105] train a diffusion model for mesh texture generation of specific object categories. Although Shap·E [106] is proposed to directly generate the parameters of implicit functions that can be rendered as both textured meshes and neural radiance fields, it cannot generalize to incorporate arbitrary text prompts. Moreover, the generated textures tend to be over smoothed and low quality when compared with that from the text to image model [15].

2.5 Lifting pre-trained 2D generative models to 3D

Initially, the process of distilling 3D objects from pre-trained 2D models has been enhanced by the development of joint text-image embedding, such as Contrastive Language-Image Pre-training (CLIP) [107]. For example, CLIP-Mesh [108] learns to generate a mesh with the guidance of CLIP text embedding and the corresponding image embedding of the diffusion model. However, since the CLIP guidance is rather sparse, the generated 3D models for CLIP-based approaches [109–111] are rather coarse.

Recently, researchers have leveraged large-scale 2D T2I diffusion models to distil individual 3D objects in the form of neural radiance fields. Among various distilling approaches, a dominant one is Score Distillation Sampling (SDS) [112]. SDS pioneered the approach with many follow-up [113–119]. For example, Magic3D [113] proposes a coarse-to-fine strategy to improve the quality of generated objects. Latent-NeRF [114] performs distillation in the latent space of latent diffusion model (LDM) [15]. A crucial drawback of this line of work is that SDS typically requires strong guidance, resulting in low diversity and over-saturation of the generated textures. ProlificDreamer [116] addresses this issue with a Variational Score Distillation (VSD) algorithm that adopts a particle-based variational inference to estimate the distribution of 3D scenes instead of a single point as in SDS. Yet, it still suffers from issues like blurry edges and color artefacts.

Texture Synthesis with Multiview Denoising. Instead of relying on the lengthy optimization of score distillation pipelines, an alternative research direction is directly leveraging the sampling process in diffusion models to synthesize UV textures. TEXTure [120] and Text2tex [121] adopt a depth-aware diffusion model [15] to progressively paint the mesh surface from different views and aggregate the images generated from the T2I model of sampled views into the texture map. While rich textures and

details can be faithfully synthesized, there are obvious seams on the aggregated texture map due to error accumulation in the process of the autoregressive view update. To further reduce view inconsistencies, TexFusion [122] interleaves texture aggregation with denoising steps in different camera views and maintains a latent texture map at each sampling step. To convert latent features to RGB textures, an intermediate neural color field is optimized on the decoding of 2D rendering of the latent texture which blurs the rich details [123]. In this thesis, the proposed approach distinguishes itself from previous methods with its ability to generate 3D-consistent textures while preserving rich details.

2.6 Spectral Reconstruction from RGB

Conventional methods: The spectral reflectance of a scene can be represented by a linear combination of several base spectra [124]. Conventional methods mainly focus on learning the base spectra and the corresponding representation coefficients [43, 44, 125–127]. For example, Arad and Ben-Shahar [43] create an over complete hyperspectral dictionary using K-SVD and learn the representation coefficients from the RGB counterpart. Fu *et al.* [44] first cluster the hyperspectral data and create a dictionary for each cluster, and the spectral reflectance of each pixel is learned from its nearest cluster. Jia *et al.* [126] utilize a low-dimensional manifold to represent the high-dimensional spectral data, which is able to learn a well-conditioned three-to-three mapping between a RGB vector and a 3D point in the embedded natural spectra. Akhtar and Mian [125] also cluster the spectral data but replace the dictionary with Gaussian processes.

DNN-based methods: Recently, DNN-based methods have dominated this area owing to the encouraging results of external learning [28–30, 128–135]. Shi *et al.* [128] stack multiple residual blocks or dense blocks for end-to-end spectral reconstruction. Lin *et al.* [28] separate the spectra into the sub-space and the null-space of the CSS for plausible reconstruction, where the sub-space component signifies the projection

of the spectra onto the CSS matrix, while the null-space component represents the remaining portion. Our approach builds upon this concept by extending it to the recovery of spectral reflectance in cases where the CSS is unknown. Zhang *et al.* [130] generate basis functions from different receptive fields and fuse them with learned pixel-wise weights. Sun *et al.* [29] estimate the spectral reflectance and the illumination spectrum simultaneously with a learnable IR-cut filter. Hang *et al.* [136] decompose the spectral bands into groups based on the correlation coefficients and estimate each group separately using a neural network. A self-supervised loss further constrains the reconstruction. Li *et al.* [30] exploit channel-wise attention to refine the degraded RGB images. Cai *et al.* [129] exploit the spectral-wise self-attention to capture inter-spectra correlations. Li *et al.* [137] learn a quantized diffractive optical element (DOE) to improve the hyperspectral imaging of RGB cameras. Zhang *et al.* [138] exploit the implicit neural representation that maps a spatial coordinate to the corresponding continuous spectrum using a multi-layer perceptron (MLP) whose parameters are generated from a convolution network. Some methods guide the reconstruction with a low-resolution hyperspectral image [139–141], which are different from the scope of this paper. All of the above mentioned methods do not considered the internal information from testing cases.

2.7 Meta-auxiliary Learning

In contrast to the term "meta-auxiliary learning (MAXL)" in image classification [142], which is designed to improve the generalization of classification models by using meta-learning [143] to discover optimal labels for auxiliary tasks without the need of manually-labelled auxiliary data [144], Chi *et al.* [49] redefine MAXL as a combination of model-agnostic meta-learning (MAML) [145] and auxiliary-learning (AL) [146] for test-time fast adaptation. In this thesis, the definition of the latter is used.

Model-agnostic meta-learning (MAML): The aim of MAML is to train models capable of fast adaptation to a new task with only a few steps of gradient descent,

which can be applied to few-shot learning [147]. Park *et al.* [148] and Soh *et al.* [149] adopt the MAML for super-resolution. They first initialize the model by training on external datasets like other DNN-based super resolution methods [150], then conduct MAML to further optimize the model for unseen kernels. During testing, a low resolution input and its down-scaled version are represented as a new training pair to fine-tune the model. Although the targets are similar (spatial/spectral upsampling), directly applying the MAML to SRR is infeasible because the three RGB channels of an image cannot be further downsampled.

Auxiliary-learning (AL): AL is to assist the optimization of primary tasks with at least one auxiliary task for better generalization and performance. Guo *et al.* [151] reconstruct low resolution images for real-world super resolution. Valada *et al.* [152] learn to estimate visual odometry and global pose simultaneously for higher efficiency. Lu *et al.* [153] solve the depth completion problem with image reconstruction to extract more semantic cues. AL can also stabilize the training of GAN for image synthesis [154]. Sun *et al.* [146] choose the rotation prediction as the auxiliary task to update pre-trained parameters for test-time adaptation. Nevertheless, simply updating the pre-trained parameters with only auxiliary tasks may result in catastrophic forgetting [49], where the model exhibits overfitting in the auxiliary tasks, leading to a loss of previously acquired knowledge from the primary task during the training process.

To leverage both the MAML and AL, we follow the strategy of Chi *et al.* [49] using self-supervised RGB reconstruction as the auxiliary task and MAML to avoid catastrophic forgetting. The auxiliary task also avoids downsampling RGB images to generate training pairs for fine-tuning.

2.8 Transparent Object/Glass Recognition

Methods using RGB only: Traditional algorithms detect glass by analyzing local edge/region characteristics, which exhibit issues in the wild [155–158]. Since the

recent progress of deep learning in computer vision, researchers start to collect large-scale transparent object datasets [56, 57, 159, 160] and train their proposed neural networks for glass-like object detection and segmentation from a single RGB image, where contextual feature [56, 161] and boundary supervision [58, 159, 162] are both popular ways for boosting accuracy. Multi-task learning also has been adopted for transparent object detection and segmentation, where object segmentation is jointly tackled with other related problems such as refractive flow estimation [163], reflection detection [57] and scene understanding [55].

Methods using modalities beyond RGB: The seemingly simple visual appearance of the glass is deceptive because its appearance interacts with the environment, which motivates the use of more reliable sensors other than RGB cameras. Existing methods have utilized refractive distortions captured by a light-field camera [60] and the high contrast of edges in a polarization image [61] for transparent object segmentation. RGB-Depth (RGB-D) fusion has been used in both traditional optimization methods [59] and recent learning-based methods [53, 164]. However, depth cameras suffer from severe sensor failures for transparent surfaces due to light refraction. A backlight with AprilTag is employed to enhance RGB-D 3D scanning [165]. In this thesis, a low-cost but robust solution by integrating RGB and thermal images is adopted.

2.9 Salient Object Detection

Salient object detection (SOD) aims to segment the most prominent object in a given scene. Early methods [166–168] heavily rely on hand-crafted features from a single RGB image. Recent data-driven methods [18, 169–172] have dominated this field. Integrating RGB and depth images has significantly improved the performance of SOD methods. Among them, direct concatenation [173–176], addition [177], spatial/channel attention [178–182], prediction guidance [183], affine transformation [184, 185], message passing [186], mutual information minimization [187], and self/cross atten-

tion [188–190] have been utilized for feature fusion.

In addition to depth images, thermal images are also exploited to compensate RGB images for SOD. Tu *et al.* [191] regard deep features of superpixels as graph nodes to cluster the foreground and background with collaborative graph learning. Zhang *et al.* [192] learn to generate spatial attention mask for modality fusion of multi-scale features. Zhou *et al.* [193] utilize different dilation rates to extract features from two modalities and combine spatial and channel attention for modality fusion. Zhou *et al.* [194] adopt a three-branch architecture to generate salient masks from RGB, thermal and fusion features separately and use the weighted summation of three masks as the final results. Sun *et al.* [195] utilize sine-cosine functions to extract features from two modalities. This paper focuses on extracting binary masks for glass to tackle the same task of binary segmentation as in SOD. Wu *et al.* [196] exploit channel attention weights learned from one modality to enhance the other which can better complement features from two modalities. Tu *et al.* [197] propose a modality alignment module for weakly alignment-free RGB-T image pairs.

2.10 RGB-T Fusion Applications

RGB-T image pairs are widely used in many vision tasks to compensate the low-quality of RGB images under poor illumination or occlusion, such as object tracking [198–204], moving object detection [205–208], face recognition [209], semantic segmentation [210–216], scene understanding [217–221], crowd counting [222–225] and salient object detection [226–233]. To the best of my knowledge, the proposed method is the first RGB-T method for glass segmentation. As well, I also collected the first dataset for such an application.

Chapter 3

Blind Non-Uniform Motion Deblurring

When we are taking photos using a camera, especially the one on a mobile device, non-uniform motion blur is one of the most common types of undesirable artifacts caused by object motion and camera shake [83]. Removing such blur to recover the original sharp image plays a critical role in many high-level vision tasks, e.g. computational photography [234], image classification [235], object detection [236], and face recognition [237], because motion blur severely degrades the image quality. In this chapter, I propose a new *Atrous Spatial Pyramid Deformable Convolution* (ASPDC) module for region-specific convolution and for integrating features from different sizes of receptive fields, which is more suitable for non-uniform deblurring. I also propose a new reblurring network to reblur the output, which is helpful in constraining the solution space [151] of deblurring with a new deblurring-reblurring consistency loss. Note that the reblurring network is used during training only, which needs both of the blurred image and the corresponding sharp (deblurred) image. Extensive experimental results demonstrate the effectiveness of the proposed method compared to that of other SOTA methods on the benchmark datasets.

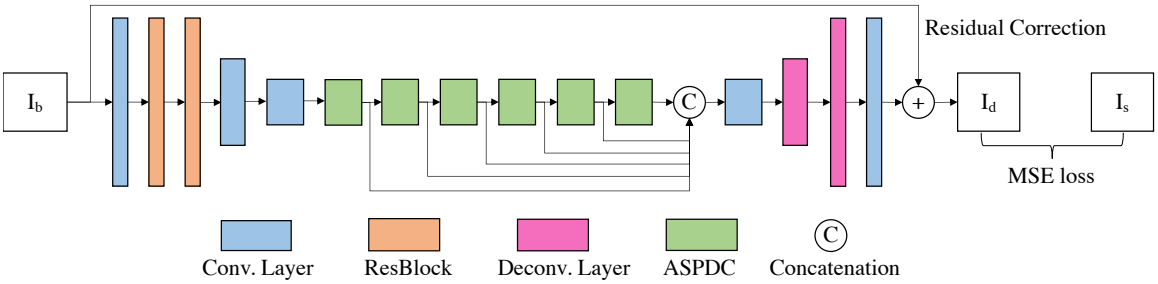


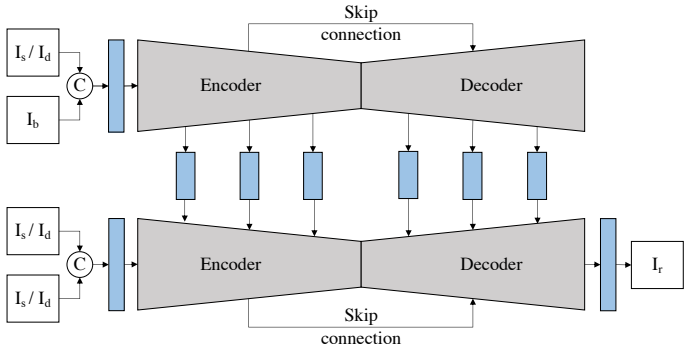
Figure 3.1: Overview of the deblurring network architecture.

3.1 Proposed Method

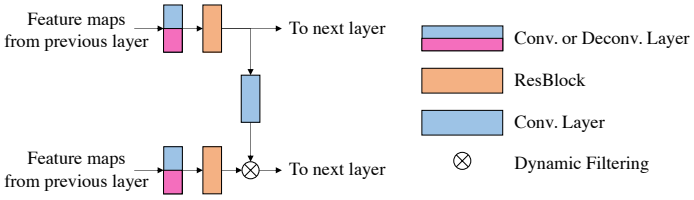
3.1.1 Overview

An overviews of the proposed deblurring and reblurring network architectures are illustrated in Figure 3.1 and Figure 3.2, respectively. To make training more stable, the two networks are trained separately. The deblurring network attempts to recover the sharp image I_s from the blurred input I_b, then the deblurred output I_d is reblurred by the reblurring network. The reblurring network takes both I_s and I_b as input and outputs the reblurred image I_r. When the training of the two networks converges, I replace the input I_s of the reblurring network with the deblurred output I_d, and fine-tune the I_d with the deblurring-reblurring consistency loss. I do not use any kind of normalization (e.g. batch normalization [238] or instance normalization [239]).

Our deblurring-reblurring consistency is inspired by Guo *et al.* [151]. Deblurring and reblurring can be regarded as a pair of dual tasks. The former is the primary task and the latter the corresponding dual task, which corresponds to upsampling and downsampling in super-resolution [33]. Guo *et al.* [151] prove that the generalization bound of the dual regression (in our case, consistency) is lower than that of the primary regression (only deblurring). Therefore, it leads to more accurate deblurring results. However, simply mapping the deblurred output to the original blurred input is highly ill-posed. Lu *et al.* [240] show that the sharp image only contains the sharp content without any blur information, so it needs the blur information from the corresponding blurred image as the extra input for reblurring. Thus, we also utilize



(a) Reblurring network



(b) Feed blurred information from the upper to lower branch

Figure 3.2: Overview of the reblurring network architecture.

blur information.

3.1.2 Deblurring Network

We use two residual blocks (ResBlocks) [2] and strided convolution layers to extract high dimensional features at the beginning, and two deconvolution layers to recover the spatial dimension in the end. The last convolution layer reduces the channel size of the feature map to 3 (RGB). We find that learning the residual correction instead of directly learning the latent sharp image can make the training more stable and faster. To explain our intuition, note that the blurred image contains all of the signals from the sensor during the exposure time. One of the sharp images $I_{S(t)}$, say at time t^*, is the corresponding target while many features extracted from I_b could be from times other than t^*, which can be formulated as:

$$I_b = g\left(\frac{1}{T}\int_{t=0}^{T} I_{S(t)}dt\right), \tag{3.1}$$

in which I_b is the blurred image of the dynamic scene, T is the period of the exposure time, $I_{S(t)}$ is the sharp snapshot at timestamp t and $g()$ represents the Camera

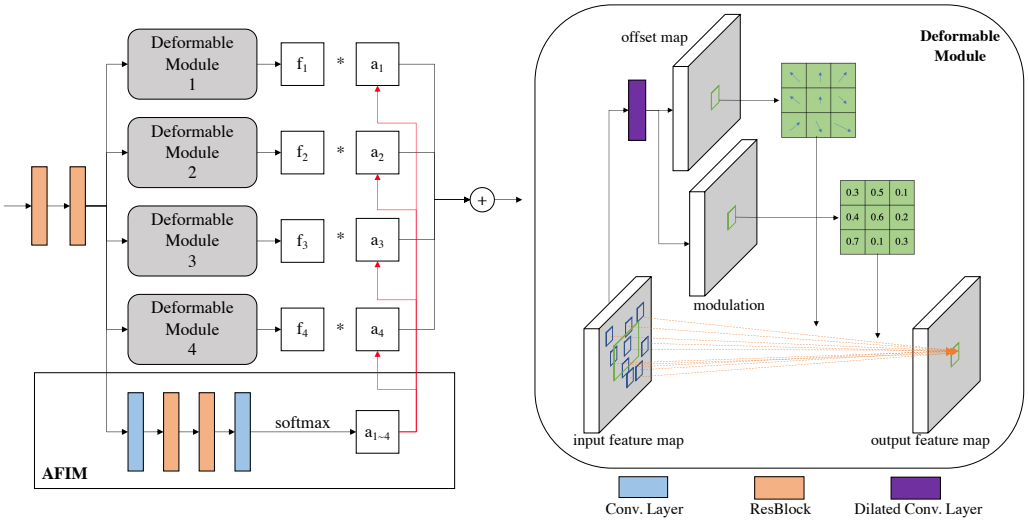


Figure 3.3: Schematic of the ASPDC module.

Response Function (CRF). Hence, learning features of the other times will be easier than directly learning a specific one.

In the middle of the deblurring network, we stack six ASPDC modules (as shown in Figure 3.3) in which we extend the work of the deformable convolution network v2 (DCNv2) [241]. The original DCNv2 applies different convolution kernels to different regions by learning an offset map Δp and a modulation Δm. But a fixed size is used for the receptive field of each region used to generate Δp and Δm. Since Δp represents the shift of each pixel, it can be regarded as the local optical flow [242] corresponding to the motion of the object and the camera. For a non-uniform blurred image, some of the regions might have only small variations while other regions might have large movements and overlaps. In this case, the original DCNv2 uses a single convolution layer to generate Δp and Δm and treats these regions similarly, which is not an optimal choice for this problem.

To make deformable convolution more flexible, in our ASPDC module, we build four branches with different dilation rates [243] to generate four offset maps Δp and modulations Δm with different receptive fields, and four deformable convolution outputs. As shown in Figure 3.3, the dilation rates of dilated convolution layers in

deformable modules 2∼4 are 1, 2, and 4, respectively. Deformable module 1 also uses the dilation rate 1 but it ignores the offsets (by setting Δp as zero). Such a special module is used to recover static regions.

The outputs of four branches in an ASPDC module are fused by an attention feature integration module (AFIM) [244]. We can write it as:

$$f_o = \sum_{i=1}^{4} a_i * f_i, \tag{3.2}$$

$$\sum_{i=1}^{4} a_{ij} = 1, 1 \leq j \leq h \times w, \tag{3.3}$$

in which f_i is the output of the i^{th} branch, a_i is a single-channel attention map generated from the AFIM, j is the index of the pixel, h and w are, respectively, the height and width of the attention map, $*$ represents the element-wise multiplication and f_o is the output of the ASPDC module. To make sure that the channel-wise sum of attention maps is 1, we utilize the softmax activation function. In this case, each region that integrates information from different receptive fields significantly boosts the performance. The output feature maps of six ASPDC modules are further concatenated to stabilize training.

We use the Mean Squared Error (MSE) loss as our final deblurring loss:

$$L_{deblurring} = ||I_s - I_d||_F^2, \tag{3.4}$$

where I_s and I_d are the sharp target and the deblurred output, respectively.

3.1.3 Reblurring Network

In order to narrow down the solution space of deblurring and to refine the deblurred output I_d, we build an end-to-end reblurring network to reblur the deblurred output and calculate the deblurring-reblurring consistency loss. Simply mapping the sharp image back to the blurred image is not impossible but difficult, because the non-uniform blurred image domain is much larger than the sharp image domain. Hence,

we need the blur information from the blurred image to assist the mapping. However, directly inputting sharp and blurred images together and outputting the reblurred image is difficult to train the model, since the training procedure is unstable and easy to collapse to an identity mapping. The network may choose to output the blurred input directly and ignore the sharp input, which is definitely undesirable.

To handle the above problem, we utilize an architecture which is able to take full use of the blur information and avoid training collapse. As shown in Figure 3.2a, the network contains two encoder-decoder branches, and the weights of the two branches are shared for reducing the number of parameters. The architecture of the encoder-decoder is simple, which consists of multiple conv/deconv-resblock pairs (convolution layers for the encoder and deconvolution layers for the decoder) as shown in Figure 3.2b. The concatenation of blurred and sharp image is used as the input of the upper branch, and two duplicate sharp images are input into the lower branch for matching the channel dimension.

The upper branch learns to compare the blurred and the sharp image and passes feature maps with blur information to the lower branch after each conv/deconv-resblock pair. In the lower branch, a convolution layer reduces the channels of feature maps from the upper branch to generate a $K \times K$ dynamic local filter [245] for each pixel. For reducing the computational cost of dynamic filtering, we set K as 3 and apply the same filter to all the channels of feature maps from the lower branch. The dynamic local filters are regarded as the spatial-variant blur kernels which gradually reblur the feature maps of the lower branch from beginning to end. The detailed architecture of the reblurring network is shown in Appendix A.

Similar to the deblurring network, we use the Mean Squared Error (MSE) loss here:

$$L_{reblurring} = ||I_r - I_b||_F^2. \tag{3.5}$$

3.1.4 Fine-tuning

After the training of the deblurring and reblurring networks converges, we replace I_s in the reblurring network with the deblurred output I_d to refine I_d by the deblurring-reblurring consistency loss. The loss function is defined as:

$$L_{consistency} = L_{deblurring} + \lambda L_{reblurring}, \tag{3.6}$$

where λ is the weight of the reblurring loss, and we empirically set $\lambda = 0.1$.

3.2 Experiments

3.2.1 Datasets

We follow the literature [10, 20, 22, 24, 45] to train our model on 2103 training images from the GoPro dataset [24]. We then use 1111 testing images from the GoPro dataset and 2025 testing images from the HIDE dataset [246] as our testing set. We also do qualitative comparison on the Real World Blurred Image (RWBI) dataset [247].

3.2.2 Implementation Details

The method is implemented in PyTorch [248] and evaluated on a single NVIDIA RTX 2080 Ti GPU with 11 GB of memory. During training, we use Adam optimizer [249] with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-18}$. All parameters are initialized using Xavier normalization [250]. We randomly crop the training images into 256×256 patch pairs and set the batch size as 6. The learning rate is initialized as 10^{-4} and halved every 1000 epochs. The training procedure is terminated when the learning rate reaches 10^{-6}. During fine-tuning, we set the learning rate as 10^{-5} and it is halved every 200 epochs. We stop the fine-tuning when the learning rate reaches 10^{-6}. The size of all convolution filters is 3×3. We set the initial number of channels for all convolution layers and residual blocks to 32 in the deblurring network and 16 in the reblurring network, and we double (halve) them every time we downscale (upscale) the spatial dimension.

Method	Xu [70]	Sun [251]	Nah [24]	Kupyn [10]	Tao [21]	Zhang [81]	Kupyn [20]	Aljadaan [252]
PSNR	20.30	25.31	28.49	28.70	30.26	29.19	29.55	30.35
SSIM	0.741	0.851	0.917	0.927	0.934	0.931	0.934	0.961
Method	Zhang [22]	Suin [45]	Park [253]	Yuan [242]	Purohit [254]	Zhang [247]	Ours	Ours+
PSNR	31.20	32.02	31.15	29.81	31.76	31.10	31.97	32.09
SSIM	0.945	0.953	0.945	0.937	0.953	0.942	0.957	0.959

Table 3.1: Quantitative comparison on the GoPro dataset [24]. Ours/Ours+ represents our deblurring network without/with fine-tuning on the reblurring network. The best results are in red and the second best in blue.

Method	Kupyn [10]	Tao [21]	Zhang [22]	Park [253]	Suin [45]	Shen [246]	Kupyn [20]	Ours	Ours+
PSNR	24.51	28.36	29.09	29.16	29.98	28.89	26.61	29.98	30.04
SSIM	0.871	0.915	0.924	0.933	0.930	0.930	0.875	0.944	0.945

Table 3.2: Quantitative comparison on the HIDE dataset [246]. Ours/Ours+ represents our deblurring network without/with fine-tuning on the reblurring network. The best results are in red and the second best in blue.

3.2.3 Quantitative Comparison

We first compare our method ("ours+") with others on the 1111 testing images from the GoPro dataset, including a conventional method (Xu *et al.* [70]), and several deep learning based methods (Sun *et al.* [251], Nah *et al.* [24], Kupyn *et al.* [10], Tao *et al.* [21], Zhang *et al.* [81], Kupyn *et al.* [20], Aljadaany *et al.* [252], Zhang *et al.* [22], Suin *et al.* [45], Park *et al.* [253], Yuan *et al.* [242], Purohit *et al.* [254] and Zhang *et al.* [247]). We also evaluate our deblurring network without fine-tuning on the reblurring network ("ours"). We use PSNR [255] and SSIM [256] as evaluation metrics. All the methods are trained on the GoPro dataset following the same strategy.

The results are shown in Table 3.1. Our method outperforms most of the existing SOTA methods even without fine-tuning, and fine-tuning on the reblurring network can further improve the performance. In terms of PSNR, Ours+ is ranked first and is 0.07db better than the second [45]. Although our SSIM is slightly lower than the first [252], our PSNR far surpasses it which demonstrates that our method is good at both evaluation metrics. Note that we use the mean squared error loss without a sophisticated GAN [257] and still achieve good performance.

We further compare with some of the methods on the HIDE dataset in Table 3.2.

λ	0.01	0.1	1
PSNR	32.17	32.22	32.15
SSIM	0.960	0.960	0.959

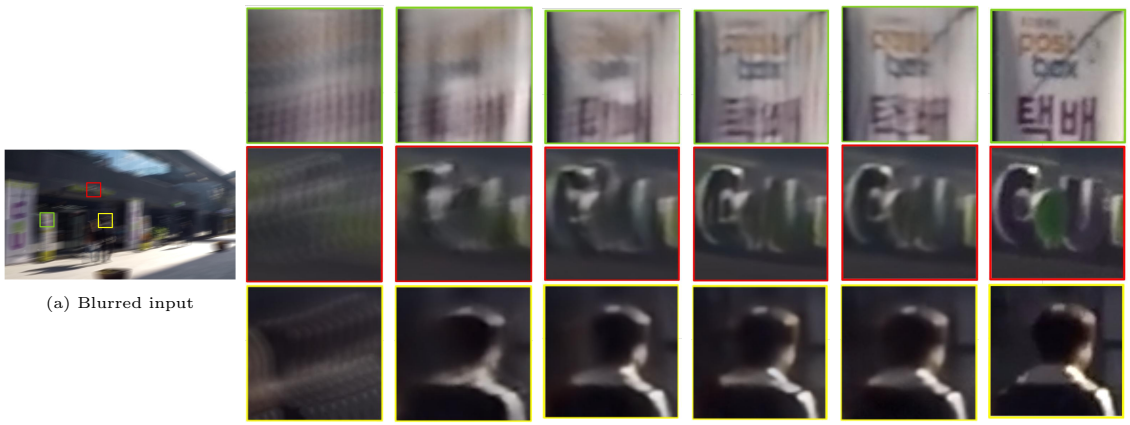
Table 3.3: Performance of fine-tuning with different λ.

Method	Kupyn [20]	Zhang [22]	Nah [24]
Time (sec)	1.68	0.40	0.93
GPU (GB)	2.41	2.10	9.70
Method	Tao [21]	Park [253]	Ours
Time (sec)	0.78	0.05	0.28
GPU (GB)	6.09	8.49	2.25

Table 3.4: Average testing time and GPU usage of images of size 1280×720 on a single NVIDIA RTX 2080 Ti GPU.

Ours+ remains the top and Ours is ranked second. Note that unlike all other methods that follow the same strategy of training on the GoPro dataset but tested on the HIDE dataset, the method of Shen *et al.* [246] is trained on the HIDE dataset directly but it cannot perform better than our proposed method.

The average testing time and GPU memory usage on the GoPro dataset is reported in Table 3.4. Although the testing time of Park *et al.* [253] is the lowest, its GPU memory usage is almost four times of ours, and its performance (Table 3.1) is lower than ours. Our method is 30% faster than Zhang *et al.* [22] while only increasing the GPU memory usage by 7%. Since the architecture of Kupyn *et al.* [20] is much deeper and wider than that of other listed methods, its testing time is the highest even if it uses only a single stage. Our proposed method is a good trade-off between performance and efficiency in both memory and computation.



(a) Blurred input (b) Blurred (c) Park[253] (d) Zhang[22] (e) Prohit[254] (f) Ours+ (g) Sharp GT

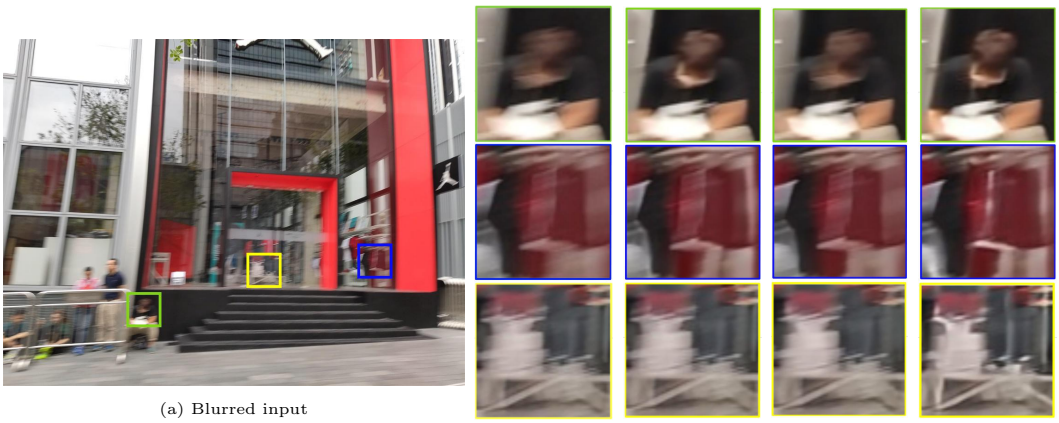
Figure 3.4: Qualitative comparison on the GoPro dataset: a) Blurred input image. b-g) Magnified crops of the blurred input and deblurred outputs of compared methods, and the sharp ground truth.

(a) Blurred input (b) Park [253] (c) Zhang [22] (d) Ours+ (e) Sharp GT

Figure 3.5: Qualitative comparison on the HIDE dataset: a) Blurred input image. b-g) Magnified crops of the blurred input, deblurred outputs of compared methods, and the sharp ground truth.

3.2.4 Qualitative Comparison

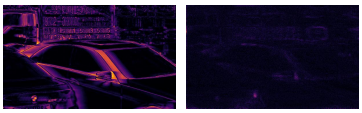
Following a similar strategy as in the quantitative comparison, we first compare our method against others on the testing images of the GoPro dataset [24]. We compare with the two best performing and most recent methods [22, 253] with published well-trained models, along with the published results of Purohit *et al.* [254] on the same dataset. As apparent in Figure 3.4, the output of our method is most similar to the ground truth sharp image, comparing to others. The alphabets written on the banner of the first row are almost clearly visible for ours, while others fail to deblur them correctly. Similarly, the sign over the shop is best deblurred by ours, while the result of Purohit *et al.* [254] is also reasonably good. On the third row, our method restores the skin and bright colors better. We also use the well-trained models on the GoPro dataset to test on the HIDE dataset [246]. As shown in Figure 3.5, our method does



(a) Blurred input (b) Blurred (c) Park[253] (d) Zhang[22] (e) Ours+

Figure 3.6: Qualitative comparison on the RWBI dataset: a) The blurred input image. b-e) Magnified crops of the blurred input and the deblurred outputs of compared methods. Note that no ground truth is available for RWBI.

(a) Sharp (b) Reblurred (c) Blurred



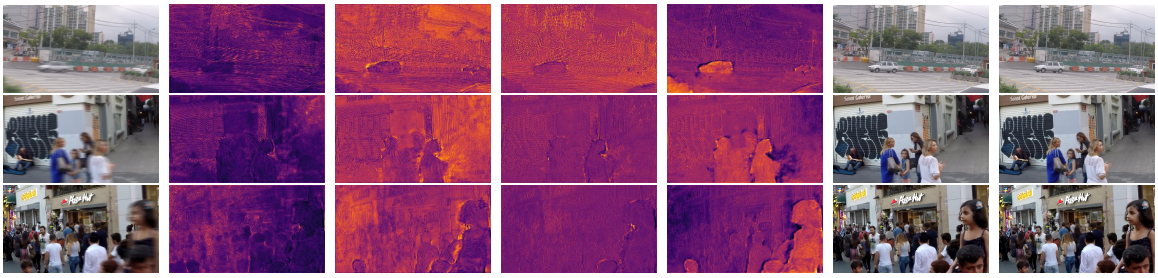
(d) Diff-S (e) Diff-R

Figure 3.7: Comparison of a reblurred output with its corresponding blurred and sharp images. Diff-S (Diff-R) represents the difference map between the sharp image (reblurred output) and the blurred ground truth.

a good job deblurring the bicycles in the image. We further test the same models on the RWBI dataset [247]. As seen in Figure 3.6, our method generally outperforms the compared methods in terms of deblurring quality. For a more complete qualitative study, please refer to Appendix A.

PSNR	SSIM	Mean	variance
55.71	0.9997	0.18	0.22

Table 3.5: Performance evaluation of reblurring network on the GoPro dataset.



(a) Blurred (b) a_1 (c) a_2 (d) a_3 (e) a_4 (f) Deblurred (g) Sharp

Figure 3.8: Attention maps of the last ASPDC module. Values are within [0, 1] and the brighter the higher.

Version	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
Module 1	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Module 2		✓			✓	✓		×3			✓	✓
Module 3			✓		✓		✓		×3		✓	✓
Module 4				✓		✓	✓			×3	✓	✓
AFIM		✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
PSNR	30.24	30.65	30.94	30.85	31.82	31.73	31.53	31.83	31.93	31.72	31.16	32.12
SSIM	0.942	0.944	0.948	0.947	0.957	0.956	0.954	0.957	0.958	0.955	0.950	0.959

Table 3.6: Performance of different versions of the ASPDC module.

3.2.5 Reblurring Evaluation

In addition to evaluating the deblurring network, we evaluate the performance of the reblurring network, which is critical for fine-tuning. As shown in Table 3.5, PSNR and SSIM of reblurred images are quite high, and the mean and variance of the difference map $|I_r - I_b|$ are almost 0. The experimental results illustrate that reblurred images are quite close to the original blurred images.

One of the reblurred outputs is shown in Figure 3.7 with the original sharp and blurred images. As we can see, the difference between the reblurred and blurred images is small enough and negligible.

3.2.6 Ablation Studies

To evaluate the effectiveness of each component in the ASPDC module, we compare multiple versions of it. We randomly select 200 testing images from the GoPro dataset for validation. All versions are trained with the mean squared error loss only without fine-tuning. With regard to the performance and efficiency trade-off, we find having

four ASPDC modules is optimal.

The experimental results are shown in Table 3.6. Version 1 has a deformable module 1 without the offset Δp, which is used as our baseline. The $\times 3$ in Version 8$\sim$10 represents three duplicated modules (the same dilation rate but no parameter-sharing). As we can see, the deformable module 2$\sim$4 is critical for improving performance, especially module 3. Simply duplicating modules cannot get results as good as combining different modules. It demonstrates that fusing information from different sizes of receptive fields is meaningful and justified. Version 11 demonstrates that features from different receptive fields should be fused properly. We visualize the attention maps of the last (sixth) ASPDC module in Figure 3.8. It shows that the attention maps of small receptive fields (a_1 and a_2) focus more on static objects or objects with small movements, while the attention maps of large receptive fields (especially a_4) pay more attention to objects with large movements.

For the choice of the hyperparameter λ in Eqn 3.6, we evaluate values in the range of 0.01 to 1 in Table 3.3. As we can see, the deblurring term is overwhelmed by the reblurring term when the value of λ is too large. Inversely, a small value of λ, such as 0.01, limits the effect of the reblurring term on the improvement of performance. To fully utilize the reblurring term, we set $\lambda = 0.1$ in all our experiments.

3.3 Summary

In this chapter, I propose to reduce the domain gap between the training and testing datasets by adopting adaptive convolutions. However, the model still needs training on paired synthetic data. In the next chapter, I solve this problem using deep image prior (DIP) which needs a single degraded image only for the optimization of the network parameters. By utilizing the DIP, I avoid the domain gap between the training and testing datasets especially when the testing data is quite different from the synthetic training data.

Chapter 4

Blind Image Deconvolution Using Variational Deep Image Prior

Blind image deconvolution is aimed at recovering the latent sharp image based on a single blurred image without knowing the blur kernel. When the blur kernel is spatially invariant, it can be modeled as

$$I_b = k \otimes I_s + n, \tag{4.1}$$

where I_b denotes the blurred image, k the blur kernel, $\otimes$ the convolution operator, I_s the latent sharp image and n the additive noise. Most conventional methods utilize maximum a posteriori (MAP) to alternatively solve for k and I_s, which is formulated as

$$\arg\max_{I_s,k} P(I_s, k|I_b) = \arg\max_{I_s,k} P(I_b|I_s, k)P(I_s)P(k) \tag{4.2}$$

where $P(I_b|I_s, k)$ is the likelihood term, $P(I_s)$ and $P(k)$ are the prior distributions of the latent sharp image and the blur kernel, respectively. However, MAP suffers from the problem of trivial solutions. In this chapter, I adopt variational Bayes-based (VB) methods to the deep image prior (DIP), so that not only the optimization is constrained but also the problems of the MAP can be avoided. Conventional VB-based methods [62, 68, 72] utilize a trivial (e.g., Gaussian) distribution to directly approximate the posterior distribution (the left term of Eqn. 4.2) by minimizing the Kullback–Leibler (KL) divergence [258] instead of using MAP. Although the accurate

posterior distribution is hard to obtain, the approximated one is good enough and much more robust than the result of MAP. In order to combine the DIP with VB, I propose a new variational deep image prior (VDIP) to learn the distributions of all latent variables (sharp images and blur kernels) which is motivated by the idea of variational auto-encoder [97]. The experimental results show that the proposed VDIP can significantly improve over the DIP in both quantitative results on benchmark datasets and the quality of the generated sharp images.

4.1 Proposed Method

In this section, we provide the mathematical analysis of the feasibility of our proposed methods. More derivation details are given in Appendix B.

4.1.1 Super-Gaussian Distribution

Conventional image priors can be formulated as a super-Gaussian distribution:

$$P(I_s) = W \exp\left(-\frac{\rho(F_x(I_s)) + \rho(F_y(I_s))}{2}\right), \tag{4.3}$$

where W is the normalization coefficient, and $\rho()$ is the penalty function to constrain the sparsity of $F_x(I_s)$ and $F_y(I_s)$. For sparse image priors, $F_x()$ and $F_y()$ are gradient kernels $[-1, 1]^T$ and $[-1, 1]$. When $\rho()$ is quadratic, $P(I_s)$ degenerates to a Gaussian distribution. Since $\rho(\sqrt{x})$ has to be increasing and concave for $x \in (0, \infty)$ when x follows the super-Gaussian distribution [259], we can decouple $\rho()$ and I_s using the concave conjugate of $\rho(\sqrt{F_x(I_s)})$ and $\rho(\sqrt{F_y(I_s)})$ following the strategy of Babacan *et al.* [68], and the upper bound of $\rho(F_x(I_s))$ and of $\rho(F_y(I_s))$ are represented as

$$\begin{aligned} \rho(F_x(I_s)) &\leq \frac{1}{2}\xi_x(F_x(I_s))^2 - \rho^*\left(\frac{1}{2}\xi_x\right), \\ \rho(F_y(I_s)) &\leq \frac{1}{2}\xi_y(F_y(I_s))^2 - \rho^*\left(\frac{1}{2}\xi_y\right), \end{aligned} \tag{4.4}$$

where $\rho^*(\frac{1}{2}\xi_x)$ and $\rho^*(\frac{1}{2}\xi_y)$ denote the concave conjugates of $\rho(\sqrt{F_x(I_s)})$ and $\rho(\sqrt{F_y(I_s)})$, respectively, and ξ_x and ξ_y are the variational parameters. We replace $\rho(F_x(I_s))$ and

$\rho(F_y(I_s))$ in Eqn. 4.3 with their upper bounds in $P(I_s)$

$$\begin{aligned} P(I_s) \geq & W \exp\left(-\frac{\xi_x(F_x(I_s))^2 + \xi_y(F_y(I_s))^2}{4}\right) \\ & \cdot \exp\left(\frac{\rho^*(\frac{1}{2}\xi_x) + \rho^*(\frac{1}{2}\xi_y)}{2}\right). \end{aligned} \tag{4.5}$$

Since the right-hand side of each inequality in Eqn. 4.4 is a convex quadratic function with a single global minimum, by calculating the derivative with respect to $F_x(I_s)$ and to $F_y(I_s)$, respectively, in Eqn. 4.4, equality is attained when

$$\xi_x = \frac{\rho'(F_x(I_s))}{|F_x(I_s)|}, \xi_y = \frac{\rho'(F_y(I_s))}{|F_y(I_s)|}, \tag{4.6}$$

where $\rho'()$ is the derivative of $\rho()$. As shown in Eqn. 4.5, irrespective of the form of $\rho()$, $P(I_s|\xi_x, \xi_y)$ becomes a trivial Gaussian distribution when equality is attained, which simplifies the derivation and the implementation because other penalty functions are discontinuous and the integral is too complicated to obtain (*e.g.*, $|x|$, $\ln|x|$). Besides, a Gaussian distribution is usually utilized to approximate the real distribution in VB-based methods, and the multiplication of two Gaussian distributions is much easier to calculate.

4.1.2 Variational Inference

Due to the extra variational parameters ξ_x and ξ_y, the problem can be reformulated as

$$\arg\max_{I_s,k,\xi_x,\xi_y} P(I_s, k, \xi_x, \xi_y|I_b) = \arg\max_{I_s,k,\xi_x,\xi_y} \frac{P(I_b|I_s, k)P(I_s|\xi_x, \xi_y)P(\xi_x, \xi_y)P(k)}{P(I_b)}. \tag{4.7}$$

Directly calculating $P(I_s, k, \xi_x, \xi_y|I_b)$ is challenging because the true distribution of I_b is difficult to obtain. The most common strategy is to use MAP, which estimates the posterior distribution by maximizing it as shown in Eqn. 4.7. However, MAP with the sparse image prior favors a trivial solution. An alternative strategy is to use VB, which uses a trivial distribution $Q(I_s, k, \xi_x, \xi_y)$ (e.g., Gaussian) to approximate the posterior distribution $P(I_s, k, \xi_x, \xi_y|I_b)$ by minimizing the KL divergence between

these two distributions, which can be written as

$$\begin{aligned} & D_{KL}(Q(I_s, k, \xi_x, \xi_y)||P(I_s, k, \xi_x, \xi_y|I_b)) \\ =& \ln P(I_b) - \int Q(I_s, k, \xi_x, \xi_y) \ln \frac{P(I_s, k, \xi_x, \xi_y, I_b)}{Q(I_s, k, \xi_x, \xi_y)} dI_s dk d\xi_x d\xi_y \\ =& \ln P(I_b) - L(I_s, k, \xi_x, \xi_y, I_b), \end{aligned} \tag{4.8}$$

where D_{KL} represents the KL divergence, and $L(I_s, k, \xi_x, \xi_y, I_b)$ is the variational lower bound. Since $\ln P(I_b)$ is constant and D_{KL} is non-negative, minimizing D_{KL} is equivalent to maximizing $L(I_s, k, \xi_x, \xi_y, I_b)$. By assuming that I_s and k are independent, the variational lower bound can be rewritten as

$$\begin{aligned} & L(I_s, k, \xi_x, \xi_y, I_b) \\ =& \int Q(k) \ln \frac{P(k)}{Q(k)} dk - \int Q(I_s) \ln Q(I_s) dI_s \\ & + \int Q(I_s) Q(\xi_x, \xi_y) \ln P(I_s|\xi_x, \xi_y) dI_s d\xi_x d\xi_y \\ & + \int Q(\xi_x, \xi_y) \ln \frac{P(\xi_x, \xi_y)}{Q(\xi_x, \xi_y)} d\xi_x d\xi_y + E_{Q(I_s,k)} \left[\ln P(I_b|I_s, k)\right], \end{aligned} \tag{4.9}$$

where $P(I_s|\xi_x, \xi_y)$ can be obtained from Eqn. 4.5, $P(k)$ is set as the standard Gaussian distribution $\mathcal{N}(0,\, I)$. Based on the mean field theory [68, 260], it is more convenient to simply assume that pixels on images and kernels are all independent. We can

further rewrite Eqn. 4.9 as

$$
\begin{aligned}
& L(I_s, k, \xi_x, \xi_y, I_b) \\
=& \frac{1}{2}\sum_{i=1}^{I}\sum_{j=1}^{J}(2\ln S(k(i,j)) - E^2(k(i,j)) - S^2(k(i,j))) \\
& + \frac{1}{2}\sum_{m=1}^{M}\sum_{n=1}^{N} 2\ln S(I_s(m,n)) \\
& - \frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N} E((F_x(I_s)(m,n))^2)E(\xi_x(m,n)) \\
& - \frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N} E((F_y(I_s)(m,n))^2)E(\xi_y(m,n)) \\
& + E_{Q(I_s,k)}\left[\ln P(I_b|I_s,k)\right] \\
& + \int Q(\xi_x,\xi_y)\ln\frac{P(\xi_x,\xi_y)}{Q(\xi_x,\xi_y)}d\xi_x d\xi_y \\
& + \frac{1}{2}\int Q(\xi_x,\xi_y)(\rho^*(\frac{1}{2}\xi_x) + \rho^*(\frac{1}{2}\xi_y))d\xi_x d\xi_y \\
& + Constant,
\end{aligned}
\tag{4.10}
$$

where $S()$ and $E()$ denote the standard deviation and the expectation, respectively, of distribution $Q()$, (i,j) is the pixel index of k, (m,n) is the pixel index of I_s and ξ. Since only the expectation of ξ_x and ξ_y are related to I_s, we do not need to consider their distributions so that the last three rows in Eqn. 4.10 can be ignored. Following Babacan *et al.* [68], $E(\xi_x)$ and $E(\xi_y)$ can be simply calculated by

$$
\begin{aligned}
E(\xi_x(m,n)) &= \frac{\rho'(v_x(m,n))}{v_x(m,n)}, \\
E(\xi_y(m,n)) &= \frac{\rho'(v_y(m,n))}{v_y(m,n)},
\end{aligned}
\tag{4.11}
$$

$$
\begin{aligned}
v_x(m,n) &= \sqrt{E((F_x(I_s)(m,n))^2)}, \\
v_y(m,n) &= \sqrt{E((F_y(I_s)(m,n))^2)}.
\end{aligned}
\tag{4.12}
$$

For the sparse image prior, $F_x(I_s)(m,n)$ and $F_y(I_s)(m,n)$ can be reformulated as

$$
\begin{aligned}
F_x(I_s)(m,n) &= I_s(m,n) - I_s(m-1,n), \\
F_y(I_s)(m,n) &= I_s(m,n) - I_s(m,n-1),
\end{aligned}
\tag{4.13}
$$

where $I_s(0,\cdot)$ and $I_s(\cdot,0)$ denote paddings.

Our VDIP can also be extended to the extreme channel prior. For the extreme channel prior, $F_x(I_s)(m,n)$ and $F_y(I_s)(m,n)$ can be reformulated as

$$\begin{aligned} F_x(I_s)(m,n) &= \min_{i\in\Omega(m,n)}(\min_{c\in(r,g,b)}(I_s^c(i))), \\ F_y(I_s)(m,n) &= 1-\max_{i\in\Omega(m,n)}(\max_{c\in(r,g,b)}(I_s^c(i))) \end{aligned} \tag{4.14}$$

where $\Omega(m,n)$ denotes a local patch centered at (m,n), and I_s^c is a color channel of I_s. Further derivation of $E((F_x(I_s)(m,n))^2)$ and $E((F_y(I_s)(m,n))^2)$ are shown in Appendix B.

4.1.3 Variational Deep Image Prior

Conventional variational inference solves Eqn. 4.10 by calculating the closed-form expectation with respect to each variable over all the other variables to get the distribution [260], but it is challenging to apply this strategy to deep learning since the networks are highly non-convex. Hence, we use two networks to learn the distribution of the latent sharp image and the blur kernel, respectively, in an unsupervised manner. For simplification, we assume that the standard deviation of the blur kernel $S(k)$ is constant. We also assume that the additive noise is white Gaussian noise. Then, we only need to learn the expectation of the image $E(I_s)$, the expectation of the kernel $E(k)$, and the standard deviation of the image $S(I_s)$.

We utilize an encoder-decoder as the image generator $G_I()$, a fully-connected network as the kernel generator $G_k()$, and random noises Z_I and Z_k as inputs. The image generator outputs both $E(I_s)$ and $S(I_s)$, and the kernel generator outputs $E(k)$. We can now approximate $E_{Q(I_s,k)}[\ln P(I_b|I_s,k)]$ in Eqn. 4.9 and 4.10 by Monte Carlo estimation using sampling [97]

$$\begin{aligned} E_{Q(I_s,k)}[\ln P(I_b|I_s,k)] &\approx \frac{1}{A}\sum_{a=1}^{A}\frac{||I_b-\hat{k}\otimes\hat{I_s^a}||_2^2}{2\sigma^2}, \\ \hat{k}=E(k), \hat{I_s^a} &= E(I_s)+\epsilon^a\odot S(I_s), \epsilon^a(m,n)\sim\mathcal{N}(0,\,I), \end{aligned} \tag{4.15}$$

where A is the number of samples, σ is the noise level, $\odot$ represents the element-wise multiplication, and $\epsilon^a(m,n)$ is a random scalar sampled from a standard Gaussian

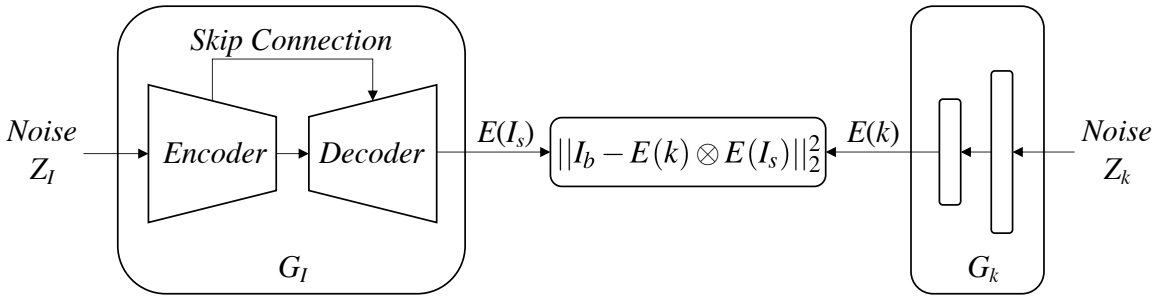
Algorithm 1: Blind Image Deconvolution Using Variational Deep Image Prior

Input: blurred image I_b, image generator $G_I()$, kernel generator $G_k()$
Output: estimated sharp image I_s^* and blur kernel k^*
Initialization: fixed noise inputs z_I and z_k, parameters of two generators $\theta_I^{(0)}$ and $\theta_k^{(0)}$ to be optimized
for $t = 1, 2, \ldots, T$ **do**
1. generate $E(I_s)^{(t)}$, $S(I_s)^{(t)}$ by $G_I(z_I, \theta_I^{(t-1)})$ and $E(k)^{(t)}$ by $G_k(z_k, \theta_k^{(t-1)})$
2. calculate $E(\xi_x^{(t)})$ and $E(\xi_y^{(t)})$ using Eqn. 4.11
3. sample $\hat{I}_s^{(t)}$ A times and approximate $E_{Q(I_s,k)}\left[\ln P(I_b|I_s,k)\right]^{(t)}$ using Eqn. 4.15
4. calculate $L(I_s, k, \xi_x, \xi_y, I_b)^{(t)}$ using Eqn. 4.10
5. update $\theta_I^{(t-1)}$ and $\theta_k^{(t-1)}$ by maximizing $L(I_s, k, \xi_x, \xi_y, I_b)^{(t)}$

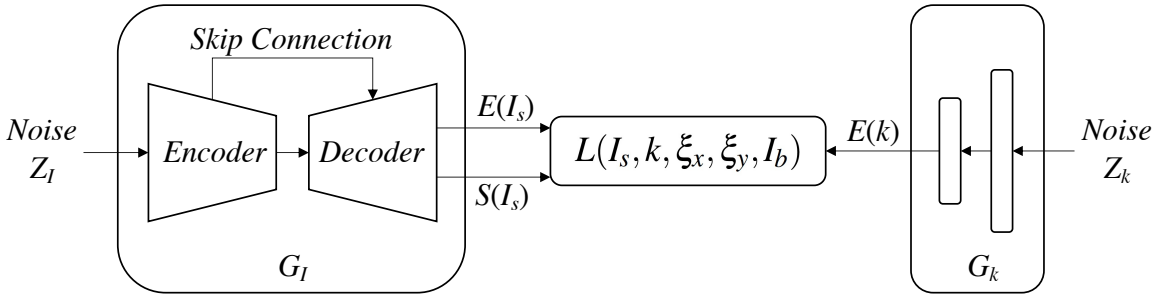
end for
$[E(I_s)^{(T+1)}, S(I_s)^{(T+1)}] = G_I(z_I, \theta_I^{(T)})$
$E(k)^{(T+1)} = G_k(z_k, \theta_k^{(T)})$
$I_s^* = E(I_s)^{(T+1)}$, $k^* = E(k)^{(T+1)}$

distribution for the pixel (m, n). The more samplings, the more accurate distribution will be obtained. Using Monte Carlo estimation, the expectation term is now differentiable. Our final algorithm is shown in Alg. 1.

The overview comparison of the DIP [51] and our proposed method is shown in Fig. 4.1. We can see that the DIP only generates a single value $E(I_s)$ for each pixel instead of $E(I_s)$ and $S(I_s)$ in our VDIP, and the target is minimizing the mean square error $||I_b - E(k) \otimes E(I_s)||_2^2$. The target of the DIP only focuses on maximizing $P(I_b|I_s, k)$ in Eqn. 4.7, so that $P(I_s|\theta_I)$ and $P(I_k|\theta_k)$ are not properly constrained. In contrast, in our proposed method, we apply a Gaussian prior and a sparse image prior to constrain $P(I_k|\theta_k)$ and $P(I_s|\theta_I)$, respectively, as shown in Eqn. 4.10. Simply exploiting the additive priors for optimizing Eqn. 4.7 can lead to suboptimal solutions of sparse MAP. Thus, we adpot the VB to avoid such a problem by introducing the standard deviation $S(I_s)$ to the optimization target. It is noteworthy that Eqn. 4.10 degenerates to the sparse MAP when we fix $S(I_s)$ as zero. It shows the limitation of optimizing the sparse MAP that its solution is difficult to achieve a large variational



(a) Overview of the DIP [51]



(b) Overview of our proposed VDIP

Figure 4.1: Comparison of the DIP [51] and our proposed VDIP. The number of decoder outputs are doubled and the loss function is replaced with the variational lower bound.

lower bound, because $\ln S(I_s)$ is negative infinity. The VB can nicely avoid it by considering non-zero $S(I_s)$. Besides, the values of $E(\xi)$ act as the penalty weights of gradients. In particular, small weights for large gradients and large weights for small gradients. Zero $S(I_s)$ may result in over-penalty in regions with small gradients.

4.2 Experiments

4.2.1 Implementation Details

Our proposed method is implemented in PyTorch [248] and evaluated on a single RTX A6000 GPU with 48GB of memory. The learning rate of the image generator and of the kernel generator are set as 1×10^{-2} and 1×10^{-4}, respectively, and the number of optimization steps T is 5000. In Eqn. 4.15, the number of samples A is set as 1.

Method	Manmade	Natural	People	Saturated	Text	Average
Cho *et al.* [64]	17.08/0.482	21.15/0.615	20.96/0.630	14.32/0.531	16.01/0.522	17.91/0.556
Levin *et al.* [261]	15.12/0.284	18.76/0.419	19.55/0.528	13.98/0.487	14.44/0.372	16.37/0.418
Krishnan *et al.* [66]	16.32/0.476	20.13/0.587	22.59/0.709	14.41/0.545	15.78/0.518	17.85/0.567
Xu *et al.* [70]	19.11/0.686	22.70/0.754	26.42/0.856	14.97/0.586	20.56/0.789	20.75/0.734
Perrone *et al.* [262]	18.66/0.676	22.78/0.786	24.79/0.828	14.46/0.531	18.35/0.673	19.81/0.699
Michaeli *et al.* [73]	18.27/0.509	21.93/0.614	25.74/0.791	14.46/0.539	16.59/0.503	19.40/0.591
Pan *et al.* [39]	20.00/0.714	24.47/0.801	26.70/0.811	17.46/0.680	21.13/0.762	21.95/0.753
Dong *et al.* [69]	18.88/0.567	23.42/0.702	25.53/0.769	16.72/0.611	20.05/0.682	20.92/0.666
Tao *et al.* [21]	17.11/0.381	20.18/0.492	22.12/0.651	15.41/0.545	15.76/0.469	18.12/0.508
Kupyn *et al.* [20]	17.47/0.414	20.71/0.520	22.71/0.682	15.67/0.565	16.22/0.503	18.55/0.537
Wen *et al.* [263]	18.06/0.550	22.51/0.669	25.59/0.769	17.79/0.672	17.85/0.598	20.36/0.652
Zamir *et al.* [11]	17.12/0.392	20.30/0.506	21.50/0.631	15.49/0.547	14.75/0.415	17.83/0.498
Huo *et al.* [264]	17.11/0.380	20.27/0.495	21.69/0.636	15.45/0.545	15.84/0.478	18.07/0.507
Zamir *et al.* [12]	17.19/0.389	20.26/0.493	21.67/0.636	15.52/0.545	15.36/0.460	18.00/0.505
Chen *et al.* [265]	16.89/0.371	20.10/0.484	21.51/0.642	15.59/0.544	14.87/0.401	17.79/0.488
Ren *et al.* [51] (DIP)	18.12/0.506	21.77/0.608	26.00/0.789	16.64/0.613	20.79/0.686	20.67/0.640
DIP-Extreme	19.90/0.708	21.48/0.656	27.90/0.862	18.10/0.690	24.57/0.840	22.39/0.751
DIP-Sparse	17.59/0.494	23.30/0.723	25.44/0.744	15.95/0.632	20.36/0.703	20.53/0.659
VDIP-Std	18.52/0.542	21.61/0.607	26.61/0.813	16.37/0.596	21.26/0.699	20.87/0.651
VDIP-Extreme	20.50/0.768	25.36/0.882	**30.83/0.938**	18.09/0.723	25.90/0.892	24.14/0.841
VDIP-Sparse	**22.86/0.868**	**26.18/0.895**	30.76/0.927	**18.55/0.727**	**27.24/0.927**	**25.12/0.869**

Table 4.1: Quantitative comparison (PSNR↑/SSIM↑) on the synthetic dataset from Lai *et al.* [266].

We use $\ln|x|$ as our penalty function $\rho(x)$. Note that the architectures of $G_I()$ and $G_k()$ are the same as those of DIP [51] for fair comparison, except the output layers of $G_I()$ are doubled (half for $E(I_s)$ and half for $S(I_s)$). Different from the original DIP [51] that adds additive random Gaussian noise to Z_I and Z_k to avoid the local minima, we do not add additive random noise to the inputs.

4.2.2 Quantitative Comparison

We first evaluate different versions of DIP including our VDIP for image deconvolution on the synthetic dataset from Lai *et al.* [266] and compare with several conventional methods including Cho and Lee [64], Levin *et al.* [261], Krishnan *et al.* [66], Xu *et al.* [70], Perrone *et al.* [262], Michaeli and Irani [73], Pan *et al.* [39], Dong *et al.* [69], and Wen *et al.* [263], and several deep-learning-based methods including Tao *et al.* [21], Kupyn *et al.* [20], Zamir *et al.* [11], Huo *et al.* [264], Zamir *et al.* [12] and Chen *et al.* [265]. To be specific, these deep-learning-based methods are trained on external datasets [24, 267]. DIP-Extreme and DIP-Sparse represent the

Method	NIQE↓	BRISQUE↓	PIQE↓
Cho *et al.* [64]	4.0050	36.2829	48.6227
Levin *et al.* [261]	3.6594	36.5006	46.7037
Krishnan *et al.* [66]	3.8696	37.9942	50.4024
Xu *et al.* [70]	3.9536	37.3240	49.5436
Perrone *et al.* [262]	4.0397	39.7997	51.7650
Michaeli *et al.* [73]	3.5852	35.1205	46.7085
Pan *et al.* [39]	4.8790	36.3792	68.9470
Dong *et al.* [69]	4.7557	37.1199	64.1972
Tao *et al.* [21]	3.5612	40.1954	53.0908
Kupyn *et al.* [20]	**3.2937**	35.8382	40.0545
Wen *et al.* [263]	4.9210	33.1731	58.3326
Zamir *et al.* [11]	3.7926	42.4894	52.1181
Huo *et al.* [264]	3.5222	40.1037	47.0717
Zamir *et al.* [12]	3.7401	42.9266	50.6804
Chen *et al.* [265]	4.6754	46.3900	74.0267
Ren *et al.* [51] (DIP)	4.2460	38.5827	45.8822
DIP-Extreme	4.7763	33.3678	36.6031
DIP-Sparse	7.9063	41.9810	54.9295
VDIP-Std	4.1260	37.0199	42.3010
VDIP-Extreme	4.5072	34.4400	36.1535
VDIP-Sparse	3.8882	**32.4120**	**34.3614**

Table 4.2: Quantitative comparison on the real blurred dataset from Lai *et al.* [266].

DIP [51] with the extreme channel prior and the sparse image prior, respectively. Our VDIP-Std, VDIP-Extreme and VDIP-Sparse are the corresponding versions of DIP, DIP-Extreme and DIP-Sparse with non-zero $S(I_s)$.

The quantitative comparison is shown in Tab. 4.1. We can see that DIP-Sparse even perform worse than DIP, which is consistent with the suboptimal problem of sparse MAP. And non-zero $S(I_s)$ without additive priors can only slightly improve the performance. The combination of additive priors and non-zero $S(I_s)$ significantly increases the evaluation results, where the former helps to constrain the optimization and the latter avoids the local minimum resulting from the former. For gradient-based priors, a sparser constrain can lead to better performance comparing L0 norm [70], L1 norm [262] and L2 norm [64, 261], but the outliers on saturated images should be properly handled as in [69]. Image-based priors [39, 263] are more robust to outliers, and the comparison of DIP-Extreme and DIP-Sparse follows this observation. Additionally, our VDIP-sparse takes advantage of gradient-based priors without explicitly handling the outliers of saturated images and performs even better than VDIP-Extreme, which shows the effectiveness of utilizing variational Bayes.

Method	Manmade	Natural	People	Saturated	Text	Average
Cho *et al.* [64]	0.00138	0.00121	0.00145	0.00164	0.00139	0.00141
Levin *et al.* [261]	0.00099	0.00107	0.00117	0.00124	0.00117	0.00113
Krishnan *et al.* [66]	0.00125	0.00114	0.00128	0.00134	0.00118	0.00124
Xu *et al.* [70]	0.00114	0.00084	0.00073	0.00144	0.00074	0.00098
Perrone *et al.* [262]	0.00108	0.00091	0.00111	0.00135	0.00102	0.00109
Michaeli *et al.* [73]	0.00131	0.00118	0.00102	0.00169	0.00148	0.00134
Pan *et al.* [39]	0.00078	**0.00060**	**0.00083**	**0.00099**	0.00071	**0.00078**
Dong *et al.* [69]	0.00097	0.00078	0.00096	0.00111	0.00082	0.00093
Wen *et al.* [263]	0.00113	0.00092	0.00089	0.00074	0.00098	0.00093
Ren *et al.* [51] (DIP)	0.00168	0.00168	0.00164	0.00172	0.00144	0.00163
DIP-Extreme	0.00117	0.00122	0.00084	0.00153	0.00086	0.00113
DIP-Sparse	0.00159	0.00148	0.00136	0.00142	0.00135	0.00144
VDIP-Std	0.00163	0.00167	0.00157	0.00171	0.00140	0.00160
VDIP-Extreme	0.00104	0.00101	0.00098	0.00147	0.00061	0.00102
VDIP-Sparse	**0.00073**	0.00095	0.00084	0.00146	**0.00060**	0.00092

Table 4.3: Average kernel recovery error on the synthetic dataset from Lai *et al.* [266].

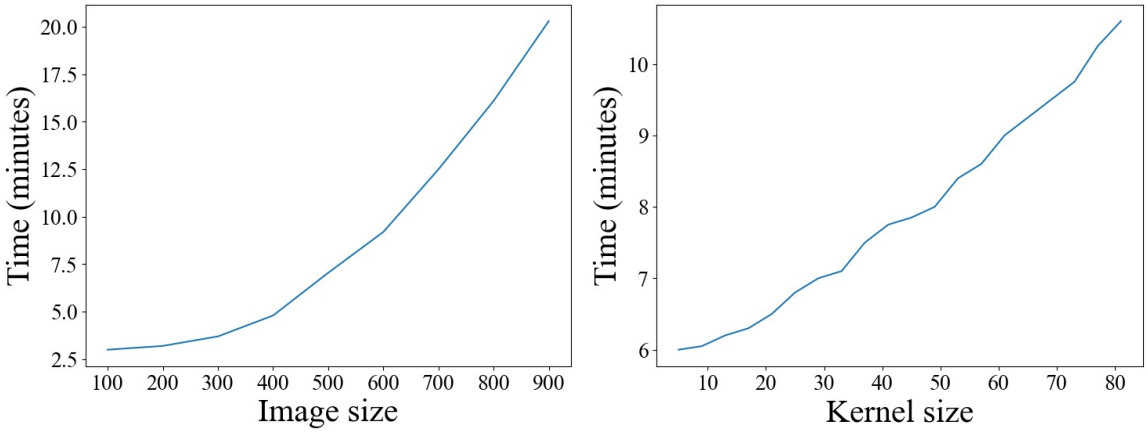


Figure 4.2: The optimization time corresponding to the image size and kernel size. The kernel size is fixed as 31×31 for evaluating the image size, and the image size is fixed as 500×500 for evaluating the kernel size.

To evaluate the estimated kernel, we calculate the average kernel recovery error [268] and report the results in Tab. 4.3. Note that the compared deep-learning-based methods do not estimate the blur kernels. Although the evaluated kernel of Pan *et al.* [39] is more accurate than VDIP-Extreme, our VDIP-Extreme performs better, which demonstrate that a proper deconvolution method is important even with accurate estimated blur kernels.

We also evaluate the above mentioned methods on the real blurred dataset from Lai *et al.* [266]. Since there is no ground truth sharp image, we utilize three no-

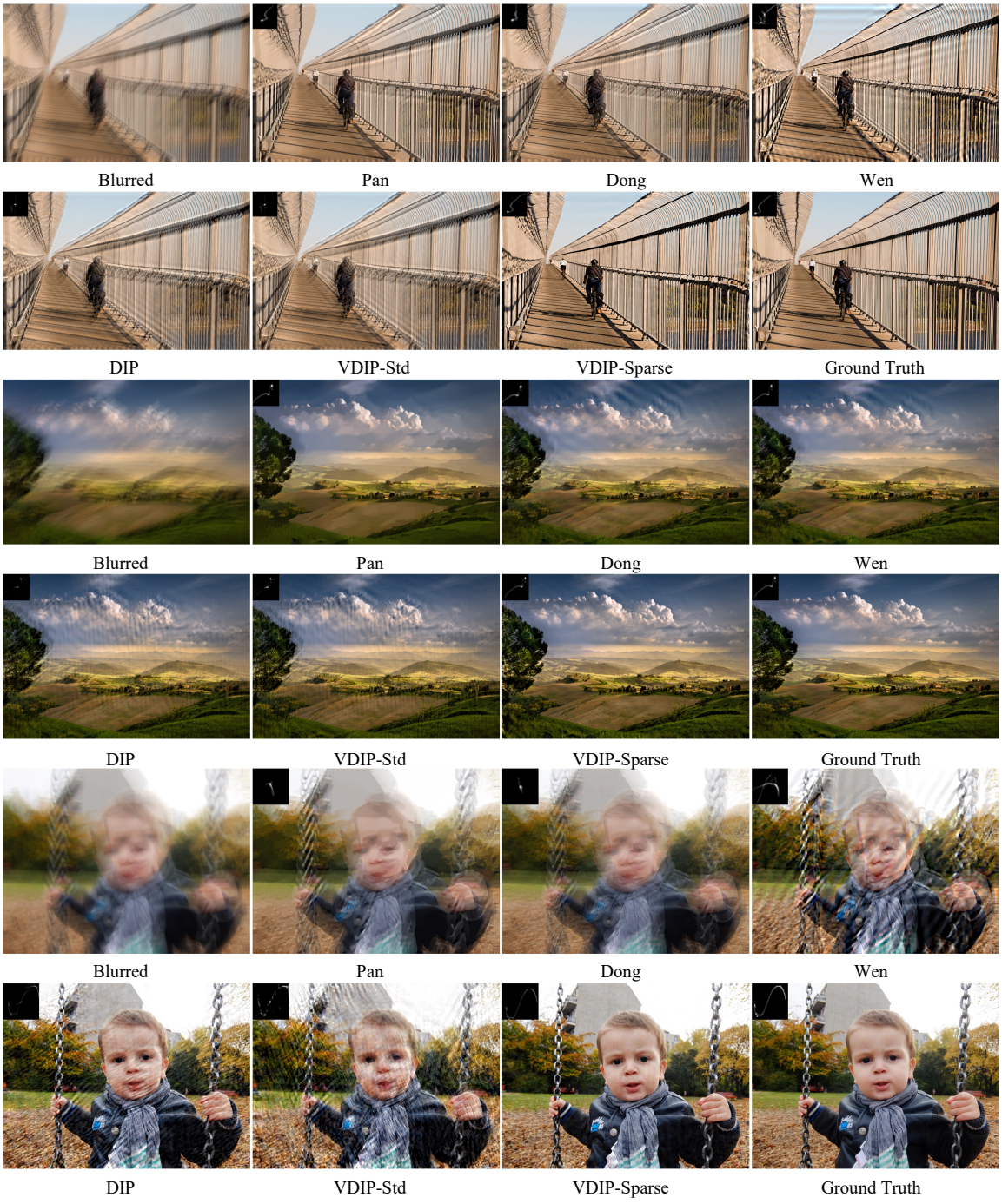


Figure 4.3: Qualitative comparison on the synthetic dataset from Lai *et al.* [266]. The estimated blur kernels are pasted at the top-left corners of the corresponding deblurred results.

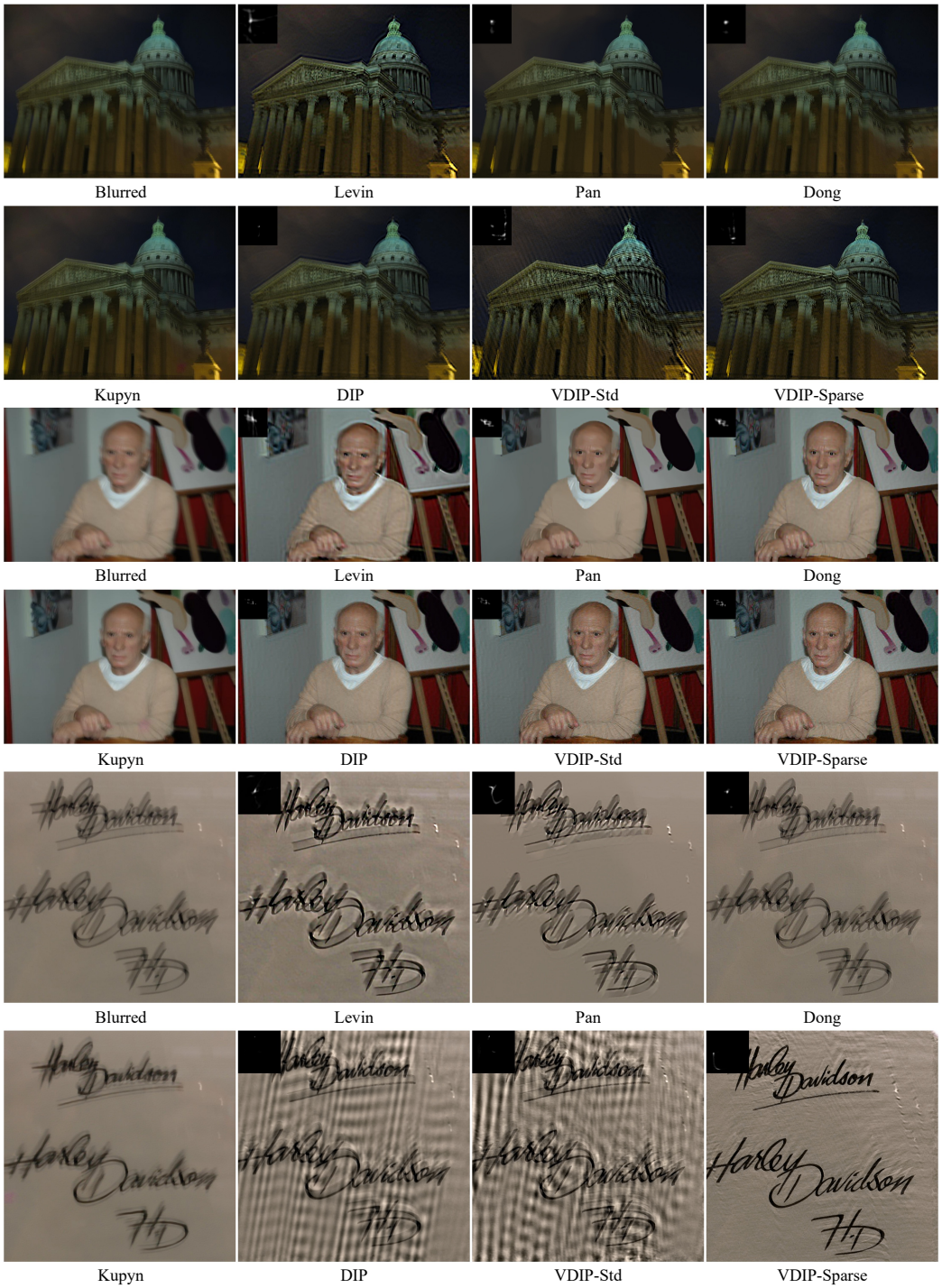


Figure 4.4: Qualitative comparison on the real blurred dataset from Lai *et al.* [266]. The estimated blur kernels are pasted at the top-left corners of the corresponding deblurred results.

reference image quality assessment metrics, in particular, Naturalness Image Quality Evaluator (NIQE) [269], Blind/Referenceless Image Spatial Quality Evaluator (BRISQUE) [270], and Perception based Image Quality Evaluator (PIQE) [271] to quantitatively evaluate the results. As shown in Tab. 4.2, our method can generate images of the highest quality based on BRISQUE and PIQE among all compared methods. Similar to all of the compared conventional methods and DIP, our proposed method is also designed for spatially invariant (uniform) blur. However, it even performs better than deep-learning-based methods that are trained for spatially variant blur. We think this is because the compared deep-learning-based methods are all trained on synthetic datasets where the blurred images are generated by averaging consecutive frames from a high-frame-rate video. The performance of these methods are limited on the real data with more artifacts because of the domain-shift issue.

4.2.3 Optimization Time

To evaluate the relation between the optimization time and the size of images and kernels, we run the optimization with varying image size and fixed kernel size, and then run the optimization with varying kernel size and fixed image size. All of the experiments are run on a single RTX A6000 GPU with 48GB of memory. As shown in Fig. 4.2, the optimization time is proportional to the quadratic of image size and kernel size.

4.2.4 Qualitative Comparison

Some of the qualitative comparisons are shown in Fig. 4.3 and 4.4. Our VDIP-Sparse can generate sharper results with less noise and artifacts than other methods including DIP. Specifically, Pan *et al.* [39] are able to obtain correct blur kernels in some cases but the deconvolution results are over-smoothed. Dong *et al.* [69], Wen *et al.* [263] and DIP [51] are over-enhanced with many artifacts. Since the blur on the real images are spatially variant (non-uniform), obtaining perfect results with

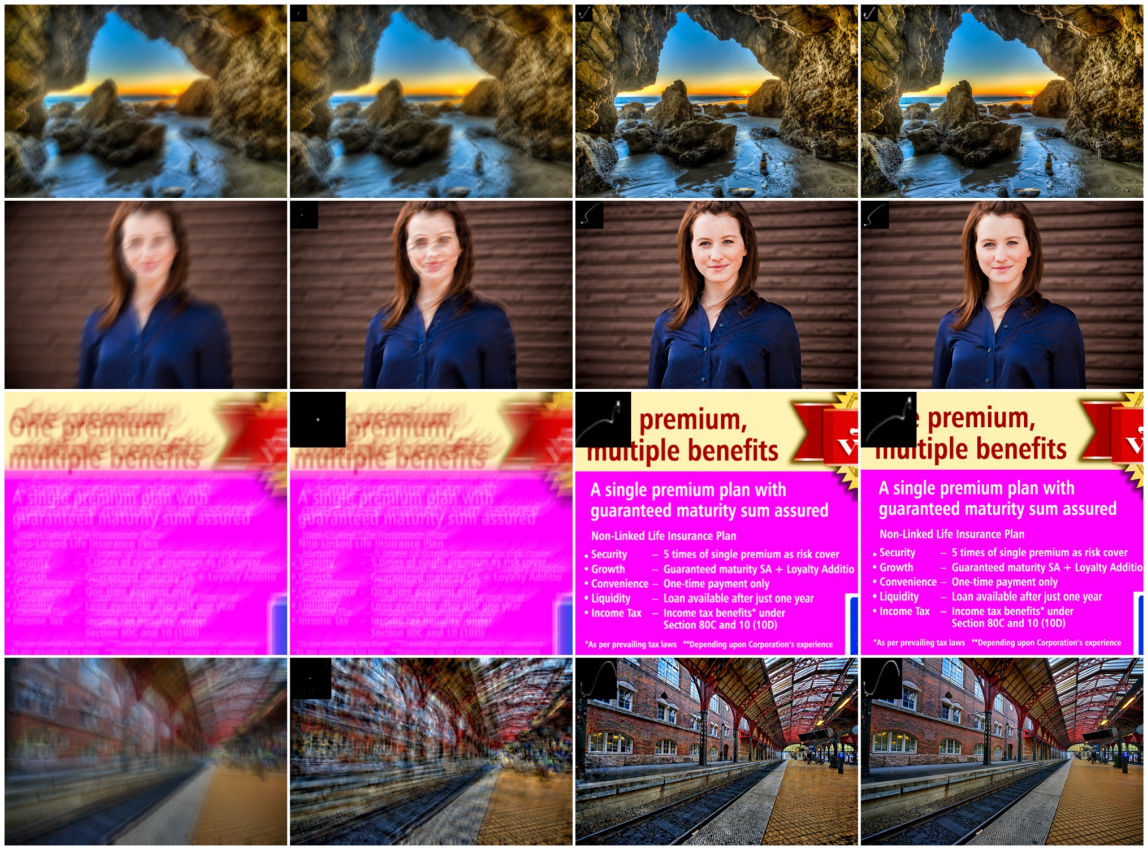


Figure 4.5: Qualitative comparison of MAP (DIP) and VB (VDIP-Sparse). The estimated blur kernels are pasted at the top-left corners of the corresponding deblurred results where the estimated kernels of MAP are all delta kernels.

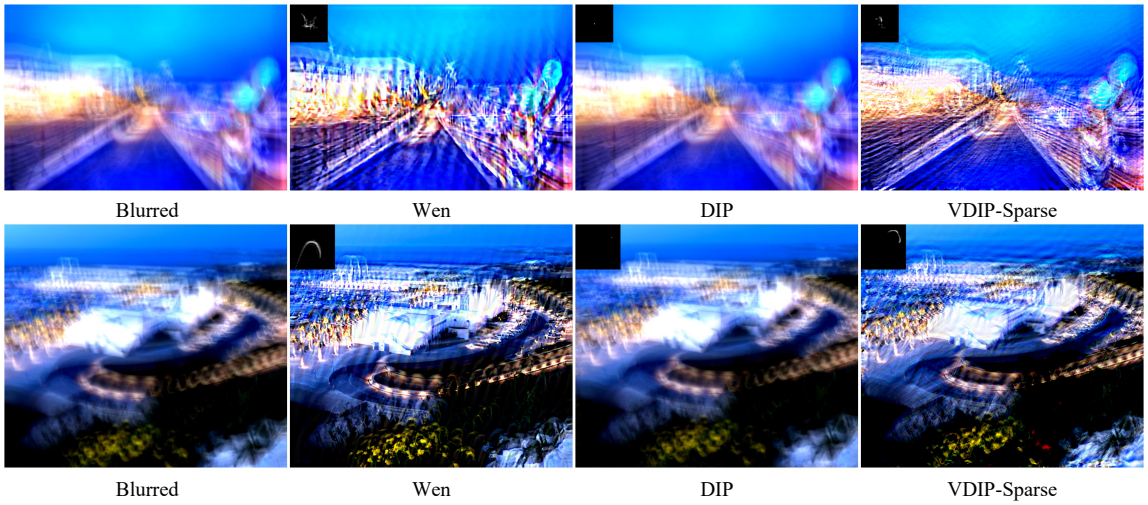


Figure 4.6: Failure Cases.

uniform deconvolution methods is difficult, if not impossible. But our method still performs better than Kupyn *et al.* [20] trained on non-uniform blurred datasets [24], showing the limited generalization ability of external training and the importance of image-specific information.

As outlined in Section 4.1, when a sparse image prior is employed in conjunction with Maximum a Posteriori (MAP) estimation, the resulting solution favors a trivial outcome, wherein the generated kernel is a delta kernel. Fig. 4.5 demonstrates the effectiveness of our improved approach utilizing Variational Bayes (VB) over the MAP method. It displays the trivial solution obtained by MAP, where the estimated kernels collapse to a single white dot (delta kernel). In contrast, VB successfully avoids such solutions, resulting in more accurate estimations.

4.2.5 Failure Cases

As shown in Fig. 4.6, our VDIP does not perform well on small images with complex scenes, due to the lack of enough information to properly optimize the network.

4.3 Summary

In this chapter, I propose VDIP to better constrain the optimization of DIP. Along with the previous chapter, I make efforts on reducing the domain-shift problem of solving 2D spatial degradation tasks. In the next chapter, we address the 3D spatial degradation which takes advantage of the high-quality generation results of the pretrained text to image models.

Chapter 5

Text-Guided Texture Generation for 3D Objects

Generating high-quality 3D content is an essential component of visual applications in films, games, and upcoming AR/VR industries. While many prior works on 3D synthesis have focused on the geometric components of the assets, textures have garnered less attention which play a vital role in enhancing the realism of 3D assets. In this chapter, our objective is to achieve automatic text-driven 3D texture synthesis for arbitrary meshes.

5.1 Proposed Method

5.1.1 Overview

In this section, we present an overview of our proposed Progressive Texture Sampling Scheme to synthesize view-consistent textures from a pre-trained T2I generation model. We first introduce the sampling process of the Denoising Diffusion Implicit Models (DDIM) [272], which forms the basis of our texture sampling approach.

DDIM Sampling. Assuming we sequentially sample N distinct views around a 3D mesh, the DDIM sampling process for each sampled viewpoint i at the denoising step t can be described as follows:

$$\hat{x}_0^i(x_t^i) = \frac{x_t^i - \sqrt{1-\alpha_t} \cdot \epsilon_\theta(x_t^i)}{\sqrt{\alpha_t}}, \tag{5.1}$$

$$x_{t-1}^i = \sqrt{\alpha_{t-1}} \cdot \hat{x}_0^i(x_t^i) + \sqrt{1-\alpha_{t-1}} \cdot \epsilon_\theta(x_t^i), \tag{5.2}$$

where x_t^i represents the noisy latent feature, and $\epsilon_\theta(x_t^i)$ represents the estimated noise from the pre-trained T2I diffusion model. At each denoising step t, we calculate $\hat{x}_0^i(x_t^i)$, representing the predicted x_0^i and dubbed as the *denoised observation* of x_t^i. α_t is the total noise variance parameterized via denoising step t.

Texture Sampling. To address the inconsistencies arising from independently generated different views, our texture sampling scheme is proposed by leveraging the sequential nature of the denoising process of the diffusion model and maintaining the 3D consistency of the generated texture at each denoising step.

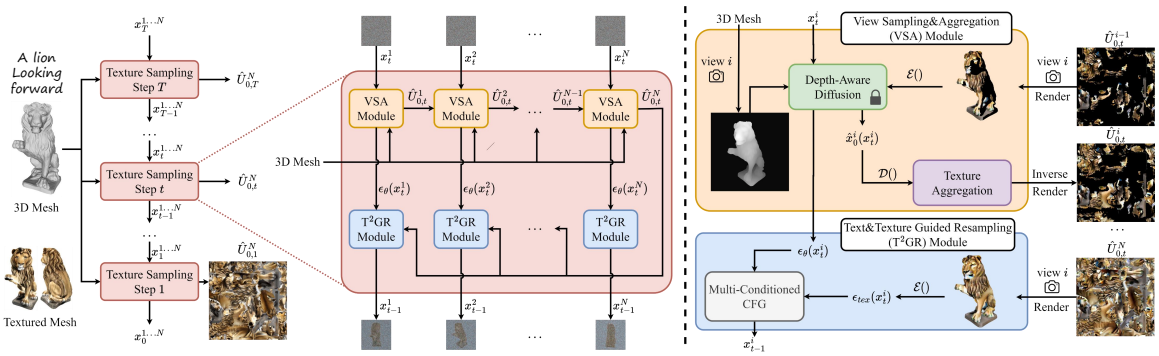
In detail, at each denoising step t, we conduct the following two steps. First, we compute a view-consistent denoised observation of a texture map $\hat{U}_0(x_t^{1\cdots N})$ by progressively aggregating the *denoised observations* $\hat{x}_0^i(x_t^i)$ where $i = 1, \ldots, N$ via an attention-guided view aggregation pipeline (Sec. 5.1.2). For brevity, we denote $\hat{U}_{0,t}^i$ as the partial texture map $\hat{U}_0(x_t^{1\cdots i})$ and $\hat{U}_{0,t}^N$ as the complete texture map $\hat{U}_0(x_t^{1\cdots N})$, both at denoising step t. Second, the calculation of the noisy latent feature for the upcoming denoising step $t-1$ is based on the current latent feature x_t^i, the *denoised observation* $\hat{x}_0^i(x_t^i)$, as well as the current texture map $\hat{U}_{0,t}^N$ as detailed in Sec. 5.1.3,

$$x_{t-1}^i \sim q(x_{t-1}^i | x_t^i, \hat{x}_0^i(x_t^i), \hat{U}_{0,t}^N). \tag{5.3}$$

Following the DDIM sampling, we go through the above process with T denoising steps to arrive at the final generated texture map $\hat{U}_{0,1}^N$. The texture sampling scheme is further illustrated in Fig. 5.1. We present the above-mentioned two major steps in the following sections.

5.1.2 View Sampling&Aggregation (VSA)

The objective of this stage is to determine the denoised observation of the texture map $\hat{U}_{0,t}^N$, based on the latent features of $x_t^{1\cdots N}$ at each viewpoint for the current step.



(a) Illustration of our texture sampling scheme

(b) Details of a denoising step

Figure 5.1: Overview of our proposed method. First of all, we sample N viewpoints across the objects. Our texture sampling scheme is an interleaved process of multi-view texture aggregation and diffusion denoising. Specifically, our texture sampling process is structured into T steps of diffusion process. As shown in (a), at denoising step t, it takes the noisy latent features of sampled views ($x_t^{1\dots N}$) as input to predict the noisy features for the next denoising step ($x_{t-1}^{1\dots N}$) as well as a time-dependent texture map ($\hat{U}_{0,t}^N$). Upon completing T steps of sampling, the final texture map ($\hat{U}_{0,1}^N$) will be achieved. To elaborate more on each denoising step, we present two novel modules: View Sampling&Aggregation (VSA) module and Text&Texture Guided Resampling (T^2GR) module. As shown in (b), for view i, the VSA module is used to generate denoised observation $\hat{x}_0^i(x_t^i)$ which will be aggregated onto texture map to form $\hat{U}_{0,t}^{i+1}$. After iterating over all sampled views starting from $i = 1$ to N, we obtain $\hat{U}_{0,t}^N$ for each denoising step. Conditioned on the current estimation of texture map $\hat{U}_{0,t}^N$, the T^2GR module will update the noise estimations $\epsilon_{tex}(x_t^i)$ to calculate the noisy latent feature x_{t-1}^i for the next denoising step.

View Sampling

Following DDIM sampling, for each sampled view i at time step t, the *denoised observation* $\hat{x}_0^i(x_t^i)$ can be computed as in Eq. 5.1. The latent features are then decoded into images I_t^i in the RGB space via the VAE decoder $\mathcal{D}$ of the pre-trained stable diffusion [15],

$$I_t^i = \mathcal{D}(\hat{x}_0^i(x_t^i)). \tag{5.4}$$

A naive solution of generating the denoised observation of texture map $\hat{U}_{0,t}^N$ is to directly fuse I_t^i (for $i = 1, \dots, N$), after inverse rendering onto the UV space. However, it will lead to noticeable seams between adjacent views due to the separate and view-inconsistent generation of each I_t^i.

View Aggregation

To address the issue of view inconsistency, we adopt an autoregressive generation strategy. This involves generating the *denoised observation* $\hat{x}_0^i(x_t^i)$ based on previous denoised views $\hat{x}_0^{1\ldots i-1}(x_t^{1\ldots i-1})$. Starting with the first viewpoint $i = 1$, we perform inverse rendering to map I_t^i onto the UV space, obtaining the partial texture map $\hat{U}_{0,t}^i$. Then for the subsequent viewpoint $i+1$, the prediction of $\hat{x}_0^{i+1}(x_t^{i+1})$ depends on the current partial texture map $\hat{U}_{0,t}^i$. More specifically, we render the partial texture map $\hat{U}_{0,t}^i$ onto viewpoint $i+1$, denoted as $Render^{i+1}(\hat{U}_{0,t}^i)$. The rendered output is then fed as input to the VAE encoder $\mathcal{E}$ for obtaining the latent features $G_{0,t}^{i+1}$,

$$G_{0,t}^{i+1} = \mathcal{E}(Render^{i+1}(\hat{U}_{0,t}^i)). \tag{5.5}$$

The computation of the *denoised observation* for viewpoint $i+1$ at step t is performed as follows:

$$\hat{x}_0^{i+1}(x_t^{i+1}) = \frac{x_t^{i+1} - \sqrt{1-\alpha_t} \cdot \epsilon_\theta(x_t^{i+1})}{\sqrt{\alpha_t}}, \tag{5.6}$$

with

$$\begin{aligned} x_t^{i+1} =& x_t^{i+1} \odot \mathcal{M}_{\downarrow}^{i+1} + (\sqrt{\alpha_t} \cdot G_{0,t}^{i+1} \\ &+ \sqrt{1-\alpha_t} \cdot \epsilon) \odot (1 - \mathcal{M}_{\downarrow}^{i+1}), \end{aligned} \tag{5.7}$$

where $\mathcal{M}^{i+1}$ denotes the mask for regions that are viewed for the first time at view $i+1$ in RGB space, and $\downarrow$ symbolizes downsampling to the resolution of latent features.

Attention-Guided View Synthesis

As indicated in Fig. 5.6, sequential generation across different viewpoints often fails to ensure appearance consistency. To address this, we introduce a novel attention-guided cross-view generation strategy. Drawing inspiration from the work of Cao *et al.* [273], we believe that Key and Value features in the self-attention module of the stable diffusion encapsulate the local contents and textures of generated images. In detail, we regard the front view as the reference view and propagate the Key and Value of the reference view to other views. The process can be outlined as follows:

$$\epsilon_\theta(x_t^{ref}),\, Q_t^{ref},\, K_t^{ref},\, V_t^{ref} \leftarrow Unet_\theta(x_t^{ref}), \tag{5.8}$$

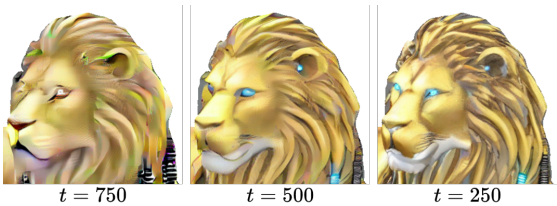


Figure 5.2: *Denoised observation* $\hat{x_0}(x_t^i)$ with text prompt "A Cyber Punk lion". The high-frequency information is gradually generated during sampling.

$$\epsilon_\theta(x_t^i) \leftarrow Unet_\theta(x_t^i, K_t^{ref}, V_t^{ref}). \tag{5.9}$$

Herein, Q_t^{ref}, K_t^{ref}, and V_t^{ref} denote the Query, Key, and Value features from the self-attention module of the reference view, respectively. In Eq. 5.9, the Key and Value features for each viewpoint are substituted with those from the reference view to calculate its estimated noise. Following this substitution, for each viewpoint i, the *denoised observation* $\hat{x}_0^i(x_t^i)$ is updated in accordance with Eq. 5.1. As shown in Fig. 5.2, the texture details will gradually appear in the *denoised observation* as the diffusion process proceeds.

5.1.3 Text&Texture Guided Resampling (T^{2}GR)

Upon obtaining the current denoised observation of the texture map $\hat{U}_{0,t}^N$, we perform Text&Texture Guided Resampling (T^{2}GR) to update the noisy latent features $x_{t-1}^{1\cdots N}$ for the next denoising step.

As shown in Eq. 5.2, the derivation of x_{t-1}^i depends on the estimated noise $\epsilon_\theta(x_t^i)$ and the *denoised observation* $\hat{x}_0^i(x_t^i)$. Given that $\hat{x}_0^i(x_t^i)$ is expected to exhibit view consistency, as it is maintained by the texture map $\hat{U}_{0,t}^N$, recalculating the noise map $\epsilon_\theta(x_t^i)$ under the guidance of current denoised observation of texture map $\hat{U}_{0,t}^N$ ensures to preserve the view consistency. Specifically, in Eq. 5.1 we set $\hat{x}_0^i(x_t^i)$ equal to the current encoded render of the texture map $\hat{U}_{0,t}^N$ at view i. From this, we derive the recalculated noise map $\hat{\epsilon}_{tex}(x_t^i)$ as:

$$\hat{\epsilon}_{tex}(x_t^i) = \frac{x_t^i - \sqrt{\alpha_t} \cdot \mathcal{E}(Render^i(\hat{U}_{0,t}^N))}{\sqrt{1-\alpha_t}}. \tag{5.10}$$

This recalculated noise map is then utilized in place of $\epsilon_\theta(x_t^i)$ in Eq. 5.1 and Eq. 5.2 for the computation of x_{t-1}^i.

While our noise map update strategy ensures view consistency, it tends to result in over-smoothed images (as shown in Fig. 5.7). This is primarily because the VAE encoder $\mathcal{E}$ in the stable diffusion model compresses high-frequency details, referred to as *imperceptible details*, as noted by [15]. The repeated use of the encoder $\mathcal{E}$ leads to an accumulation of this detail compression, affecting the overall image quality.

To avoid over-smoothness, we take $\hat{U}_{0,t}^N$ as an additional condition to the diffusion model rather than directly replacing $\epsilon_\theta(x_t^i)$ with $\epsilon_{tex}(x_t^i)$. Basically, we want to compute a texture-conditioned noise estimation which we denote as $\epsilon_{tex}(x_t^i|\hat{U}_{0,t}^N)$. By analyzing the formulation of $\epsilon_\theta(x_t^i)$, we see that it is essentially a weighted combination of conditional noise prediction $\epsilon_\theta(x_t^i|c)$ and unconditional noise prediction $\epsilon_\theta(x_t^i|\varnothing)$, following the Classifier-Free Guidance (CFG) introduced in [274]:

$$\epsilon_\theta(x_t^i) = \epsilon_\theta(x_t^i|\varnothing) + \omega(\epsilon_\theta(x_t^i|c) - \epsilon_\theta(x_t^i|\varnothing)), \tag{5.11}$$

where c and $\varnothing$ represent the text prompt and null-text prompt, respectively, and ω is a user-specified weight.

Similarly, $\epsilon_{tex}(x_t^i)$ should follow the same formulation of CFG,

$$\epsilon_{tex}(x_t^i) = \epsilon_\theta(x_t^i|\varnothing) + \omega(\epsilon_{tex}(x_t^i|\hat{U}_{0,t}^N) - \epsilon_\theta(x_t^i|\varnothing)). \tag{5.12}$$

Thus, to disentangle the texture-conditioned noise estimation $\epsilon_{tex}(x_t^i|\hat{U}_{0,t}^N)$, we subtract the null-text conditioned noise estimation from $\epsilon_{tex}(x_t^i)$. Here we set $\epsilon_{tex}(x_t^i) = \hat{\epsilon}_{tex}(x_t^i)$. The computation for the texture-conditioned noise estimation $\epsilon_{tex}(x_t^i|\hat{U}_{0,t}^N)$ is as follows:

$$\epsilon_{tex}(x_t^i|\hat{U}_{0,t}^N) = \frac{1}{\omega}(\hat{\epsilon}_{tex}(x_t^i) - \epsilon_\theta(x_t^i|\varnothing)) + \epsilon_\theta(x_t^i|\varnothing). \tag{5.13}$$

In the end, we formulate our multi-conditioned CFG for the final noise estimation, which is conditioned on both the textual prompt and texture map:

$$\epsilon_m(x_t^i) = \epsilon_\theta(x_t^i|\varnothing) + \omega_1(\epsilon_\theta(x_t^i|c) - \epsilon_\theta(x_t^i|\varnothing)) + \omega_2(\epsilon_{tex}(x_t^i|\hat{U}_{0,t}^N) - \epsilon_\theta(x_t^i|\varnothing)), \tag{5.14}$$

Figure 5.3: Visual comparison of our proposed method against TEXTure [120] and Text2Tex [121].

where $\omega_1 + \omega_2 = \omega$. We exploit a large ω_2 for early sampling steps, which decreases linearly from ω to 0 in the process of denoising. The comprehensive derivation of Eq. 5.14 can be found in Appendix C. Finally, we compute x_{t-1}^i for subsequent denoising steps by letting $\epsilon_\theta(x_t^i) = \epsilon_m(x_t^i)$ in Eq. 5.1 and Eq. 5.2.

5.2 Experiments

5.2.1 Implementation Details

We employ the depth-aware diffusion model provided by ControlNet [52] as our T2I backbone with the number of denoising steps $T = 40$. To render an object, we take eight different viewpoints around the object. The pose is sampled in spherical coordinates, with the elevation angle set to zero and the azimuth angles is uniformly sampled between [0°, 360°]. An additional top view is sampled. Additionally, we employ the Xatlas [275] tool to compute the UV atlas for a given mesh.

Figure 5.4: Visual comparison of our proposed method against Fantasia3D [115] and ProlificDreamer [116].

Dataset

Our experiment incorporates a diverse collection of 45 meshes, sourced from various datasets such as Objaverse [276] and ThreeDScans [277], with 2 to 3 distinct prompts for each mesh. Please refer to Appendix C for details.

5.2.2 Compared Methods

We conduct experimental comparison with several state-of-the-art approaches, including TEXTure [120], Text2Tex [121], Fantasia3D [115], ProlificDreamer [116] and TexFusion [122]. For TEXTure, Text2Tex, and Fantasia3D, we use their respective publicly available codebase. As the official code for ProlificDreamer is not yet accessible, we adopt the implementation of ThreeStudio [278] and replace its backbone with ControlNet [52] to recognize the depth. In the case of TexFusion, where the implementation is not available, our analysis is limited to a qualitative assessment using results extracted directly from the original paper. Notably, for all the compared

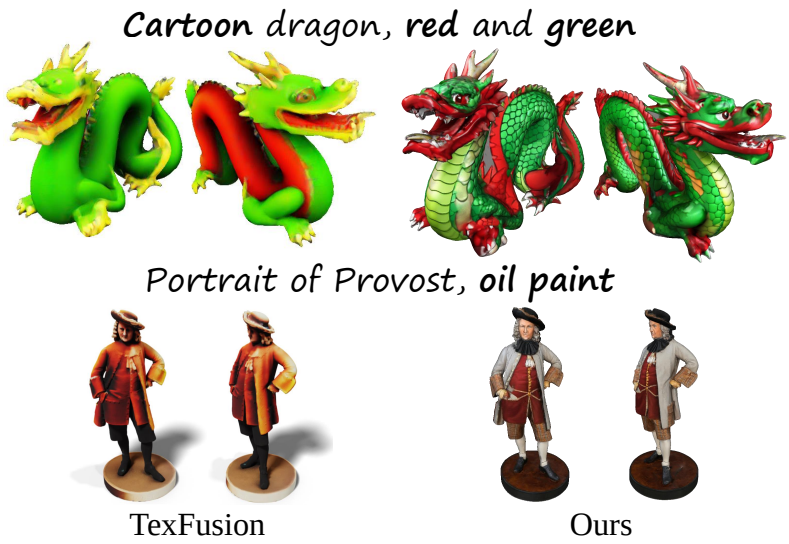


Figure 5.5: Visual comparison of our proposed method against TexFusion [122]. The results of TexFusion are directly copied from its original paper

approaches, the geometry remains fixed during texture generation.

5.2.3 Qualitative Comparison

We provide visual comparison in Fig. 5.3 and Fig. 5.4. Specifically, in Fig. 5.3, we showcase the robustness of our approach in addressing fragmented textures against progressively texture aggregation approaches, namely TEXTure [120] and Text2Tex [121]. This improvement is credited to our use of attention-guided view aggregation, combined with a distinct text&texture guided resampling approach that maintains view consistency at each denoising step to persistently enchance 3D consistency.

In Fig. 5.4, we compare with score distillation based approaches, namely Fantasia3D [115] and ProlificDreamer [116]. As demonstrated, Fantasia3D typically produces textures that are over-smoothed and over-saturated, while ProlificDreamer, though with more details and higher contrast, is marred by evident artifacts of blurry edges. In contrast, our method surpasses these distillation-based methods by generating more realistic high-quality results.

We also present a qualitative comparison of our method with TexFusion [122] in Fig. 5.5. TexFusion employs instantNGP [123] to mitigate inconsistencies post-

Methods	FID ↓	KID×10^{-3} ↓	CLIPScore ↑
TEXTure	99.06	7.23	19.73
Text2Tex	109.94	7.17	21.26
Fantasia3D	108.58	7.52	21.14
ProlificDreamer	94.51	7.00	21.25
Ours	**84.65**	**4.27**	**22.83**

Table 5.1: Quantitative comparison on generated textures.

decoding of latent features into RGB space, which often leads to over-smoothed results. In contrast, our method effectively generates textures that are consistent across views and retain rich details. Please refer to Appendix C for more visual results.

5.2.4 Quantitative Comparison

Evaluation Metrics

For quantitative evaluation of the generated texture, we employ two widely used image quality and diversity evaluation metrics, including Frechet Inception Distance (FID) [279] and Kernel Inception Distance (KID) [280]. These metrics are instrumental in measuring the distribution similarity between two sets of images. For each compared method, we render a set of images by uniformly sampling 32 different views of the generated textured mesh. To establish a ground truth image set, we follow the approach outlined by Cao *et al.* [122], who use a depth-conditioned ControlNet to synthesize images conditioned on rendered depth maps and corresponding textual prompts. The background pixels have been removed from all images to mitigate the influence caused by unconstrained background. Additionally, we incorporate the CLIPScore metric [281] to assess the congruence and resemblance between the generated images and their associated text prompts. Specifically, for each method, we calculate the average CLIPScore across all rendered images relative to the given text prompts.

	TEXTure ↑	Text2Tex ↑	Fantasia3D ↑	ProlificDreamer ↑
Ours	64.72%	71.46%	70.97%	69.18%

Table 5.2: User Study Preference: The entries in the table indicate our preference over other methods. A higher value represents a greater preference.

We present the quantitative evaluations of the above-mentioned methods on FID, KID and CLIPScore in Tab. 5.1. Notably, our approach demonstrates superior performance, outstripping the other methods by at least 10.4% in FID and 39.0% in KID. The figures showcase our method's capability to generate textures that not only are more realistic but also exhibit a wide variety of appearances across diverse objects.

User Studies

To analyze the quality of the generated textures and their fidelity to the corresponding text prompts, we conducted a detailed user study of our method against four baseline methods. We randomly select 40 meshes from our collected data and feed them along with a text prompt as the input for each method. For each of these 40 selections, we generate 360° rotating view videos using both our method and one of the baseline methods and display them side-by-side. Participants in the study are then requested to select the video that not only better matched the given caption but also exhibited superior quality. The user study yielded a dataset of 2,480 responses from 62 participants. We report the user preferences in Tab. 5.2. The results indicate that our method is notably more effective in producing high-quality textures that are preferred by human evaluators.

5.2.5 Ablation Studies

We first visually evaluate the impact of the attention-guided view synthesis as shown in Fig. 5.6. The results demonstrate that our proposed method with attention-guided view synthesis is able to generate textures with consistent appearance in different

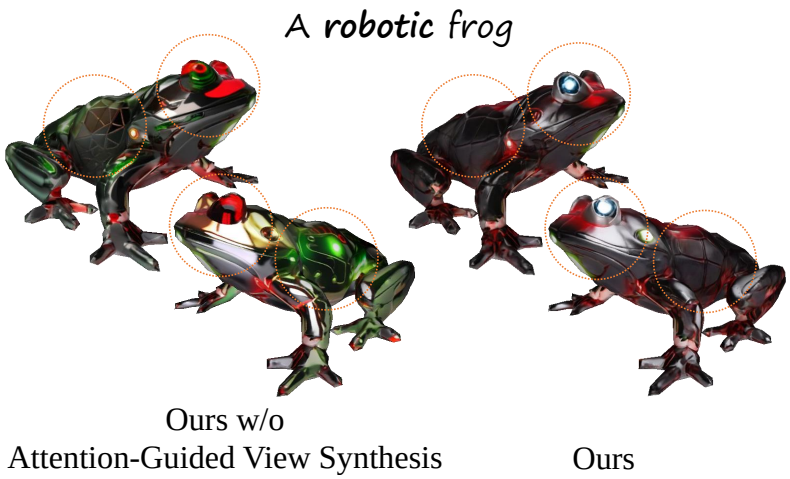


Figure 5.6: Visual comparison of ablation study over attention-guided view synthesis. Without attention-guided view synthesis, the frog has different appearance patterns and color tones over different sides such as eyes and back.

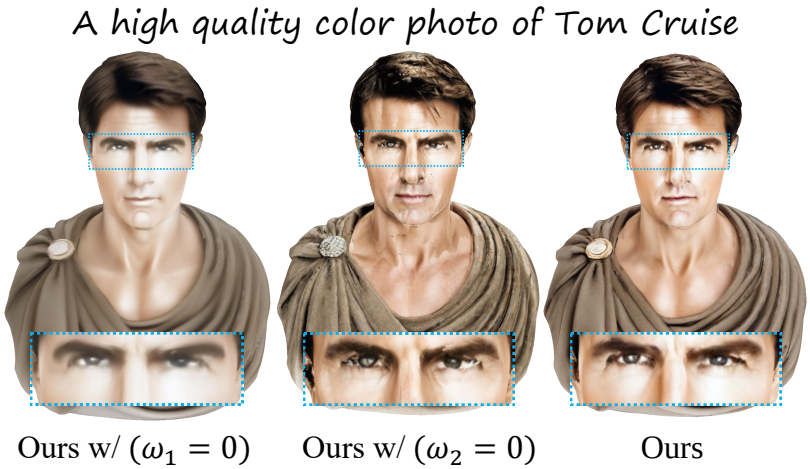


Figure 5.7: Visual comparison of ablation study over T^2GR module.

viewpoints.

We also evaluate the impact of the T^2GR by keeping $\omega_1 = 0$ and $\omega_2 = 0$ in Eqn. 5.14, respectively. As shown in Fig. 5.7, the first figure with $\omega_1 = 0$ lacks high-frequency details and tends to be over-smoothed, while the middle figure with $\omega_2 = 0$ lacks texture guidance and the aggregated texture from all viewpoints is fragmented. We also evaluate the generation quality using FID and KID in Tab. 5.3, which shows that our method with attention-guided view synthesis and T^2GR outperforms other variants by a large margin.

Methods	FID ↓	KID×10^{-3} ↓
Ours w/o AGVS	98.62	5.01
Ours w/ ($\omega_1 = 0$)	95.97	4.60
Ours w/ ($\omega_2 = 0$)	99.46	5.22
Ours	**84.65**	**4.27**

Table 5.3: Ablation study over Attention-Guided View Synthesis (AGVS) and T^2GR.

Figure 5.8: Applications of our proposed texture sampling scheme for text-driven texture editing.

5.2.6 Applications

Our proposed texture sampling scheme can also be applied to texture editing, as shown in Fig. 5.8. It shares the same pipeline with texture generation, but here we replace the depth-aware ControlNet with the MultiControlNet [52] that combines both the depth-guided and edge-guided generation to preserve the original identity, where the canny edges are extracted from the generated views.

5.2.7 Failure Cases

Fig. 5.9 show a failure case of our method caused by significant semantic mismatch between text prompts and geometry.

Figure 5.9: A failure case of our method.

5.3 Summary

In this chapter, I propose a novel texture sampling strategy to generate high quality textures for 3D objects with the help of the pretrained text to image generation models. In the next chapter, I solve the domain-shift problem of the spectral degradation where more than 90% of the spectral information has been lost on RGB images. Since there is no pretrained models for spectral reflectance generation, and a single image is not enough to optimize the network parameters of DIP when the information loss is such high. Thus, I propose a novel learning strategy for spectral reflectance recovery by combining both the information from the training and testing domains.

Chapter 6

Spectral Reflectance Recovery from RGB Images

Unlike traditional RGB images with only three bands (red, green, and blue), the spectral reflectance captured by a hyperspectral imaging system has a higher sampling rate in wavelength and provides more spectral information of the scene. The spectral reflectance of an object is independent of the illumination so that it describes the distinctive intrinsic characteristics of an object's materials, which is widely used in many applications such as remote sensing [282, 283], agriculture [284, 285], medical imaging [286–289], and food quality evaluation [290, 291]. In this chapter, I utilize meta-auxiliary learning (MAXL) [49] to take advantage of both internal and external information for SRR under known illuminations, with the goal to rapidly adapt the trained parameters to an unseen image using only a few steps of gradient descent at test time. In particular, we design a neural architecture featuring two tasks: the primary task focuses on recovering spectral reflectance from RGB images, while the auxiliary task involves reconstructing RGB inputs from the recovered spectral reflectance. We adopt both tasks to train the model on paired inputs and outputs (referred to as external information), and fine-tune the pretrained parameters using a single testing input (referred to as internal information) leveraging solely the auxiliary task. Notably, the fine-tuning process eliminates the need for paired ground truth. Experiments show that MAXL significantly boosts the performance on real data,

which demonstrates the effectiveness of MAXL in reducing domain gap.

6.1 Proposed Method

6.1.1 Overview

The relationship between an RGB image of a scene and its spectral reflectance can be expressed as

$$I^c(x,y) = \int_\lambda S^c(\lambda)L(\lambda)R(x,y,\lambda)d\lambda, \tag{6.1}$$

where I^c represents channel c of the RGB image ($c \in \{Red, Green, Blue\}$), R the spectral reflectance, S^c the CSS of channel c, and L the illumination spectrum. λ refers to the wavelength, and (x, y) are the spatial coordinates. Assume that the number of pixels and the number of sampled spectral bands are N and B, respectively, Eqn. 6.1 can be discretized and represented in matrix form as

$$\mathbf{I} = (\mathbf{S} \odot \mathbf{L}) \cdot \mathbf{R}, \tag{6.2}$$

where $\mathbf{I} \in \mathbb{R}^{3\times N}$ is the RGB image, $\mathbf{R} \in \mathbb{R}^{B\times N}$ is the spectral reflectance, $\mathbf{S} \in \mathbb{R}^{3\times B}$ denotes the CSS, and $\mathbf{L} \in \mathbb{R}^{1\times B}$ denotes the illumination spectrum. $\odot$ is the Hadamard product, and $\cdot$ is the matrix multiplication.

Since the system is under-determined, more images of the same scene under different and independent $\mathbf{L}$ can help to reduce the unknown, which can be formulated as

$$\boldsymbol{\mathcal{I}} = \boldsymbol{\mathcal{H}} \cdot \mathbf{R},\ \boldsymbol{\mathcal{I}} = \begin{bmatrix} \mathbf{I}_1 \\ \vdots \\ \mathbf{I}_M \end{bmatrix},\ \boldsymbol{\mathcal{H}} = \begin{bmatrix} \mathbf{S} \odot \mathbf{L}_1 \\ \vdots \\ \mathbf{S} \odot \mathbf{L}_M \end{bmatrix}, \tag{6.3}$$

where M is the number of illuminations, $\boldsymbol{\mathcal{I}} \in \mathbb{R}^{3M\times N}$ is the stack of RGB images of the same scene. Our goal is to learn a mapping $F(\cdot)$ from $\boldsymbol{\mathcal{I}}$ to $\mathbf{R}$ with known illuminations and unknown CSSs, as

$$\hat{\mathbf{R}} = F(\boldsymbol{\mathcal{I}}, \mathbf{L}_1, \ldots, \mathbf{L}_M). \tag{6.4}$$

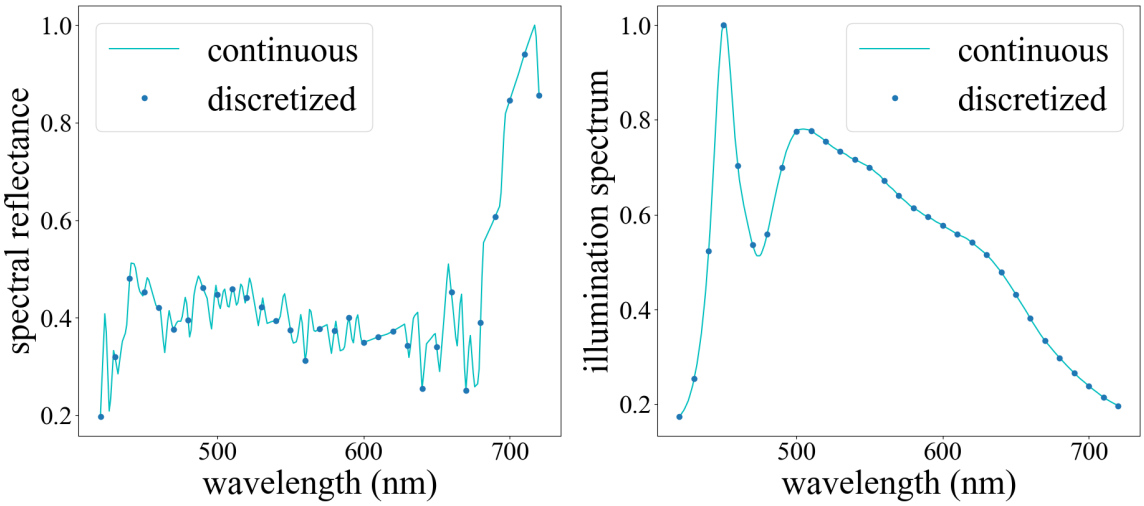


Figure 6.1: The left and right figures show a spectral reflectance curve and the illumination spectrum of a white LED, respectively. We can see that discretization loses high-frequency information.

Instead of naively learning an end-to-end mapping between $\boldsymbol{\mathcal{I}}$ and $\mathbf{R}$, we attempt to take $\boldsymbol{\mathcal{H}}$ into consideration so that the physical relationship of $\boldsymbol{\mathcal{I}}$ and $\mathbf{R}$ can be exploited.

Lin *et al.* [28] prove that all possible solutions of $\hat{\mathbf{R}}$ shares the same component $\hat{\mathbf{R}}^{\parallel}$ within the sub-space of $\boldsymbol{\mathcal{H}}$, where

$$\hat{\mathbf{R}}^{\parallel} = \boldsymbol{\mathcal{H}}^{T} \cdot (\boldsymbol{\mathcal{H}} \cdot \boldsymbol{\mathcal{H}}^{T})^{-1} \cdot \boldsymbol{\mathcal{I}}. \tag{6.5}$$

As we can see, $\hat{\mathbf{R}}^{\parallel}$ can be directly calculated from $\boldsymbol{\mathcal{H}}$ and $\boldsymbol{\mathcal{I}}$, so that they aim at learning the other component $\hat{\mathbf{R}}^{\perp}$ within the null-space of $\boldsymbol{\mathcal{H}}$, and the recovered result is $\hat{\mathbf{R}}^{\parallel} + \hat{\mathbf{R}}^{\perp}$. Nevertheless, simply adopting this strategy to our problem may lead to the following issues:

- Real RGB images are integrated from continuous spectra as in Eqn. 6.1, the discretized form in Eqn. 6.2 and 6.3 are obtained by sub-sampling, resulting in information loss in RGB images (as shown in Fig. 6.1);
- $\mathbf{S}$ is unknown because it varies from sensor to sensor. We have to train an extra network to estimate it from $\boldsymbol{\mathcal{I}}$ and approximate the matrix $\boldsymbol{\mathcal{H}}$;

- The real intensity of illumination depends on the standard exposure settings [28], but our illumination spectra are normalized to $[0, 1]$ which need to be rescaled with factor ω;
- We empirically observe that the back-propagation of the null-space is extremely unstable.

To solve the above mentioned problems, the recovered result needs to be reformulated as

$$\hat{\mathbf{R}} = \hat{\omega}\hat{\mathbf{R}}_{\hat{\boldsymbol{\mathcal{H}}}} + \Delta\hat{\mathbf{R}}, \tag{6.6}$$

$$\hat{\mathbf{R}}_{\hat{\boldsymbol{\mathcal{H}}}} = \hat{\boldsymbol{\mathcal{H}}}^T \cdot (\hat{\boldsymbol{\mathcal{H}}} \cdot \hat{\boldsymbol{\mathcal{H}}}^T)^{-1} \cdot \boldsymbol{\mathcal{I}},\ \hat{\boldsymbol{\mathcal{H}}} = \begin{bmatrix} \hat{\mathbf{S}} \odot \mathbf{L}_1 \\ \vdots \\ \hat{\mathbf{S}} \odot \mathbf{L}_M \end{bmatrix}, \tag{6.7}$$

where $\hat{\omega}$ is the estimated rescaling factor for the illumination, and $\hat{\mathbf{S}}$ denotes the estimated CSS. We directly generate $\Delta\hat{\mathbf{R}} \in \mathbb{R}^{B\times N}$ using the network to avoid the back-propagation of the null-space.

Theorem 1 *All possible solutions of* $\hat{\mathbf{R}}$ *share the same* $\hat{\omega}\hat{\mathbf{R}}_{\hat{\boldsymbol{\mathcal{H}}}}$ *component.*

Proof 1 *Let* $\Delta\boldsymbol{\mathcal{I}}$ *be the lost information of RGB images by discretization,* $\Delta\boldsymbol{\mathcal{H}}$ *be the difference between* $\boldsymbol{\mathcal{H}}$ *and* $\hat{\boldsymbol{\mathcal{H}}}$*, and* $\Delta\omega$ *be the difference between* ω *and* $\hat{\omega}$*.* $\hat{\mathbf{R}}$ *can be rewritten as*

$$\begin{aligned}
\hat{\mathbf{R}} =& \hat{\mathbf{R}}^{\parallel} + \hat{\mathbf{R}}^{\perp} \\
=& (\hat{\omega} + \Delta\omega)(\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}})^T \\
& \cdot ((\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}}) \cdot (\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}})^T)^{-1} \cdot (\boldsymbol{\mathcal{I}} - \Delta\boldsymbol{\mathcal{I}}) + \hat{\mathbf{R}}^{\perp} \\
=& \hat{\omega} \underbrace{\hat{\boldsymbol{\mathcal{H}}}^T \cdot (\hat{\boldsymbol{\mathcal{H}}} \cdot \hat{\boldsymbol{\mathcal{H}}}^T)^{-1} \cdot \boldsymbol{\mathcal{I}}}_{\hat{\mathbf{R}}_{\hat{\boldsymbol{\mathcal{H}}}}} + \Delta\hat{\mathbf{R}}.
\end{aligned} \tag{6.8}$$

■

In addition to the primary task $F(\cdot)$, we utilize the self-supervised RGB reconstruction as the auxiliary task $G(\cdot)$ for test-time adaptation, as

$$\hat{\boldsymbol{\mathcal{I}}} = G(\boldsymbol{\mathcal{I}}, \hat{\mathbf{R}}), \tag{6.9}$$

where the ground truth $\mathbf{R}$ is not needed. We empirically show in our experimental results that the auxiliary task also benefits the primary task, which coincides with the observation reported in [49].

In this paper, $M = 1$ or 2, and $\mathbf{I}_1$ and $\mathbf{I}_2$ represent a pair of RGB images from the same scene illuminated by a white LED $\mathbf{L}_1$ and an amber LED $\mathbf{L}_2$, respectively[1]. The number of sampled spectral bands is 31 from 420nm to 720nm at 10nm increments. More detailed derivations are shown in Appendix D.

6.1.2 Architecture

The overview of our proposed architecture is shown in Fig. 6.2(a). It takes two RGB images $\mathbf{I}_1$ and $\mathbf{I}_2$ as inputs, and utilizes two separate conv layers to extract features. Feature maps from $\mathbf{I}_2$ are simply discarded for $M = 1$ or concatenated with those from $\mathbf{I}_1$ for $M = 2$. The channel size of the initial conv layers are set as 31 and are doubled/halved after downsampling/upsampling. All conv kernels are of size 3×3 and are followed by a LeakyReLU function [292] except those before the concatenations, the element-wise operations $(+, -, \times)$, and the outputs. The output channel size of the auxiliary task is 3 for $M = 1$ and 6 for $M = 2$.

We adopt an encoder-decoder architecture for SRR. Each scale of the encoder contains a conv layer followed by three resblocks. The decoder is similar but has an extra deconv layer to upscale the spatial dimension and a concatenation for skip-connection. We utilize four spectral-attention blocks [129] to extract spectral correlation after the encoder.

[1]Since the spectrum of an amber LED has a narrow band and is zero in most wavelengths, it can only serve as an auxiliary light source instead of the main one.

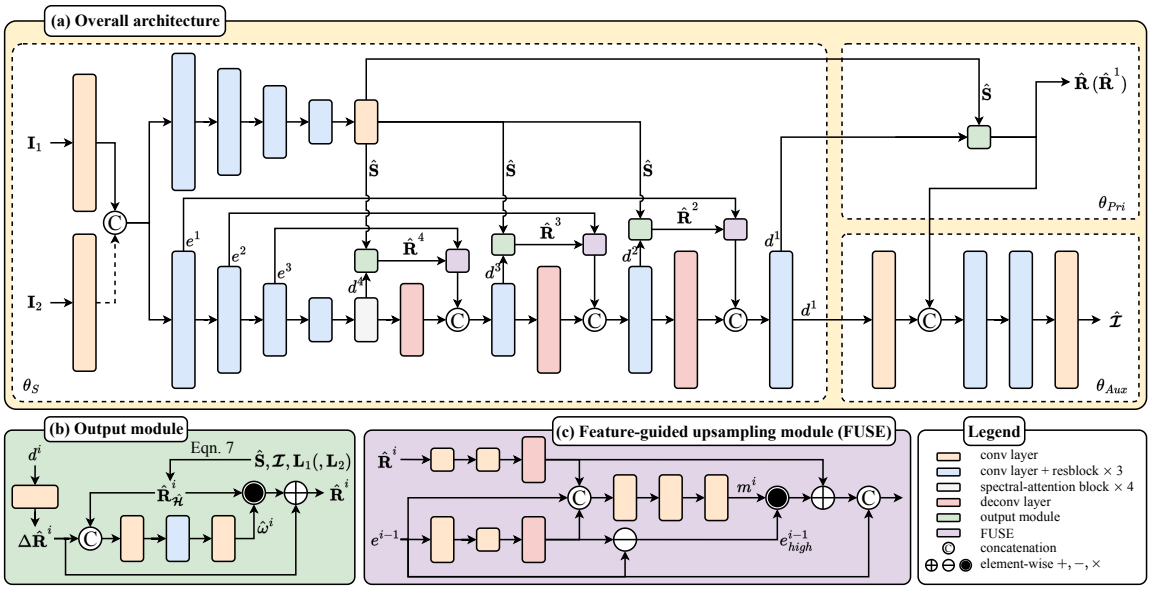


Figure 6.2: Our proposed network architecture for SRR and meta-auxiliary learning. e^i and d^i denote the feature map from the encoder and the decoder, respectively, of scale i ($i \in \{1, 2, 3, 4\}$), $\hat{\mathbf{R}}^i$ is the recovered reflectance of scale i and $\hat{\mathbf{R}}^1$ represents the final recovered result $\hat{\mathbf{R}}$. The RGB image stack $\boldsymbol{\mathcal{I}}$ is downsampled to the corresponding scale before calculating $\hat{\mathbf{R}}^i_{\hat{\boldsymbol{\mathcal{H}}}}$. θ_{Pri} and θ_{Aux} denote the task-specific parameters for the primary task and the auxiliary task, respectively, and θ_S denotes the shared parameters. Our network consists of an encoder network to estimate the CSS, an encoder-decoder architecture for SRR, four spectral-attention layers to extract spectral correlation, output modules to generate $\hat{\mathbf{R}}^i$, and feature-guided upsampling modules (FUSEs) to upsample $\hat{\mathbf{R}}^i$ with the guidance of e^{i-1}. The global average pooling before $\hat{\mathbf{S}}$ is omitted to simplify the illustration.

To explicitly estimate the CSS, we utilize the same encoder as the feature extractor and a conv layer to reduce the channel size to 3×31, following which is a global average pooling layer. Despite using the same architecture, we do not share the parameters of the two encoders because we empirically observe that the network is difficult to converge.

In the decoder, we adopt a pyramid scheme [21, 293, 294] by generating a spectral reflectance at the end of each scale, which can act as a "hint" for the prediction of finer scales. As shown in Fig. 6.2(b), we downsample the RGB image stack $\boldsymbol{\mathcal{I}}$ with bilinear interpolation to match the spatial dimension at scale i ($i \in \{1, 2, 3, 4\}$) and calculate $\hat{\mathbf{R}}^i_{\hat{\boldsymbol{\mathcal{H}}}}$ with the estimated CSS $\hat{\mathbf{S}}$. The rescaling factor $\hat{\omega}^i$ is learned from

the concatenation of $\hat{\mathbf{R}}^{i}_{\hat{\boldsymbol{\mathcal{H}}}}$ and $\Delta\hat{\mathbf{R}}^{i}$. $\hat{\mathbf{R}}^{i}$ is obtained by Eqn. 6.8 and $\hat{\mathbf{R}}^{1}$ is our final recovered result $\hat{\mathbf{R}}$ in Eqn. 6.4.

A simple approach to fuse $\hat{\mathbf{R}}^{i}$ with features from scale $i-1$ is to directly upsample $\hat{\mathbf{R}}^{i}$ to scale $i-1$ with deconv layers and then concatenate them together [293]. Nevertheless, the upsampled spectral reflectance lacks high-frequency information which needs further refinement. Inspired by the generalized Laplacian pyramid algorithm [295] that fuses a high-resolution panchromatic image with a low-resolution multispectral image by feeding the weighted high-frequency information from the panchromatic image to the multispectral image, we propose a new **F**eature-g**U**ided up**S**ampling modul**E** (FUSE) that utilizes the feature e^{i-1} from scale $i-1$ of the encoder to guide the upsampling. As shown in Fig. 6.2(c), we exploit a downsampling-upsampling scheme to get the low-pass components of e^{i-1} and then subtract it from e^{i-1} for high-pass components e^{i-1}_{high}. The remaining low-pass components are concatenated with the upsampled recovery output $\hat{\mathbf{R}}^{i}$ and e^{i-1} to extract local correlation, and generate local gain factor m^{i} to reweight high-pass components which supplement the upsampled $\hat{\mathbf{R}}^{i}$ for refinement.

Most parameters of the two tasks are shared. As shown in Fig. 6.2(a), we separate the parameters θ of the whole network into three components, θ_S, θ_{Pri} and θ_{Aux}, where θ_S represents the shared parameters, θ_{Pri} and θ_{Aux} represent the task-specific parameters for the primary task and the auxiliary task, respectively. We feed the output of the last shared resblock into two branches, one for generating the spectral reflectance $\hat{\mathbf{R}}$ (primary task), and the other with $\hat{\mathbf{R}}$ as an extra input to reconstruct the original RGB images as in Eqn. 6.9 (auxiliary task), so that the parameters of the primary task can be updated with only the auxiliary loss during test time. We adopt the L1 loss for both tasks as

$$\mathcal{L}_{Pri}(\theta_S, \theta_{Pri}) = \left\| \mathbf{S} - \hat{\mathbf{S}} \right\|_1 + \sum_{i=1}^{4} \left\| \mathbf{R}^{i} - \hat{\mathbf{R}}^{i} \right\|_1, \tag{6.10}$$

$$\mathcal{L}_{Aux}(\theta_S, \theta_{Pri}, \theta_{Aux}) = \left\| \boldsymbol{\mathcal{I}} - \hat{\boldsymbol{\mathcal{I}}} \right\|_1. \tag{6.11}$$

Algorithm 2: Meta-auxiliary Training

Input: $(\boldsymbol{\mathcal{I}}, \mathbf{S}, \mathbf{R})$ triples
α, β: learning rates
Output: θ: meta-auxiliary trained parameters

Randomly initialize θ, $\theta = \{\theta_S, \theta_{Pri}, \theta_{Aux}\}$
while *not converged* **do**
  Sample a batch of triples $\{\boldsymbol{\mathcal{I}}^k, \mathbf{S}^k, \mathbf{R}^k\}_{k=1}^K$
  Evaluate pre-training loss $\mathcal{L}_{Pre}$ by Eqn. 6.12
  Update θ with respect to $\mathcal{L}_{Pre}$
end
while *not converged* **do**
  Sample a batch of triples $\{\boldsymbol{\mathcal{I}}^k, \mathbf{S}^k, \mathbf{R}^k\}_{k=1}^K$
  for *each* k **do**
    Evaluate auxiliary loss $\mathcal{L}_{Aux}$ by Eqn. 6.11
    Compute adapted parameters θ^k with gradient descent by Eqn. 6.13
    Update θ_{Aux} by Eqn. 6.16
  end
  Update θ_S and θ_{Pri} by Eqn. 6.15
end

Directly updating the randomly initialized parameters with meta-auxiliary learning is time-consuming and unstable. Hence, we first initialize all the parameters by pre-training with the summation of the primary and the auxiliary losses following [49], which is formulated as

$$\mathcal{L}_{Pre}(\theta) = \mathcal{L}_{Pri}(\theta_S, \theta_{Pri}) + \mathcal{L}_{Aux}(\theta_S, \theta_{Pri}, \theta_{Aux}). \tag{6.12}$$

6.1.3 Meta-auxiliary Learning

The goal of meta-learning is to learn a general model for different tasks, which is able to rapidly adapt to new tasks with only a few steps [145]. In our case, we regard each triple $(\boldsymbol{\mathcal{I}}^k, \mathbf{S}^k, \mathbf{R}^k)$ (k represents the index) as a task[2] $\mathcal{T}^k$ of meta-learning.

[2]To distinguish from the primary and auxiliary tasks, we utilize "meta-task" in the following text.

Meta-auxiliary training

Given a meta-task $\mathcal{T}^k$, we first adapt the pre-trained parameters θ using several gradient descent updates based on only the auxiliary loss

$$\theta^k = \theta - \alpha \nabla_\theta \mathcal{L}_{Aux}^{\mathcal{T}^k}(\theta_S, \theta_{Pri}, \theta_{Aux}), \tag{6.13}$$

where α represents the adaptation learning rate. The update of Eqn. 6.13 includes all the parameters with only $\boldsymbol{\mathcal{I}}^k$ utilized.

The key of making the pre-trained parameters θ suitable for test-time adaptation is to update θ_S and θ_{Pri} of the primary task in the direction of minimizing the auxiliary loss. Thus, the meta-objective can be defined as

$$\arg\min_{\theta_S, \theta_{Pri}} \sum_{k=1}^{K} \mathcal{L}_{Pri}^{\mathcal{T}^k}(\theta_S^k, \theta_{Pri}^k), \tag{6.14}$$

where K is the number of sampled meta-tasks. The meta-optimization is then performed on Eqn. 6.14 via stochastic gradient descent

$$\theta \leftarrow \theta - \beta \sum_{k=1}^{K} \nabla_\theta \mathcal{L}_{Pri}^{\mathcal{T}^k}(\theta_S^k, \theta_{Pri}^k), \tag{6.15}$$

where β represents the meta-learning rate. Note that the gradient in Eqn. 6.15 is calculated based on θ^k but updates the original θ in Eqn. 6.13. The full algorithm is demonstrated in Alg. 2. Only θ_S and θ_{Pri} are updated in the outer loop, and θ_{Aux} is updated in the inner loop as

$$\theta_{Aux} \leftarrow \theta_{Aux} - \alpha \nabla_\theta \mathcal{L}_{Aux}^{\mathcal{T}^k}(\theta_{Aux}). \tag{6.16}$$

Test-time adaptation

At test-time, we simply fine-tune the meta-learned parameters on a testing $\boldsymbol{\mathcal{I}}$ with Eqn. 6.13 using several steps of gradient descent as shown in Alg. 3.

Algorithm 3: Test-time Adaptation

Input: A testing RGB image stack $\boldsymbol{\mathcal{I}}$
n: number of gradient updates
α: adaptation learning rate
Output: Recovered spectral reflectance $\hat{\mathbf{R}}$

Initialize network parameters with meta-learned θ
for n *steps* **do**
  Evaluate auxiliary loss $\mathcal{L}_{Aux}$ by Eqn. 6.11
  Update $\theta \leftarrow \theta - \alpha \nabla_{\theta} \mathcal{L}_{Aux}(\theta_S, \theta_{Pri}, \theta_{Aux})$
end
return $\hat{\mathbf{R}}$ from Eqn. 6.4

6.2 Experiments

6.2.1 Datasets

Synthetic data

TokyoTech [296] contains 16 spectral reflectance images from 420nm to 1000nm at 10nm increments, and we utilize the first 31 bands. ICVL [43] contains 201 hyperspectral images under daylight illumination from 400nm to 1000nm at 1.5nm increments. We divide the hyperspectral images by the daylight illumination spectrum [297] to simulate the spectral reflectance, then downsample from 420nm to 720nm at 10nm increments. We randomly select 75% images from two datasets for training and the rest for testing. Jiang *et al.* [298] provide 28 CSSs and we randomly select 23 for generating training inputs and the rest for testing. The illumination spectra of white and amber LEDs are collected with a Specim IQ mobile hyperspectral camera and are downsampled using the same scheme. We normalize two illumination spectra to the range $[0, 1]$ and keep their relative intensity. To simulate the continuous spectra, we interpolate the spectral reflectance spectra, CSSs and illumination spectra at 1nm increments before generating RGB images with Eqn. 6.2.

Real data

To evaluate the robustness of models trained on synthetic data, we collect 25 spectral reflectance images with a Specim IQ and the corresponding RGB images under white and amber LEDs with a Canon 6D camera which is not included in the training data. The illumination spectra are represented as the spectral radiance of a white reference panel under two LEDs. The reflectance spectra are downsampled from 420nm to 720nm at 10nm increments. We first convert the downsampled spectra to RGB using a randomly selected CSS from [298], then we adopt feature matching with SIFT features [299] to align the images of two cameras. Note that images without enough features for matching are removed. Feasibility analysis of data capture in real world is shown in Appendix D.

6.2.2 Implementation Details

All images are linearly rescaled to the range $[0, 1]$. Training images are cropped into 128×128 patches with a stride of 64, and are augmented by random flips. The batch size is set to 64. We adopt the Adam optimizer [249] for pre-training with a learning rate 10^{-4} and the Cosine Annealing scheme [300] for 300 epochs. During the meta-auxiliary learning, we set α and β to 1×10^{-2} and 5×10^{-5}, respectively. For test-time adaptation, we perform $n = 5$ gradient descent updates. All experiments are conducted on a single NVIDIA RTX A6000 GPU with 48GB of RAM.

We adopt the mean absolute error (MAE), rooted mean square error (RMSE), spectral angle similarity (SAS [301]), peak signal-to-noise ratio (PSNR [302]) and structural similarity (SSIM [256]) as the metrics to evaluate the performance of SRR.

6.2.3 Quantitative Comparison

We first evaluate the performance of $M = 1$. We compare our method with 6 state-of-the-art methods for spectral reconstruction from a single RGB image, including HSCNN+ [128], MSDCNN [134], PADFMN [130], QDO [137], MST++ [129], and

Methods	Synthetic data					Real data				
	MAE↓	RMSE↓	SAS↓	PSNR↑	SSIM↑	MAE↓	RMSE↓	SAS↓	PSNR↑	SSIM↑
HSCNN+ [128]	0.1261	0.1594	0.1418	16.96	0.7837	0.3107	0.3526	0.5521	9.11	0.3877
MSDCNN [134]	0.0877	0.1124	0.1027	19.76	0.8400	0.3136	0.3563	0.5585	9.02	0.3821
PADFMN [130]	0.0851	0.1102	0.1010	20.15	0.8257	0.2746	0.3214	0.5217	9.93	0.3770
QDO [137]	0.1494	0.1889	0.1295	15.14	0.7759	0.4665	0.5330	0.6139	5.52	0.2883
MST++ [129]	0.0724	0.0927	0.0865	21.72	0.8611	0.2400	0.2944	0.5312	10.69	0.3383
DRCRN [30]	0.0750	0.0998	0.0894	20.98	0.8429	0.2717	0.3154	0.5501	10.09	0.3992
Ours (pre-trained)	0.0625	0.0828	0.0748	22.91	0.8818	0.2313	0.2783	0.5174	11.19	0.4721
Ours	**0.0607**	**0.0809**	**0.0734**	**23.09**	**0.8833**	**0.2136**	**0.2590**	**0.4934**	**11.84**	**0.4947**
Ours† (pre-trained)	0.0580	0.0778	0.0696	23.67	0.8891	0.1657	0.2137	0.4426	13.56	0.5641
Ours†	**0.0575**	**0.0771**	**0.0691**	**23.72**	**0.8905**	**0.1536**	**0.1997**	**0.4095**	**14.23**	**0.5796**

Table 6.1: Quantitative evaluations. All compared methods are trained on the synthetic data. Ours and Ours† represent the $M = 1$ (white LED only) and $M = 2$ (white&amber LEDs), respectively. "pre-trained" represents the model without meta-auxiliary training and test-time adaptation.

DRCRN [30]. For fair comparison, we remove the DOE optimization of QDO. All of these competing methods are retrained with our selected synthetic data. The evaluation results are listed in the first part of Tab. 6.1. We can see that our proposed architecture outperforms other methods even with only the pre-trained model, and the MAXL obviously improves the performance especially on the challenging real data (0.65dB), which demonstrates the importance of utilizing internal information. Since our model is trained on the synthetic data, it is reasonable that the performance gain of MAXL on the synthetic data is not as much as that on the real data.

We also evaluate the effectiveness of the extra illumination ($M = 2$). As reported in the second part of Tab. 6.1, it demonstrates 0.63dB and 2.39dB improvement over $M = 1$ on synthetic data and real data, respectively.

The evaluation of computational complexity on images of size 1392×1303 is shown in Tab. 6.2. We can see that our method without MAXL is faster than most of the other methods with comparable number of parameters, and the test-time adaptation only takes seconds.

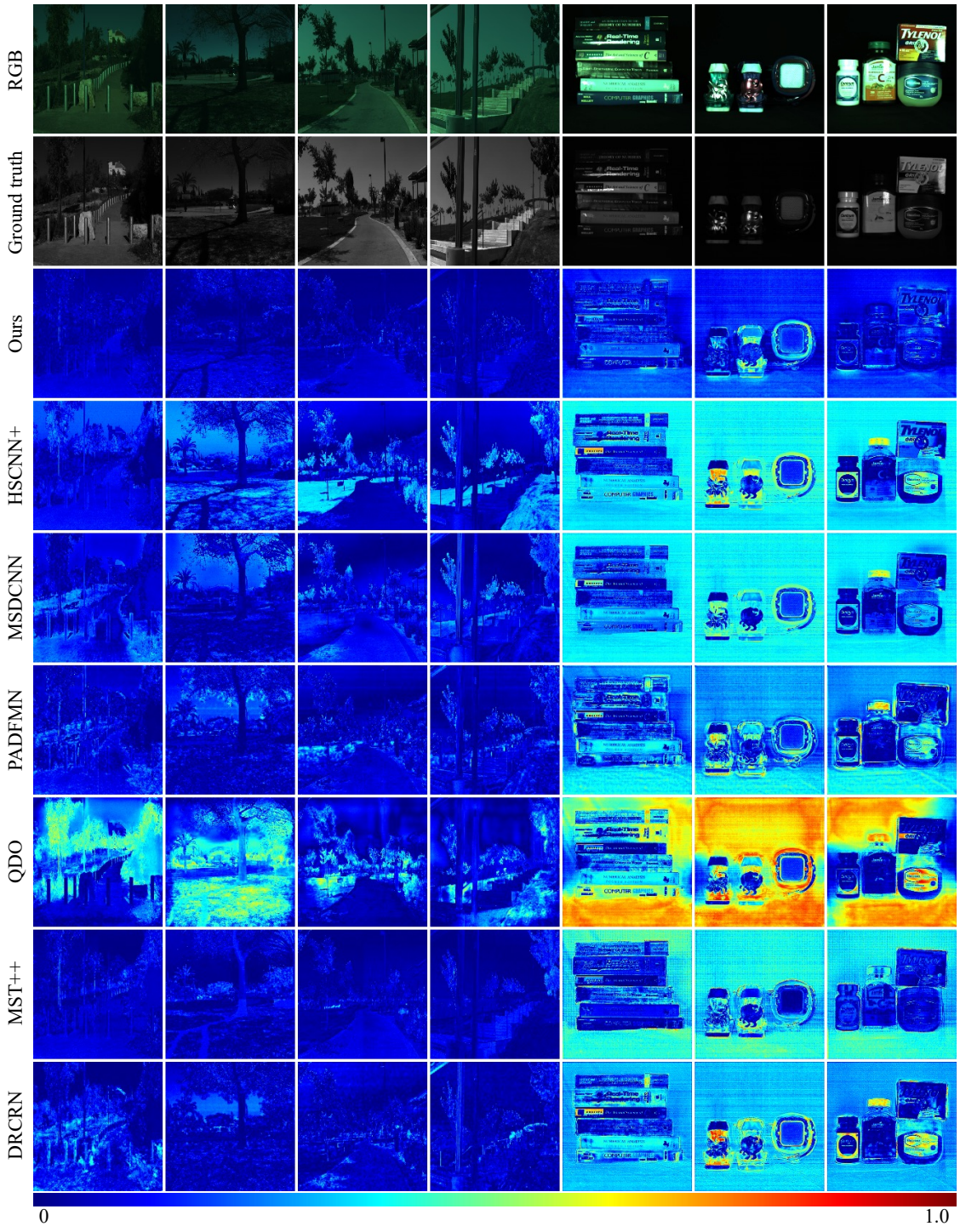


Figure 6.3: Qualitative comparison of error maps (MAE between the recovered results and the ground truth) with state-of-the-art approaches. The first four columns are from the synthetic data and last three columns are from our collected real data.

Figure 6.4: Qualitative comparison of error maps (MAE between the recovered results and the ground truth) of our method with/without MAXL for $M = 1$ and $M = 2$ on real data.

Methods	#Params	FLOPs	Inference time
HSCNN+ [128]	**7.98×10^5**	2.88×10^{12}	**0.020 sec**
MSDCNN [134]	2.67×10^7	2.27×10^{12}	0.023 sec
PADFMN [130]	3.17×10^7	9.02×10^{12}	0.334 sec
QDO [137]	1.47×10^9	1.38×10^{12}	0.308 sec
MST++ [129]	1.62×10^6	**1.20×10^{12}**	0.239 sec
DRCRN [30]	9.48×10^6	3.23×10^{13}	0.538 sec
Ours (pre-trained)	2.41×10^7	5.03×10^{12}	0.145 sec
Ours	2.41×10^7	2.57×10^{13}	6.018 sec
Ours† (pre-trained)	2.42×10^7	5.10×10^{12}	0.153 sec
Ours†	2.42×10^7	2.61×10^{13}	6.082 sec

Table 6.2: Evaluations of computational complexity. Ours and Ours† represent the $M = 1$ (white LED only) and $M = 2$ (white&amber LEDs), respectively. "pre-trained" represents the model without meta-auxiliary training and test-time adaptation. All evaluations are calculated on images of size 1392×1303.

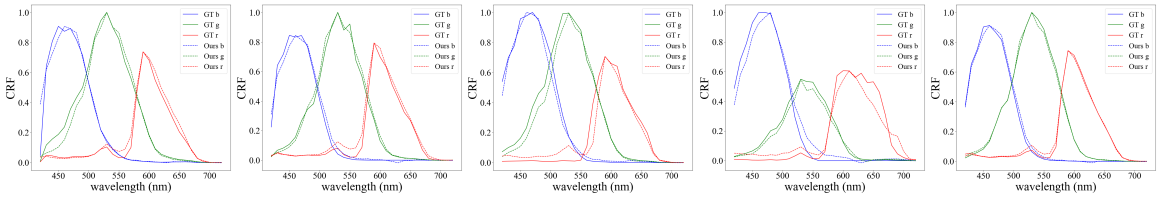


Figure 6.5: Visual comparison of the ground truth and our estimated CSSs.

6.2.4 Qualitative Comparison

The qualitative comparison results of the 630nm band of the spectral reflectance are shown in Fig. 6.3. The RGB images under white LED, the ground truth, and the error maps of all competing methods are shown from top to bottom. The first four columns and last three columns show the results from synthetic data and real data, respectively. We can see that our method with MAXL performs better than others and is more robust on real data. More qualitative evaluations are shown in the Appendix D.

Methods	MAE↓	RMSE↓	SAS↓	PSNR↑	SSIM↑
Ours (pre-trained)	**0.0625**	**0.0828**	**0.0748**	**22.91**	**0.8818**
w/o pyramid	0.1277	0.1585	0.1432	16.58	0.7420
w/o FUSE	0.0676	0.0887	0.0803	22.27	0.8738
w/ zero m^i in FUSE	0.0711	0.0922	0.0830	22.08	0.8696
w/o $\hat{\mathbf{R}}^i_{\hat{\boldsymbol{\mathcal{H}}}}$	0.0669	0.0876	0.0798	22.55	0.8770
w/o $\hat{\omega}^i$	0.0691	0.0909	0.0824	22.24	0.8726
w/o $\Delta\hat{\mathbf{R}}^i$	0.3485	17.4572	0.8131	1.19	0.3836
w/ ground truth CSSs	**0.0621**	**0.0824**	**0.0744**	**22.95**	**0.8818**
w/o spectral-attention	0.0730	0.0958	0.0894	21.44	0.8678
w/o auxiliary task	0.0674	0.0888	0.0796	22.46	0.8703

Table 6.3: Ablation studies of network components.

Methods	MAE↓	RMSE↓	SAS↓	PSNR↑	SSIM↑
Ours (pre-trained)	0.0625	0.0828	0.0748	22.91	0.8818
w/ meta-auxiliary training	0.0611	0.0814	0.0739	23.03	0.8828
w/ test-time adaptation	0.0624	0.0827	0.0745	22.92	0.8822
w/ MAXL	**0.0607**	**0.0809**	**0.0734**	**23.09**	**0.8833**

Table 6.4: Ablation studies of learning strategies.

Fig. 6.4 visualizes the effect of using one ($M = 1$) or two ($M = 2$) illuminations with/without MAXL on real data. It shows that the extra illumination can help to reduce the overall error of the entire image, and the MAXL benefits some local details.

To evaluate the generated CSSs of five selected testing cameras, we display the visual comparison between the ground truth and our estimation in Fig. 6.5. It demonstrates that our proposed method can accurately estimate the CSSs that are unseen during the training.

Methods	MAE↓	RMSE↓	SAS↓	PSNR↑	SSIM↑
$n = 0$	0.0625	0.0828	0.0748	22.91	0.8818
$n = 1$	0.0613	0.0816	0.0737	23.01	0.8826
$n = 2$	0.0612	0.0812	0.0736	23.02	0.8828
$n = 3$	0.0611	0.0814	0.0736	23.03	0.8828
$n = 4$	0.0611	0.0812	0.0735	23.03	0.8830
$n = 5$	**0.0607**	**0.0809**	**0.0734**	**23.09**	**0.8833**
$n = 6$	0.0609	0.0813	**0.0734**	23.07	0.8827

Table 6.5: Ablation studies of number of gradient descent updates n.

Methods	MAE↓	RMSE↓	SAS↓	PSNR↑	SSIM↑
$M = 1$	0.0625	0.0828	0.0748	22.91	0.8818
$M = 2$	0.0580	0.0778	0.0696	23.67	0.8891
$M = 3$	**0.0555**	**0.0742**	**0.0674**	**23.97**	**0.8939**

Table 6.6: Ablation studies of number of different illuminations M.

6.2.5 Ablation Studies

We conduct ablation studies on the synthetic data. As shown in Tab. 6.3, pyramid learning (multi-scale outputs) plays a vital role in the performance, and a proper fusion strategy is also important compared with no FUSE (simply output encoder feature e^{i-1} in FUSE) and zero m^i. We also remove $\hat{\mathbf{R}}^i_{\hat{\boldsymbol{\mathcal{H}}}}$ (use $\Delta\hat{\mathbf{R}}^i$ as $\hat{\mathbf{R}}^i$), $\Delta\hat{\mathbf{R}}^i$ (use $\hat{\omega}^i\hat{\mathbf{R}}^i_{\hat{\boldsymbol{\mathcal{H}}}}$ as $\hat{\mathbf{R}}^i$) and $\hat{\omega}^i$ (use $\hat{\mathbf{R}}^i_{\hat{\boldsymbol{\mathcal{H}}}} + \Delta\hat{\mathbf{R}}^i$ as $\hat{\mathbf{R}}^i$) in all output modules to investigate the impact of the subspace component. We can see that the physical properties of the spectral reflectance can benefit its recovery, but its subspace component alone is insufficient. Utilizing the ground truth CSSs to calculate the $\hat{\boldsymbol{\mathcal{H}}}$ can further improve the results. Besides, the performance gain from the spectral-attention blocks illustrates the effectiveness of spectral correlation. The experimental results without the auxiliary task also demonstrate that it can help the optimization of the primary task.

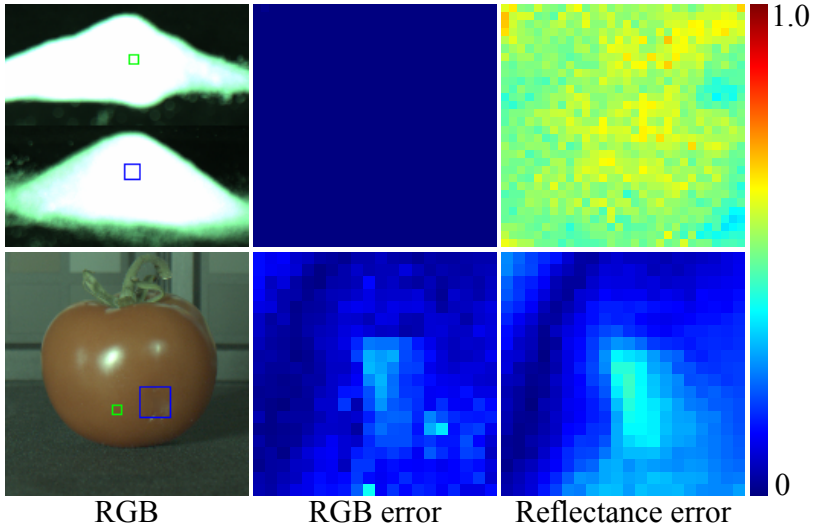


Figure 6.6: The application results of recovered spectral reflectance. In each row, we randomly extract a pixel from the green box as the reference (the source material) and regard pixels from the blue box as the observation (the target material). A smaller green box is to reduce the variance of the reference. Then we calculate the error maps (MAE) between the reference and the observation for both RGB values and recovered spectral reflectances. The green and the blue box in the first row represent the salt and the sugar, respectively. The green and the blue box in the second row represent the flawless tomato peel and the region with a puncture, respectively.

As shown in Tab. 6.4, after fine-tuning the pre-trained model with meta-auxiliary training, the evaluation results show an improvement but are still sub-optimal. We also evaluate the performance of direct test-time adaptation without meta-auxiliary training. While the performance improvement is minor, we do not observe the catastrophic forgetting as mentioned in a previous research [49].

We also investigate the effect of gradient descent update step n as reported in Tab. 6.5. We choose $n = 5$ for the best performance. More update steps may lead to the overfitting on the auxiliary task. Note that we utilize the same n during training and testing.

In Section 6.2.3, we illustrate the performance of using one ($M = 1$) or two ($M = 2$) illuminations. To further demonstrate the robustness of our proposed architecture with more illuminations, we utilize the spectrum of a halogen light as the illumination $\mathbf{L}_3$ to synthesize the input RGB image $\mathbf{I}_3$. Reported in Tab. 6.6 are the results with

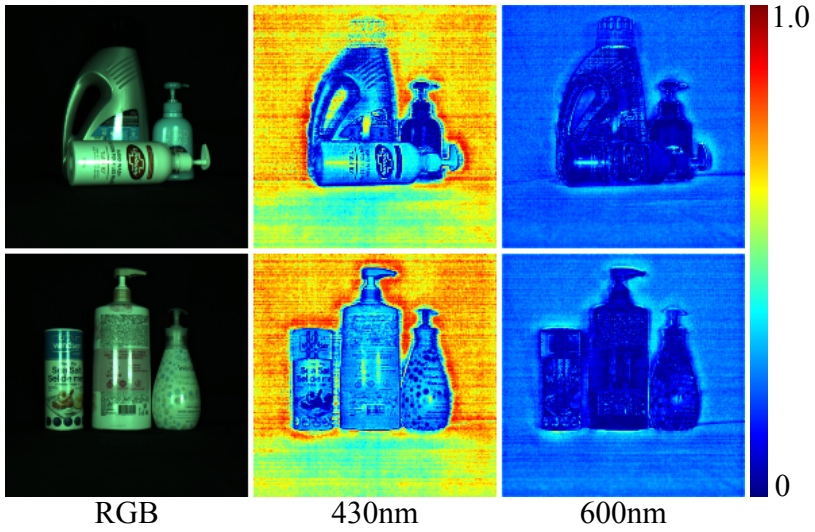


Figure 6.7: Error maps of our recovered 430nm and 600nm bands.

$1 \sim 3$ illuminations. As we can see, utilizing a third illumination can further improve the performance of recovery.

6.2.6 Applications

Spectral reflectance describes the distinctive intrinsic characteristics of an object's material or composition and is widely leveraged for material recognition [303–305]. For example, it has been found to be a more reliable cue for assessing the quality of food, particularly fruits, compared to RGB images [290]. To demonstrate that our recovered spectral reflectance possesses the same property, we conduct experiments of distinguishing between salt and sugar, and detecting fruit puncture in a tomato. The objects in each case have similar RGB colors, with salt and sugar both appearing white, and the tomato peel and pulp both appearing red.

Fig. 6.6 shows the two example results. We can see that the discrepancy of different materials (salt and sugar, tomato peel and pulp) are more distinguishable on our recovered spectral reflectance than that on the original RGB image. For example, the error between the salt and sugar on RGB images is only 1.78×10^{-5} but 0.53 on the recovered spectral reflectance, and the error between the tomato peel and pulp on RGB images and spectral reflectance are 0.09 and 0.17, respectively.

6.2.7 Failure Cases

Our method is limited on bands that have little impact on the RGB images, such as marginal bands, which is a common issue for most approaches. As shown in Fig. 6.1 and Fig. 6.5, the illumination spectra and CSSs are heterogeneous, and the intensity of marginal bands (*e.g.*, 430nm) is much lower than that of central bands (*e.g.*, 600nm). As a result, marginal bands are harder to recover. The error maps of our recovered 430nm and 600nm bands are shown in Fig. 6.7, which illustrate that the errors on the marginal bands are higher than that of central bands.

6.3 Summary

This chapter proposes a novel meta-auxiliary learning scheme for spectral reflectance recovery from RGB images. Although the domain-shift problem between the synthetic and real data has not been totally solved, the performance on the real collected data is significantly better than models without the meta-auxiliary learning. By the end of this chapter, I focus on solving the problem of the low level vision tasks. In the next chapter, I propose to address the glass surface segmentation where the existing semantic segmentation methods fail to detect the glass correctly.

Chapter 7

Glass Segmentation with RGB-Thermal Image Pairs

In this chapter, I first analyse the physical feasibility of combining visible and infrared light for glass segmentation. A novel transformer-based architecture is proposed for multi-modality fusion. Experiments show that utilizing RGB-T image pairs performs significantly better than only using RGB or thermal image.

7.1 Physical Analysis

Glass could have different meanings, broad or narrow, and different types of glass have different properties and applications. In this section, we first explain the different glass types, their applications and optical properties. Then we narrow down the glass type to daily transparent glass as the focus of this paper which is also the most difficult to segment. Lastly, we introduce our RGB-T fusion idea and discuss the limitations of other alternative solutions.

Glass is non-crystalline and can be categorized into two types based on its compositions [307]. (1) Silicate glass, based on silicon dioxide (SiO_2) which is abundant on earth in the form of quartz and beach sands. SiO_2 has a very high melting temperature ($\sim 1700°C$), which is hard to work with, and hence, other substances are often added, *e.g.*Na_2O, CaO, *etc.*, to lower the melting temperature and to tailor to different applications: vitreous silica (100% SiO_2) is used for furnace tubes, soda-lime

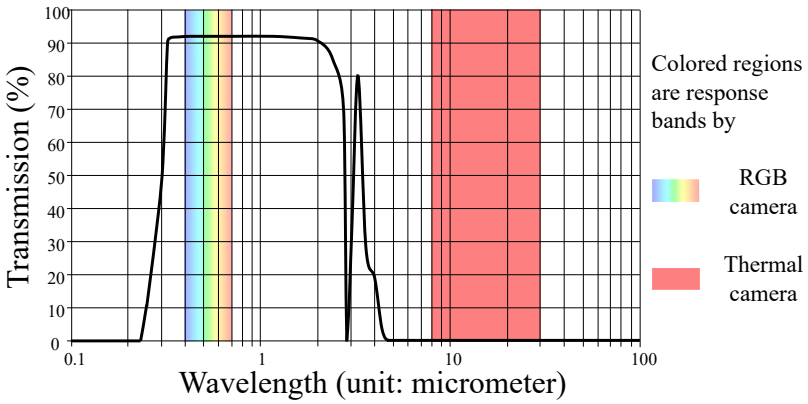


Figure 7.1: Typical glass spectral transmission curve [306] and response bands of RGB and thermal cameras (colored regions).

silicate (72% SiO_2 with Na_2O and CaO) is used for windows, containers and tableware, "crown" (69% SiO_2 with Na_2O, B_2O_3 and K_2O) is used for optical lens, and aluminosilicate (53% SiO_2 with Al_2O_3) is used for fibreglass, *etc.*. (2) Non-silicate based glass includes amorphous metals, chalcogenides, fluorides, polymers, *etc.*

For optical property, glass is transparent to visible light because there are no grain boundaries (the interface between two crystals formed during cooling) which scatter light in polycrystalline materials [308]. Meanwhile, silicate glasses absorb light with wavelengths longer than $4\mu m$, which makes them opaque to long infrared light[1] (Fig. 7.1). Non-silicate glass could sometimes transmit long infrared light, *e.g.*, fluoride glasses and chalcogenide glasses can transmit light up to $7\mu m$ and $18\mu m$, respectively, but they are used for infrared imaging, infrared fiber, and CD/DVDs.

Some processing can change the transparency slightly. Glass *equally* transmits visible light at different wavelengths (Fig. 7.1), but can appear tinted after adding some metallic oxides that absorb light of certain wavelengths, *e.g.*, blue by cobalt, green by chromium. Sandblasting or acid etching clear glass creates frosted glass which is translucent. Low-emissivity (Low-e) glass is glass with a thin coating layer to prevent transmission of light over $780nm$ and is usually used as a window to the

[1]In this paper, these words are interchangeable, "thermal radiation, thermal energy, long infrared light, long-wave infrared light", all referring to electromagnetic radiation of 8 to 12 μm.

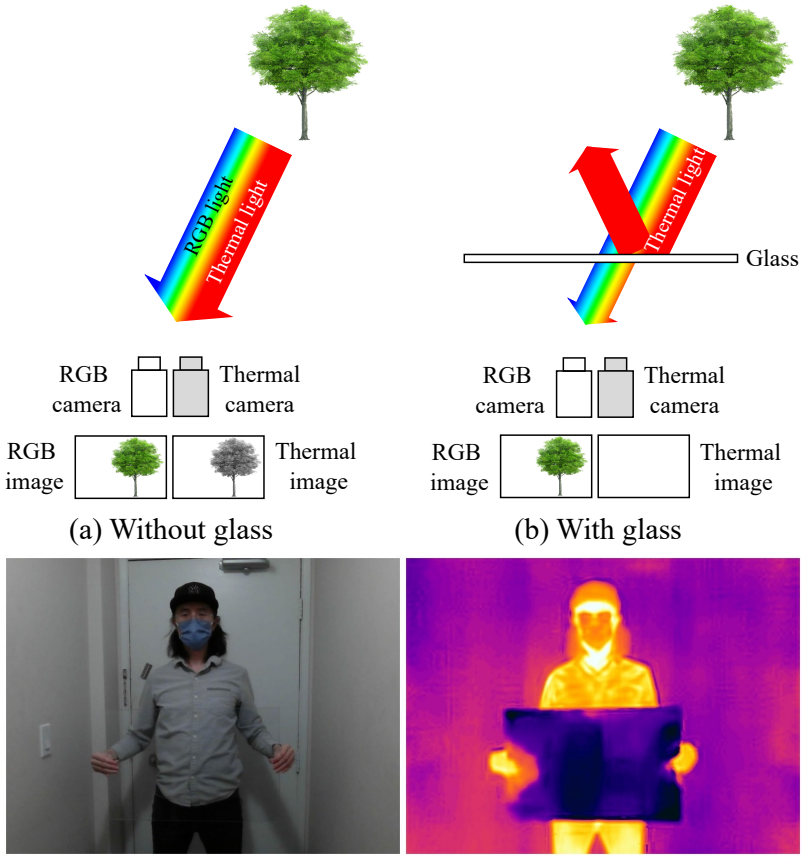


Figure 7.2: A toy illustration for the imaging models of RGB and thermal cameras without and with glass in the scene (a,b). The glass plate held by the person is invisible in the RGB image while visible in the thermal image (c).

outside. Note that no coating can increase the transmission rate.

In this thesis, **we limit the scope to daily transparent glass**, *i.e.*, the plate glass normally seen in our daily life such as glass windows, doors and tables, which is the most difficult to detect and segment. The daily glass in human-made environments is mostly silicate-based, more specifically, soda-lime silicate (SiO_2 + Na_2O + CaO). Silica glass is like a band-pass optical filter which has a cut-on wavelength $350nm$ and a cut-off wavelength $4\mu m$ (Fig. 7.1). It has high transmission in the visible band ($0.4\mu m \sim 0.7\mu m$) but low in the thermal band ($8\mu m \sim 12\mu m$).

Motivated by this fact, **we propose to use an RGB-T image pair to detect and segment glass**. A toy example is shown in Fig. 7.2(a, b). The tree behind the

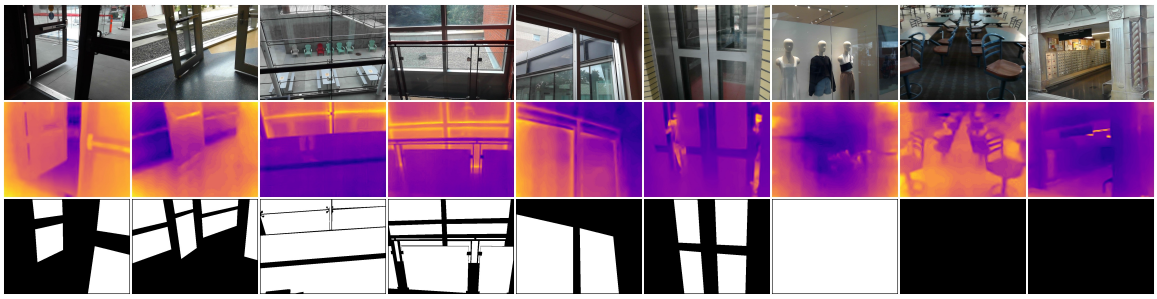


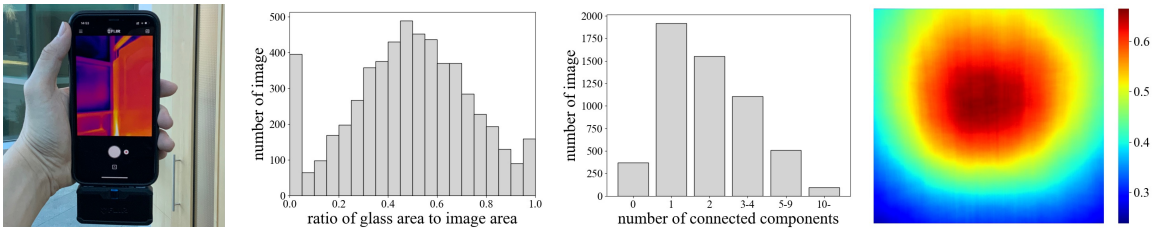
Figure 7.3: Examples of RGB-T image pairs with GT masks (bottom row) in our dataset. The last three columns show images with glass at all pixels or without glass. Please note that the image border of each mask is set to black for better visualization.

glass is visible to the RGB camera but not to the thermal camera, which is not the case when the glass is removed. As a result, images captured by thermal cameras have less arbitrary textures than RGB cameras on glass regions, while keeping similar textures on non-glass regions. Fig. 7.2(c) shows a real example.

RGB-T is better than the alternative methods. The problem of RGB-UV is that there is limited UV light indoor. The problem of RGB-MWIR (middle-wave IR) is that the MWIR camera is very costly ($\$10^{-1}$ per pixel [309]). RGB-Near IR (NIR) only works for some low-e glass. In contrast, RGB-T works for any silicate glass, works indoors and is low cost ($< \$500$), and it is extensible to traditional glass alternatives. We do not know whether some glass in our collected data is made of polymers like plastics and acrylic because we could not check, but acrylic and plastics of several millimeters thick are also opaque to thermal radiation. In the future, there could be new technologies to make glass windows and glass doors; RGB-T should still work because the new materials are expected to have high visibility and low heat transmission, *i.e.*opaque to heat to keep heat out in summer and to prevent heat loss in winter.

7.2 Proposed Dataset

Coupled RGB-T image pairs is a new idea for robust glass segmentation, for which this paper contributes a new RGB-T dataset with GT segmentation masks (see Fig.



(a) Image acquisition device (b) Glass area distribution (c) Connected components distribution (d) Glass location distribution

Figure 7.4: The RGB-T image acquisition device (a) and statistical analysis of our dataset (b,c,d). See text for details.

7.3).

Dataset construction

We capture RGB-T pairs with a FLIR ONE Pro camera [310], which consists of a nearly collocated RGB sensor and a thermal sensor. The thermal and RGB images are aligned with the FLIR Thermal Studio software [311] and both are saved at a resolution of 640×480[2]. During capturing, the camera is connected to an iPhoneXR for real-time image display (See Fig 7.4(a)). We use LabelMe [312] to manually annotate segmentation masks over RGB images. Pixels of raw thermal images (only one channel) represent the absolute temperature in the range $-20°C \sim 120°C$, we calculate the min and max values of each thermal image and normalize it to $[0, 1]$ using min-max normalization for obtaining the relative temperature before inputting to the algorithm, which is less sensitive to the variation of ambient temperature. Pseudocolor of thermal images shown in the paper is rendered from grayscale using Matplotlib [313].

Our new dataset covers a variety of scenes, such as libraries, shopping malls, galleries, train stations, museums, streets and houses, yielding 5551 RGB-T image pairs from 40 scenes. Among them, we capture 370 pairs without glass in 7 scenes. To generate the train and test split, we randomly select 23 scenes with glass and 5 scenes without glass for training and the others are used for testing (4427/1124 RGB-T

[2]The raw thermal image resolution is 160×120. The camera software performs super resolution for the thermal image and resizes it to 640×480.

pairs for train/test). Note that for the test split, we manually select some images with structures that visually look like glass, *e.g.*, door openings, holes in a wall. Such challenging examples increase the segmentation difficulty for RGB-only solutions and better reflect the merit of our RGB-T fusion idea (*e.g.*, door openings have very different appearances with glass in thermal images).

Dataset statistics

Following [56, 57], below we conduct some statistical analyses using GT segmentation masks from our dataset:

Glass area distribution. For each mask, we calculate the ratio of the glass area to the entire image area, where a ratio of 0 and 1 indicate, respectively, images without glass and images with glass at all pixels. As shown in Fig. 7.4(b), most of the captured images have ratios in the range of $[0.2, 0.8]$. There are also quite a few images with a ratio of 0 or 1, which are extreme cases captured on purpose.

Connected components distribution. An image may contain several glass regions. We compute the number of connected components in each binary mask and show the histogram in Fig. 7.4(c). The majority of images have 0$\sim$4 connected components and 52 at most. Images with more than 10 connected components are usually from complex scenes such as that in a shopping mall.

Glass location distribution. Fig. 7.4(d) shows the probability map of each pixel to be glass. The center region has the highest probability.

7.3 Proposed Method

7.3.1 Overview

Our architecture follows the standard encoder-decoder framework with skip-connections for dense segmentation [314], which consists of two encoding branches, one decoding branch and a multi-modal fusion module (MFM) as the bridge, as shown in Fig. 7.5(a). Specifically, we apply two ResNet-50 [2] encoders to convert the RGB and thermal

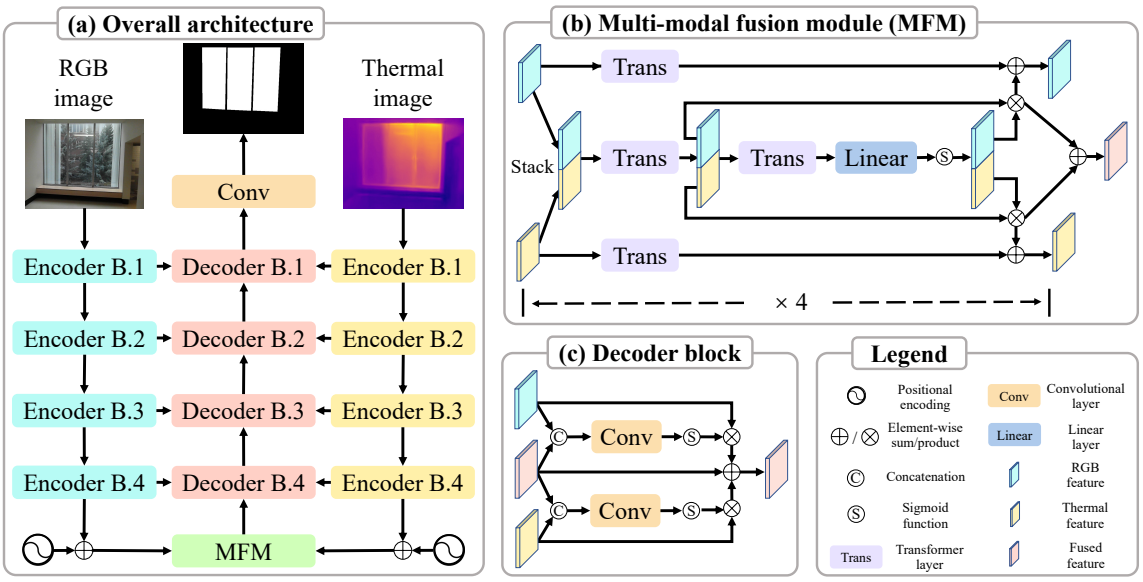


Figure 7.5: Neural network architecture for RGB-T glass segmentation. Our network consists of two separate ResNet-50 backbones as encoders for extracting high-level features from the RGB and thermal images, a transformer-based multi-modal fusion module for integrating the two modalities and a decoder for generating the segmentation result. Encoder/decoder B.i represents the ith encoder/decoder block.

input images into two $H \times W \times C$ feature volumes, where H, W and C, respectively, denote the height, the width and the channel size. In our implementation, H and W are varying and depend on the input image resolution, and $C = 256$. We supplement the two features with sinusoidal positional encodings [315] and produce a $H \times W \times C$ fused feature volume by the MFM. The fused feature further passes through a decoding branch with four decoder blocks and is progressively upsampled to the resolution of the input images. Finally, a convolution layer with a sigmoid function converts the results to the final segmentation. Note that all convolution layers other than those before the sigmoid functions are followed by a batch normalization layer and a ReLU function, which are omitted in Fig. 7.5. The Binary Cross-Entropy loss is used for training. Please refer to Appendix E for more architectural details.

7.3.2 Multi-modal Fusion Module (MFM)

The foundation of our RGB-T fusion is the attention mechanism in Transformer, which is known to be powerful in combining information from different modalities [190, 316]. As shown in Fig. 7.5(b), our MFM contains four iterative blocks and the ith ($i \in \{1, 2, 3, 4\}$) block takes in an RGB feature $\mathbf{f}_r^i$ and a thermal feature $\mathbf{f}_t^i$ both with the same resolution of $HW \times C$[3]. We stack $\mathbf{f}_r^i$ and $\mathbf{f}_t^i$ and use a transformer layer to obtain a $2HW \times C$ feature map $\mathbf{f}_{rt}^i$. We then generate a $2HW \times 1$ weight vector $\mathbf{w}^i$ by passing $\mathbf{f}_{rt}^i$ through a transformer layer followed by a linear layer and a sigmoid function. Such a weight vector is further used to update the RGB and thermal features. Formally, the above process can be summarized as:

$$\mathbf{f}_{rt}^i = \text{trans}(\text{stack}(\mathbf{f}_r^i, \mathbf{f}_t^i)), \tag{7.1}$$

$$\mathbf{w}^i = \text{sigmoid}(\text{linear}(\text{trans}(\mathbf{f}_{rt}^i))), \tag{7.2}$$

$$\mathbf{f}_r^{i+1} = \text{trans}(\mathbf{f}_r^i) + (\mathbf{w}^i \otimes \mathbf{f}_{rt}^i)[: HW, :], \tag{7.3}$$

$$\mathbf{f}_t^{i+1} = \text{trans}(\mathbf{f}_t^i) + (\mathbf{w}^i \otimes \mathbf{f}_{rt}^i)[HW :, :], \tag{7.4}$$

where trans(), linear(), sigmoid(), stack() denote the transformer layer, the linear layer, the sigmoid function and the stack operation, respectively. The symbol $\otimes$ denotes the element-wise multiplication. The transformer layer consists of a multi-head self-attention and a feed-forward network [317] (See Appendix E for details). Lastly, at the 4th iteration, the fused feature is computed as $(\mathbf{w}^4 \otimes \mathbf{f}_{rt}^4)[: HW, :] + (\mathbf{w}^4 \otimes \mathbf{f}_{rt}^4)[HW :, :]$ and reshaped back to $H \times W \times C$.

In our MFM, the transformer blocks from the top and the bottom branches extract non-local intra-modality dependencies from the RGB feature $\mathbf{f}_r^i$ and the thermal feature $\mathbf{f}_t^i$, respectively, and we utilize an additional transformer block in the middle branch to extract non-local inter-modality dependencies from the stack of $\mathbf{f}_r^i$ and $\mathbf{f}_t^i$. Considering the discrepancy of two modalities, deficiencies of one modality should be properly compensated by the other. Instead of directly unstacking the multi-modal

[3] $\mathbf{f}_r^1$ and $\mathbf{f}_t^1$ are reshaped from the $H \times W \times C$ feature volumes from the two encoders.

Method	#Params	Inference time	Model size	Images with glass				Images without glass			All images
				MAE ↓	IOU ↑	F_β ↑	BER ↓	MAE ↓	IOU* ↑	FPR ↓	MAE ↓
RTFNet [214]	185.23M	0.016s	708Mb	0.068	88.92	0.936	6.675	0.188	83.69	0.89	0.058
ShapeConv [319]	41.21M	0.534s	161Mb	0.059	87.65	0.930	6.940	0.019	98.24	0.41	0.054
ESANet [320]	46.88M	0.022s	359Mb	0.051	90.15	0.945	5.863	0.030	97.70	0.25	0.040
CMX [321]	66.56M	0.035s	255Mb	0.031	92.40	0.956	5.421	0.004	99.70	0.06	0.029
Segformer [322]	34.07M	0.017s	131Mb	0.052	89.22	0.934	7.208	0.063	93.72	0.34	0.053
Segmenter [323]	103.15M	1.067s	784Mb	0.072	85.83	0.912	8.451	0.303	92.06	0.31	0.072
MCNet [324]	54.64M	0.266s	210Mb	0.177	67.45	0.782	19.934	0.118	89.51	0.60	0.172
DPANet [325]	92.40M	0.019s	354Mb	0.241	71.51	0.838	15.051	0.291	83.56	0.90	0.154
HAINet [326]	59.82M	0.023s	229Mb	0.078	87.16	0.932	7.247	0.069	94.44	0.48	0.062
Zhang *et al.* [192]	38.87M	0.053s	445Mb	0.156	75.13	0.842	14.141	0.488	51.57	1.00	0.163
SSF [179]	32.93M	0.028s	126Mb	0.081	84.53	0.909	8.068	0.356	64.47	0.96	0.097
UCNet [174]	31.26M	0.026s	120Mb	0.079	84.70	0.913	8.324	0.028	97.24	0.14	0.071
CoNet [327]	43.66M	0.021s	168Mb	0.118	79.94	0.876	9.336	0.533	46.83	1.00	0.145
ATSA [189]	32.16M	0.021s	124Mb	0.088	83.87	0.903	8.185	0.276	73.57	1.00	0.098
DANet [175]	26.68M	0.007s	102Mb	0.082	85.63	0.915	7.982	0.045	96.36	0.36	0.069
HDFNet [177]	54.77M	0.019s	419Mb	0.055	89.56	0.941	5.673	0.019	98.49	0.59	0.048
FRDT [328]	72.81M	0.013s	279Mb	0.094	82.53	0.890	9.050	0.313	69.49	0.96	0.107
RD3D [176]	46.90M	0.013s	180Mb	0.064	88.94	0.938	6.610	0.037	97.13	0.26	0.045
DCFNet [182]	108.49M	0.064s	373Mb	0.059	88.13	0.930	6.757	0.036	96.45	0.21	0.056
UTA [329]	48.61M	0.023s	98Mb	0.051	89.59	0.933	6.060	0.069	93.00	0.23	0.052
EBS [330]	118.96M	0.036s	467Mb	0.041	91.24	0.946	5.528	0.031	96.97	0.21	0.040
VST [190]	83.83M	0.034s	321Mb	0.050	90.03	0.939	5.857	0.026	97.47	0.15	0.044
CLNet [187]	246.13M	0.142s	941Mb	0.045	91.01	0.945	4.983	0.021	98.03	0.14	0.041
SPNet [185]	175.29M	0.058s	671Mb	0.044	90.76	0.947	5.064	0.035	96.68	0.40	0.041
EBLNet [58]	48.36M	0.012s	185Mb	0.113	80.54	0.880	10.301	0.129	88.67	0.72	0.104
Ours (Thermal-only)	65.14M	0.039s	238Mb	0.189	68.63	0.781	19.315	0.136	86.68	0.55	0.177
Ours (RGB-only)	65.14M	0.039s	238Mb	0.056	88.94	0.929	6.618	0.016	98.42	0.11	0.052
Ours (RGB-T)	85.02M	0.048s	328Mb	0.027	93.80	0.965	4.078	0.003	99.73	0.08	0.024

Table 7.1: Quantitative evaluations. All compared methods (7 for semantic segmentation, 17 for salient object detection and 1 for glass segmentation) are retrained with our dataset. The performances on images with and without glass are separately evaluated. We also list the results of our thermal-only and RGB-only variants. The colors blue and cyan represent the best and the second best methods, respectively.

features $\mathbf{f}_{rt}^{i}$ along the first dimension and feed them to each modality, we generate a weight vector $\mathbf{w}^{i}$ as the spatial attention mask to remove the detrimental features and keep the beneficial ones. We apply the weight vector (spatial attention mask) $\mathbf{w}^{i}$ on multi-modal features using residual multiplication following Lee *et al.* [318].

7.3.3 Decoder

Our decoder consists of four blocks, where each block takes in (1) an RGB feature volume $\mathbf{e}_r$ from the RGB encoder, (2) a thermal feature $\mathbf{e}_t$ from the thermal encoder, and (3) a fused feature $\mathbf{d}$ from the previous decoder block. We concatenate $\mathbf{e}_r$ and $\mathbf{d}$

and use three convolution layers and a sigmoid function to compute a single-channel weight volume $\mathbf{w}_r$, which has the same height and width as that of $\mathbf{d}$. We apply the same operation to $\mathbf{e}_t$ and $\mathbf{d}$ and obtain another weight $\mathbf{w}_t$. Finally, the output of the block is calculated as $\mathbf{w}_r \otimes \mathbf{e}_r + \mathbf{w}_t \otimes \mathbf{e}_t + \mathbf{d}$. Similar to the weight vector $\mathbf{w}^i$ in MFM, $\mathbf{w}_r$ and $\mathbf{w}_t$ are used as the spatial attention mask for feature fusion.

Note that we could have chosen transformer layers for feature fusion in the decoder as that in MFM, but we opt for a convolution-based spatial attention scheme due to the intensive memory constraints of self-attention in high resolution [331]. In addition, each decoder block is further followed by a convolution layer and a bilinear upsampling to gradually recover the spatial dimension, the detail of which is omitted in Fig. 7.5(c).

7.4 Experiments

7.4.1 Implementation Details

We have implemented our method in PyTorch [248]. The ImageNet-pretrained ResNet-50 backbone (encoder block 1∼4) is loaded from torchvision. The batch size is 16. The learning rate is initialized as 10^{-4} and changed to 10^{-5} after 200 epochs. We use random flipping, resizing and cropping for data augmentation. Our model is trained with the AdamW [332] optimizer for 300 epochs, which takes around 35 hours on an NVIDIA RTX A6000 GPU with 48GB of RAM.

We adopt standard segmentation metrics: mean absolute error (MAE), intersection over union (IOU), maximum F-measure (F_β) and balanced error rate (BER). These four metrics are commonly used in previous glass segmentation papers [56–58], whereas only MAE is valid for evaluating images without glass (*i.e.*, GT mask is black). To assess our performance on those images without glass in our new dataset, together with MAE, we use inverse intersection over union (IOU*) and false positive rate (FPR). IOU* takes inverse masks and is defined as IOU$(1 - \text{result}, 1 - \text{GT})$.

Method	MAE↓	IOU↑	BER↓
MirrorNet [333]	0.094	81.3	8.98
GDNet [56]	0.088	82.6	8.42
Translab [159]	0.081	85.1	7.43
EBLNet [58]	0.074	86.0	6.90
Ours (RGB-only)	0.072	86.6	7.35

Table 7.2: Quantitative evaluations on RGB-only glass segmentation dataset GDD [56]. Evaluation results except ours are directly copied from [58]. The colors blue and cyan represent the best and the second best methods, respectively.

FPR is calculated as the ratio of the number of false positives (*i.e.*, glass are wrongly detected) to the total number of images without glass. Because MAE is commonly used, we use it to evaluate on all images.

7.4.2 Quantitative Comparison

RGB-T image pairs are new cues for glass segmentation. We compare three variants of our method with 24 state-of-the-art methods from other related areas, which include ShapeConv [319], ESANet [320] and CMX [321] for RGB-D semantic segmentation, RTFNet [214] for RGB-T semantic segmentation, Segformer [322] and Segmenter [323] for RGB-only semantic segmentation, MCNet [324] for thermal-only semantic segmentation, DPANet [325], HAINet [326], SSF [179], UCNet [174], CoNet [327], ATSA [189], DANet [175], HDFNet [177], FRDT [328], RD3D [176], DCFNet [182], UTA [329], EBS [330], VST [190], CLNet [187] and SPNet [185] for RGB-D salient object detection, and Zhang *et al.* [192] for RGB-T saliency detection. All of these competing methods are retrained with our dataset using RGB-T pairs as input. We also compare with a state-of-the-art RGB-only glass segmentation method EBLNet[4] [58], which is retrained with the RGB images in our dataset.

[4]Other recent glass segmentation methods [56, 57, 333] did not provide training codes for their models.

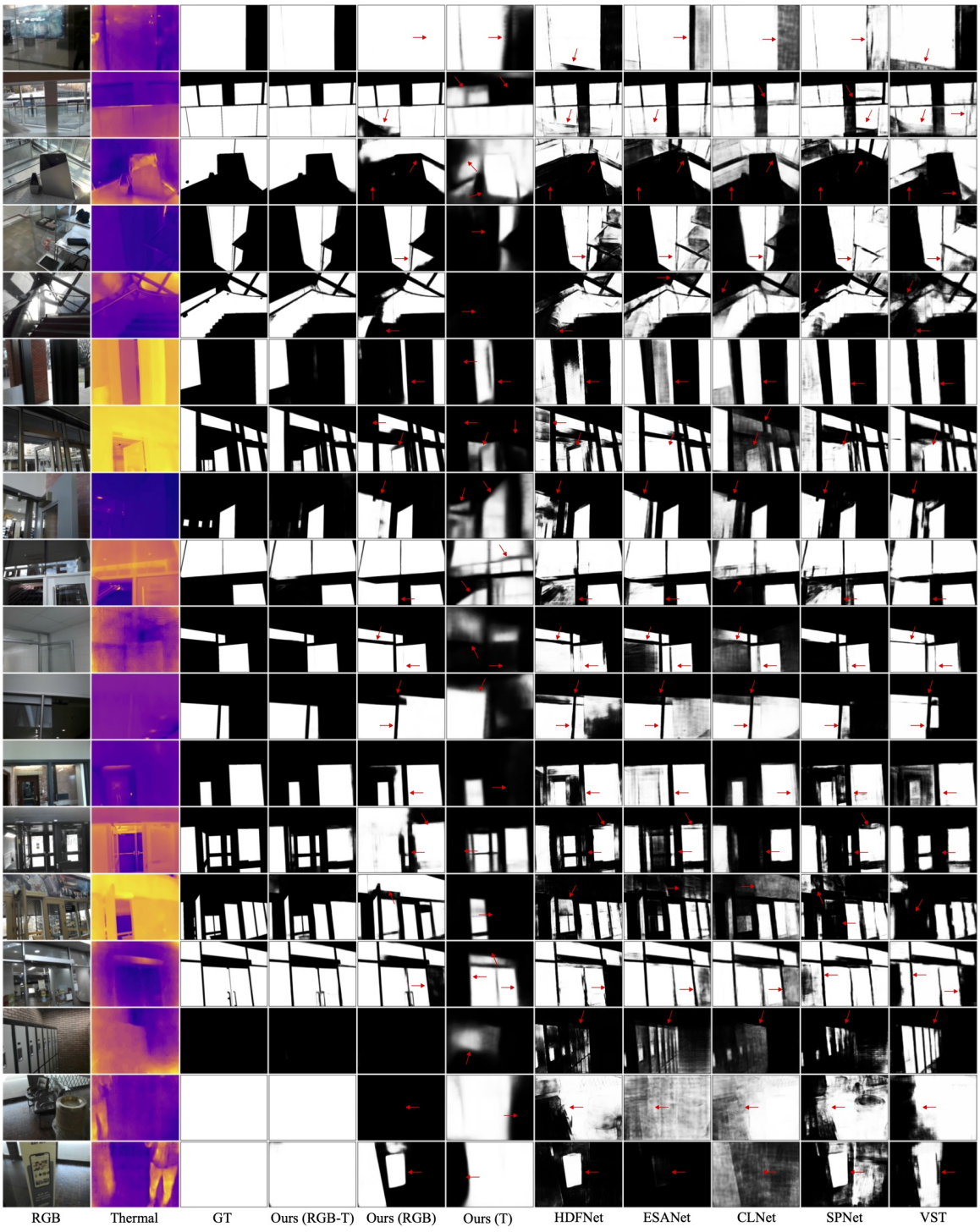


Figure 7.6: Qualitative comparison of our method and 5 state-of-the-art methods (HDFNet [177], ESANet [320], CLNet [187], SPNet [185], and VST [190]). Results of our RGB-only and thermal-only variants are also displayed. For better visualization, we set the image border of each mask to black. The superiority of our method can be clearly validated at various places, as highlighted by the red arrows.

As shown in Table 7.1, the last three rows give the ablation results of our full RGB-T method and its two variants that use either thermal or RGB data only. Albeit the thermal-only variant is the worst, it can significantly boost the performance when it is combined with RGB images. For example, the MAE of our RGB-T method on glass images is 86% and 52% better than the two variants, which demonstrates the effectiveness of our RGB-T fusion idea for glass segmentation. Please refer to Section 7.4.6 for more ablation studies.

Our approach consistently outperforms previous methods on most metrics by a large margin. Similar to our method, CMX, EBS and Segformer utilize a combination of convolution and transformer (hybrid) in their architectures, which achieve superior performance in the evaluations of RGB-T and RGB-only, respectively. We attribute it to the hybrid architecture [7, 334], while pure convolution (*e.g.*, ESANet, CLNet, SPNet, EBLNet) or transformer (*e.g.*, VST, Segmenter) architectures obtain inferior results.

The performance of our thermal-only variant and MCNet is much worse than that of other RGB-only and RGB-T methods. We believe that there are two reasons. First, the resolution of thermal images is lower than that of RGB images. Hence, the segmentation results of thermal images are poor. Second, even though the glass is opaque to the thermal light, we need to compare it with the RGB images to see the contrast. Hence, it is difficult, if not impossible, to distinguish glass and other opaque objects using a single thermal image.

7.4.3 Qualitative Comparison

Figure 7.6 shows the qualitative comparison results. Our method is able to accurately segment the glass regions in various challenging scenes, while previous methods and the RGB-only variant produce a plethora of noticeable errors: (1) blurry segmentation results and fuzzy boundaries, (2) under-segmentation masks due to the influence by the background behind glass, and (3) over-segmentation results where cabinet or

Method	VT5000 [335]			VT1000 [191]		
	MAE ↓	S_m ↑	F_β ↑	MAE ↓	S_m ↑	F_β ↑
MTMR [227]	0.114	0.680	0.662	0.119	0.706	0.755
M3S-NIR [228]	0.188	0.631	0.607	0.151	0.717	0.734
SGDL [191]	0.089	0.750	0.738	0.090	0.787	0.807
ADF [335]	0.048	0.864	0.864	0.034	0.910	0.923
MIDD [233]	0.043	0.868	0.872	0.027	0.915	0.926
APNet [194]	0.035	0.876	0.875	0.021	0.921	0.930
ECFFNet [193]	0.038	0.874	0.872	0.021	0.923	0.930
MIA-DPD [336]	0.040	0.879	0.880	0.025	0.924	0.935
DCNet [197]	0.035	0.872	0.870	0.021	0.922	0.928
Ours	0.036	0.881	0.881	0.020	0.929	0.941

Table 7.3: Quantitative evaluations on RGB-T SOD dataset VT5000 [335] and VT1000 [191]. The colors blue and cyan represent the best and the second best methods, respectively.

door openings are wrongly identified as glass. The last two rows show two extreme cases where the scenes are completely covered by glass. The human bodies in the thermal images are reflections of the photographer, which are invisible in the RGB images, further validating the different transmission models of visual and thermal light through glass. Our thermal-only variant tends to recognize the distinct boundaries which may mislead the segmentation process. We provide more visual results in Appendix E.

7.4.4 Evaluations on GDD dataset [56]

Table 7.2 compares our RGB-only variant with four state-of-the-art glass segmentation methods: MirrorNet [333], GDNet [56], Translab [159] and EBLNet [58]. Both our variant and compared methods are trained and tested on the RGB-only glass segmentation dataset GDD [56]. Our RGB-only variant removes the thermal branch from

Method	Images with glass				Images without glass			All images
	MAE ↓	IOU ↑	F_β ↑	BER ↓	MAE ↓	IOU* ↑	FPR ↓	MAE ↓
Thermal-only	0.189	68.63	0.781	19.315	0.136	86.68	0.55	0.018
Dual-Thermal	0.189	68.47	0.783	19.120	0.120	88.34	0.50	0.018
RGB-only	0.056	88.94	0.929	6.618	0.016	98.42	0.11	0.052
Dual-RGB	0.055	89.21	0.932	6.250	0.018	98.25	0.10	0.050
RGB-T (Ours)	0.027	93.80	0.965	4.078	0.003	99.73	0.07	0.024

Table 7.4: Ablation studies on input. The colors blue and cyan represent the best and the second best methods, respectively.

our RGB-T architecture and the resulting RGB-only architecture is a combination of convolution and transformer, while the four existing methods adopt convolution only. Our RGB-only variant achieves the best in IOU and MAE and the second best in BER, echoing recent findings of other recognition methods (*e.g.*, object detection [7], wireframe parsing [334]), which also demonstrate the effectiveness of the hybrid structure of convolution and transformer.

7.4.5 Evaluations on RGB-T SOD datasets

To show the versatility of our method on other RGB-T tasks, we retrain our model on the 2500 training images from VT5000 dataset [335] for RGB-T salient object detection (SOD), then evaluate on the 2500 testing images from VT5000 dataset and 1000 images from VT1000 dataset [191]. Nine state-of-the-art RGB-T SOD methods are compared including three conventional graph-based methods (MTMR [227], M3S-NIR [228], SGDL [191]) and six deep learning based methods (ADF [335], MIDD [233], APNet [194], ECFFNet [193], MIA-DPD [336], DCNet [197]). We take the mean absolute error (MAE), S-measure (S_m) [337], and maximum F-measure (F_β) to evaluate the SOD results. As shown in Table 7.3, our method is comparable to methods specifically designed for RGB-T SOD.

Method	Images with glass				Images without glass			All images
	MAE ↓	IOU ↑	F_β ↑	BER ↓	MAE ↓	IOU* ↑	FPR ↓	MAE ↓
SFS	0.058	89.88	0.946	5.865	0.041	96.08	0.29	0.045
SFC	0.052	90.47	0.945	5.588	0.025	98.15	0.36	0.043
PAF [61]	0.038	91.73	0.954	5.188	0.010	99.04	0.05	0.035
AT [184]	0.038	91.96	0.953	4.885	0.016	98.48	0.20	0.033
MFM-EGFNet [218]	0.030	92.82	0.960	5.328	0.006	99.42	0.10	0.028
MFM-DS	0.031	93.06	0.959	4.392	0.004	99.59	0.05	0.027
MFM-DC	0.033	92.75	0.958	4.565	0.003	99.69	0.06	0.029
MFM (Ours)	0.027	93.80	0.965	4.078	0.003	99.73	0.07	0.024

Table 7.5: Ablation studies on MFM. The colors blue and cyan represent the best and the second best methods, respectively.

Method	Images with glass				Images without glass			All images
	MAE ↓	IOU ↑	F_β ↑	BER ↓	MAE ↓	IOU* ↑	FPR ↓	MAE ↓
Decoder-DS	0.033	92.87	0.960	4.460	0.007	99.30	0.11	0.030
Decoder-DC	0.034	92.42	0.956	4.907	0.008	99.24	0.10	0.031
Decoder (Ours)	0.027	93.80	0.965	4.078	0.003	99.73	0.07	0.024

Table 7.6: Ablation studies on decoder. The colors blue and cyan represent the best and the second best methods, respectively.

7.4.6 Ablation Studies

Below we evaluate the contributions of different components of our architecture, followed by limitation analysis.

Impact of inputs

In Table 7.1, we obtain our RGB-only or thermal-only variant by removing the corresponding encoders from our architecture. To verify that the performance decrease is not due to the architectural change, we additionally train another two models by inputting the same RGB/thermal images to both encoders (dual-RGB/thermal). Table 7.4 shows that the differences between RGB/thermal-only and dual-RGB/thermal

Method	Images with glass				Images without glass			All images
	MAE ↓	IOU ↑	F_β ↑	BER ↓	MAE ↓	IOU* ↑	FPR ↓	MAE ↓
ResNet-18	0.033	92.34	0.956	5.388	0.006	99.48	0.09	0.031
ResNet-34	0.031	92.96	0.959	5.049	0.011	98.90	0.12	0.030
ResNet-101	0.030	93.14	0.961	4.982	0.010	99.04	0.07	0.029
ResNet-50 w/o pretraining	0.057	87.29	0.924	8.063	0.064	94.02	0.50	0.057
ResNet-50 (Ours)	0.027	93.80	0.965	4.078	0.003	99.73	0.07	0.024

Table 7.7: Ablation studies on backbones. The colors blue and cyan represent the best and the second best methods, respectively.

Method	Images with glass				Images without glass			All images
	MAE ↓	IOU ↑	F_β ↑	BER ↓	MAE ↓	IOU* ↑	FPR ↓	MAE ↓
Training w/ glass images only	0.029	93.03	0.960	5.116	0.006	99.36	0.09	0.028
Training w/ all images (Ours)	0.027	93.80	0.965	4.078	0.003	99.73	0.07	0.024

Table 7.8: Ablation studies on images without glass. The colors blue and cyan represent the best and the second best methods, respectively.

are negligible, validating that the accuracy gap is because of using single modality rather than architectural differences.

Effectiveness of MFM

We compare our method with four simple baselines by replacing the MFM with the simple feature summation (SFS), simple feature concatenation (SFC), the pixel-wise attention fusion (PAF) [61] and affine transform (AT) [184]. The multi-modal fusion module from EGFNet [218] for RGB-T fusion are also utilized to replace our MFM. We also modify our MFM by replacing the weighted summation with direct summation (DS) or direct concatenation (DC), where the attention weights are discarded in the two variants. As shown in Table 7.5, our variants using the MFM (the last row) achieves the best performance.

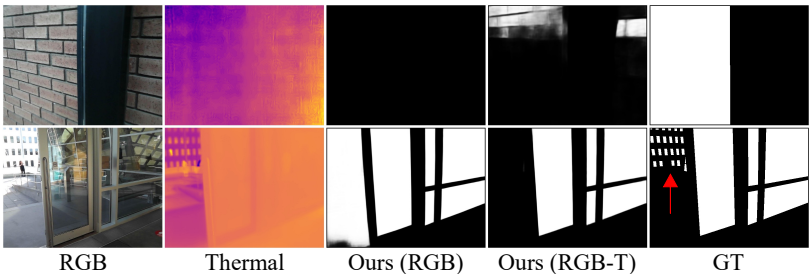


Figure 7.7: Two typical failure examples. The red arrow highlights small glass regions.

Selection of backbones

Following other related methods [174, 179, 325–327], we exploit a pretrained ResNet backbone. In Table 7.7, we evaluate different pretrained backbones and the ResNet-50 backbone without pretraining. We can see that pretraining is critical for the performance and the ResNet-50 yields the best results. We believe that the performance decrease of ResNet-101 is due to over-parameterization which makes it hard to train.

Variants of our decoder

Similar to the evaluation of the weighted summation in MFM, we also replace the weighted summation in decoder blocks with direct summation (DS) or direct concatenation (DC). As shown in Table 7.6, our decoder that uses weighted summation outperforms the other two variants.

Images without glass

To reduce the false positive segmentation, our training data includes 370 RGB-thermal images pairs without any glass. As shown in Table 7.8, the model trained on only glass images has an obvious performance decrease on both images with and without glass, which demonstrates the necessity of such samples.

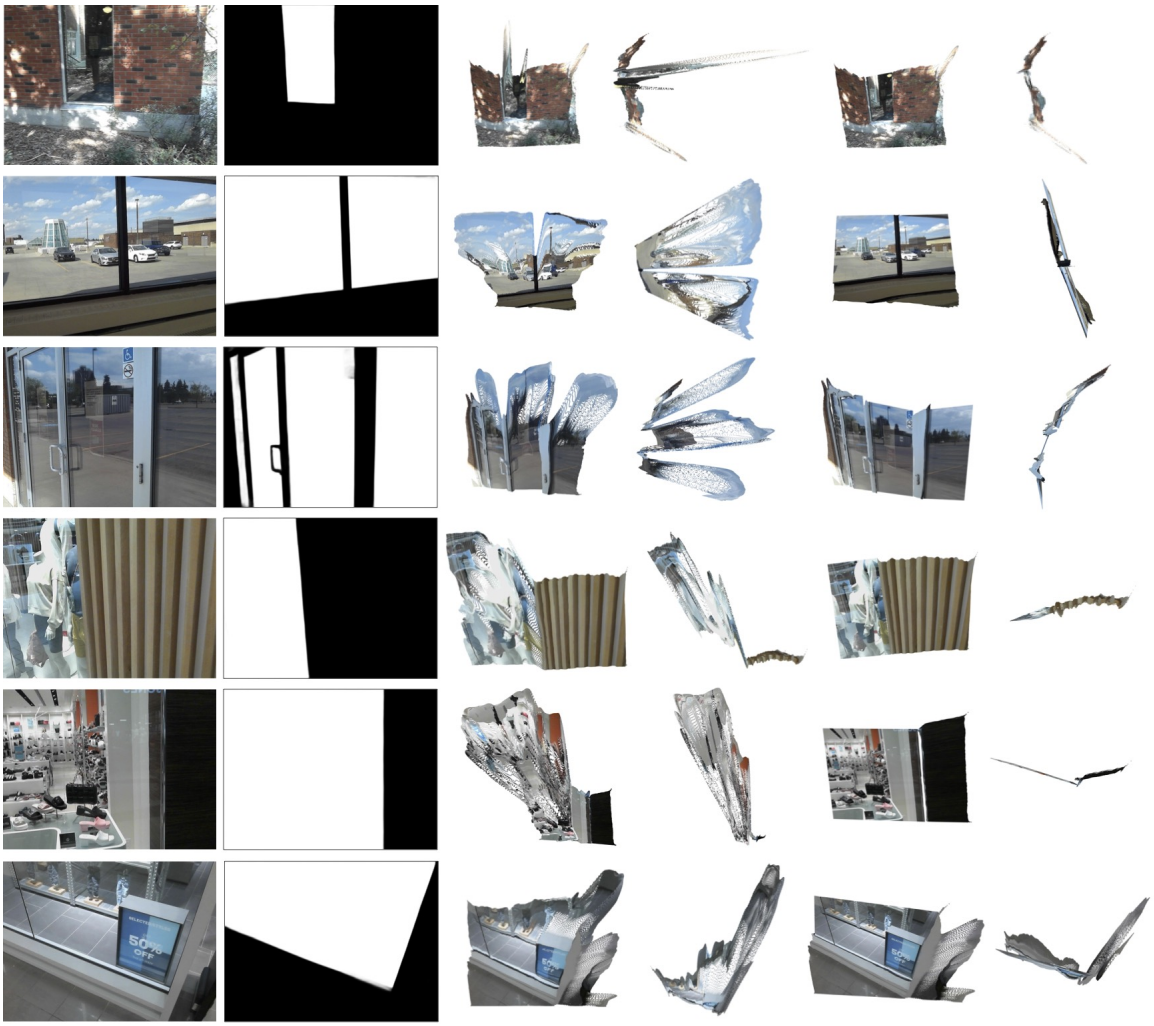


Figure 7.8: Application of monocular 3D reconstruction. From left to right, it shows the RGB images, the glass segmentation masks by our method, the raw reconstructed point clouds by Adabins [338] and our corrected point clouds, respectively.

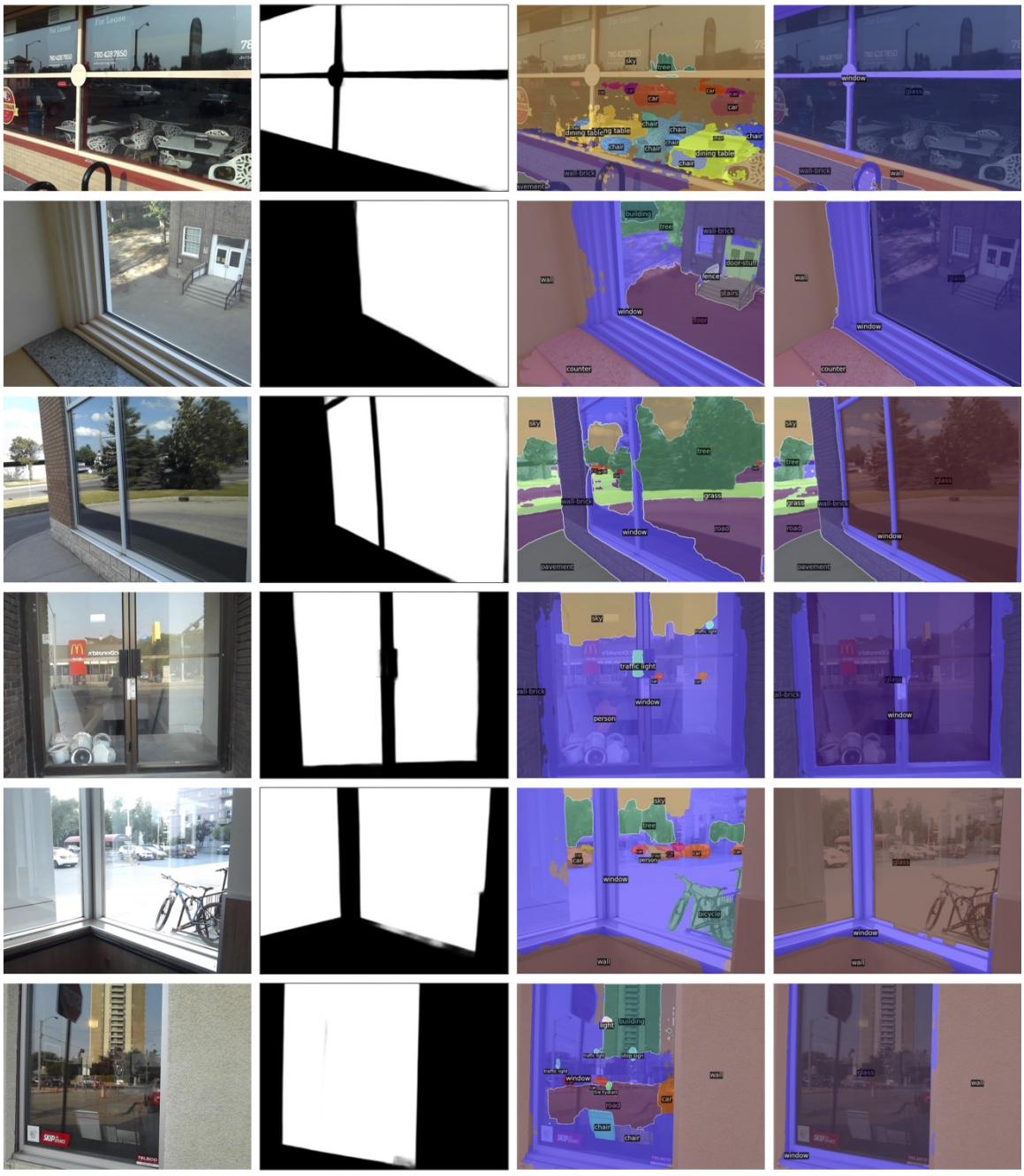


Figure 7.9: Application of semantic image segmentation. From left to right, it shows the RGB images, the glass segmentation masks by our method, the raw semantic segmenation results of DETR [7] and our refined results, respectively.

7.4.7 Applications

Monocular 3D Reconstruction

State-of-the-art 3D reconstruction approaches including recent deep-learning-based ones [338–340] exhibit challenges when handling scenes with glass (*e.g.*, urban buildings, indoor scenes). While it seems that glass elements occupy a relatively small region in an entire scene, inaccurate glass geometry is catastrophic to the overall 3D reconstruction performance (see Fig. 7.8), leading to undesirable geometry artifacts and subsequent unpleasant user experiences in downstream applications including augmented reality, gaming, navigation, rendering.

To correct such reconstruction errors in glass regions, we apply our RGB-T glass segmentation method and recover each glass region as a 3D plane. Specifically, we first adopt a recent monocular reconstruction method [338] for depth estimation. We then follow Mirror3D [341] and estimate the 3D glass plane parameters based on the depths of the boundary pixels of glass regions. The final 3D point clouds are generated with pre-defined camera intrinsics in Open3D [342]. As shown in Fig. 7.8, by accurately segmenting glass surfaces, our plane-based depth correction strategy significantly improves 3D reconstruction results compared to the raw point clouds from [338].

Semantic Segmentation

Similar problems also arise in semantic image segmentation [7, 210, 343], as shown in Fig. 7.9 where the reflections on glass surfaces are mis-recognized. To correct those errors, we first utilize our RGB-T segmentation method to obtain glass masks and set the glass regions of RGB images to zero, which serve as inputs to a transformer architecture DETR [7] (designed for both object detection and semantic segmentation) to get the refined semantic segmentation results. As shown in Fig. 7.9, such a simple strategy effectively eliminate inaccurate semantic segmentation.

7.4.8 Failure Cases

The first row in Fig. 7.7 shows that our method fails when the visual appearances of glass and non-glass regions are highly similar in both RGB and thermal images, which is also a difficult, if not impossible, task for the human eye to differentiate without looking at the GT mask beforehand. Our method also fails to detect small glass regions, as highlighted by the red arrow in the second row.

7.5 Summary

Since existing semantic segmentation methods are mainly trained on images with only opaque objects, they usually fail to detect glass surfaces. In this chapter, I solve the problem of existing semantic segmentation methods on glass scenes by proposing a new glass segmentation method. An extra thermal image is combined with the RGB image for better segmentation. Our proposed glass segmentation method is further aggregated into a pretrained semantic segmentation methods (DETR) generalize it on glass scenes without retraining.

Chapter 8

Conclusion and Future Work

In Chapter 3, I propose a novel end-to-end blind non-uniform motion deblurring network with new ASPDC modules, which are able to apply region-specific convolution to each pixel and integrate features from different receptive fields. Compared to SOTA methods, my method achieves better performance with high efficiency. In addition, the performance can be further improved by fine-tuning on the proposed reblurring network. In the future, I plan to address the fact that none of the existing methods perform well when the magnitude of motion is large, resulting in issues such as color degradation or even failure in deblurring. I further plan to study deblurring-reblurring consistency of non-uniform deblurring in an unsupervised setting without access to blurred-sharp pairs for the reblurring network.

In Chapter 4, I propose a new variational deep image prior (VDIP) for blind image deconvolution, which achieves a better performance than that of the DIP. One common issue of optimizing a model using a single image is high inference time compared with methods trained on external datasets, which makes it hard to adopt the method to large testing datasets. My method is also limited when the single degraded image cannot provide enough information. In my future work, I plan to adopt meta-learning [149] to train the networks on external datasets and fine-tune on each test image, which can take advantage of the information from other images and obtain an image-specific model with only several iterations.

In Chapter 5, I present a novel texture sampling scheme for text-driven texture generation on 3D meshes, leveraging depth-aware diffusion models. To address the significant challenges in 3D content generation, particularly in producing textures that are consistent across views and rich in detail, I first propose to maintain a time-dependent texture map that evolves with each denoising step to progressively reduce the view discrepancy. Specifically, at each denoising step, the texture is aggregated from the *denoised observations* of sampled views under my attention-guided inpainting process. It is then utilized in my Text&Texture Guided noise resampling procedure to further guide the estimated noise fed into the next denoising step. The effectiveness of my method is evident in its ability to generate superior-quality textures for diverse 3D objects as well as in its adaptability for texture editing purposes.

Chapter 6 presents a novel architecture motivated by the physical relationship between RGB images and the corresponding spectral reflectances, by which I estimate the components within the sub-space of the degradation matrix $\hat{\boldsymbol{\mathcal{H}}}$ to compensate for the final output. My proposed architecture can be easily adapted to RGB images illuminated by more than one light source with only the output size of the auxiliary task needs to be changed. I also adopt meta-auxiliary learning to make use of the internal information of the input RGB images at test-time. Qualitative and quantitative evaluations demonstrate that my method surpasses state-of-the-art approaches by a large margin. Extensive ablation studies further justify the significant contribution of each component in my proposed method.

Chapter 7 presents the idea of leveraging RGB-T image pairs for glass segmentation. I propose a novel neural architecture for fusing features of the RGB and thermal modalities. I also contribute the first RGB-T glass scene dataset with GT masks. My extensive evaluations reveal the significantly better performance of using an RGB-T pair over using a single RGB image, and demonstrate the superiority of my cross-modality fusion method against existing fusion methods using the same RGB-T input. Polarization methods have drawbacks in that detection is sparse for a

plate glass, which can be confused by glass and other dielectric surfaces, and might not work when transmission $\gg$ reflection. Since polarization and my method use completely different cues, they can be combined as "RGB + polarization + thermal" (polarized RGB like [61] or polarized thermal [344]) to further improve the result, which is an interesting direction for future work. I could also add an invisible thermal light to address limitations of my method, which makes it an active method. As shown in the first row of Fig. 7.7, glass and non-glass regions are very similar in both RGB and thermal images. Considering the different smoothness of glass surface and other materials like brick wall, the thermal camera should obtain clear reflection of the thermal light source on glass while a blurry light cone on others.

Bibliography

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[3] A. Dosovitskiy *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 1, pp. 142–158, 2015.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[6] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[7] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision*, Springer, 2020, pp. 213–229.

[8] A. Kirillov *et al.*, "Segment anything," *arXiv preprint arXiv:2304.02643*, 2023.

[9] X. Wang, L. Xie, C. Dong, and Y. Shan, "Real-esrgan: Training real-world blind super-resolution with pure synthetic data," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 1905–1914.

[10] O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas, "Deblurgan: Blind motion deblurring using conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8183–8192.

[11] S. W. Zamir *et al.*, "Multi-stage progressive image restoration," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 14 821–14 831.

[12] S. W. Zamir, A. Arora, S. Khan, M. Hayat, F. S. Khan, and M.-H. Yang, "Restormer: Efficient transformer for high-resolution image restoration," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5728–5739.

[13] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4401–4410.

[14] A. Van Den Oord, O. Vinyals, *et al.*, "Neural discrete representation learning," *Advances in neural information processing systems*, vol. 30, 2017.

[15] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695.

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, Ieee, 2009, pp. 248–255.

[17] C. Schuhmann *et al.*, "Laion-5b: An open large-scale dataset for training next generation image-text models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 25 278–25 294, 2022.

[18] X. Qin, Z. Zhang, C. Huang, C. Gao, M. Dehghan, and M. Jagersand, "Basnet: Boundary-aware salient object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7479–7489.

[19] M. Jia *et al.*, "Visual prompt tuning," in *European Conference on Computer Vision*, Springer, 2022, pp. 709–727.

[20] O. Kupyn, T. Martyniuk, J. Wu, and Z. Wang, "Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 8878–8887.

[21] X. Tao, H. Gao, X. Shen, J. Wang, and J. Jia, "Scale-recurrent network for deep image deblurring," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8174–8182.

[22] H. Zhang, Y. Dai, H. Li, and P. Koniusz, "Deep stacked hierarchical multi-patch network for image deblurring," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5978–5986.

[23] Z. Zhong, Y. Gao, Y. Zheng, and B. Zheng, "Efficient spatio-temporal recurrent neural network for video deblurring," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, Springer, 2020, pp. 191–207.

[24] S. Nah, T. Hyun Kim, and K. Mu Lee, "Deep multi-scale convolutional neural network for dynamic scene deblurring," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3883–3891.

[25] S. Nah *et al.*, "Ntire 2019 challenge on video deblurring and super-resolution: Dataset and study," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[26] R. Liu, R. Wu, B. Van Hoorick, P. Tokmakov, S. Zakharov, and C. Vondrick, "Zero-1-to-3: Zero-shot one image to 3d object," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 9298–9309.

[27] E. R. Chan *et al.*, "Generative novel view synthesis with 3d-aware diffusion models," *arXiv preprint arXiv:2304.02602*, 2023.

[28] Y.-T. Lin and G. D. Finlayson, "Physically plausible spectral reconstruction from rgb images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 532–533.

[29] B. Sun, J. Yan, X. Zhou, and Y. Zheng, "Tuning ir-cut filter for illumination-aware spectral reconstruction from rgb," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 84–93.

[30] J. Li, S. Du, C. Wu, Y. Leng, R. Song, and Y. Li, "Drcr net: Dense residual channel re-calibration network with non-local purification for spectral super resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1259–1268.

[31] T.-Y. Lin *et al.*, "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, Springer, 2014, pp. 740–755.

[32] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*, IEEE, 2012, pp. 3354–3361.

[33] W. Yang, X. Zhang, Y. Tian, W. Wang, J.-H. Xue, and Q. Liao, "Deep learning for single image super-resolution: A brief review," *IEEE Transactions on Multimedia*, vol. 21, no. 12, pp. 3106–3121, 2019.

[34] J. Koh, J. Lee, and S. Yoon, "Single-image deblurring with neural networks: A comparative survey," *Computer Vision and Image Understanding*, vol. 203, p. 103 134, 2021.

[35] Y. Mo, Y. Wu, X. Yang, F. Liu, and Y. Liao, "Review the state-of-the-art technologies of semantic segmentation based on deep learning," *Neurocomputing*, vol. 493, pp. 626–646, 2022.

[36] Y. Xu, X. Zhou, S. Chen, and F. Li, "Deep learning for multiple object tracking: A survey," *IET Computer Vision*, vol. 13, no. 4, pp. 355–368, 2019.

[37] T. Michaeli and M. Irani, "Nonparametric blind super-resolution," in *ICCV*, 2013, pp. 945–952.

[38] L. Yu, S. Cao, J. He, B. Sun, and F. Dai, "Single-image super-resolution based on regularization with stationary gradient fidelity," in *CISP-BMEI*, IEEE, 2017, pp. 1–5.

[39] J. Pan, D. Sun, H. Pfister, and M.-H. Yang, "Blind image deblurring using dark channel prior," in *CVPR*, 2016.

[40] Y. Yan, W. Ren, Y. Guo, R. Wang, and X. Cao, "Image deblurring via extreme channels prior," in *CVPR*, 2017.

[41] S. Avidan and A. Shashua, "Novel view synthesis in tensor space," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE, 1997, pp. 1034–1040.

[42] S. Avidan and A. Shashua, "Novel view synthesis by cascading trilinear tensors," *IEEE transactions on visualization and computer graphics*, vol. 4, no. 4, pp. 293–306, 1998.

[43] B. Arad and O. Ben-Shahar, "Sparse recovery of hyperspectral signal from natural rgb images," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VII 14*, Springer, 2016, pp. 19–34.

[44] Y. Fu, Y. Zheng, L. Zhang, and H. Huang, "Spectral reflectance recovery from a single rgb image," *IEEE Transactions on Computational Imaging*, vol. 4, no. 3, pp. 382–394, 2018.

[45] M. Suin, K. Purohit, and A. Rajagopalan, "Spatially-attentive patch-hierarchical network for adaptive motion deblurring," in *CVPR*, 2020, pp. 3606–3615.

[46] E. Park, J. Yang, E. Yumer, D. Ceylan, and A. C. Berg, "Transformation-grounded image generation network for novel 3d view synthesis," in *Proceedings of the ieee conference on computer vision and pattern recognition*, 2017, pp. 3500–3509.

[47] T. Zhou, S. Tulsiani, W. Sun, J. Malik, and A. A. Efros, "View synthesis by appearance flow," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, Springer, 2016, pp. 286–301.

[48] A. Shocher, N. Cohen, and M. Irani, ""zero-shot" super-resolution using deep internal learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3118–3126.

[49] Z. Chi, Y. Wang, Y. Yu, and J. Tang, "Test-time fast adaptation for dynamic scene deblurring via meta-auxiliary learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9137–9146.

[50] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *CVPR*, 2018, pp. 9446–9454.

[51] D. Ren, K. Zhang, Q. Wang, Q. Hu, and W. Zuo, "Neural blind deconvolution using deep priors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3341–3350.

[52] L. Zhang, A. Rao, and M. Agrawala, "Adding conditional control to text-to-image diffusion models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 3836–3847.

[53] S. Sajjan *et al.*, "Clear grasp: 3d shape estimation of transparent objects for manipulation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 3634–3642.

[54] S. S. Martínez, J. G. Ortega, J. G. García, A. S. García, and E. E. Estévez, "An industrial vision system for surface quality inspection of transparent parts," *The International Journal of Advanced Manufacturing Technology*, vol. 68, no. 5-8, pp. 1123–1136, 2013.

[55] J. Zhang, K. Yang, A. Constantinescu, K. Peng, K. Muller, and R. Stiefelhagen, "Trans4trans: Efficient transformer for transparent object segmentation to help visually impaired people navigate in the real world," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 1760–1770.

[56] H. Mei *et al.*, "Don't hit me! glass detection in real-world scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3687–3696.

[57] J. Lin, Z. He, and R. W. Lau, "Rich context aggregation with reflection prior for glass surface detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 13 415–13 424.

[58] H. He *et al.*, "Enhanced boundary learning for glass-like object segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[59] T. Wang, X. He, and N. Barnes, "Glass object segmentation by label transfer on joint depth and appearance manifolds," in *2013 IEEE International Conference on Image Processing*, IEEE, 2013, pp. 2944–2948.

[60] Y. Xu, H. Nagahara, A. Shimada, and R.-i. Taniguchi, "Transcut: Transparent object segmentation from a light-field image," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3442–3450.

[61] A. Kalra, V. Taamazyan, S. K. Rao, K. Venkataraman, R. Raskar, and A. Kadambi, "Deep polarization cues for transparent object segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8602–8611.

[62] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman, "Removing camera shake from a single photograph," in *SIGGRAPH*, ACM, 2006.

[63] Q. Shan, J. Jia, and A. Agarwala, "High-quality motion deblurring from a single image," *TOG*, vol. 27, no. 3, pp. 1–10, 2008.

[64] S. Cho and S. Lee, "Fast motion deblurring," in *SIGGRAPH Asia*, 2009.

[65] L. Xu and J. Jia, "Two-phase kernel estimation for robust motion deblurring," in *ECCV*, 2010.

[66] D. Krishnan, T. Tay, and R. Fergus, "Blind deconvolution using a normalized sparsity measure," in *CVPR*, 2011.

[67] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, "Understanding and evaluating blind deconvolution algorithms," in *CVPR*, 2009.

[68] S. D. Babacan, R. Molina, M. N. Do, and A. K. Katsaggelos, "Bayesian blind deconvolution with general sparse image priors," in *ECCV*, 2012, pp. 341–355.

[69] J. Dong, J. Pan, Z. Su, and M.-H. Yang, "Blind image deblurring with outlier handling," in *ICCV*, 2017, pp. 2478–2486.

[70] L. Xu, S. Zheng, and J. Jia, "Unnatural l0 sparse representation for natural image deblurring," in *CVPR*, 2013.

[71] L. Chen, F. Fang, S. Lei, F. Li, and G. Zhang, "Enhanced sparse model for blind deblurring," in *ECCV*, 2020.

[72] L. Yang and H. Ji, "A variational em framework with adaptive edge selection for blind motion deblurring," in *CVPR*, 2019.

[73] T. Michaeli and M. Irani, "Blind deblurring using internal patch recurrence," in *ECCV*, 2014.

[74] W.-S. Lai, J.-J. Ding, Y.-Y. Lin, and Y.-Y. Chuang, "Blur kernel estimation using normalized color-line prior," in *CVPR*, 2015.

[75] W. Ren, X. Cao, J. Pan, X. Guo, W. Zuo, and M.-H. Yang, "Image deblurring via enhanced low-rank prior," *TIP*, vol. 25, no. 7, pp. 3426–3437, 2016.

[76] L. Pan, R. Hartley, M. Liu, and Y. Dai, "Phase-only image based kernel estimation for single image blind deblurring," in *CVPR*, 2019.

[77] Y. Bai, H. Jia, M. Jiang, X. Liu, X. Xie, and W. Gao, "Single-image blind deblurring using multi-scale latent structure prior," *CSVT*, vol. 30, no. 7, pp. 2033–2045, 2019.

[78] L. Chen, F. Fang, T. Wang, and G. Zhang, "Blind image deblurring with local maximum gradient prior," in *CVPR*, 2019.

[79] A. Chakrabarti, "A neural approach to blind motion deblurring," in *ECCV*, 2016.

[80] S. Liu, J. Pan, and M.-H. Yang, "Learning recursive filters for low-level vision via a hybrid neural network," in *ECCV*, 2016, pp. 560–576.

[81] J. Zhang *et al.*, "Dynamic scene deblurring using spatially variant recurrent neural networks," in *CVPR*, 2018.

[82] T. M. Nimisha, A. Kumar Singh, and A. N. Rajagopalan, "Blur-invariant deep learning for blind-deblurring," in *ICCV*, 2017.

[83] D. Gong *et al.*, "From motion blur to motion flow: A deep learning solution for removing heterogeneous motion blur," in *CVPR*, 2017.

[84] X. Xu, J. Pan, Y.-J. Zhang, and M.-H. Yang, "Motion blur kernel estimation via deep learning," *TIP*, vol. 27, no. 1, pp. 194–205, 2017.

[85] J. Liu, W. Sun, and M. Li, "Recurrent conditional generative adversarial network for image deblurring," *IEEE Access*, vol. 7, pp. 6186–6193, 2018.

[86] D. Park, D. U. Kang, J. Kim, and S. Y. Chun, "Multi-temporal recurrent neural networks for progressive non-uniform single image deblurring with incremental temporal training," in *ECCV*, 2020.

[87] M. Asim, F. Shamshad, and A. Ahmed, "Blind image deconvolution using deep generative priors," *IEEE Transactions on Computational Imaging*, vol. 6, pp. 1493–1506, 2020.

[88] P. Tran, A. T. Tran, Q. Phung, and M. Hoai, "Explore image deblurring via encoded blur kernel space," in *CVPR*, 2021.

[89] C. Li, C. Zhu, J. Zhang, B. Luo, X. Wu, and J. Tang, "Learning local-global multi-graph descriptors for rgb-t object tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 10, pp. 2913–2926, 2018.

[90] X. Pan, X. Zhan, B. Dai, D. Lin, C. C. Loy, and P. Luo, "Exploiting deep generative prior for versatile image restoration and manipulation," *TPAMI*, pp. 1–1, 2021. DOI: 10.1109/TPAMI.2021.3115428.

[91] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte, "Plug-and-play image restoration with deep denoiser prior," *TPAMI*, 2021.

[92] Z. Cheng, M. Gadelha, S. Maji, and D. Sheldon, "A bayesian perspective on the deep image prior," in *CVPR*, 2019.

[93] K. Hoa, A. Gilberta, H. Jinb, and J. Collomossea, "Neural architecture search for deep image prior," *Computers & Graphics*, 2021.

[94] Y. Gandelsman, A. Shocher, and M. Irani, ""double-dip": Unsupervised image decomposition via coupled deep-image-priors," in *CVPR*, 2019.

[95] S. Abu-Hussein, T. Tirer, S. Y. Chun, Y. C. Eldar, and R. Giryes, "Image restoration by deep projected gsure," *arXiv preprint arXiv:2102.02485*, 2021.

[96] G. Mataev, P. Milanfar, and M. Elad, "Deepred: Deep image prior powered by red," in *ICCV*, 2019.

[97] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[98] A. Vahdat and J. Kautz, "NVAE: A deep hierarchical variational autoencoder," in *NeurIPS*, 2020.

[99] X. Zeng *et al.*, "Lion: Latent point diffusion models for 3d shape generation," *arXiv preprint arXiv:2210.06978*, 2022.

[100] S. Luo and W. Hu, "Diffusion probabilistic models for 3d point cloud generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2837–2845.

[101] L. Zhou, Y. Du, and J. Wu, "3d shape generation and completion through point-voxel diffusion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5826–5835.

[102] A. Nichol, H. Jun, P. Dhariwal, P. Mishkin, and M. Chen, "Point-e: A system for generating 3d point clouds from complex prompts," *arXiv preprint arXiv:2212.08751*, 2022.

[103] G. Nam, M. Khlifi, A. Rodriguez, A. Tono, L. Zhou, and P. Guerrero, "3d-ldm: Neural implicit 3d shape generation with latent diffusion models," *arXiv preprint arXiv:2212.00842*, 2022.

[104] T. Wang *et al.*, "Rodin: A generative model for sculpting 3d digital avatars using diffusion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 4563–4573.

[105] X. Yu, P. Dai, W. Li, L. Ma, Z. Liu, and X. Qi, "Texture generation on 3d meshes with point-uv diffusion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4206–4216.

[106] H. Jun and A. Nichol, "Shap-e: Generating conditional 3d implicit functions," *arXiv preprint arXiv:2305.02463*, 2023.

[107] A. Radford *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, PMLR, 2021, pp. 8748–8763.

[108] N. Mohammad Khalid, T. Xie, E. Belilovsky, and T. Popa, "Clip-mesh: Generating textured meshes from text using pretrained image-text models," in *SIGGRAPH Asia 2022 conference papers*, 2022, pp. 1–8.

[109] A. Sanghi *et al.*, "Clip-sculptor: Zero-shot generation of high-fidelity and diverse shapes from natural language," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 18 339–18 348.

[110] O. Michel, R. Bar-On, R. Liu, S. Benaim, and R. Hanocka, "Text2mesh: Text-driven neural stylization for meshes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 13 492–13 502.

[111] J. Lei, Y. Zhang, K. Jia, *et al.*, "Tango: Text-driven photorealistic and robust 3d stylization via lighting decomposition," *Advances in Neural Information Processing Systems*, vol. 35, pp. 30 923–30 936, 2022.

[112] B. Poole, A. Jain, J. T. Barron, and B. Mildenhall, "Dreamfusion: Text-to-3d using 2d diffusion," *arXiv preprint arXiv:2209.14988*, 2022.

[113] C.-H. Lin *et al.*, "Magic3d: High-resolution text-to-3d content creation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 300–309.

[114] G. Metzer, E. Richardson, O. Patashnik, R. Giryes, and D. Cohen-Or, "Latent-nerf for shape-guided generation of 3d shapes and textures," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 12 663–12 673.

[115] R. Chen, Y. Chen, N. Jiao, and K. Jia, "Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation," *arXiv preprint arXiv:2303.13873*, 2023.

[116] Z. Wang *et al.*, "Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation," *arXiv preprint arXiv:2305.16213*, 2023.

[117] J. Tang *et al.*, "Make-it-3d: High-fidelity 3d creation from a single image with diffusion prior," *arXiv preprint arXiv:2303.14184*, 2023.

[118] J. Sun *et al.*, "Dreamcraft3d: Hierarchical 3d generation with bootstrapped diffusion prior," *arXiv preprint arXiv:2310.16818*, 2023.

[119] Y. Guo *et al.*, "Decorate3d: Text-driven high-quality texture generation for mesh decoration in the wild," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[120] E. Richardson, G. Metzer, Y. Alaluf, R. Giryes, and D. Cohen-Or, "Texture: Text-guided texturing of 3d shapes," *arXiv preprint arXiv:2302.01721*, 2023.

[121] D. Z. Chen, Y. Siddiqui, H.-Y. Lee, S. Tulyakov, and M. Nießner, "Text2tex: Text-driven texture synthesis via diffusion models," *arXiv preprint arXiv:2303.11396*, 2023.

[122] T. Cao, K. Kreis, S. Fidler, N. Sharp, and K. Yin, "Texfusion: Synthesizing 3d textures with text-guided image diffusion models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4169–4181.

[123] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Transactions on Graphics (ToG)*, vol. 41, no. 4, pp. 1–15, 2022.

[124] A. Chakrabarti and T. Zickler, "Statistics of real-world hyperspectral images," in *CVPR 2011*, IEEE, 2011, pp. 193–200.

[125] N. Akhtar and A. Mian, "Hyperspectral recovery from rgb images using gaussian processes," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 1, pp. 100–113, 2018.

[126] Y. Jia *et al.*, "From rgb to spectrum for natural scenes via manifold-based mapping," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 4705–4713.

[127] R. Timofte, V. De Smet, and L. Van Gool, "A+: Adjusted anchored neighborhood regression for fast super-resolution," in *Computer Vision–ACCV 2014: 12th Asian Conference on Computer Vision, Singapore, Singapore, November 1-5, 2014, Revised Selected Papers, Part IV 12*, Springer, 2015, pp. 111–126.

[128] Z. Shi, C. Chen, Z. Xiong, D. Liu, and F. Wu, "Hscnn+: Advanced cnn-based hyperspectral recovery from rgb images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 939–947.

[129] Y. Cai *et al.*, "Mst++: Multi-stage spectral-wise transformer for efficient spectral reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 745–755.

[130] L. Zhang *et al.*, "Pixel-aware deep function-mixture network for spectral super-resolution," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 12 821–12 828.

[131] Z. Xiong, Z. Shi, H. Li, L. Wang, D. Liu, and F. Wu, "Hscnn: Cnn-based hyperspectral image recovery from spectrally undersampled projections," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 518–525.

[132] Y. Fu, T. Zhang, Y. Zheng, D. Zhang, and H. Huang, "Joint camera spectral sensitivity selection and hyperspectral image recovery," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 788–804.

[133] Y. Zhao, L.-M. Po, Q. Yan, W. Liu, and T. Lin, "Hierarchical regression network for spectral reconstruction from rgb images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 422–423.

[134] Y. Yan, L. Zhang, J. Li, W. Wei, and Y. Zhang, "Accurate spectral super-resolution from single rgb image using multi-scale cnn," in *Pattern Recognition and Computer Vision: First Chinese Conference, PRCV 2018, Guangzhou, China, November 23-26, 2018, Proceedings, Part II 1*, Springer, 2018, pp. 206–217.

[135] T. Stiebel, S. Koppers, P. Seltsam, and D. Merhof, "Reconstructing spectral images from rgb-images using a convolutional neural network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 948–953.

[136] R. Hang, Q. Liu, and Z. Li, "Spectral super-resolution network guided by intrinsic properties of hyperspectral imagery," *IEEE Transactions on Image Processing*, vol. 30, pp. 7256–7265, 2021.

[137] L. Li, L. Wang, W. Song, L. Zhang, Z. Xiong, and H. Huang, "Quantization-aware deep optics for diffractive snapshot hyperspectral imaging," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 19 780–19 789.

[138] K. Zhang, D. Zhu, X. Min, and G. Zhai, "Implicit neural representation learning for hyperspectral image super-resolution," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, pp. 1–12, 2022.

[139] W. Dong, C. Zhou, F. Wu, J. Wu, G. Shi, and X. Li, "Model-guided deep hyperspectral image super-resolution," *IEEE Transactions on Image Processing*, vol. 30, pp. 5754–5768, 2021.

[140] W. Wang, W. Zeng, Y. Huang, X. Ding, and J. Paisley, "Deep blind hyperspectral image fusion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4150–4159.

[141] Z. Zhu, J. Hou, J. Chen, H. Zeng, and J. Zhou, "Hyperspectral image super-resolution via deep progressive zero-centric residual learning," *IEEE Transactions on Image Processing*, vol. 30, pp. 1423–1438, 2020.

[142] S. Liu, A. Davison, and E. Johns, "Self-supervised generalisation with meta auxiliary learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[143] M. Andrychowicz *et al.*, "Learning to learn by gradient descent by gradient descent," *Advances in neural information processing systems*, vol. 29, 2016.

[144] Y. Zhang, H. Tang, and K. Jia, "Fine-grained visual categorization using meta-learning optimization with sample selection of auxiliary data," in *Proceedings of the european conference on computer vision (ECCV)*, 2018, pp. 233–248.

[145] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International conference on machine learning*, PMLR, 2017, pp. 1126–1135.

[146] Y. Sun, X. Wang, Z. Liu, J. Miller, A. Efros, and M. Hardt, "Test-time training with self-supervision for generalization under distribution shifts," in *International conference on machine learning*, PMLR, 2020, pp. 9229–9248.

[147] Q. Sun, Y. Liu, T.-S. Chua, and B. Schiele, "Meta-transfer learning for few-shot learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 403–412.

[148] S. Park, J. Yoo, D. Cho, J. Kim, and T. H. Kim, "Fast adaptation to super-resolution networks via meta-learning," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVII 16*, Springer, 2020, pp. 754–769.

[149] J. W. Soh, S. Cho, and N. I. Cho, "Meta-transfer learning for zero-shot super-resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3516–3525.

[150] Z. Wang, J. Chen, and S. C. Hoi, "Deep learning for image super-resolution: A survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 10, pp. 3365–3387, 2020.

[151] Y. Guo *et al.*, "Closed-loop matters: Dual regression networks for single image super-resolution," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 5407–5416.

[152] A. Valada, N. Radwan, and W. Burgard, "Deep auxiliary learning for visual localization and odometry," in *2018 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2018, pp. 6939–6946.

[153] K. Lu, N. Barnes, S. Anwar, and L. Zheng, "From depth what can you see? depth completion via auxiliary image reconstruction," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 306–11 315.

[154] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," in *International conference on machine learning*, PMLR, 2017, pp. 2642–2651.

[155] M. Fritz, G. Bradski, S. Karayev, T. Darrell, and M. Black, "An additive latent feature model for transparent object recognition," *Advances in Neural Information Processing Systems*, vol. 22, pp. 558–566, 2009.

[156] K. McHenry and J. Ponce, "A geodesic active contour framework for finding glass," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, IEEE, vol. 1, 2006, pp. 1038–1044.

[157] K. McHenry, J. Ponce, and D. Forsyth, "Finding glass," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, IEEE, vol. 2, 2005, pp. 973–979.

[158] C. J. Phillips, K. G. Derpanis, and K. Daniilidis, "A novel stereoscopic cue for figure-ground segregation of semi-transparent objects," in *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, IEEE, 2011, pp. 1100–1107.

[159] E. Xie, W. Wang, W. Wang, M. Ding, C. Shen, and P. Luo, "Segmenting transparent objects in the wild," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*, Springer, 2020, pp. 696–711.

[160] E. Xie *et al.*, "Segmenting transparent object in the wild with transformer," *arXiv preprint arXiv:2101.08461*, 2021.

[161] C. Zheng *et al.*, "Glassnet: Label decoupling-based three-stream neural network for robust image glass detection," *arXiv preprint arXiv:2108.11117*, 2021.

[162] Y. Zhu, J. Qiu, and B. Ren, "Transfusion: A novel slam method focused on transparent objects," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6019–6028.

[163] G. Chen, K. Han, and K.-Y. K. Wong, "Tom-net: Learning transparent object matting from a single image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9233–9241.

[164] V. Seib, A. Barthen, P. Marohn, and D. Paulus, "Friend or foe: Exploiting sensor failures for transparent object localization and classification," in *2016 International Conference on Robotics and Machine Vision*, International Society for Optics and Photonics, vol. 10253, 2017, p. 102530I.

[165] T. Whelan *et al.*, "Reconstructing scenes with mirror and glass surfaces.," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 102–1, 2018.

[166] Y. Li, X. Hou, C. Koch, J. M. Rehg, and A. L. Yuille, "The secrets of salient object segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 280–287.

[167] F. Perazzi, P. Krähenbühl, Y. Pritch, and A. Hornung, "Saliency filters: Contrast based filtering for salient region detection," in *2012 IEEE conference on computer vision and pattern recognition*, IEEE, 2012, pp. 733–740.

[168] Q. Yan, L. Xu, J. Shi, and J. Jia, "Hierarchical saliency detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 1155–1162.

[169] Y. Pang, X. Zhao, L. Zhang, and H. Lu, "Multi-scale interactive network for salient object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9413–9422.

[170] H. Zhou, X. Xie, J.-H. Lai, Z. Chen, and L. Yang, "Interactive two-stream decoder for accurate and fast saliency detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9141–9150.

[171] L. Tang, B. Li, Y. Zhong, S. Ding, and M. Song, "Disentangled high quality salient object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 3580–3590.

[172] A. Siris, J. Jiao, G. K. Tam, X. Xie, and R. W. Lau, "Scene context-aware salient object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 4156–4166.

[173] K. Fu, D.-P. Fan, G.-P. Ji, and Q. Zhao, "Jl-dcf: Joint learning and densely-cooperative fusion framework for rgb-d salient object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 3052–3062.

[174] J. Zhang *et al.*, "Uc-net: Uncertainty inspired rgb-d saliency detection via conditional variational autoencoders," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 8582–8591.

[175] X. Zhao, L. Zhang, Y. Pang, H. Lu, and L. Zhang, "A single stream network for robust and real-time rgb-d salient object detection," in *European Conference on Computer Vision*, Springer, 2020, pp. 646–662.

[176] Q. Chen, Z. Liu, Y. Zhang, K. Fu, Q. Zhao, and H. Du, "Rgb-d salient object detection via 3d convolutional neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, 2021, pp. 1063–1071.

[177] Y. Pang, L. Zhang, X. Zhao, and H. Lu, "Hierarchical dynamic filtering network for rgb-d salient object detection," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV 16*, Springer, 2020, pp. 235–252.

[178] Y. Piao, Z. Rong, M. Zhang, W. Ren, and H. Lu, "A2dele: Adaptive and attentive depth distiller for efficient rgb-d salient object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9060–9069.

[179] M. Zhang, W. Ren, Y. Piao, Z. Rong, and H. Lu, "Select, supplement and focus for rgb-d saliency detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3472–3481.

[180] D.-P. Fan, Y. Zhai, A. Borji, J. Yang, and L. Shao, "Bbs-net: Rgb-d salient object detection with a bifurcated backbone strategy network," in *European Conference on Computer Vision*, Springer, 2020, pp. 275–292.

[181] P. Sun, W. Zhang, H. Wang, S. Li, and X. Li, "Deep rgb-d saliency detection with depth-sensitive attention and automatic multi-modal fusion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1407–1417.

[182] W. Ji *et al.*, "Calibrated rgb-d salient object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9471–9481.

[183] S. Chen and Y. Fu, "Progressively guided alternate refinement network for rgb-d salient object detection," in *European Conference on Computer Vision*, Springer, 2020, pp. 520–538.

[184] C. Li, R. Cong, Y. Piao, Q. Xu, and C. C. Loy, "Rgb-d salient object detection with cross-modality modulation and selection," in *European Conference on Computer Vision*, Springer, 2020, pp. 225–241.

[185] T. Zhou, H. Fu, G. Chen, Y. Zhou, D.-P. Fan, and L. Shao, "Specificity-preserving rgb-d saliency detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 4681–4691.

[186] A. Luo, X. Li, F. Yang, Z. Jiao, H. Cheng, and S. Lyu, "Cascade graph neural networks for rgb-d salient object detection," in *European Conference on Computer Vision*, Springer, 2020, pp. 346–364.

[187] J. Zhang *et al.*, "Rgb-d saliency detection via cascaded mutual information minimization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 4338–4347.

[188] N. Liu, N. Zhang, and J. Han, "Learning selective self-mutual attention for rgb-d saliency detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 13 756–13 765.

[189] M. Zhang, S. X. Fei, J. Liu, S. Xu, Y. Piao, and H. Lu, "Asymmetric two-stream architecture for accurate rgb-d saliency detection," in *European Conference on Computer Vision*, Springer, 2020, pp. 374–390.

[190] N. Liu, N. Zhang, K. Wan, L. Shao, and J. Han, "Visual saliency transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 4722–4732.

[191] Z. Tu, T. Xia, C. Li, X. Wang, Y. Ma, and J. Tang, "Rgb-t image saliency detection via collaborative graph learning," *IEEE Transactions on Multimedia*, vol. 22, no. 1, pp. 160–173, 2019.

[192] Q. Zhang, T. Xiao, N. Huang, D. Zhang, and J. Han, "Revisiting feature fusion for rgb-t salient object detection," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 5, pp. 1804–1818, 2020.

[193] W. Zhou, Q. Guo, J. Lei, L. Yu, and J.-N. Hwang, "Ecffnet: Effective and consistent feature fusion network for rgb-t salient object detection," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 3, pp. 1224–1235, 2021.

[194] W. Zhou, Y. Zhu, J. Lei, J. Wan, and L. Yu, "Apnet: Adversarial learning assistance and perceived importance fusion network for all-day rgb-t salient object detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2021.

[195] F. Sun, W. Zhou, L. Ye, and L. Yu, "Hierarchical decoding network based on swin transformer for detecting salient objects in rgb-t images," *IEEE Signal Processing Letters*, vol. 29, pp. 1714–1718, 2022.

[196] J. Wu, W. Zhou, X. Qian, J. Lei, L. Yu, and T. Luo, "Mfenet: Multitype fusion and enhancement network for detecting salient objects in rgb-t images," *Digital Signal Processing*, vol. 133, p. 103 827, 2023.

[197] Z. Tu, Z. Li, C. Li, and J. Tang, "Weakly alignment-free rgbt salient object detection with deep correlation network," *IEEE Transactions on Image Processing*, 2022.

[198] C. Li, X. Liang, Y. Lu, N. Zhao, and J. Tang, "Rgb-t object tracking: Benchmark and baseline," *Pattern Recognition*, vol. 96, p. 106 977, 2019.

[199] L. Zhang, M. Danelljan, A. Gonzalez-Garcia, J. van de Weijer, and F. Shahbaz Khan, "Multi-modal fusion for end-to-end rgb-t tracking," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[200] C. Wang *et al.*, "Cross-modal pattern-propagation for rgb-t tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7064–7073.

[201] P. Zhang, J. Zhao, C. Bo, D. Wang, H. Lu, and X. Yang, "Jointly modeling motion and appearance cues for robust rgb-t tracking," *IEEE Transactions on Image Processing*, vol. 30, pp. 3335–3347, 2021.

[202] F. Zhang, S. Ma, Z. Li, and Y. Zhang, "Rgb-t tracking via multi-expert correlation filters using spatial-temporal robustness," in *2020 5th International Conference on Mechanical, Control and Computer Engineering (ICMCCE)*, IEEE, 2020, pp. 360–364.

[203] Z. Tu, W. Pan, Y. Duan, J. Tang, and C. Li, "Rgbt tracking via reliable feature configuration," *Science China Information Sciences*, vol. 65, no. 4, pp. 1–13, 2022.

[204] X. Zhang, P. Ye, S. Peng, J. Liu, and G. Xiao, "Dsiammft: An rgb-t fusion tracking method via dynamic siamese networks using multi-layer feature fusion," *Signal Processing: Image Communication*, vol. 84, p. 115 756, 2020.

[205] C. Li, X. Wang, L. Zhang, J. Tang, H. Wu, and L. Lin, "Weighted low-rank decomposition for robust grayscale-thermal foreground detection," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 4, pp. 725–738, 2016.

[206] B. Zhao, Z. Li, M. Liu, W. Cao, and H. Liu, "Infrared and visible imagery fusion based on region saliency detection for 24-hour-surveillance systems," in *2013 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, IEEE, 2013, pp. 1083–1088.

[207] S. Yang, B. Luo, C. Li, G. Wang, and J. Tang, "Fast grayscale-thermal foreground detection with collaborative low-rank decomposition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 2574–2585, 2017.

[208] T. Alldieck, C. H. Bahnsen, and T. B. Moeslund, "Context-aware fusion of rgb and thermal imagery for traffic monitoring," *Sensors*, vol. 16, no. 11, p. 1947, 2016.

[209] S. Gundimada, V. K. Asari, and N. Gudur, "Face recognition in multi-sensor images based on a novel modular feature selection technique," *Information Fusion*, vol. 11, no. 2, pp. 124–132, 2010.

[210] W. Zhou, J. Liu, J. Lei, L. Yu, and J.-N. Hwang, "Gmnet: Graded-feature multilabel-learning network for rgb-thermal urban scene semantic segmentation," *IEEE Transactions on Image Processing*, vol. 30, pp. 7790–7802, 2021.

[211] Q. Zhang, S. Zhao, Y. Luo, D. Zhang, N. Huang, and J. Han, "Abmdrnet: Adaptive-weighted bi-directional modality difference reduction network for rgb-t semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2633–2642.

[212] Q. Ha, K. Watanabe, T. Karasawa, Y. Ushiku, and T. Harada, "Mfnet: Towards real-time semantic segmentation for autonomous vehicles with multi-spectral scenes," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2017, pp. 5108–5115.

[213] S. S. Shivakumar, N. Rodrigues, A. Zhou, I. D. Miller, V. Kumar, and C. J. Taylor, "Pst900: Rgb-thermal calibration, dataset and segmentation network," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 9441–9447.

[214] Y. Sun, W. Zuo, and M. Liu, "Rtfnet: Rgb-thermal fusion network for semantic segmentation of urban scenes," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2576–2583, 2019.

[215] Z. Kütük and G. Algan, "Semantic segmentation for thermal images: A comparative survey," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 286–295.

[216] Y. Sun, W. Zuo, P. Yun, H. Wang, and M. Liu, "Fuseseg: Semantic segmentation of urban scenes based on rgb and thermal data fusion," *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 3, pp. 1000–1011, 2020.

[217] T. Gong, W. Zhou, X. Qian, J. Lei, and L. Yu, "Global contextually guided lightweight network for rgb-thermal urban scene understanding," *Engineering Applications of Artificial Intelligence*, vol. 117, p. 105 510, 2023.

[218] W. Zhou, S. Dong, C. Xu, and Y. Qian, "Edge-aware guidance fusion network for rgb–thermal scene parsing," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, 2022, pp. 3571–3579.

[219] W. Zhou, X. Lin, J. Lei, L. Yu, and J.-N. Hwang, "Mffenet: Multiscale feature fusion and enhancement network for rgb–thermal urban road scene parsing," *IEEE Transactions on Multimedia*, vol. 24, pp. 2526–2538, 2021.

[220] W. Zhou, S. Dong, J. Lei, and L. Yu, "Mtanet: Multitask-aware network with hierarchical multimodal fusion for rgb-t urban scene understanding," *IEEE Transactions on Intelligent Vehicles*, 2022.

[221] Y.-L. Hou, Y. Jia, Z. Hou, X. Hao, and Y. Shen, "Iaffnet: Illumination-aware feature fusion network for all-day rgb-thermal semantic segmentation of road scenes," *IEEE Access*, 2022.

[222] W. Zhou, Y. Pan, J. Lei, L. Ye, and L. Yu, "Defnet: Dual-branch enhanced feature fusion network for rgb-t crowd counting," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 24 540–24 549, 2022.

[223] L. Liu, J. Chen, H. Wu, G. Li, C. Li, and L. Lin, "Cross-modal collaborative representation learning and a large-scale rgbt benchmark for crowd counting," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 4823–4833.

[224] E. Pahwa, S. Kapadia, A. Luthra, and S. Sheeranali, "Conditional rgb-t fusion for effective crowd counting," in *2022 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2022, pp. 376–380.

[225] Z. Liu, Y. Tan, W. Wu, and B. Tang, "Dilated high-resolution network driven rgb-t multi-modal crowd counting," *Signal Processing: Image Communication*, p. 116 915, 2022.

[226] Q. Zhang, N. Huang, L. Yao, D. Zhang, C. Shan, and J. Han, "Rgb-t salient object detection via fusing multi-level cnn features," *IEEE Transactions on Image Processing*, vol. 29, pp. 3321–3335, 2019.

[227] G. Wang, C. Li, Y. Ma, A. Zheng, J. Tang, and B. Luo, "Rgb-t saliency detection benchmark: Dataset, baselines, analysis and a novel approach," in *Chinese Conference on Image and Graphics Technologies*, Springer, 2018, pp. 359–369.

[228] Z. Tu, T. Xia, C. Li, Y. Lu, and J. Tang, "M3s-nir: Multi-modal multi-scale noise-insensitive ranking for rgb-t saliency detection," in *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, IEEE, 2019, pp. 141–146.

[229] W. Gao, G. Liao, S. Ma, G. Li, Y. Liang, and W. Lin, "Unified information fusion network for multi-modal rgb-d and rgb-t salient object detection," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[230] Q. Guo, W. Zhou, J. Lei, and L. Yu, "Tsfnet: Two-stage fusion network for rgb-t salient object detection," *IEEE Signal Processing Letters*, vol. 28, pp. 1655–1659, 2021.

[231] F. Huo, X. Zhu, L. Zhang, Q. Liu, and Y. Shu, "Efficient context-guided stacked refinement network for rgb-t salient object detection," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[232] J. Wang, K. Song, Y. Bao, L. Huang, and Y. Yan, "Cgfnet: Cross-guided fusion network for rgb-t salient object detection," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[233] Z. Tu, Z. Li, C. Li, Y. Lang, and J. Tang, "Multi-interactive dual-decoder for rgb-thermal salient object detection," *IEEE Transactions on Image Processing*, vol. 30, pp. 5678–5691, 2021.

[234] R. Lukac, *Computational photography: methods and applications*. CRC press, 2017.

[235] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural Computation*, vol. 29, no. 9, pp. 2352–2449, 2017.

[236] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[237] I. Adjabi, A. Ouahabi, A. Benzaoui, and A. Taleb-Ahmed, "Past, present, and future of face recognition: A review," *Electronics*, vol. 9, no. 8, p. 1188, 2020.

[238] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015.

[239] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," *arXiv preprint arXiv:1607.08022*, 2016.

[240] B. Lu, J.-C. Chen, and R. Chellappa, "Unsupervised domain-specific deblurring via disentangled representations," in *CVPR*, 2019.

[241] X. Zhu, H. Hu, S. Lin, and J. Dai, "Deformable convnets v2: More deformable, better results," in *CVPR*, 2019, pp. 9308–9316.

[242] Y. Yuan, W. Su, and D. Ma, "Efficient dynamic scene deblurring using spatially variant deconvolution network with optical flow guided training," in *CVPR*, 2020, pp. 3555–3564.

[243] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *ICLR*, 2016.

[244] Z. Deng *et al.*, "Deep multi-model fusion for single-image dehazing," in *ICCV*, 2019.

[245] X. Jia, B. De Brabandere, T. Tuytelaars, and L. V. Gool, "Dynamic filter networks," in *NeurIPS*, 2016.

[246] Z. Shen *et al.*, "Human-aware motion deblurring," in *ICCV*, 2019.

[247] K. Zhang *et al.*, "Deblurring by realistic blurring," in *CVPR*, 2020, pp. 2737–2746.

[248] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, pp. 8026–8037, 2019.

[249] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[250] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *AISTATS*, 2010.

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

[252] R. Aljadaany, D. K. Pal, and M. Savvides, "Douglas-rachford networks: Learning both the image prior and data fidelity terms for blind image deconvolution," in *CVPR*, 2019, pp. 10 235–10 244.

[253] D. Park, D. U. Kang, J. Kim, and S. Y. Chun, "Multi-temporal recurrent neural networks for progressive non-uniform single image deblurring with incremental temporal training," in *ECCV*, Springer, 2020, pp. 327–343.

[254] K. Purohit and A. Rajagopalan, "Region-adaptive dense network for efficient motion deblurring," in *AAAI*, 2020, pp. 11 882–11 889.

[255] M. Nadipally, "Chapter 2 - optimization of methods for image-texture segmentation using ant colony optimization," in *Intelligent Data Analysis for Biomedical Applications*, ser. Intelligent Data-Centric Systems, D. J. Hemanth, D. Gupta, and V. Emilia Balas, Eds., Academic Press, 2019, pp. 21 –47, ISBN: 978-0-12-815553-0. DOI: https://doi.org/10.1016/B978-0-12-815553-0.00002-1. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780128155530000021.

[256] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[257] I. Goodfellow *et al.*, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[258] S. Kullback, *Information theory and statistics*. Courier Corporation, 1997.

[259] J. A. Palmer, K. Kreutz-Delgado, and S. Makeig, "Strong sub-and super-gaussianity," in *International Conference on Latent Variable Analysis and Signal Separation*, 2010.

[260] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[261] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, "Efficient marginal likelihood optimization in blind deconvolution," in *CVPR*, 2011.

[262] D. Perrone and P. Favaro, "Total variation blind deconvolution: The devil is in the details," in *CVPR*, 2014.

[263] F. Wen, R. Ying, Y. Liu, P. Liu, and T.-K. Truong, "A simple local minimal intensity prior and an improved algorithm for blind image deblurring," *CSVT*, 2020.

[264] D. Huo, A. Masoumzadeh, and Y.-H. Yang, "Blind non-uniform motion deblurring using atrous spatial pyramid deformable convolution and deblurring-reblurring consistency," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 437–446.

[265] L. Chen, X. Chu, X. Zhang, and J. Sun, "Simple baselines for image restoration," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part VII*, Springer, 2022, pp. 17–33.

[266] W.-S. Lai, J.-B. Huang, Z. Hu, N. Ahuja, and M.-H. Yang, "A comparative study for single image blind deblurring," in *CVPR*, 2016.

[267] S. Nah, S. Son, S. Lee, R. Timofte, and K. M. Lee, "Ntire 2021 challenge on image deblurring," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 149–165.

[268] Y. Zhang, Y. Lau, H.-w. Kuo, S. Cheung, A. Pasupathy, and J. Wright, "On the global geometry of sphere-constrained sparse blind deconvolution," in *CVPR*, 2017.

[269] A. Mittal, R. Soundararajan, and A. C. Bovik, "Making a "completely blind" image quality analyzer," *SPL*, vol. 20, no. 3, pp. 209–212, 2012.

[270] A. Mittal, A. K. Moorthy, and A. C. Bovik, "Blind/referenceless image spatial quality evaluator," in *ASILOMAR*, IEEE, 2011, pp. 723–727.

[271] N Venkatanath, D Praneeth, M. C. Bh, S. S. Channappayya, and S. S. Medasani, "Blind image quality evaluation using perception based features," in *NCC*, IEEE, 2015, pp. 1–6.

[272] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," *arXiv preprint arXiv:2010.02502*, 2020.

[273] M. Cao, X. Wang, Z. Qi, Y. Shan, X. Qie, and Y. Zheng, "Masactrl: Tuning-free mutual self-attention control for consistent image synthesis and editing," *arXiv preprint arXiv:2304.08465*, 2023.

[274] J. Ho and T. Salimans, "Classifier-free diffusion guidance," *arXiv preprint arXiv:2207.12598*, 2022.

[275] J. Young, "Xatlas," in *github.com/jpcy/xatlas*, 2016.

[276] M. Deitke *et al.*, "Objaverse: A universe of annotated 3d objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 13 142–13 153.

[277] *Three d scans*, https://threedscans.com, 2012.

[278] Y.-C. Guo *et al.*, *Threestudio: A unified framework for 3d content generation*, https://github.com/threestudio-project/threestudio, 2023.

[279] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in neural information processing systems*, vol. 30, 2017.

[280] M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton, "Demystifying mmd gans," *arXiv preprint arXiv:1801.01401*, 2018.

[281] J. Hessel, A. Holtzman, M. Forbes, R. L. Bras, and Y. Choi, "Clipscore: A reference-free evaluation metric for image captioning," *arXiv preprint arXiv:2104.08718*, 2021.

[282] A. F. Goetz, G. Vane, J. E. Solomon, and B. N. Rock, "Imaging spectrometry for earth remote sensing," *science*, vol. 228, no. 4704, pp. 1147–1153, 1985.

[283] M. R. Nanni and J. A. M. Demattê, "Spectral reflectance methodology in comparison to traditional soil analysis," *Soil science society of America journal*, vol. 70, no. 2, pp. 393–407, 2006.

[284] G. A. Carter, "Responses of leaf spectral reflectance to plant stress," *American journal of botany*, vol. 80, no. 3, pp. 239–243, 1993.

[285] A. Martínez-Usó, F. Pla, and P. García-Sevilla, "Multispectral image segmentation by energy minimization for fruit quality estimation," in *Pattern Recognition and Image Analysis: Second Iberian Conference, IbPRIA 2005, Estoril, Portugal, June 7-9, 2005, Proceedings, Part II 2*, Springer, 2005, pp. 689–696.

[286] N. Tsumura, Y. Miyake, and F. H. Imai, "Medical vision: Measurement of skin absolute spectral-reflectance image and the application to component analysis," in *Proceedings of the 3rd International Conference on Multispectral Color Science (MCS'01)*, Citeseer, 2001, pp. 25–28.

[287] S. J. Preece and E. Claridge, "Monte carlo modelling of the spectral reflectance of the human eye," *Physics in Medicine & Biology*, vol. 47, no. 16, p. 2863, 2002.

[288] S. Kim *et al.*, "Smartphone-based multispectral imaging: System development and potential for mobile skin diagnosis," *Biomedical optics express*, vol. 7, no. 12, pp. 5294–5307, 2016.

[289] Q. He and R. Wang, "Hyperspectral imaging enabled by an unmodified smartphone for analyzing skin morphological features and monitoring hemodynamics," *Biomedical optics express*, vol. 11, no. 2, pp. 895–910, 2020.

[290] N.-N. Wang, D.-W. Sun, Y.-C. Yang, H. Pu, and Z. Zhu, "Recent advances in the application of hyperspectral imaging for evaluating fruit quality," *Food analytical methods*, vol. 9, no. 1, pp. 178–191, 2016.

[291] G. ElMasry, D. F. Barbin, D.-W. Sun, and P. Allen, "Meat quality evaluation by hyperspectral imaging technique: An overview," *Critical reviews in food science and nutrition*, vol. 52, no. 8, pp. 689–711, 2012.

[292] A. L. Maas, A. Y. Hannun, A. Y. Ng, *et al.*, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. icml*, Atlanta, Georgia, USA, vol. 30, 2013, p. 3.

[293] A. Dosovitskiy *et al.*, "Flownet: Learning optical flow with convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2758–2766.

[294] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8934–8943.

[295] B. Aiazzi, L. Alparone, S. Baronti, and A. Garzelli, "Context-driven fusion of high spatial and spectral resolution images based on oversampled multiresolution analysis," *IEEE Transactions on geoscience and remote sensing*, vol. 40, no. 10, pp. 2300–2312, 2002.

[296] Y. Monno, H. Teranaka, K. Yoshizaki, M. Tanaka, and M. Okutomi, "Single-sensor rgb-nir imaging: High-quality system design and prototype implementation," *IEEE Sensors Journal*, vol. 19, no. 2, pp. 497–507, 2018.

[297] J. Roby and M. Aubé, *Lspdd: Lamp spectral power distribution database*, Retrieved from https://lspdd.org/app/en/lamps/2629, 2021.

[298] J. Jiang, D. Liu, J. Gu, and S. Süsstrunk, "What is the space of spectral sensitivity functions for digital color cameras?" In *2013 IEEE Workshop on Applications of Computer Vision (WACV)*, IEEE, 2013, pp. 168–179.

[299] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the seventh IEEE international conference on computer vision*, Ieee, vol. 2, 1999, pp. 1150–1157.

[300] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.

[301] P. E. Dennison, K. Q. Halligan, and D. A. Roberts, "A comparison of error metrics and constraints for multiple endmember spectral mixture analysis and spectral angle mapper," *Remote Sensing of Environment*, vol. 93, no. 3, pp. 359–367, 2004.

[302] J. Korhonen and J. You, "Peak signal-to-noise ratio revisited: Is simple beautiful?" In *2012 Fourth International Workshop on Quality of Multimedia Experience*, IEEE, 2012, pp. 37–38.

[303] S. Li, W. Song, L. Fang, Y. Chen, P. Ghamisi, and J. A. Benediktsson, "Deep learning for hyperspectral image classification: An overview," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 9, pp. 6690–6709, 2019.

[304] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi, and J. Chanussot, "Hyperspectral remote sensing data analysis and future challenges," *IEEE Geoscience and remote sensing magazine*, vol. 1, no. 2, pp. 6–36, 2013.

[305] P. S. Thenkabail, I. Mariotto, M. K. Gumma, E. M. Middleton, D. R. Landis, and K. F. Huemmrich, "Selection of hyperspectral narrowbands (hnbs) and composition of hyperspectral twoband vegetation indices (hvis) for biophysical characterization and discrimination of crop types using field reflectance and hyperion/eo-1 data," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 6, no. 2, pp. 427–439, 2013.

[306] M. Vollmer and K.-P. Möllmann, *Infrared thermal imaging: fundamentals, research and applications*. Germany, Wiley, 2017, p. 61.

[307] A. K. Varshneya, "Industrial glass," *Encyclopedia Britannica*, May 10, 2016.

[308] C. B. Carter and M. G. Norton, *Ceramic materials: science and engineering*. United Kingdom, Springer, 2007, p. 583.

[309] M. Gehm and D. Brady, "Compressive sensing in the eo/ir," *Applied optics*, vol. 54, no. 8, pp. C14–C22, 2015.

[310] FLIR, *Flir one pro*, Retrieved from https://www.flir.com/products/flir-one-pro/, 2021.

[311] FLIR, *Flir thermal studio suite*, Retrieved from https://www.flir.com/products/flir-thermal-studio-suite/, 2021.

[312] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "Labelme: A database and web-based tool for image annotation," *International journal of computer vision*, vol. 77, no. 1-3, pp. 157–173, 2008.

[313] S. Tosi, *Matplotlib for Python developers*. Packt Publishing Ltd, 2009.

[314] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, Springer, 2015, pp. 234–241.

[315] N. Parmar *et al.*, "Image transformer," in *International Conference on Machine Learning*, PMLR, 2018, pp. 4055–4064.

[316] A. Prakash, K. Chitta, and A. Geiger, "Multi-modal fusion transformer for end-to-end autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7077–7087.

[317] A. Vaswani *et al.*, "Attention is all you need," *NeurIPS*, vol. 30, 2017.

[318] Y. Lee and J. Park, "Centermask: Real-time anchor-free instance segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 13 906–13 915.

[319] J. Cao, H. Leng, D. Lischinski, D. Cohen-Or, C. Tu, and Y. Li, "Shapeconv: Shape-aware convolutional layer for indoor rgb-d semantic segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 7088–7097.

[320] D. Seichter, M. Köhler, B. Lewandowski, T. Wengefeld, and H.-M. Gross, "Efficient rgb-d semantic segmentation for indoor scene analysis," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, pp. 13 525–13 531.

[321] H. Liu, J. Zhang, K. Yang, X. Hu, and R. Stiefelhagen, "Cmx: Cross-modal fusion for rgb-x semantic segmentation with transformers," *arXiv preprint arXiv:2203.04838*, 2022.

[322] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12 077–12 090, 2021.

[323] R. Strudel, R. Garcia, I. Laptev, and C. Schmid, "Segmenter: Transformer for semantic segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 7262–7272.

[324] H. Xiong, W. Cai, and Q. Liu, "Mcnet: Multi-level correction network for thermal image semantic segmentation of nighttime driving scene," *Infrared Physics & Technology*, vol. 113, p. 103 628, 2021.

[325] Z. Chen, R. Cong, Q. Xu, and Q. Huang, "Dpanet: Depth potentiality-aware gated attention network for rgb-d salient object detection," *IEEE Transactions on Image Processing*, 2020.

[326] G. Li, Z. Liu, M. Chen, Z. Bai, W. Lin, and H. Ling, "Hierarchical alternate interaction network for rgb-d salient object detection," *IEEE Transactions on Image Processing*, vol. 30, pp. 3528–3542, 2021.

[327] W. Ji, J. Li, M. Zhang, Y. Piao, and H. Lu, "Accurate rgb-d salient object detection via collaborative learning," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*, Springer, 2020, pp. 52–69.

[328] M. Zhang, Y. Zhang, Y. Piao, B. Hu, and H. Lu, "Feature reintegration over differential treatment: A top-down and adaptive fusion network for rgb-d salient object detection," in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020, pp. 4107–4115.

[329] Y. Zhao, J. Zhao, J. Li, and X. Chen, "Rgb-d salient object detection with ubiquitous target awareness," *IEEE Transactions on Image Processing*, vol. 30, pp. 7717–7731, 2021.

[330] J. Zhang, J. Xie, N. Barnes, and P. Li, "Learning generative vision transformer with energy-based latent space for saliency prediction," *Advances in Neural Information Processing Systems*, vol. 34, pp. 15 448–15 463, 2021.

[331] N. Kitaev, L. Kaiser, and A. Levskaya, "Reformer: The efficient transformer," in *International Conference on Learning Representations*, 2019.

[332] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[333] X. Yang, H. Mei, K. Xu, X. Wei, B. Yin, and R. W. Lau, "Where is my mirror?" In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 8809–8818.

[334] Y. Xu, W. Xu, D. Cheung, and Z. Tu, "Line segment detection using transformers without edges," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4257–4266.

[335] Z. Tu, Y. Ma, Z. Li, C. Li, J. Xu, and Y. Liu, "Rgbt salient object detection: A large-scale dataset and benchmark," *IEEE Transactions on Multimedia*, 2022.

[336] Y. Liang, G. Qin, M. Sun, J. Qin, J. Yan, and Z. Zhang, "Multi-modal interactive attention and dual progressive decoding network for rgb-d/t salient object detection," *Neurocomputing*, vol. 490, pp. 132–145, 2022.

[337] D.-P. Fan, M.-M. Cheng, Y. Liu, T. Li, and A. Borji, "Structure-measure: A new way to evaluate foreground maps," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 4548–4557.

[338] S. F. Bhat, I. Alhashim, and P. Wonka, "Adabins: Depth estimation using adaptive bins," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4009–4018.

[339] M. Ramamonjisoa, M. Firman, J. Watson, V. Lepetit, and D. Turmukhambetov, "Single image depth prediction with wavelet decomposition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 089–11 098.

[340] A. Zhang and J. Sun, "Joint depth and defocus estimation from a single image using physical consistency," *IEEE Transactions on Image Processing*, vol. 30, pp. 3419–3433, 2021.

[341] J. Tan, W. Lin, A. X. Chang, and M. Savva, "Mirror3d: Depth refinement for mirror surfaces," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 990–15 999.

[342] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3d: A modern library for 3d data processing," *arXiv preprint arXiv:1801.09847*, 2018.

[343] A. Douillard, Y. Chen, A. Dapogny, and M. Cord, "Plop: Learning without forgetting for continual semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4040–4050.

[344] J.-H. Zhang, Y. Zhang, and Z. Shi, "Long-wave infrared polarization feature extraction and image fusion based on the orthogonality difference method," *Journal of Electronic Imaging*, vol. 27, no. 2, p. 023 021, 2018.

[345] Wikipedia, *Normal distribution*, https://en.wikipedia.org/wiki/Normal_distribution, 2022.

[346] H. V. Henderson and S. R. Searle, "On deriving the inverse of a sum of matrices," *Siam Review*, vol. 23, no. 1, pp. 53–60, 1981.

[347] iApplePro, *How to change flashlight color on iphone xr/xs/12*, Retrieved from https://www.youtube.com/watch?v=jwkwtCZ_MrM, 2021.

[348] J.-I. Park, M.-H. Lee, M. D. Grossberg, and S. K. Nayar, "Multispectral imaging using multiplexed illumination," in *2007 IEEE 11th International Conference on Computer Vision*, IEEE, 2007, pp. 1–8.

[349] H. He *et al.*, "Enhanced boundary learning for glass-like object segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 859–15 868.

Appendix A: Blind Non-Uniform Motion Deblurring

A.1 More Qualitative Comparison

In this section, we demonstrate more qualitative comparisons. Firstly, we look at the GoPro dataset [24]. In Figure A.1, Ours+ can recover the face and the body of the girl with less distortion. In Figure A.2, the original image contains extremely large motion, but Ours+ can remove most of the artifacts and obtain the sharpest image among the listed methods. In Figure A.3, Ours+ can recover clearer and smoother numbers on the plate.

Using the models trained on the GoPro dataset, we also compare the qualitative performance of the methods on the HIDE dataset [246], containing multiple moving human subjects in the scenes. In Figure A.4, Ours+ sharpens all the people in the image with much less artifacts compared to the other competing methods. Figure A.5 shows that Ours+ deblurs the striped jacket closest to that of the ground truth. In Figure A.6, Ours+ provides a relatively better deblurring of the car when compared with the other methods.

Lastly, we review some outputs from the Real-World Blurred Image (RWBI) dataset [247]. In Figure A.7, Our+ is the only method that gives sharp output for both the Starbucks logo and the tree leaves that are close to the camera. In Figure A.8, only Ours+ and DeblurGANv2 [20] deblur the letters written on the wall. However, DeblurGANv2 fails to deblur the grass. In Figure A.9, Ours+ can recover the poster on the background with the highest quality.

(a) Blurred input (b) Kupyn [20]

(c) Zhang [22] (d) Tao [21]

(e) Park [253] (f) Prohit [254]

(g) Ours+ (h) Sharp GT

Figure A.1: Qualitative comparison using the GoPro dataset [24].

(a) Blurred input (b) Kupyn [20]

(c) Zhang [22] (d) Tao [21]

(e) Park [253] (f) Prohit [254]

(g) Ours+ (h) Sharp GT

Figure A.2: Qualitative comparison using the GoPro dataset [24].

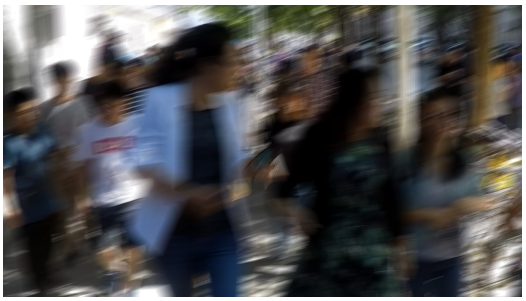
(a) Blurred input (b) Kupyn [20]

(c) Zhang [22] (d) Tao [21]

(e) Park [253] (f) Prohit [254]

(g) Ours+ (h) Sharp GT

Figure A.3: Qualitative comparison using the GoPro dataset [24].



(a) Blurred input

(b) Kupyn [20]

(c) Zhang [22]

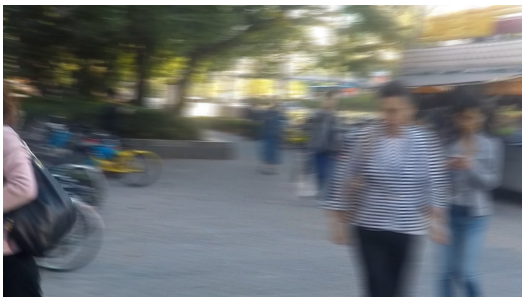
(d) Tao [21]

(e) Park [253]

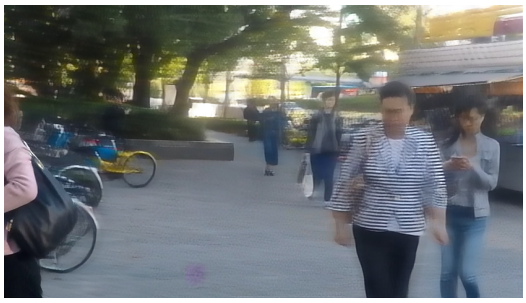
(f) Ours+

(g) Sharp GT

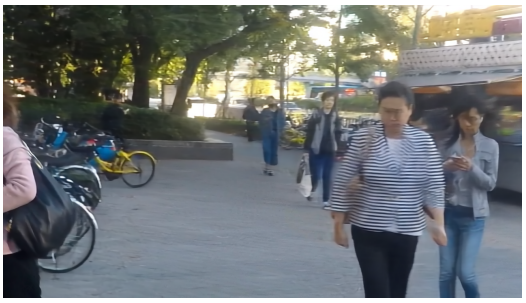
Figure A.4: Qualitative comparison using the HIDE dataset [246].



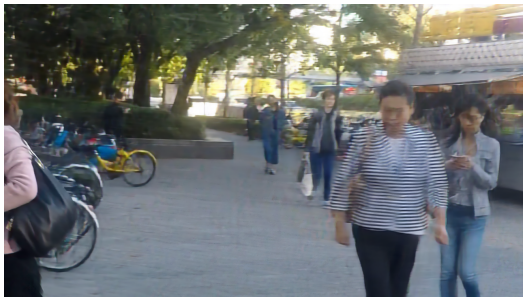
(a) Blurred input



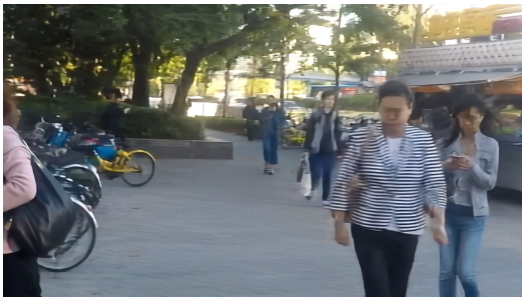
(b) Kupyn [20]



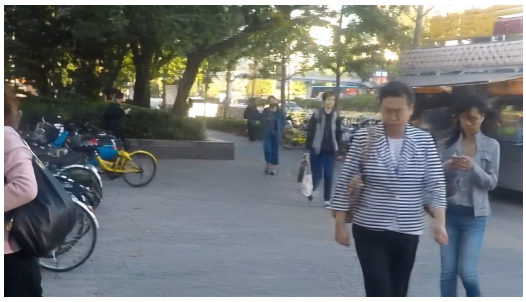
(c) Zhang [22]



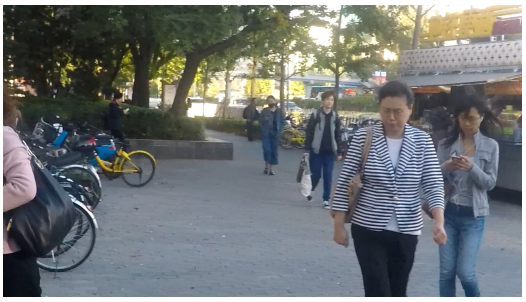
(d) Tao [21]



(e) Park [253]

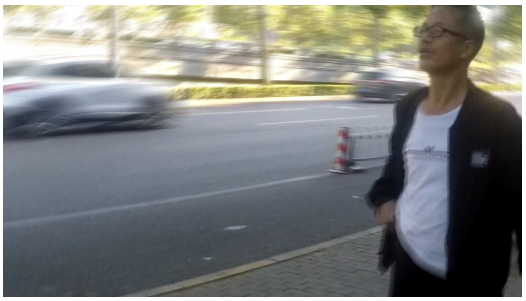


(f) Ours+



(g) Sharp GT

Figure A.5: Qualitative comparison using the HIDE dataset [246].



(a) Blurred input

(b) Kupyn [20]

(c) Zhang [22]

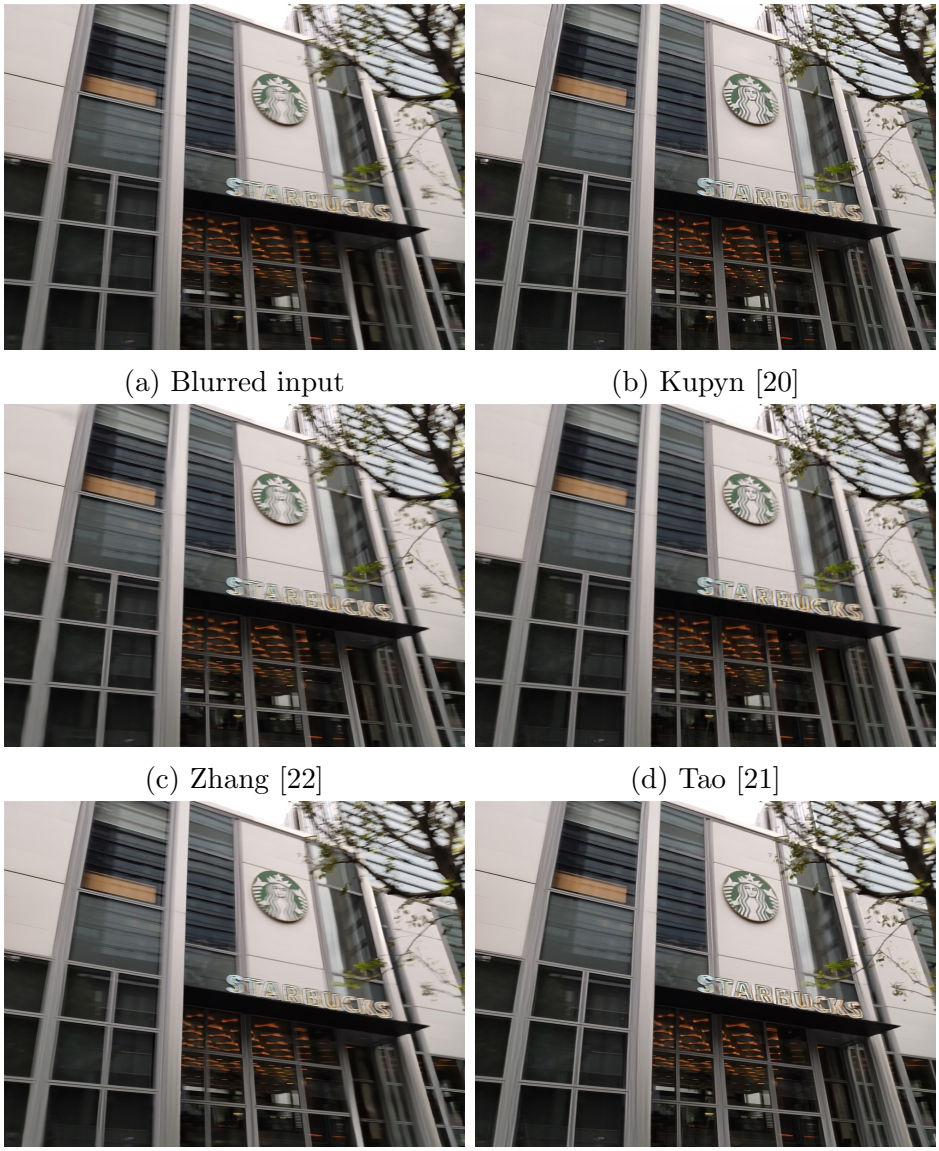
(d) Tao [21]

(e) Park [253]

(f) Ours+

(g) Sharp GT

Figure A.6: Qualitative comparison using the HIDE dataset [246].



(a) Blurred input (b) Kupyn [20]

(c) Zhang [22] (d) Tao [21]

(e) Park [253] (f) Ours+

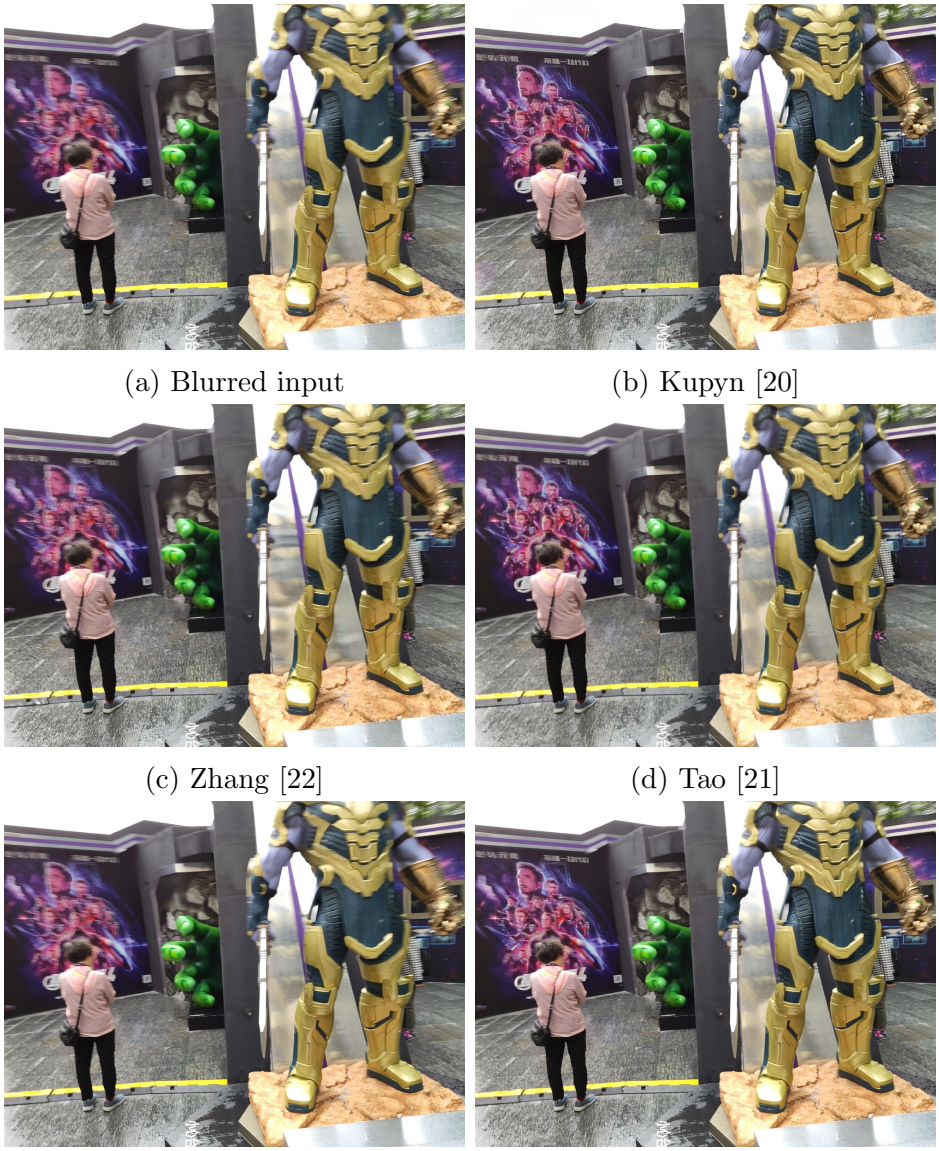
Figure A.7: Qualitative comparison using the RWBI dataset [247].

(a) Blurred input (b) Kupyn [20]

(c) Zhang [22] (d) Tao [21]

(e) Park [253] (f) Ours+

Figure A.8: Qualitative comparison using the RWBI dataset [247].



(a) Blurred input (b) Kupyn [20]

(c) Zhang [22] (d) Tao [21]

(e) Park [253] (f) Ours+

Figure A.9: Qualitative comparison using the RWBI dataset [247].

Appendix B: Blind Image Deconvolution Using Variational Deep Image Prior

B.1 Detailed Derivation

In this section, we show more detailed derivation of equations in Chapter 4.1.

B.1.1 Equation 4.8

$$
\begin{aligned}
& D_{KL}(Q(I_s,k,\xi_x,\xi_y)||P(I_s,k,\xi_x,\xi_y|I_b)) \\
&= \int Q(I_s,k,\xi_x,\xi_y)\ln\frac{Q(I_s,k,\xi_x,\xi_y)}{P(I_s,k,\xi_x,\xi_y|I_b)}dI_s dk d\xi_x d\xi_y \\
&= \int Q(I_s,k,\xi_x,\xi_y)\ln\frac{Q(I_s,k,\xi_x,\xi_y)P(I_b)}{P(I_s,k,\xi_x,\xi_y,I_b)}dI_s dk d\xi_x d\xi_y \\
&= \int Q(I_s,k,\xi_x,\xi_y)\ln P(I_b)dI_s dk d\xi_x d\xi_y - \int Q(I_s,k,\xi_x,\xi_y)\ln\frac{P(I_s,k,\xi_x,\xi_y,I_b)}{Q(I_s,k,\xi_x,\xi_y)}dI_s dk d\xi_x d\xi_y \\
&= \ln P(I_b)\int Q(I_s,k,\xi_x,\xi_y)dI_s dk d\xi_x d\xi_y - \int Q(I_s,k,\xi_x,\xi_y)\ln\frac{P(I_s,k,\xi_x,\xi_y,I_b)}{Q(I_s,k,\xi_x,\xi_y)}dI_s dk d\xi_x d\xi_y \\
&= \ln P(I_b) - \int Q(I_s,k,\xi_x,\xi_y)\ln\frac{P(I_s,k,\xi_x,\xi_y,I_b)}{Q(I_s,k,\xi_x,\xi_y)}dI_s dk d\xi_x d\xi_y \\
&= \ln P(I_b) - L(I_s,k,\xi_x,\xi_y,I_b). \qquad \text{(B.1)}
\end{aligned}
$$

B.1.2 Equation 4.9

$$
\begin{aligned}
& L(I_s, k, \xi_x, \xi_y, I_b) \\
&= \int Q(I_s, k, \xi_x, \xi_y) \ln \frac{P(I_s, k, \xi_x, \xi_y, I_b)}{Q(I_s, k, \xi_x, \xi_y)} dI_s dk d\xi_x d\xi_y \\
&= \int Q(I_s, k, \xi_x, \xi_y) \ln \frac{P(I_s, k, \xi_x, \xi_y)}{Q(I_s, k, \xi_x, \xi_y)} dI_s dk d\xi_x d\xi_y \\
&\quad + \int Q(I_s, k, \xi_x, \xi_y) \ln P(I_b | I_s, k, \xi_x, \xi_y) dI_s dk d\xi_x d\xi_y \\
&= \int Q(I_s, k, \xi_x, \xi_y) \ln \frac{P(I_s, k, \xi_x, \xi_y)}{Q(I_s, k, \xi_x, \xi_y)} dI_s dk d\xi_x d\xi_y + E_{Q(I_s,k,\xi_x,\xi_y)} \left[\ln P(I_b | I_s, k, \xi_x, \xi_y) \right] \\
&= \int Q(I_s, k, \xi_x, \xi_y) \ln \frac{P(I_s, k, \xi_x, \xi_y)}{Q(I_s, k, \xi_x, \xi_y)} dI_s dk d\xi_x d\xi_y + E_{Q(I_s,k)} \left[\ln P(I_b | I_s, k) \right] \\
&= \int Q(I_s) Q(k) Q(\xi_x, \xi_y) \cdot \ln \frac{P(k) P(I_s | \xi_x, \xi_y) P(\xi_x, \xi_y)}{Q(I_s) Q(k) Q(\xi_x, \xi_y)} dI_s dk d\xi_x d\xi_y + E_{Q(I_s,k)} \left[\ln P(I_b | I_s, k) \right] \\
&= \int Q(I_s) Q(k) Q(\xi_x, \xi_y) \left(\ln \frac{P(k)}{Q(k)} - \ln Q(I_s) + \ln P(I_s | \xi_x, \xi_y) + \ln \frac{P(\xi_x, \xi_y)}{Q(\xi_x, \xi_y)} \right) dI_s dk d\xi_x d\xi_y \\
&\quad + E_{Q(I_s,k)} \left[\ln P(I_b | I_s, k) \right] \\
&= \int Q(k) \ln \frac{P(k)}{Q(k)} dk - \int Q(I_s) \ln Q(I_s) dI_s + \int Q(I_s) Q(\xi_x, \xi_y) \ln P(I_s | \xi_x, \xi_y) dI_s d\xi_x d\xi_y \\
&\quad + \int Q(\xi_x, \xi_y) \ln \frac{P(\xi_x, \xi_y)}{Q(\xi_x, \xi_y)} d\xi_x d\xi_y + E_{Q(I_s,k)} \left[\ln P(I_b | I_s, k) \right]. \qquad \text{(B.2)}
\end{aligned}
$$

Since ξ_x and ξ_y are deterministic given I_s following Eqn. 7, we can simply set $E_{Q(I_s,k,\xi_x,\xi_y)} \left[\ln P(I_b | I_s, k, \xi_x, \xi_y) \right] = E_{Q(I_s,k)} \left[\ln P(I_b | I_s, k) \right]$.

B.1.3 Equation 4.10

$$
\begin{aligned}
& \int Q(k) \ln \frac{P(k)}{Q(k)} dk \\
&= \int \mathcal{N}(E(k), S^2(k)) \ln \frac{\mathcal{N}(0,\, I)}{\mathcal{N}(E(k), S^2(k))} dk \\
&= \int \mathcal{N}(E(k), S^2(k)) \ln \mathcal{N}(0,\, I) dk - \int \mathcal{N}(E(k), S^2(k)) \ln \mathcal{N}(E(k), S^2(k)) dk \\
&= -\frac{1}{2} \sum_{i=1}^{I} \sum_{j=1}^{J} (\ln 2\pi + E^2(k(i,j)) + S^2(k(i,j))) + \frac{1}{2} \sum_{i=1}^{I} \sum_{j=1}^{J} (\ln 2\pi + 1 + 2 \ln S(k(i,j))) \\
&= \frac{1}{2} \sum_{i=1}^{I} \sum_{j=1}^{J} (1 + 2 \ln S(k(i,j)) - E^2(k(i,j)) - S^2(k(i,j))), \qquad \text{(B.3)}
\end{aligned}
$$

$\mathcal{N}(E(), S^2())$ denotes the Gaussian distribution with mean $E()$ and variance $S^2()$, $S()$ and $E()$ denote the standard deviation and the expectation, respectively, (i, j) is the pixel index of blur kernel.

$$
\begin{aligned}
&-\int Q(I_s)\ln Q(I_s)dI_s \\
=&-\int \mathcal{N}(E(I_s),S^2(I_s))\ln \mathcal{N}(E(I_s),S^2(I_s))dI_s \\
=&\frac{1}{2}\sum_{m=1}^{M}\sum_{n=1}^{N}(\ln 2\pi+1+2\ln S(I_s(m,n))),
\end{aligned}
\tag{B.4}
$$

$\mathcal{N}(E(),S^2())$ denotes the Gaussian distribution with mean $E()$ and variance $S^2()$, $S()$ and $E()$ denote the standard deviation and the expectation, respectively, (m,n) is the pixel index of I_s and ξ.

$$
\begin{aligned}
&\int Q(I_s)Q(\xi_x,\xi_y)\ln P(I_s|\xi_x,\xi_y)dI_s d\xi_x d\xi_y \\
=&\int Q(I_s)Q(\xi_x,\xi_y)\left(\ln W-\frac{\xi_x(F_x(I_s))^2+\xi_y(F_y(I_s))^2}{4}+\frac{\rho^*(\frac{1}{2}\xi_x)+\rho^*(\frac{1}{2}\xi_y)}{2}\right)dI_s d\xi_x d\xi_y \\
=&\ln W\int Q(I_s)Q(\xi_x,\xi_y)dI_s d\xi_x d\xi_y-\int Q(I_s)Q(\xi_x,\xi_y)\left(\frac{\xi_x(F_x(I_s))^2+\xi_y(F_y(I_s))^2}{4}\right)dI_s d\xi_x d\xi_y \\
&+\int Q(I_s)Q(\xi_x,\xi_y)\left(\frac{\rho^*(\frac{1}{2}\xi_x)+\rho^*(\frac{1}{2}\xi_y)}{2}\right)dI_s d\xi_x d\xi_y \\
=&-\int Q(I_s)Q(\xi_x,\xi_y)\left(\frac{\xi_x(F_x(I_s))^2+\xi_y(F_y(I_s))^2}{4}\right)dI_s d\xi_x d\xi_y \\
&+\int Q(\xi_x,\xi_y)\left(\frac{\rho^*(\frac{1}{2}\xi_x)+\rho^*(\frac{1}{2}\xi_y)}{2}\right)d\xi_x d\xi_y+\ln W.
\end{aligned}
\tag{B.5}
$$

For the sparse image prior, $F_x(I_s)$ and $F_y(I_s)$ calculate the gradients of two directions as in Eqn. 4.13.

Let us first look at the $F_x(I_s)$ related term in Eqn. B.5.

$$
\begin{aligned}
&- \int Q(I_s)Q(\xi_x,\xi_y)\frac{\xi_x(F_x(I_s))^2}{4}dI_s d\xi_x d\xi_y \\
=&- \int Q(I_s)Q(\xi_x)\frac{\xi_x(F_x(I_s))^2}{4}dI_s d\xi_x \\
=&- \int Q(I_s)Q(\xi_x)\frac{\sum_{m=1}^{M}\sum_{n=1}^{N}\xi_x(m,n)(F_x(I_s)(m,n))^2}{4}dI_s d\xi_x \\
=&-\frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N}E((F_x(I_s)(m,n))^2)E(\xi_x(m,n)). \\
=&-\frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N}E((I_s(m,n)-I_s(m-1,n))^2)E(\xi_x(m,n)) \\
=&-\frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N}E(\xi_x(m,n))[E^2(I_s(m,n))+S^2(I_s(m,n))-2E(I_s(m,n))E(I_s(m-1,n)) \\
&+E^2(I_s(m-1,n))+S^2(I_s(m-1,n))] \\
=&-\frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N}[(E(I_s(m,n))-E(I_s(m-1,n)))^2+S^2(I_s(m,n))+S^2(I_s(m-1,n))]E(\xi_x(m,n)).
\end{aligned}
\tag{B.6}
$$

The $F_y(I_s)$ related term can be derived in a similar way.

$$
\begin{aligned}
&- \int Q(I_s)Q(\xi_x,\xi_y)\frac{\xi_y(F_y(I_s))^2}{4}dI_s d\xi_x d\xi_y \\
=&-\frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N}E((F_y(I_s)(m,n))^2)E(\xi_y(m,n)) \\
=&-\frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N}E((I_s(m,n)-I_s(m,n-1))^2)E(\xi_x(m,n)) \\
=&-\frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N}[(E(I_s(m,n))-E(I_s(m,n-1)))^2+S^2(I_s(m,n))+S^2(I_s(m,n-1))]E(\xi_y(m,n)).
\end{aligned}
\tag{B.7}
$$

Combining Eqn. B.3 $\sim$ B.7, we can get the variational lower bound as Eqn. 4.10.

Different from the sparse image prior which is differentiable and continuous, $F_x(I_s)$ and $F_y(I_s)$ (in Eqn. 4.14) are non-differentiable and discrete becasue of the max() and the min(). Thus, we cannot obtain closed-form expressions corresponding to $F_x(I_s)$ and $F_y(I_s)$ as in Eqn. B.6 and Eqn B.7. To solve this problem, we approximate $E((F_x(I_s))^2)$ and $E((F_y(I_s))^2)$ by Monte Carlo estimation using sampling [97].

Let us first look at the $F_x(I_s)$ related term in Eqn. B.5.

$$
\begin{aligned}
&- \int Q(I_s)Q(\xi_x,\xi_y)\frac{\xi_x(F_x(I_s))^2}{4}dI_s d\xi_x d\xi_y \\
=&- \int Q(I_s)Q(\xi_x)\frac{\xi_x(F_x(I_s))^2}{4}dI_s d\xi_x \\
=&- \int Q(I_s)Q(\xi_x)\frac{\sum_{m=1}^{M}\sum_{n=1}^{N}\xi_x(m,n)(F_x(I_s)(m,n))^2}{4}dI_s d\xi_x \\
=&-\frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N}E((F_x(I_s)(m,n))^2)E(\xi_x(m,n)). \\
=&-\frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N}E((\min_{i\in\Omega(m,n)}(\min_{c\in(r,g,b)}(I_s^c(i))))^2)E(\xi_x(m,n)) \\
=&-\frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N}E(\min_{i\in\Omega(m,n)}(\min_{c\in(r,g,b)}(I_s^c(i)^2)))E(\xi_x(m,n)).
\end{aligned} \tag{B.8}
$$

Based on Eqn. B.8, we need to approximate the expectation of $\min_{i\in\Omega(m,n)}(\min_{c\in(r,g,b)}(I_s^c(i)^2))$ by sampling I_s^2. Thus, we need to calculate both $E(I_s^2)$ and $S(I_s^2)$ as follows [345]:

$$E(I_s^2) = E(I_s)^2 + S(I_s)^2, \tag{B.9}$$

$$S(I_s^2) = \sqrt{4E(I_s)^2S(I_s)^2 + 2S(I_s)^4}. \tag{B.10}$$

Then the expectation can be reformulated as

$$
\begin{aligned}
&E(\min_{i\in\Omega(m,n)}(\min_{c\in(r,g,b)}(I_s^c(i)^2))) \approx \frac{1}{A}\sum_{a=1}^{A}\min_{i\in\Omega(m,n)}(\min_{c\in(r,g,b)}(\hat{I_s^2}^a(i)))) \\
&\hat{I_s^2}^a = E(I_s^2) + \epsilon^a \odot S(I_s^2), \epsilon^a \sim \mathcal{N}(0,\, I),
\end{aligned} \tag{B.11}
$$

where A is the number of samples, σ is the noise level, $\odot$ represents the element-wise multiplication, and ϵ^a is a random scalar sampled from a standard Gaussian distribution.

The $F_y(I_s)$ related term in Eqn. B.5 is slightly different.

$$
\begin{aligned}
& -\int Q(I_s)Q(\xi_x,\xi_y)\frac{\xi_y(F_y(I_s))^2}{4}dI_s d\xi_x d\xi_y \\
=& -\frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N} E((F_y(I_s)(m,n))^2)E(\xi_y(m,n)) \\
=& -\frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N} E((1-\max_{i\in\Omega(m,n)}(\max_{c\in(r,g,b)}(I_s^c(i))))^2)E(\xi_y(m,n)) \\
=& -\frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N} E((\min_{i\in\Omega(m,n)}(\min_{c\in(r,g,b)}(1-I_s^c(i))))^2)E(\xi_y(m,n)) \\
=& -\frac{1}{4}\sum_{m=1}^{M}\sum_{n=1}^{N} E(\min_{i\in\Omega(m,n)}(\min_{c\in(r,g,b)}((1-I_s^c(i))^2)))E(\xi_y(m,n)). \qquad \text{(B.12)}
\end{aligned}
$$

Based on Eqn. B.12, we need to approximate the expectation of $\min_{i\in\Omega(m,n)}(\min_{c\in(r,g,b)}((1-I_s^c(i))^2))$ by sampling $(1-I_s)^2$. Similarly, we need to calculate both $E((1-I_s)^2)$ and $S((1-I_s)^2)$ as follows:

$$E((1-I_s)^2) = (1-E(I_s))^2 + S(I_s)^2, \qquad \text{(B.13)}$$

$$S((1-I_s)^2) = \sqrt{4S(I_s)^2 + 4E(I_s)^2S(I_s)^2 + 2S(I_s)^4}. \qquad \text{(B.14)}$$

The form of expectation is the same as that shown in Eqn. B.11 except that I_s is replaced by $1-I_s$.

Appendix C: Text-Guided Texture Generation for 3D Objects

C.1 Algorithm Details

To better illustrate the working flow of our proposed method, we present the detailed algorithm in Alg. 4

C.2 Derivation of Eq. 5.14

As discussed in Eq. 5.14 of Chapter 5.1.3, we apply the classifier-free guidance (CFG) on noise estimation with two conditions: the textual prompt c and the intermediate texture map $\hat{U}^i_{0,t}$. The original text guided diffusion model targets at learning $P(x_t|c)$ where x_t denotes the noisy latent feature at time step t. Now we extend the target of the original diffusion model to $P(x^i_t|c, \hat{U}^N_{0,t})$, which has an additional condition $\hat{U}^N_{0,t}$ to constrain the generated x^i_t to be view-consistent. We assume $P(c|x^i_t, \hat{U}^N_{0,t}) = P(c|x^i_t)$. Following Bayes' theorem, $P(x^i_t|c, \hat{U}^N_{0,t})$ can be reformulated as

$$P(x^i_t|c, \hat{U}^N_{0,t}) = \frac{P(x^i_t)P(c|x^i_t)P(\hat{U}^N_{0,t}|x^i_t)}{P(c, \hat{U}^N_{0,t})}. \tag{C.1}$$

By taking logarithm on both sides of the above equation, we get

$$\log(P(x^i_t|c, \hat{U}^N_{0,t})) = \log(P(x^i_t)) + \log(P(c|x^i_t)) + \log(P(\hat{U}^N_{0,t}|x^i_t)) - log(P(c, \hat{U}^N_{0,t})). \tag{C.2}$$

As mentioned in [112], estimating $\epsilon_m(x^i_t)$ is related to predicting the score function $s_m(x^i_t)$ of the approximate marginal distribution $P(x^i_t|c, \hat{U}^N_{0,t})$, which can be formu-

Algorithm 4: Progressive Texture Sampling

Input: A 3D Mesh
A textual prompt c
A set of viewpoints v_i, $i \in \{1, \ldots, N\}$
Number of denoising step T
VAE encoder $\mathcal{E}$ and decoder $\mathcal{D}$
Depth conditioned ControlNet $Unet_\theta$

Output: Generated texture map $\hat{U}_{0,1}^N$

Randomly initialize $x_T^i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $i \in \{1, \ldots, N\}$
for $t = T, ..., 1$ **do**
  View Sampling&Aggregation (VSA):
  for $i = 1, \ldots, N$ **do**
    Substitute the Key and Value features for viewpoint i with those from reference view to calculate $\epsilon_\theta(x_t^i)$ by Eq. 5.8 and Eq. 5.9
    Obtain the $\hat{x}_0^i(x_t^i)$ with x_t^i and $\epsilon_\theta(x_t^i)$ by Eq. 5.1
    Decode the $\hat{x}_0^i(x_t^i)$ to obtain I_t^i in RGB space by Eq. 5.4
    Inverse render the I_t^i to obtain the partial texture map $\hat{U}_{0,t}^i$
    if $i < N$ **then**
      Render and encode $\hat{U}_{0,t}^i$ to obtain $G_{0,t}^{i+1}$ by Eq. 5.5
      Update x_t^{i+1} with $G_{0,t}^{i+1}$ and view aggregation mask $\mathcal{M}^{i+1}$ by Eq. 5.7
    end
  end
  Text&Texture Guided Resampling ($\mathbf{T^2GR}$):
  for $i = 1, \ldots, N$ **do**
    Calculate the $\hat{\epsilon}_{tex}(x_t^i)$ by Eq. 5.10
    Obtain the texture-conditioned noise estimation $\epsilon_{tex}(x_t^i|\hat{U}_{0,t}^N)$ by Eq. 5.13
    Combine the texture-conditioned noise estimation $\epsilon_{tex}(x_t^i|\hat{U}_{0,t}^N)$, text-conditioned noise estimation $\epsilon_\theta(x_t^i|c)$, and unconditioned noise estimation $\epsilon_\theta(x_t^i|\varnothing)$ to calculate the final noise estimation $\epsilon_m(x_t^i)$ by Eq. 5.14
    if $t > 1$ **then**
      Substitute $\epsilon_\theta(x_t^i)$ with $\epsilon_m(x_t^i)$ in Eq. 5.1 and Eq. 5.2 to calculate the x_{t-1}^i for the next denoising step
    end
  end
end

lated as:

$$s_m(x_t^i) = \nabla_{x_t^i} \log(P(x_t^i|c, \hat{U}_{0,t}^N)), \tag{C.3}$$

$$\epsilon_m(x_t^i) = -\sigma_t s_m(x_t^i), \tag{C.4}$$

where σ_t is the standard deviation of the latent noise parameterized by denoising step t. The score function $\nabla_{x_t^i} \log(P(x_t^i|c, \hat{U}_{0,t}^N))$ can be further derived from Eq. C.2 as:

$$\nabla_{x_t^i} \log(P(x_t^i|c, \hat{U}_{0,t}^N)) = \nabla_{x_t^i} \log(P(x_t^i)) + \nabla_{x_t^i} \log(P(c|x_t^i)) + \nabla_{x_t^i} \log(P(\hat{U}_{0,t}^N|x_t^i)), \tag{C.5}$$

with

$$\nabla_{x_t^i} \log(P(c|x_t^i)) = \nabla_{x_t^i} \log(P(x_t^i|c)) - \nabla_{x_t^i} \log(P(x_t^i)), \tag{C.6}$$

$$\nabla_{x_t^i} \log(P(\hat{U}_{0,t}^N|x_t^i)) = \nabla_{x_t^i} \log(P(x_t^i|\hat{U}_{0,t}^N)) - \nabla_{x_t^i} \log(P(x_t^i)), \tag{C.7}$$

which correspond to the terms in our multi-conditioned CFG in Eq. 5.14 as:

$$\epsilon_\theta(x_t^i|\varnothing) = -\sigma_t \nabla_{x_t^i} \log(P(x_t^i)), \tag{C.8}$$

$$\epsilon_\theta(x_t^i|c) - \epsilon_\theta(x_t^i|\varnothing) = -\sigma_t(\nabla_{x_t^i} \log(P(x_t^i|c)) - \nabla_{x_t^i} \log(P(x_t^i))), \tag{C.9}$$

$$\epsilon_{tex}(x_t^i|\hat{U}_{0,t}^N) - \epsilon_\theta(x_t^i|\varnothing) = -\sigma_t(\nabla_{x_t^i} \log(P(x_t^i|\hat{U}_{0,t}^N)) - \nabla_{x_t^i} \log(P(x_t^i))). \tag{C.10}$$

Following CFG [274], we apply two guidance scales ω_1 and ω_2 on two guidance terms. Finally, we have the multi-conditioned CFG as:

$$\epsilon_m(x_t^i) = \epsilon_\theta(x_t^i|\varnothing) + \omega_1(\epsilon_\theta(x_t^i|c) - \epsilon_\theta(x_t^i|\varnothing)) + \omega_2(\epsilon_{tex}(x_t^i|\hat{U}_{0,t}^N) - \epsilon_\theta(x_t^i|\varnothing)). \tag{C.11}$$

C.3 Additional Experiments

C.3.1 Inference Time

In Tab. C.1, we compare the inference time of our proposed method with that of baseline methods on a single NVIDIA Tesla V100 GPU with 32GB memory.

C.3.2 More Qualitative Evaluations

More qualitative evaluations are shown in Fig. C.1, Fig. C.2, and Fig. C.3.

Methods	Inference Time (minutes) ↓
TEXTure	3.94
Text2Tex	20.64
Fantasia3D	109.67
ProlificDreamer	483.92
Ours	46.83

Table C.1: Inference time of compared methods using images with resolution of 512×512 on a single NVIDIA Tesla V100 GPU with 32GB memory.

C.4 User Study Details

We develop a WIX-based web application for the user study. As shown in Fig. C.4, for each video pair, participants are required to choose the video that best illustrates the given textual prompt with the highest quality. They should then click the rounded check-box below the selected video and proceed to the next video pair. Finally, we determine the user preferences by counting all user selections.

C.5 Data Description

We present the details of our collected data in Tab. C.2, Tab. C.3, and Tab. C.4 with corresponding textual prompts.

Figure C.1: More texture generation results of our proposed method.

Figure C.2: Visual comparison of our proposed method against TEXTure [120] and Text2Tex [121].

Figure C.3: Visual comparison of our proposed method against Fantasia3D [115] and ProlificDreamer [116].

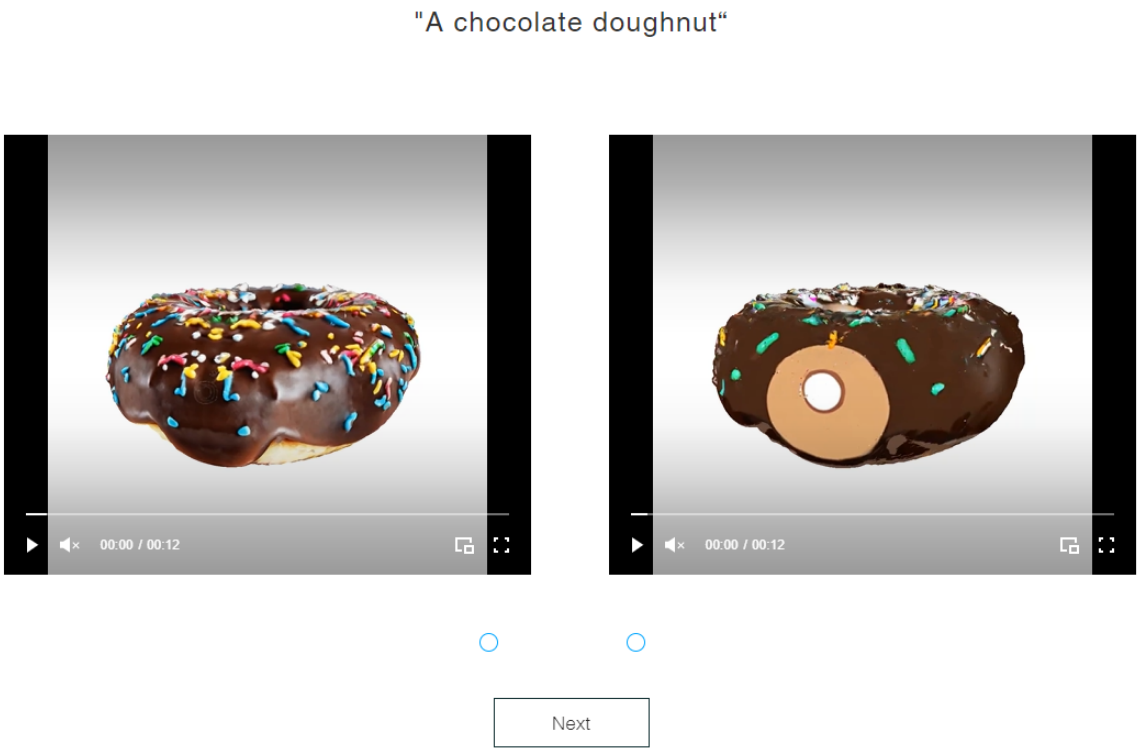


Figure C.4: Screenshot of the user study web application

Object	Source	Description	Textual Prompts
eb219212147f4d84b88f8e103af8ea10	Objaverse	frog	"A robotic frog" "A green frog"
a8813ea1e0ce47ab97a416637a7520d7	Objaverse	helmet	"A Mandalorian helmet in silver" "A black helmet"
e0417d1e05984727a50f9ab1451d162d	Objaverse	lantern	"A stone lantern" "A medieval lantern"
9fa2da2c42234b58896e8d23393cac24	Objaverse	backpack	"A backpack in ironman style" "A backpack in spiderman style" "A 3D backpack"
a51751c9989940e592eb61be41ee35cc	Objaverse	baby owl	"A baby owl with fluffy wings" "A toy owl"
f73e2e1c8ad241ff859aca7e032ec262	Objaverse	lion	"A cute 3D cartoon lion with brown hair" "A marble lion"
91c5283b27c74583900d5e26e2fcd086	Objaverse	mug	"A wooden mug surrounded by silver rings" "A mug with cloud"
b6db59bd7f10424eae54c71d19663a65	Objaverse	car	"A next gen nascar in red" "A next gen nascar"
a2832b845e4e4edd9d439342cf4fd590	Objaverse	wolf	"Statue of a wolf" "A white wolf"
b19ef2650b4347348710eb6364ca90bd	Objaverse	penguin	"A black penguin" "A penguin covered by a blue sweater"
bd384d46514548cf8c4202f1ae6ea551	Objaverse	refrigerator	"A wooden refrigerator" "A high tech refrigerator"
f1aa479977a74a608d362679ed5ca721	Objaverse	piano	"A medieval piano" "A piano with flowers"
4c4690ba918f477b829990dd2e960c21	Objaverse	lion	"A golden lion" "A cyber punk lion"
f87caf6ac5a445ccad1a97653688e16e	Objaverse	dresser	"A wooden dresser" "A marble dresser"
f15298421b3d4e0fab4c43863a7e72fd	Objaverse	shark	"A deep ocean shark" "A dark blue shark"
d4c560493a0846c5943f3aeea58acb72	Objaverse	soccer ball	"A soccer ball in black and white" "A stone soccer ball"
c6509a8fe1f44a5eac8aebe12be2699e	Objaverse	tiger	"A tiger walking on the grass" "A plastic toy tiger"
bff537fb09b641c59b2ad123da0ca3dc	Objaverse	turtle	"A metal turtle with red eyes" "A sea turtle"
d726514a97f74f168b104fd6ba538331	Objaverse	vase	"An ancient vase" "A painted vase"
01ab0842feb1448bb18e8c7b85326d11	Objaverse	pottery	"An antique pottery" "A pottery with flowers"
f2d31eb0ddac4d21944df7dcc4af6d28	Objaverse	vending machine	"A coca cola vending machine" "A silver vending machine"

Table C.2: Description of 3D Meshes in our collected data.

Object	Source	Description	Textual Prompts
e1f96691aaf648b885d927f5c3f5be61	Objaverse	apple	"A red apple" "An oil painted apple"
8a60954eccad433e987bbcafc7657140	Objaverse	armor	"A medieval armor" "A Japanese armor"
f98c5ee54c4a48f8b5eafd35a81dde4d	Objaverse	owl	"A metal owl with glowing eyes" "A wooden owl"
fadefc1eee3246a189f6b79c7c671343	Objaverse	lion	"A lion looking forward" "Statue of a lion"
9a0c52d350634e419aaf0eea1e67d9da	Objaverse	knight	"A golden knight" "A silver knight"
fa2c41a7a6c84fcb871a24016fa9a932	Objaverse	doughnut	"A chocolate doughnut" "An icecream doughnut"
f05b0c2f9bcf41cea188a4b4c848068a	Objaverse	fireplug	"A fireplug, red and yellow" "A fireplug with yellow top"
0db114d7753344d6825aa4f21ec56db9	Objaverse	crate	"A wooden crate" "A bronze crate"
72826cd5c17a42798a8e8e36c05c5035	Objaverse	clock	"A medieval clock" "A electric clock"
ac5df73de2c54239833643423a152592	Objaverse	dresser	"A wooden dresser" "A marble dresser"
90009fa6fa0b4d4bb1a1203431954097	Objaverse	keg	"A metal keg in silver" "A wooden keg"
b26a53419075442ca284cdf1d5541765	Objaverse	monitor	"A mac monitor" "An ironman monitor"
f75caead1dc1474195eb32a7f4c71117	Objaverse	control	"A game controller with black buttons on the top" "A PS5 controller"
edbeb81ef32645cea8bef89338f7e213	Objaverse	telephone	"A telephone with golden dials" "A classic telephone"
fc9cc06615084298b4c0c0a02244f356	Objaverse	piano	"A medieval piano" "A piano with flowers"
7adc9c74b75e4860b0a51c850bde9957	Objaverse	dress	"A princess dress" "A dress with spider patterns"
2fc0fc6ebe564a249c4617e6b3e6da93	Objaverse	fireplace	"A brick fireplace" "A stone fireplace"
14b8ae60eae240ff8bf1abdf9af5e49c	Objaverse	refrigerator	"A wooden refrigerator" "A high tech refrigerator"
62897c52e967469c85df9c6abdd09d16	Objaverse	doll	"A doll with yellow hairs" "A spiderman doll"
6f5480698a7a43c7a8c0a8b1e295e4a0	Objaverse	pumpkin	"A pumpkin with red eyes" "A Halloween pumpkin"

Table C.3: Description of 3D Meshes in our collected data.

Object	Source	Description	Textual Prompts
Napoleon ler	ThreeDScans	statue	"A high quality color photo of Tom Cruise" "A high quality color photo of Benedict Cumberbatch" "A high quality color photo of Robert Downey Jr."
Plastic Dragon	ThreeDScans	statue	"Cartoon dragon, red and green" "A 3D dragon"
Francois	ThreeDScans	statue	"Spiderman with white hairs" "A boy in suits"
Provost	ThreeDScans	statue	"Portrait of Provost, oil paint" "A statue of Provost"

Table C.4: Description of 3D Meshes in our collected data.

Appendix D: Spectral Reflectance Recovery from RGB Images

This appendix provides 1) the detailed derivation of Eqn. 4.14 in Sec. D.1; 2) more qualitative evaluations with competing approaches in Fig. D.1∼D.3 of Sec. D.2; 3) recovered reflectance curves and correlation coefficients (corr) between the recovered reflectance and the ground truth in Fig. D.4∼D.7 of Sec. D.2; 3) Feasibility analysis of data capture in real world in Sec. D.3.

D.1 Detailed derivation of Eqn. 4.14

$$
\begin{aligned}
\hat{\mathbf{R}} =&(\hat{\omega}+\Delta\omega)(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T\cdot((\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})\cdot(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T)^{-1}\cdot(\boldsymbol{\mathcal{I}}-\Delta\boldsymbol{\mathcal{I}})+\hat{\mathbf{R}}^{\perp}\\
=&\hat{\omega}(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T\cdot((\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})\cdot(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T)^{-1}\cdot\boldsymbol{\mathcal{I}}\\
&-\hat{\omega}(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T\cdot((\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})\cdot(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T)^{-1}\cdot\Delta\boldsymbol{\mathcal{I}}\\
&+\Delta\omega(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T\cdot((\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})\cdot(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T)^{-1}\cdot(\boldsymbol{\mathcal{I}}-\Delta\boldsymbol{\mathcal{I}})+\hat{\mathbf{R}}^{\perp}\\
=&\hat{\omega}\hat{\boldsymbol{\mathcal{H}}}^T\cdot((\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})\cdot(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T)^{-1}\cdot\boldsymbol{\mathcal{I}}\\
&+\hat{\omega}\Delta\boldsymbol{\mathcal{H}}^T\cdot((\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})\cdot(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T)^{-1}\cdot\boldsymbol{\mathcal{I}}\\
&-\hat{\omega}(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T\cdot((\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})\cdot(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T)^{-1}\cdot\Delta\boldsymbol{\mathcal{I}}\\
&+\Delta\omega(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T\cdot((\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})\cdot(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T)^{-1}\cdot(\boldsymbol{\mathcal{I}}-\Delta\boldsymbol{\mathcal{I}})+\hat{\mathbf{R}}^{\perp}\\
=&\hat{\omega}\hat{\boldsymbol{\mathcal{H}}}^T\cdot(\hat{\boldsymbol{\mathcal{H}}}\cdot\hat{\boldsymbol{\mathcal{H}}}^T+\hat{\boldsymbol{\mathcal{H}}}\cdot\Delta\boldsymbol{\mathcal{H}}^T+\Delta\boldsymbol{\mathcal{H}}\cdot\hat{\boldsymbol{\mathcal{H}}}^T+\Delta\boldsymbol{\mathcal{H}}\cdot\Delta\boldsymbol{\mathcal{H}}^T)^{-1}\cdot\boldsymbol{\mathcal{I}}\\
&+\hat{\omega}\Delta\boldsymbol{\mathcal{H}}^T\cdot((\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})\cdot(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T)^{-1}\cdot\boldsymbol{\mathcal{I}}\\
&-\hat{\omega}(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T\cdot((\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})\cdot(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T)^{-1}\cdot\Delta\boldsymbol{\mathcal{I}}\\
&+\Delta\omega(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T\cdot((\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})\cdot(\hat{\boldsymbol{\mathcal{H}}}+\Delta\boldsymbol{\mathcal{H}})^T)^{-1}\cdot(\boldsymbol{\mathcal{I}}-\Delta\boldsymbol{\mathcal{I}})+\hat{\mathbf{R}}^{\perp}. \qquad \text{(D.1)}
\end{aligned}
$$

Define the singular value decomposition of $\hat{\boldsymbol{\mathcal{H}}}\cdot\Delta\boldsymbol{\mathcal{H}}^T+\Delta\boldsymbol{\mathcal{H}}\cdot\hat{\boldsymbol{\mathcal{H}}}^T+\Delta\boldsymbol{\mathcal{H}}\cdot\Delta\boldsymbol{\mathcal{H}}^T$ as

$$
\boldsymbol{U}\cdot\boldsymbol{\Sigma}\cdot\boldsymbol{V}^T=SVD(\hat{\boldsymbol{\mathcal{H}}}\cdot\Delta\boldsymbol{\mathcal{H}}^T+\Delta\boldsymbol{\mathcal{H}}\cdot\hat{\boldsymbol{\mathcal{H}}}^T+\Delta\boldsymbol{\mathcal{H}}\cdot\Delta\boldsymbol{\mathcal{H}}^T). \qquad \text{(D.2)}
$$

Following the derivation of Henderson and Searle [346], Eqn. D.1 can be reformulated as

$$
\begin{aligned}
\hat{\mathbf{R}} =& \hat{\omega}\hat{\boldsymbol{\mathcal{H}}}^T \cdot ((\hat{\boldsymbol{\mathcal{H}}} \cdot \hat{\boldsymbol{\mathcal{H}}}^T)^{-1} - (\hat{\boldsymbol{\mathcal{H}}} \cdot \hat{\boldsymbol{\mathcal{H}}}^T)^{-1} \cdot \boldsymbol{U} \cdot (\boldsymbol{I} \\
&+ \boldsymbol{\Sigma} \cdot \boldsymbol{V}^T \cdot (\hat{\boldsymbol{\mathcal{H}}} \cdot \hat{\boldsymbol{\mathcal{H}}}^T)^{-1} \cdot \boldsymbol{U})^{-1} \cdot \boldsymbol{\Sigma} \cdot \boldsymbol{V}^T \cdot (\hat{\boldsymbol{\mathcal{H}}} \cdot \hat{\boldsymbol{\mathcal{H}}}^T)^{-1}) \cdot \boldsymbol{\mathcal{I}} \\
&+ \hat{\omega}\Delta\boldsymbol{\mathcal{H}}^T \cdot ((\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}}) \cdot (\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}})^T)^{-1} \cdot \boldsymbol{\mathcal{I}} \\
&- \hat{\omega}(\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}})^T \cdot ((\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}}) \cdot (\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}})^T)^{-1} \cdot \Delta\boldsymbol{\mathcal{I}} \\
&+ \Delta\omega(\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}})^T \cdot ((\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}}) \cdot (\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}})^T)^{-1} \cdot (\boldsymbol{\mathcal{I}} - \Delta\boldsymbol{\mathcal{I}}) + \hat{\mathbf{R}}^{\perp} \\
=& \hat{\omega}\hat{\boldsymbol{\mathcal{H}}}^T \cdot (\hat{\boldsymbol{\mathcal{H}}} \cdot \hat{\boldsymbol{\mathcal{H}}}^T)^{-1} \cdot \boldsymbol{\mathcal{I}} \\
&- \hat{\omega}\hat{\boldsymbol{\mathcal{H}}}^T \cdot ((\hat{\boldsymbol{\mathcal{H}}} \cdot \hat{\boldsymbol{\mathcal{H}}}^T)^{-1} \cdot \boldsymbol{U} \cdot (\boldsymbol{I} + \boldsymbol{\Sigma} \cdot \boldsymbol{V}^T \cdot (\hat{\boldsymbol{\mathcal{H}}} \cdot \hat{\boldsymbol{\mathcal{H}}}^T)^{-1} \cdot \boldsymbol{U})^{-1} \cdot \boldsymbol{\Sigma} \cdot \boldsymbol{V}^T \cdot (\hat{\boldsymbol{\mathcal{H}}} \cdot \hat{\boldsymbol{\mathcal{H}}}^T)^{-1}) \cdot \boldsymbol{\mathcal{I}} \\
&+ \hat{\omega}\Delta\boldsymbol{\mathcal{H}}^T \cdot ((\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}}) \cdot (\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}})^T)^{-1} \cdot \boldsymbol{\mathcal{I}} \\
&- \hat{\omega}(\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}})^T \cdot ((\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}}) \cdot (\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}})^T)^{-1} \cdot \Delta\boldsymbol{\mathcal{I}} \\
&+ \Delta\omega(\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}})^T \cdot ((\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}}) \cdot (\hat{\boldsymbol{\mathcal{H}}} + \Delta\boldsymbol{\mathcal{H}})^T)^{-1} \cdot (\boldsymbol{\mathcal{I}} - \Delta\boldsymbol{\mathcal{I}}) + \hat{\mathbf{R}}^{\perp} \\
=& \hat{\omega} \underbrace{\hat{\boldsymbol{\mathcal{H}}}^T \cdot (\hat{\boldsymbol{\mathcal{H}}} \cdot \hat{\boldsymbol{\mathcal{H}}}^T)^{-1} \cdot \boldsymbol{\mathcal{I}}}_{\hat{\mathbf{R}}_{\hat{\boldsymbol{\mathcal{H}}}}} + \Delta\hat{\mathbf{R}},
\end{aligned}
\tag{D.3}
$$

where $\mathcal{I}$ represents the identity matrix.

D.2 More Evaluation Results

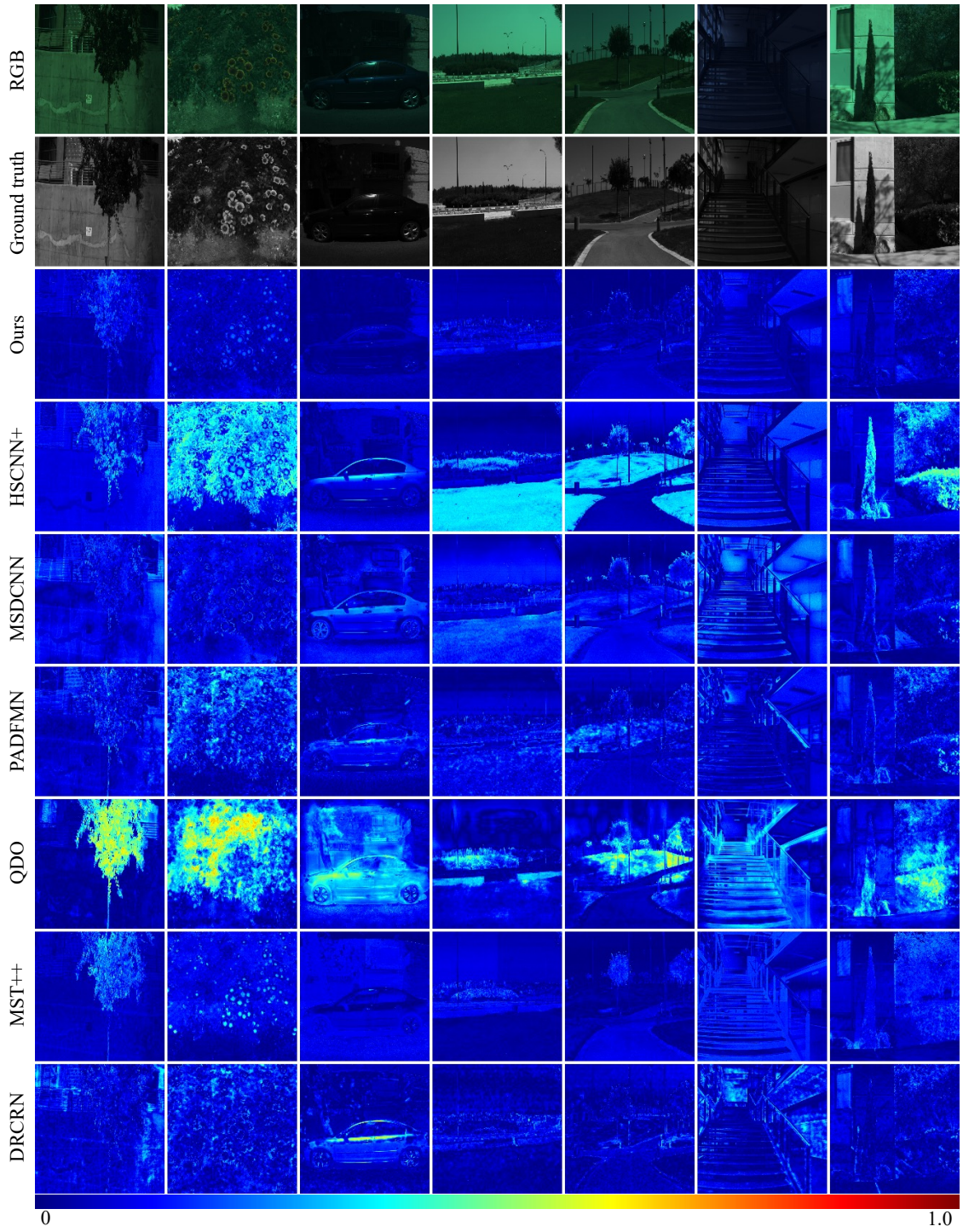


Figure D.1: More qualitative comparison of error maps (MAE between the recovered results and the ground truth) on synthetic data with state-of-the-art approaches.

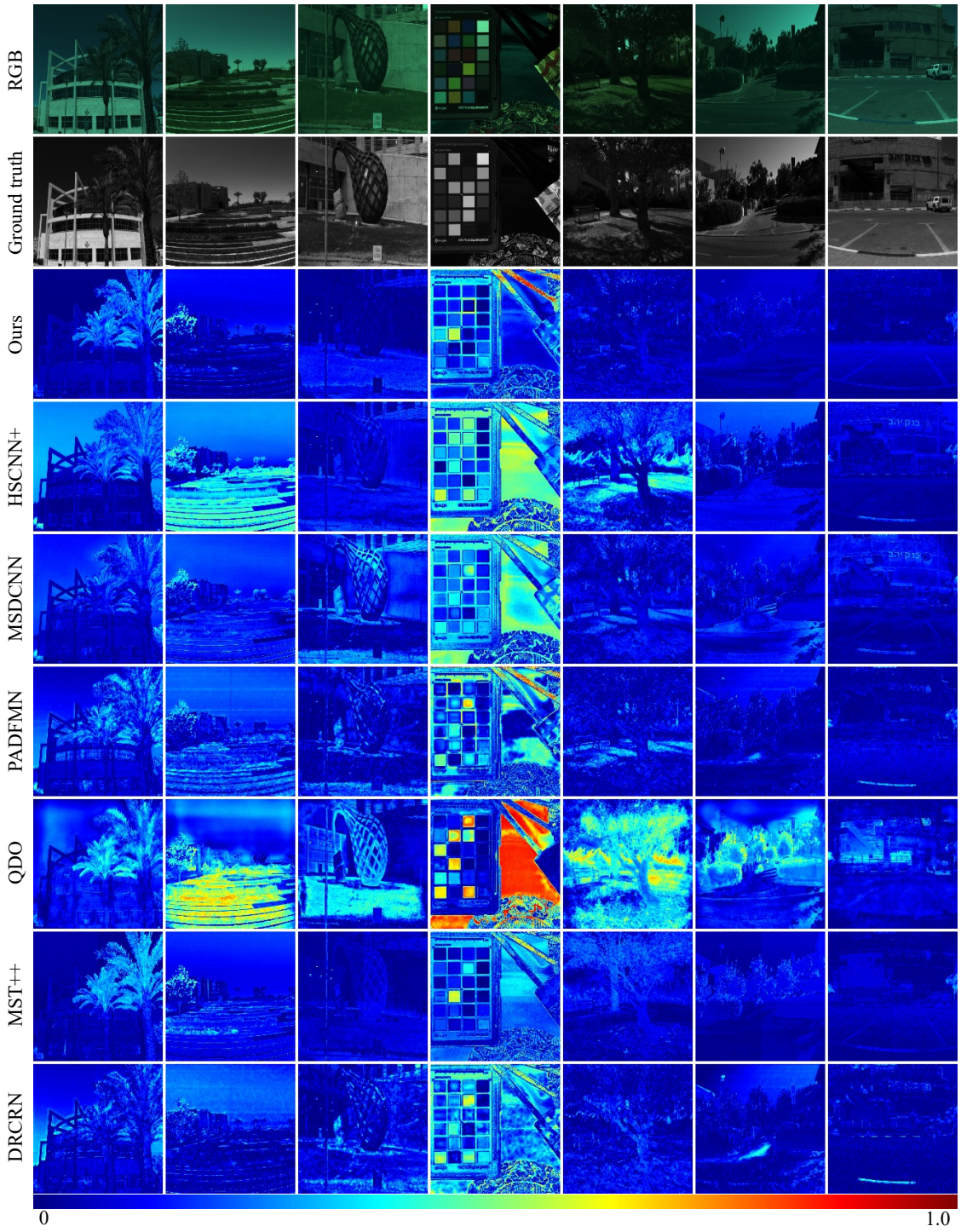


Figure D.2: More qualitative comparison of error maps (MAE between the recovered results and the ground truth) on synthetic data with state-of-the-art approaches.

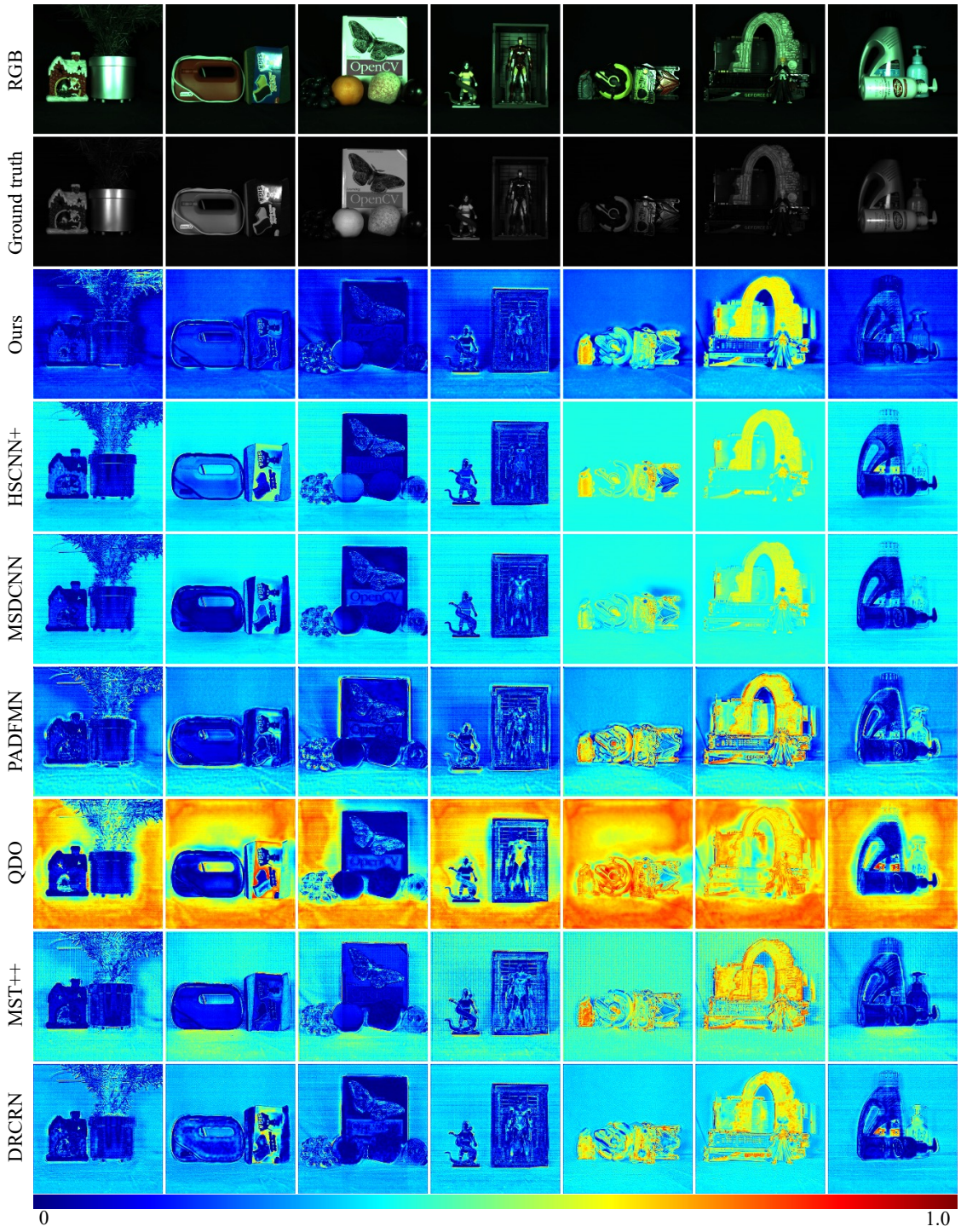


Figure D.3: More qualitative comparison of error maps (MAE between the recovered results and the ground truth) on real data with state-of-the-art approaches.

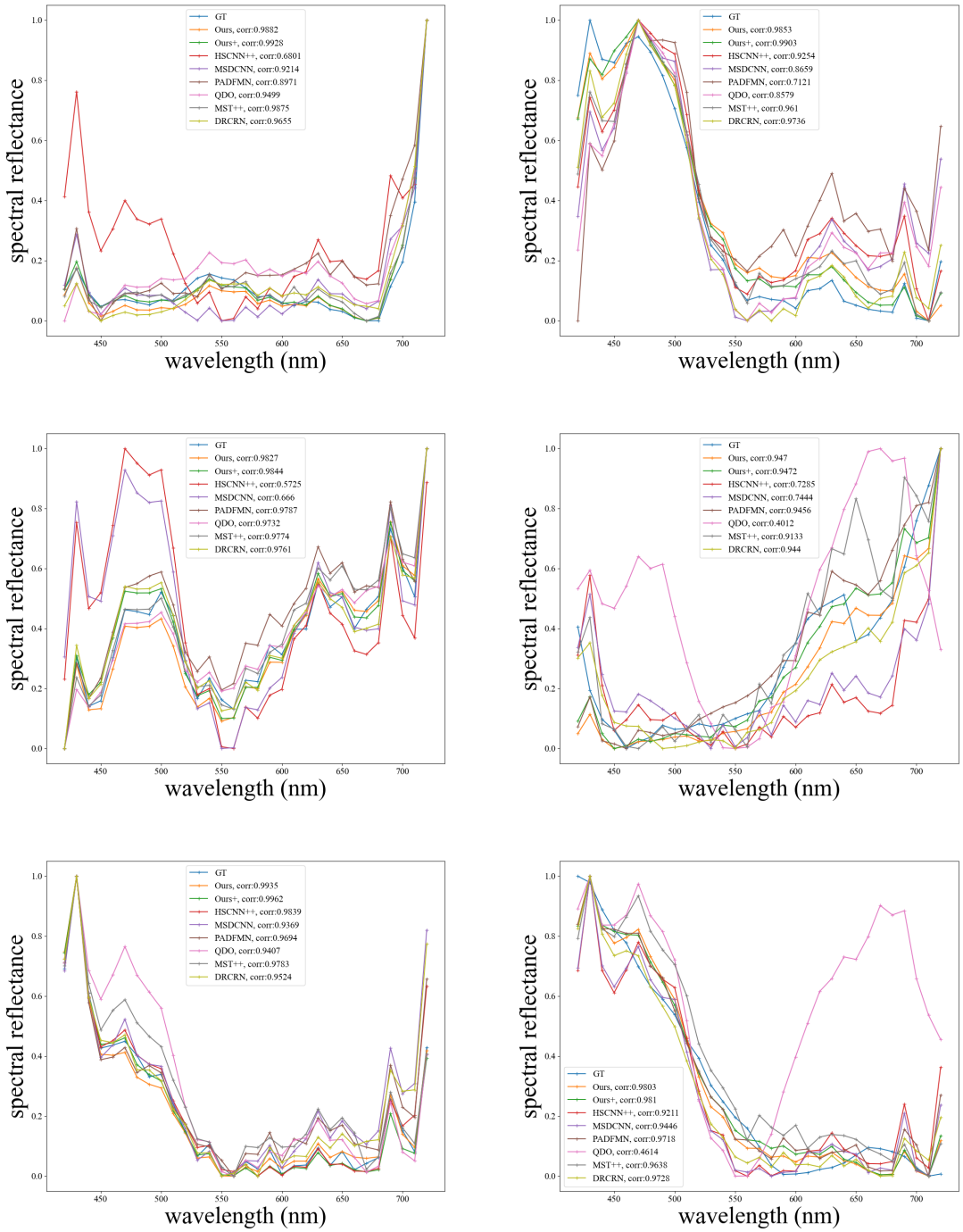


Figure D.4: Comparison of recovered spectral reflectance curves on synthetic data. We can see that our recovered spectral reflectance has higher correlation with the ground truth than that of other methods.

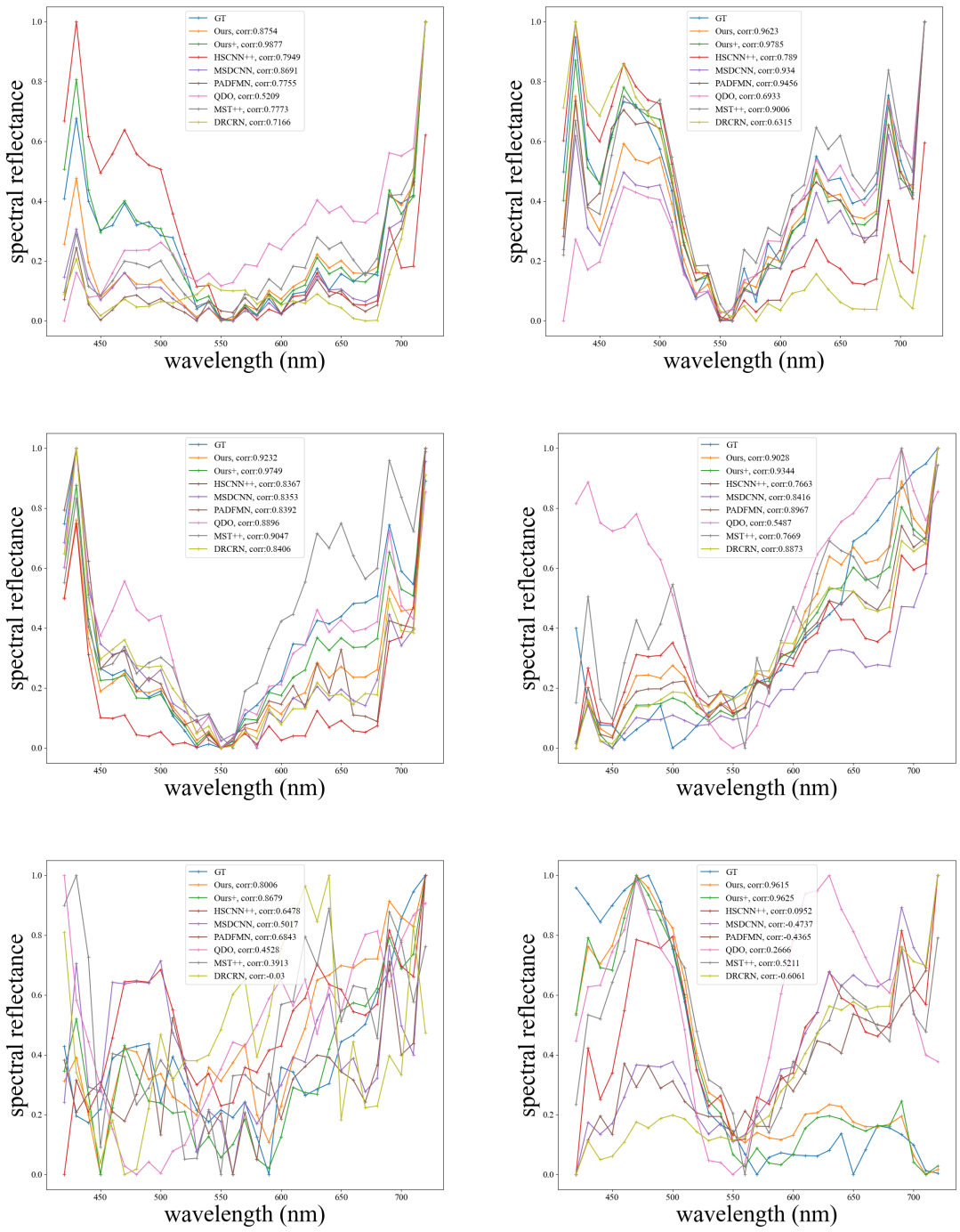


Figure D.5: Comparison of recovered spectral reflectance curves on synthetic data. We can see that when the quality of our recovered spectral reflectance under a single illumination is non-ideal, one more illumination can significantly improve the performance of our method.

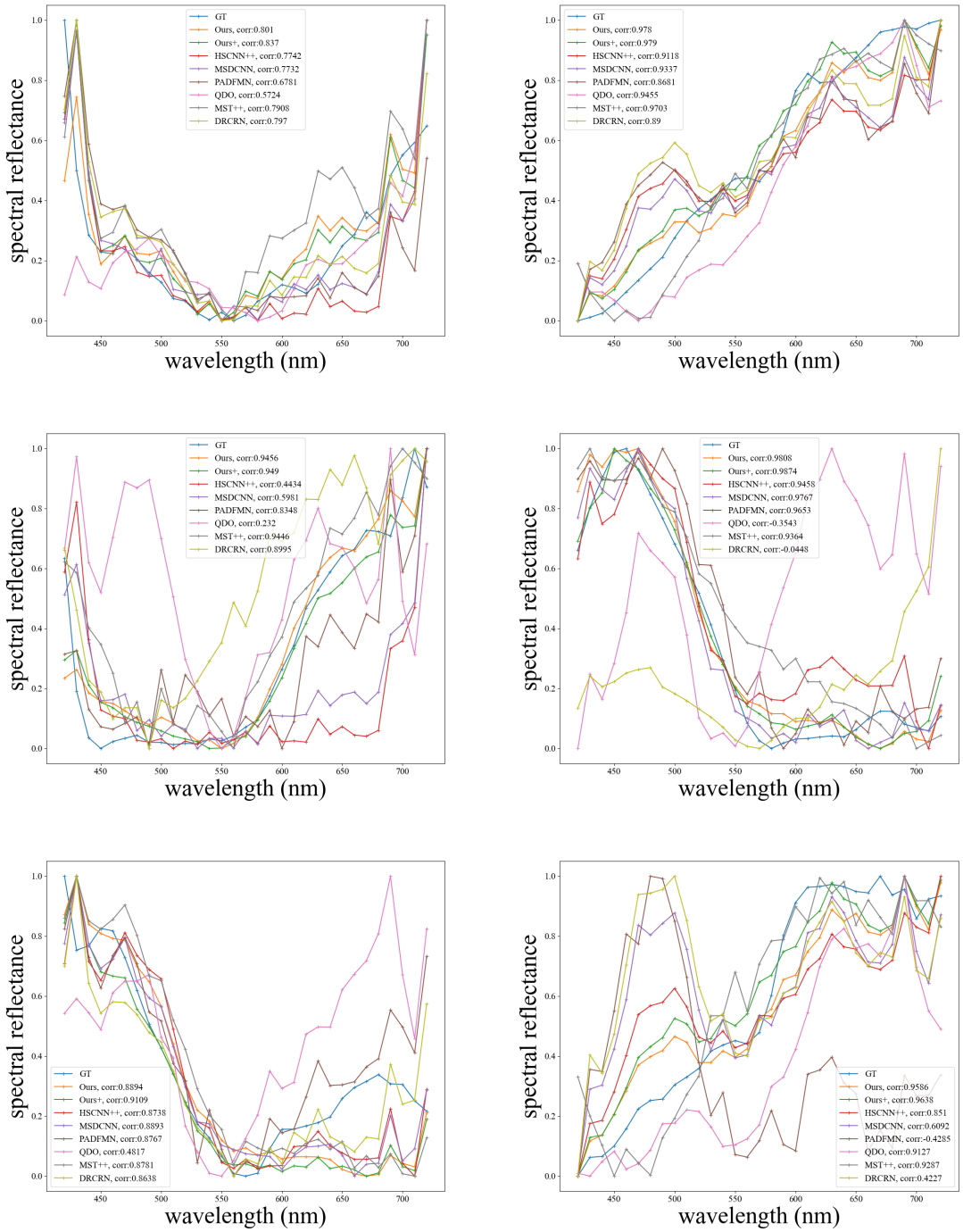


Figure D.6: Comparison of recovered spectral reflectance curves on real data. We can see that one more illumination can also help to improve the performance when testing on real data.

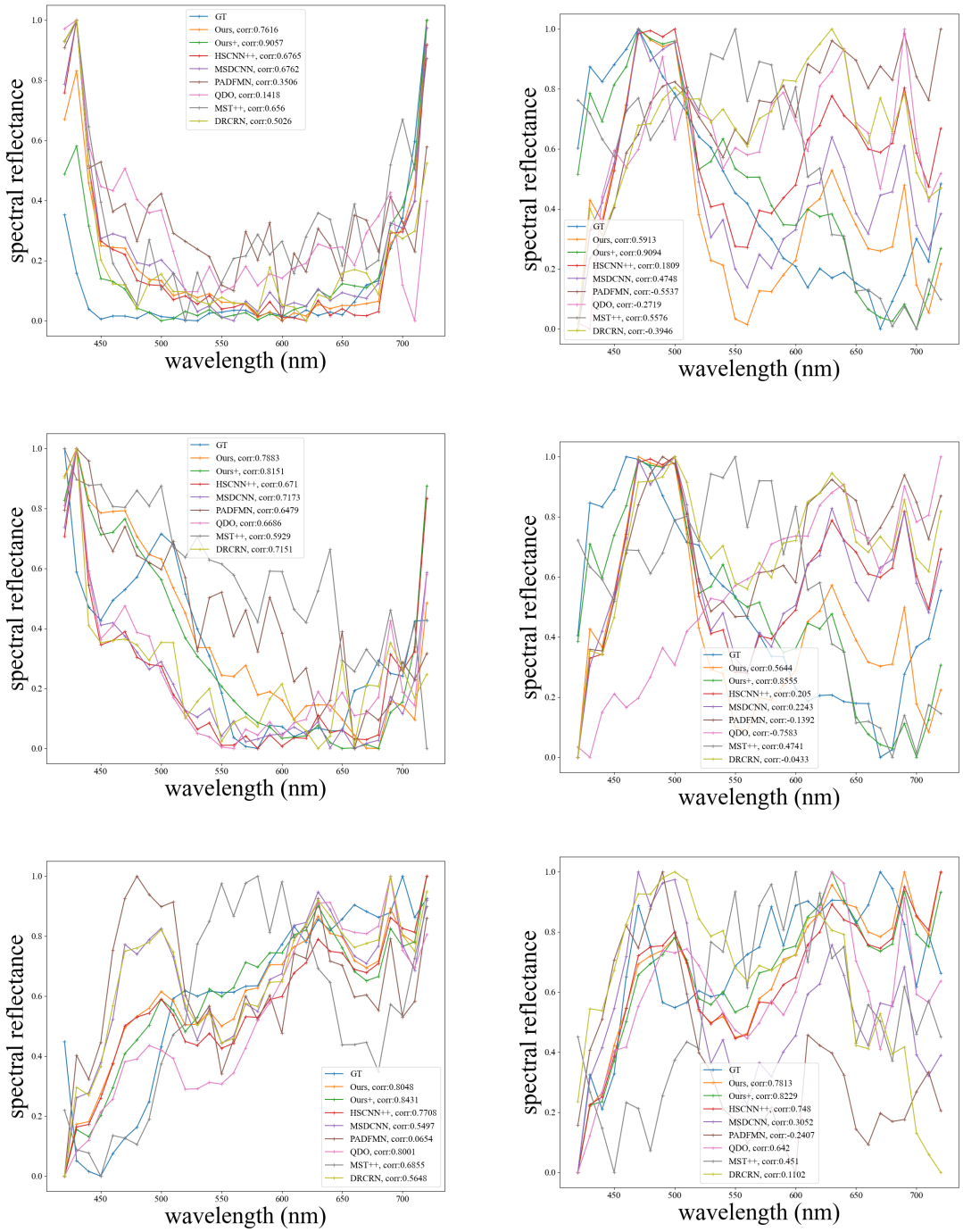


Figure D.7: Comparison of recovered spectral reflectance curves on real data. We can see that using a single illumination may still suffer from the domain gap and one more illumination can reduce this problem.

Figure D.8: True tone flash of an iPhone XR with LEDs off (left), white LEDs on (middle) and amber LEDs on (right). The middle and the right images are obtained from [347] which need jailbreak to change the color of flashlights.

D.3 Feasibility analysis of data capture

Our method needs RGB images of the same scene under different illuminations as the input. Capturing RGB images of the same scene under different illuminations is feasible and has been realized in tasks like photometric stereo. To our knowledge, two options exist. Firstly, sequential acquisition is common if the scene is static. Secondly, a commodity high-speed camera such as iPhone can be utilized. Specifically, consider the exposure time of RGB images is short, one could record high-speed videos (120/240 FPS) with alternating light sources to obtain images under different illuminations as in [348]. The iPhone flashlights consist of both white and amber LEDs (see Fig. D.8), which can be used as alternating light sources. The impact of ambient light can be removed by subtracting images captured with white/amber LEDs off. Small motion in high-speed videos is negligible. Extremely fast object/camera motion is beyond the scope of this paper. In addition to exploit the flashlight and the rear-facing camera of an iPhone, one could also explore the screen light and the front-facing camera.

Appendix E: Glass Segmentation with RGB-Thermal Image Pairs

This appendix provides 1) the full architectural specifications in Table E.1, E.2 and E.3; 2) information of competing methods in Section E.1; and 3) more qualitative comparison in Section E.2.

E.1 Information of competing methods

In Chapter 7.4 of the thesis, we compare our proposed method with 20 state-of-the-art methods which are retrained on our dataset. We show the implementation frameworks, source code links and corresponding licenses (if available) in Table E.4. For semantic segmentation methods, we change the number of output classes to 2 (glass/non-glass). For all competing methods, We change the input to RGB-T image pairs from our dataset, except that EBLNet [349] takes RGB images only. Default training settings from their source codes are used.

E.2 More qualitative results

Figure E.1, E.2, E.3, E.4, E.5 and E.6 show more qualitative comparisons of our RGB-T fusion method with a RGB-only variant and five competing methods (HDFNet [177], ESANet [320], CLNet [187], SPNet [185], and VST [190]) using our test split. The results are sorted in the order of increasing glass area. Visually, our RGB-T method consistently outperforms the RGB-only method by a large margin. Note that the five competing methods also take RGB-T pairs as input, whereas our method achieves

network	layer	k	s	channel	input	output size
	conv_r1	7	2	3/64	RGB image	240 × 320 × 64
	max_pool_r	3	2	64/64	conv_r1	120 × 160 × 64
RGB	resblock_r1			64/256	max_pool_r	120 × 160 × 256
resnet-50	resblock_r2			256/512	resblock_r1	60 × 80 × 512
encoder	resblock_r3			512/1024	resblock_r2	30 × 40 × 1024
	resblock_r4			1024/2048	resblock_r3	15 × 20 × 2048
	conv_r2	1	1	2048/256	resblock_r4	15 × 20 × 256
	conv_t1	7	2	3/64	Thermal image	240 × 320 × 64
	max_pool_t	3	2	64/64	conv_t1	120 × 160 × 64
Thermal	resblock_t1			64/256	max_pool_t	120 × 160 × 256
resnet-50	resblock_t2			256/512	resblock_t1	60 × 80 × 512
encoder	resblock_t3			512/1024	resblock_t2	30 × 40 × 1024
	resblock_t4			1024/2048	resblock_t3	15 × 20 × 2048
	conv_t2	1	1	2048/256	resblock_t4	15 × 20 × 256
	trans_r1			256/256	conv_r2	15 × 20 × 256
	trans_t1			256/256	conv_t2	15 × 20 × 256
	stack_1			256/256	conv_r2, conv_t2	30 × 20 × 256
	trans_s1			256/256	stack_1	30 × 20 × 256
	trans_m1			256/256	trans_s1	30 × 20 × 256
	linear_1			256/1	trans_m1	30 × 20 × 1
	sigmoid_1			1/1	linear_1	30 × 20 × 1
	sum_r1			256/256	trans_r1, sigmoid_1, trans_s1	15 × 20 × 256
	sum_t1			256/256	trans_t1, sigmoid_1, trans_s1	15 × 20 × 256
	trans_r2			256/256	sum_r1	15 × 20 × 256
	trans_t2			256/256	sum_t1	15 × 20 × 256
	stack_2			256/256	sum_r1, sum_t1	30 × 20 × 256
	trans_s2			256/256	stack_2	30 × 20 × 256
	trans_m2			256/256	trans_s2	30 × 20 × 256
	linear_2			256/1	trans_m2	30 × 20 × 1
MFM	sigmoid_2			1/1	linear_2	30 × 20 × 1
	sum_r2			256/256	trans_r2, sigmoid_2, trans_s2	15 × 20 × 256
	sum_t2			256/256	trans_t2, sigmoid_2, trans_s2	15 × 20 × 256
	trans_r3			256/256	sum_r2	15 × 20 × 256
	trans_t3			256/256	sum_t2	15 × 20 × 256
	stack_3			256/256	sum_r2, sum_t2	30 × 20 × 256
	trans_s3			256/256	stack_3	30 × 20 × 256
	trans_m3			256/256	trans_s3	30 × 20 × 256
	linear_3			256/1	trans_m3	30 × 20 × 1
	sigmoid_3			1/1	linear_3	30 × 20 × 1
	sum_r3			256/256	trans_r3, sigmoid_3, trans_s3	15 × 20 × 256
	sum_t3			256/256	trans_t3, sigmoid_3, trans_s3	15 × 20 × 256
	trans_r4			256/256	sum_r3	15 × 20 × 256
	trans_t4			256/256	sum_t3	15 × 20 × 256
	stack_4			256/256	sum_r3, sum_t3	30 × 20 × 256
	trans_s4			256/256	stack_4	30 × 20 × 256
	trans_m4			256/256	trans_s4	30 × 20 × 256
	linear_4			256/1	trans_m4	30 × 20 × 1
	sigmoid_4			1/1	linear_4	30 × 20 × 1
	fusion_out			256/256	sigmoid_4, trans_s4	15 × 20 × 256

Table E.1: Detailed specifications of the proposed neural network architecture. **k** is the kernel size, **s** the stride, **channel** the number of channels of input and output, **input** the input to the layer, and **output size** the size of the output in the form of *height*×*width*×*channel*. All convolution layers are each followed by a ReLU and a batch norm layer except for those before the sigmoid functions.

network	layer	k	s	channel	input	output size
Decoder	conv_d_r4	1	1	2048/256	resblock_r4	15 × 20 × 256
	conv_sigmoid_r4	3	1	$\begin{bmatrix}512/256\\256/256\\256/1\end{bmatrix}$	conv_d_r4, fusion_out	15 × 20 × 1
	conv_d_t4	1	1	2048/256	resblock_t4	15 × 20 × 256
	conv_sigmoid_t4	3	1	$\begin{bmatrix}512/256\\256/256\\256/1\end{bmatrix}$	conv_d_t4, fusion_out	15 × 20 × 1
	sum_d4			256/256	conv_sigmoid_r4, conv_sigmoid_t4, conv_d_r4, conv_d_t4, fusion_out	15 × 20 × 256
	conv_d4	3	1	256/256	sum_d4	15 × 20 × 256
	bilinear_4			256/256	conv_d4	30 × 40 × 256
	conv_d_r3	1	1	1024/256	resblock_r3	30 × 40 × 256
	conv_sigmoid_r3	3	1	$\begin{bmatrix}512/256\\256/256\\256/1\end{bmatrix}$	conv_d_r3, blinear_4	30 × 40 × 1
	conv_d_t3	1	1	1024/256	resblock_t3	30 × 40 × 256
	conv_sigmoid_t3	3	1	$\begin{bmatrix}512/256\\256/256\\256/1\end{bmatrix}$	conv_d_t3, bilinear_4	30 × 40 × 1
	sum_d3			256/256	conv_sigmoid_r3, conv_sigmoid_t3, conv_d_r3, conv_d_t3, bilinear_4	30 × 40 × 256
	conv_d3	3	1	256/128	sum_d3	30 × 40 × 128
	bilinear_3			128/128	conv_d3	60 × 80 × 128
	conv_d_r2	1	1	512/128	resblock_r2	60 × 80 × 128
	conv_sigmoid_r2	3	1	$\begin{bmatrix}256/128\\128/128\\128/1\end{bmatrix}$	conv_d_r2, blinear_3	60 × 80 × 1
	conv_d_t2	1	1	512/128	resblock_t2	60 × 80 × 128
	conv_sigmoid_t2	3	1	$\begin{bmatrix}256/128\\128/128\\128/1\end{bmatrix}$	conv_d_t2, bilinear_3	60 × 80 × 1
	sum_d2			128/128	conv_sigmoid_r2, conv_sigmoid_t2, conv_d_r2, conv_d_t2, bilinear_3	60 × 80 × 128
	conv_d2	3	1	128/64	sum_d2	60 × 80 × 64
	bilinear_2			64/64	conv_d2	120 × 160 × 64
	conv_d_r1	1	1	256/64	resblock_r1	120 × 160 × 64
	conv_sigmoid_r1	3	1	$\begin{bmatrix}128/64\\64/64\\64/1\end{bmatrix}$	conv_d_r1, blinear_2	120 × 160 × 1
	conv_d_t1	1	1	256/64	resblock_t1	120 × 160 × 64
	conv_sigmoid_t1	3	1	$\begin{bmatrix}128/64\\64/64\\64/1\end{bmatrix}$	conv_d_t1, bilinear_2	120 × 160 × 1
	sum_d1			64/64	conv_sigmoid_r1, conv_sigmoid_t1, conv_d_r1, conv_d_t1, bilinear_2	120 × 160 × 64
	conv_d1	3	1	64/32	sum_d1	120 × 160 × 32
	bilinear_1			32/32	conv_d1	480 × 640 × 32
	conv_d0	3	1	32/16	bilinear_1	480 × 640 × 16
	conv_out	1	1	16/1	conv_d0	480 × 640 × 1

Table E.2: Detailed specifications of the proposed neural network architecture (continued).

layer	**head**	**channel**	**output size**
self-attention	8	256/256	$15 \times 20 \times 256$
linear_1		256/256	$15 \times 20 \times 256$
linear_2		256/256	$15 \times 20 \times 256$

Table E.3: Details of the transformer layer [7].

Method	Original task	Framework	Code link	License
RTFNet [214]	RGB-T semantic segmentation	PyTorch	link	MIT License
ShapeConv [319]	RGB-D semantic segmentation	PyTorch	link	Apache-2.0 License
ESANet [320]	RGB-D semantic segmentation	PyTorch	link	BSD 3-Clause License
DPANet [325]	RGB-D salient object segmentation	PyTorch	link	MIT License
HAINet [326]	RGB-D salient object segmentation	PyTorch	link	-
Zhang *et al.* [192]	RGB-T salient object segmentation	TensorFlow	link	-
SSF [179]	RGB-D salient object segmentation	PyTorch	link	-
UCNet [174]	RGB-D salient object segmentation	PyTorch	link	-
CoNet [327]	RGB-D salient object segmentation	PyTorch	link	-
ASTA [189]	RGB-D salient object segmentation	Pytorch	link	-
DANet [175]	RGB-D salient object segmentation	PyTorch	link	MIT License
HDFNet [177]	RGB-D salient object segmentation	PyTorch	link	MIT License
FRDT [328]	RGB-D salient object segmentation	PyTorch	link	-
RD3D [176]	RGB-D salient object segmentation	PyTorch	link	MIT License
DCFNet [182]	RGB-D salient object segmentation	PyTorch	link	MIT License
UTA [329]	RGB-D salient object segmentation	PyTorch	link	Apache-2.0 License
VST [190]	RGB-D salient object segmentation	PyTorch	link	-
CLNet [187]	RGB-D salient object segmentation	PyTorch	link	-
SPNet [185]	RGB-D salient object segmentation	PyTorch	link	-
EBLNet [349]	RGB-only glass segmentation	PyTorch	link	-

Table E.4: Information of competing methods

cleaner and sharper segmentation results due to our powerful cross-modality fusion scheme.

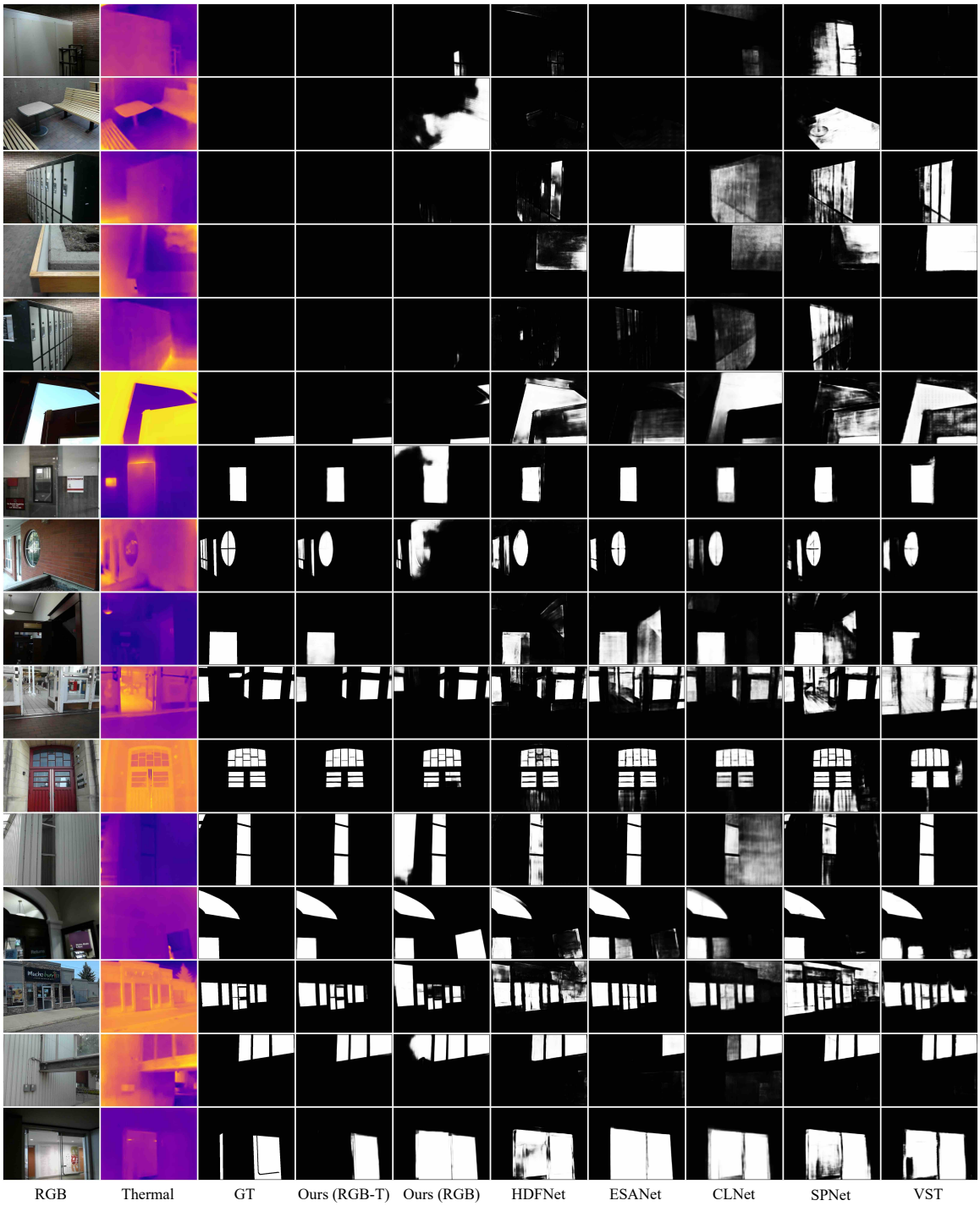


Figure E.1: More qualitative comparison results.

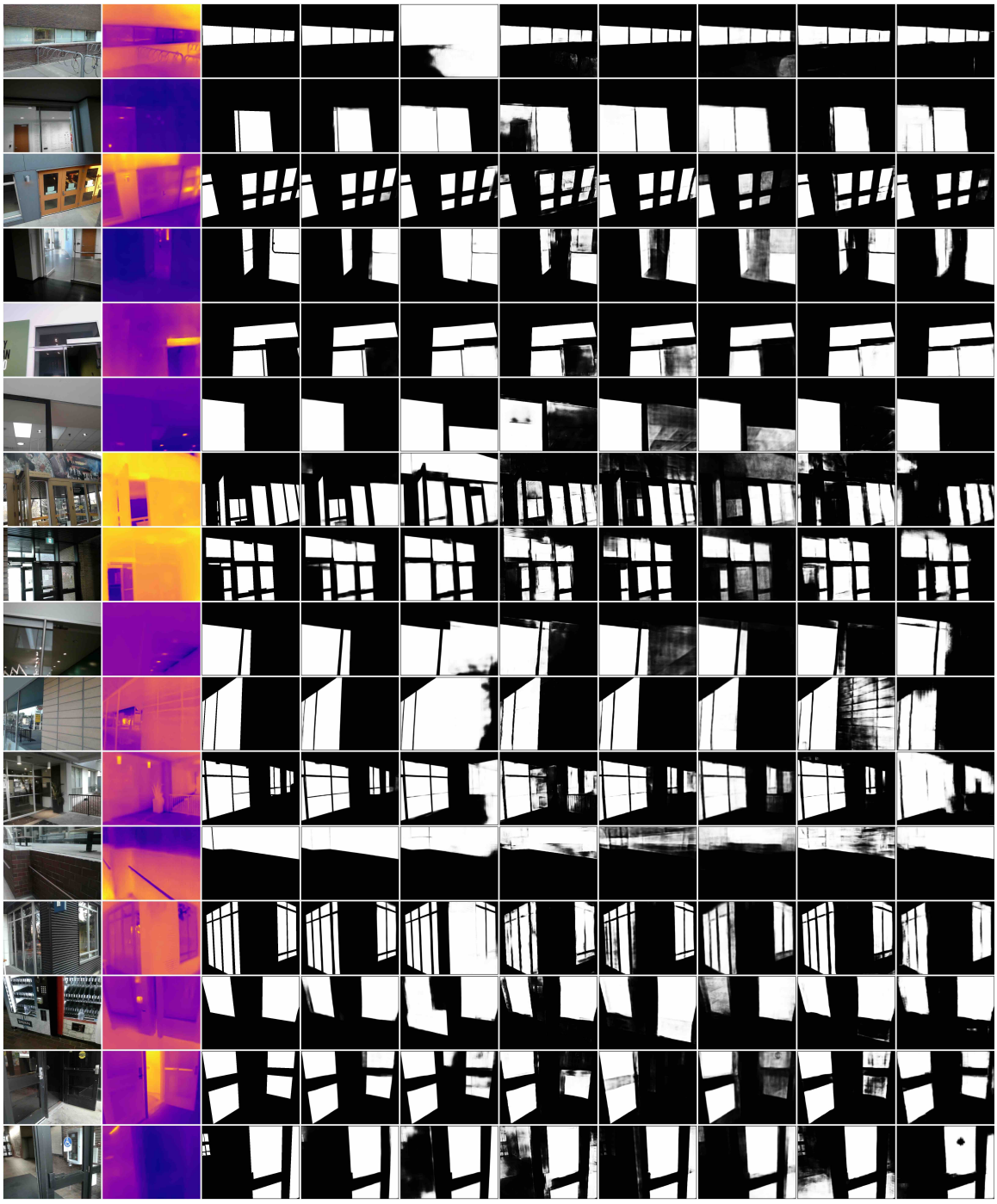


Figure E.2: More qualitative comparison results.

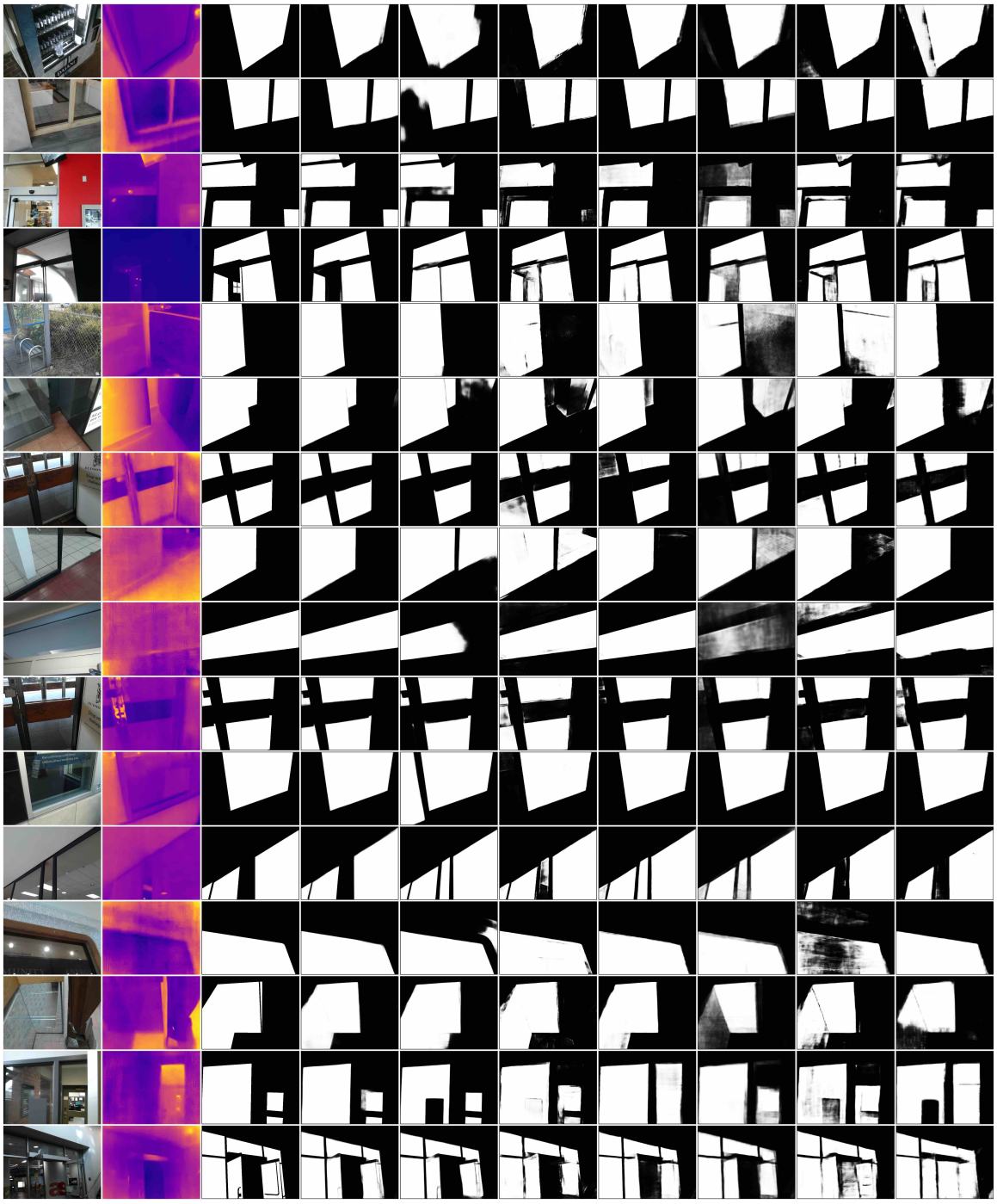


Figure E.3: More qualitative comparison results.

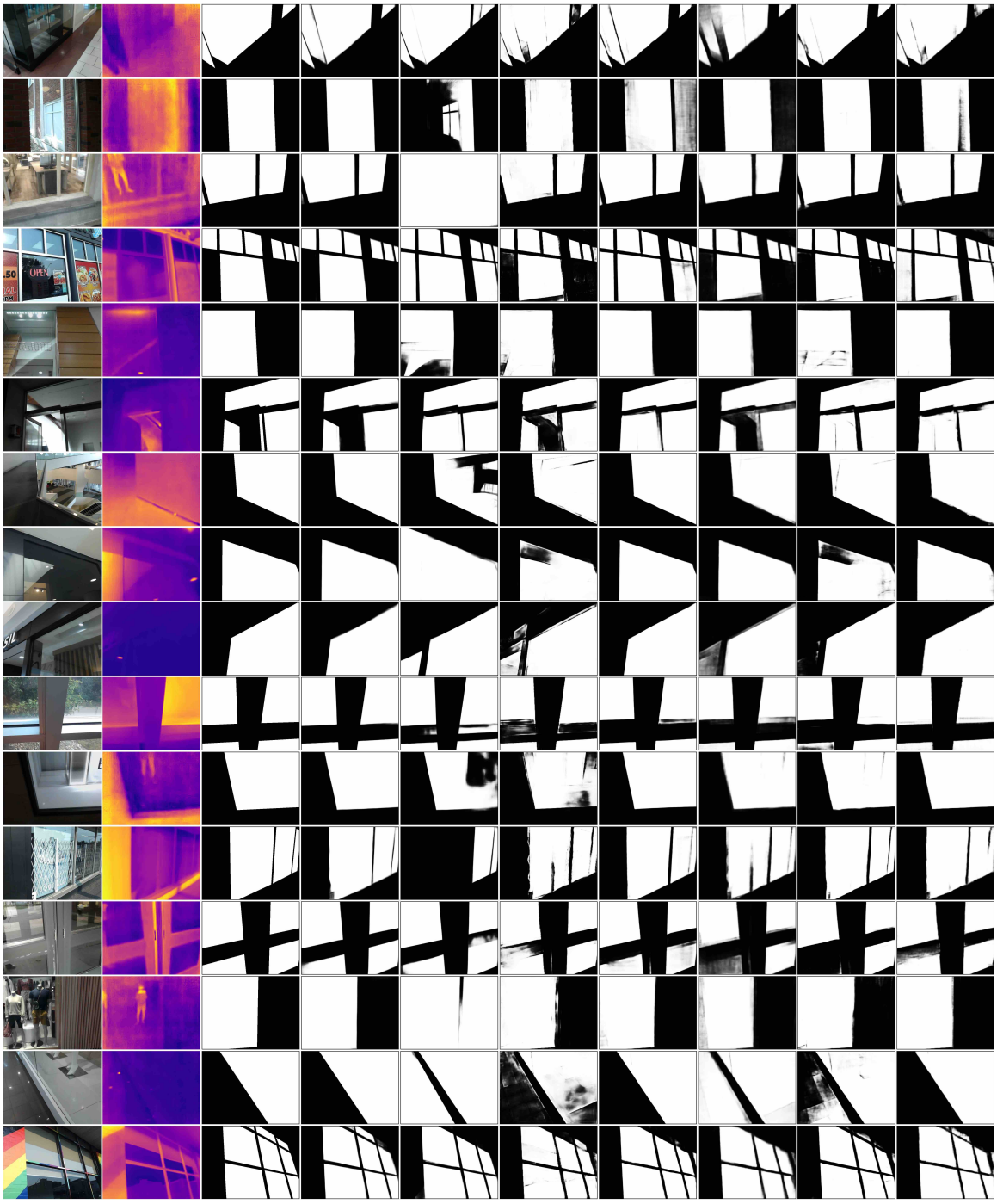


Figure E.4: More qualitative comparison results.

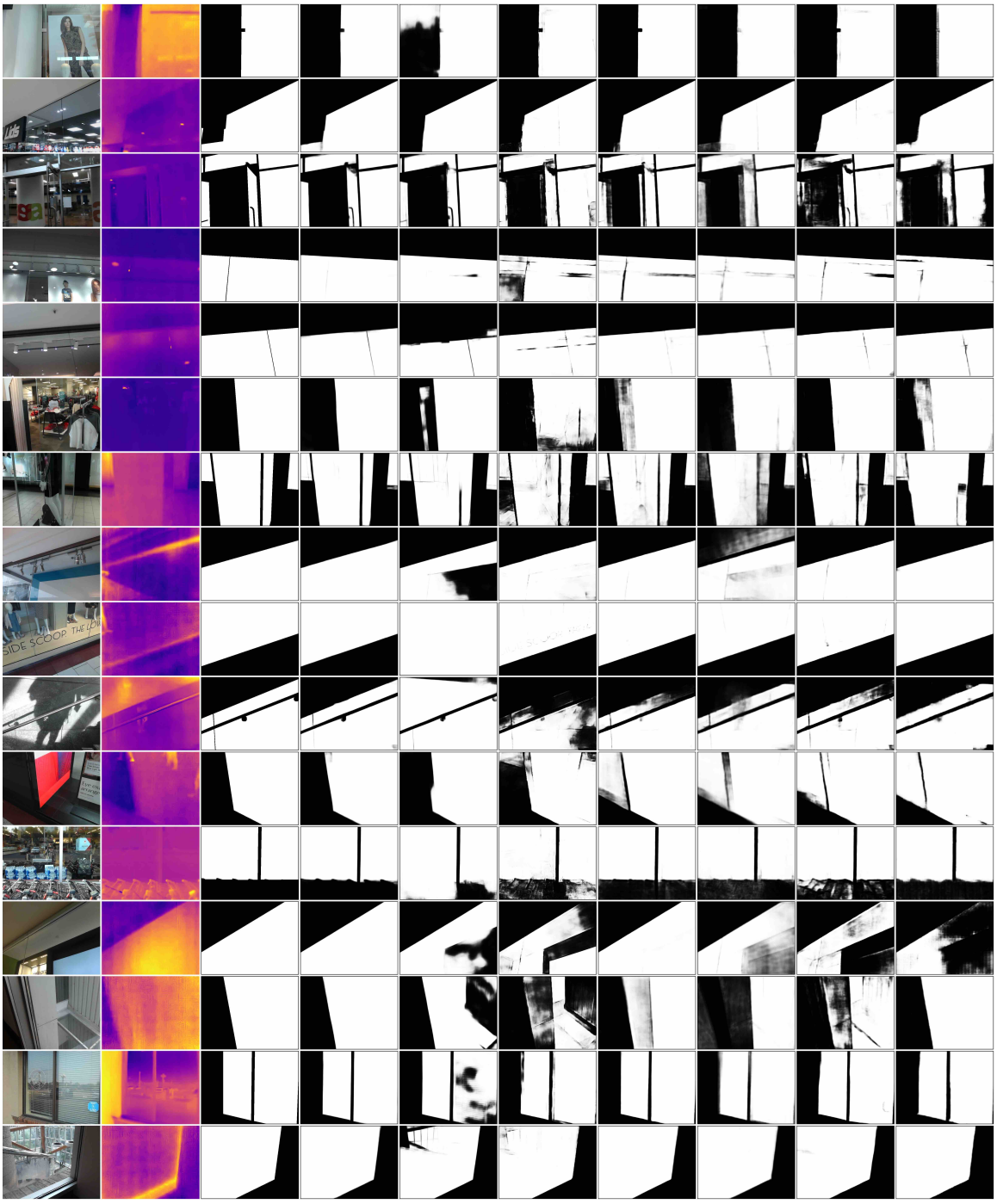


Figure E.5: More qualitative comparison results.

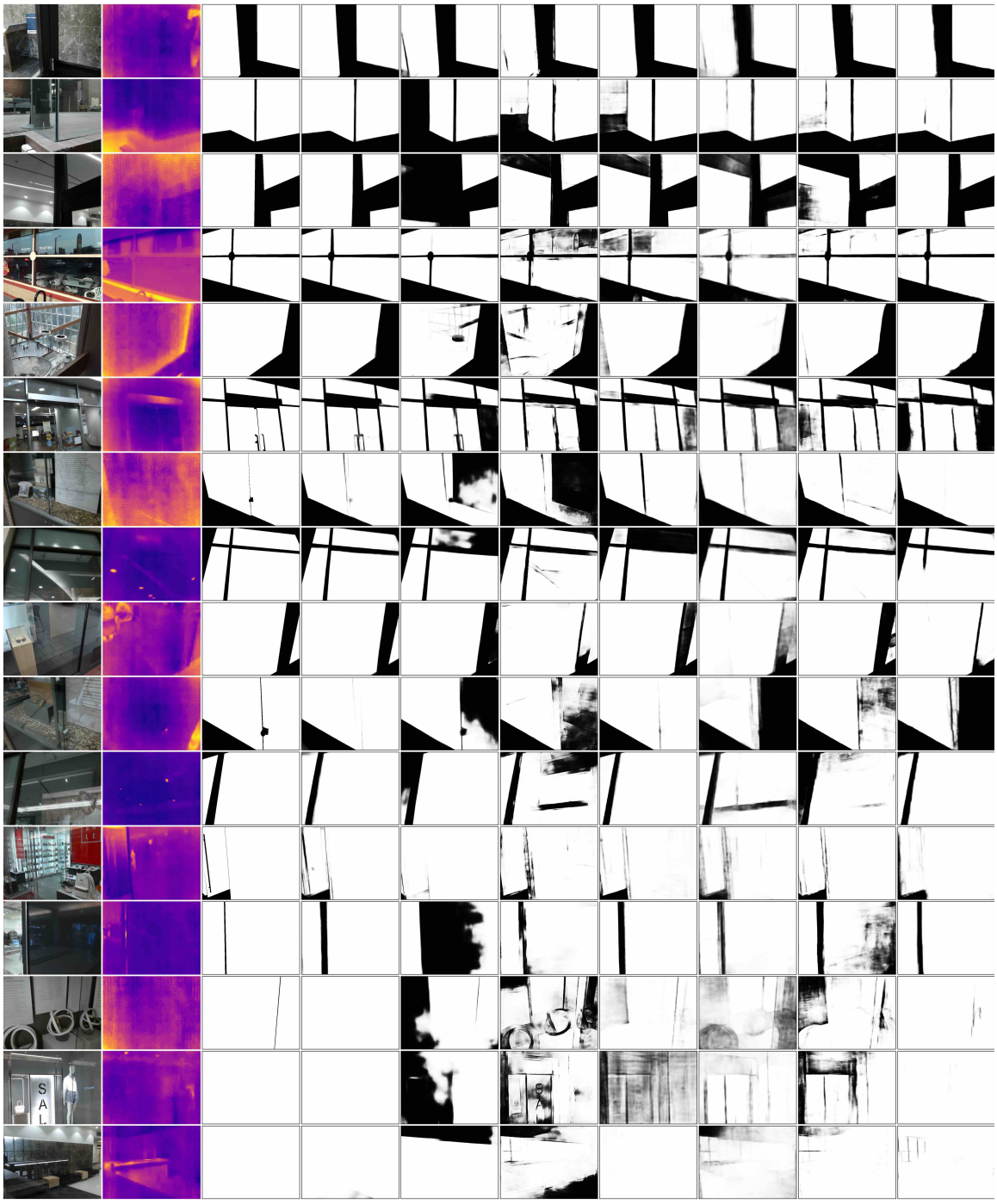


Figure E.6: More qualitative comparison results.