Image Synthesis with Appearance Decomposition

Bowei Chen
CMU-RI-TR-22-52
July 14, 2022

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Srinivasa Narasimhan, co-chair
Martial Hebert, co-chair
Jun-Yan Zhu
Mengtian Li

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

Copyright © 2022 Bowei Chen. All rights reserved.
To my family.
Abstract

Our visual world is compositional and its appearance can be decomposed into various components. Leveraging these components can be beneficial for challenging image synthesis tasks. To this end, this thesis focuses on studying how appearance decomposition can improve image synthesis methods using two examples. (1) Structural decomposition: we introduce a periodicity-aware single image framework to synthesize a scene of near-periodic patterns (NPP). In particular, the appearance of an NPP scene is decomposed into motifs and their corresponding periodicities (i.e., arrangement), which are injected into the proposed framework as a prior to synthesize the NPP scene. The proposed method can interpolate and extrapolate NPP images, in-paint large and arbitrarily shaped regions, recover blurry regions when images are remapped, segment periodic and non-periodic regions, in planar and multi-planar scenes. (2) Intrinsic decomposition: we propose a novel approach to decompose a single panorama of an empty indoor environment into four appearance components: specular, direct sunlight, diffuse and diffuse ambient without direct sunlight. This appearance decomposition enables multiple image synthesis applications including sun direction estimation, virtual furniture insertion, floor material replacement, and sun direction change. We conduct extensive experiments to demonstrate the effectiveness of both methods.
Acknowledgments

I would like to express my deepest appreciation to my advisor Srinivasa Narasimhan for his unwavering support and wonderful guidance. He taught me how to define and explore an important and unique research problem that is not well studied in the field. He also guided me to always have high-level and critical thinking for my research. He is such a good advisor and I feel extremely fortunate to work with him.

Many thanks to Martial Hebert, Sing Bing Kang, and Ivaylo Boyadzhiev for their insightful comments and valuable suggestions on my research projects. I would also like to thank the rest of my committee members, Jun-Yan Zhu and Mengtian Li.

A special thank goes to Tiancheng Zhi for his help from high-level discussion to low-level details. I thank my friends and labmates at CMU: Gaurav, Tianyuan, Sid, Shumian, Dinesh, Khiem, Robert, Adithya, Mark, Anurag, Tiffany, Zixuan, Ruoyang, Zilin, Xinjie, Jack, Andy.

I thank my parents, Jiansong and Bingjun, and my brother, Bohan, for always encouraging me to pursue my dream. Last but certainly not the least, I thank my girlfriend Jingyi for your unconditional love and support.
## Contents

1 Introduction

2 Learning Continuous Implicit Representation for Near-Periodic Patterns
   2.1 Introduction ........................................... 5
   2.2 Related Work ........................................... 7
      2.2.1 Near-Periodic Patterns Completion ................. 7
      2.2.2 Implicit Neural Representations: ................... 9
   2.3 NPP-Net ................................................. 9
      2.3.1 Periodicity-Aware Input Warping ................. 10
      2.3.2 Coordinate-Based MLP ............................. 11
      2.3.3 Single Image Optimization ....................... 12
      2.3.4 Periodicity Proposals ............................ 14
      2.3.5 Extensions ....................................... 16
      2.3.6 Implementation Details ......................... 20
   2.4 Experiments ............................................ 22
      2.4.1 Dataset ........................................... 22
      2.4.2 Metrics ........................................... 22
      2.4.3 Ablation Study ................................... 23
      2.4.4 Baselines ....................................... 24
      2.4.5 Comparison with Baselines: ..................... 24
      2.4.6 Influence of Mask Size: .......................... 25
   2.5 Limitation ............................................. 28
   2.6 Conclusion ............................................ 28

3 Semantically Supervised Appearance Decomposition for Virtual Staging from a Single Panorama
   3.1 Introduction ........................................... 29
   3.2 Related Work ........................................... 32
   3.3 Semantics for Coarse Localization of Lighting Effects .... 35
      3.3.1 Transfer Learning for Semantic Segmentation .... 36
      3.3.2 Coarse Localization of Lighting Effects ........ 37
   3.4 GAN-Based Lighting Effects Detection ................ 38
      3.4.1 Key Idea and Overall Architecture .............. 39
List of Figures

1.1 A scene of building facade. We can decompose its visual appearance into two components based on its structure: (1) *structural elements*: a motif (window) that occurs repeatedly in a scene. The yellow box visualizes one example. (2) *spatial arrangement*: how the motif tiles in a scene. In this scene, it refers to the periodicity, showing how can we tile the content in the yellow box to obtain the one in the purple box.

1.2 A scene of an indoor empty home (left). We can decompose its visual appearance into two components based on its physical properties: (1) *shading component*: an image (top right) contains two commonly presented high-frequency light effects, specular reflection and direct sunlight, in an indoor home. (2) *ambient component*: an image (bottom right) without these two light effects.

2.1 Inpainting to remove the tree, street lamp, and car from a near-periodic patterned (NPP) scene in (a). Input image (b) visualizes the mask (white unknown region) and detected (but inaccurate) NPP representation (yellow lattice). Guided by this inaccurate representation, the state of the art method BPI [68] (c) fails to generate windows occluded by the tree (orange arrow) and the white strip across the bottom (green arrow). Our NPP-Net (d) maintains global consistency and local variations, while preserving the known regions.

2.2 Initial pipeline of NPP-Net consists of three modules. (1) *Periodicity-Aware Input Warping* (pink) warps input coordinates using detected periodicity. (2) *Coordinate-based MLP* (blue) maps warped and input coordinate features to an RGB value. (3) *Single Image Optimization* (yellow) uses pixel loss and periodicity-guided patch loss on a single NPP image. Final pipeline in Section 2.3.4 shows how multiple periodicities are automatically detected and utilized.
2.3 Comparing different losses based on the final pipeline. The red, yellow and cyan dots in (a) visualize the Top-3 periodicities. Zoom-ins of the unknown area (white rectangle) are in (b)-(f). Merely using pixel loss (b) generates high-frequency artifacts across the image and periodic artifacts in the top part. Adopting only patch loss (c) removes the artifacts but has poor global structure. Using pixel loss and patch loss with random sampling (d) cannot preserve global consistency and local variations well since the ground truth patches are not sampled according to periodicity and might be far from the predicted patch. With pixel loss and periodicity-guided patch loss, NPP-Net (e) solves these issues. .................................................. 14

2.4 Final pipeline of NPP-Net modifies two modules of the initial pipeline. (1) Periodicity Proposal (green) automatically searches and augments the input periodicity to handle inaccurate periodicity detection and encourage the global consistency. (2) Coordinate-based MLP (lavender blue) has two branches: (a) for Top-1 periodicity and original coordinates, and (b) for the rest. .................................................. 15

2.5 Illustration of an NPP Remapping scenario. ...................... 18

2.6 Illustration of the multi-planar image completion extension for NPP-Net. .................................................. 20

2.7 Qualitative results for NPP completion. We show four baselines that operate on a single image. The red, yellow, and cyan dots in input images show the first, second, and third periodicity from periodicity searching module, respectively. For visualization, all periods are scaled by 2. Our NPP-Net outperforms all baselines for global consistency (rows 1-6) and local variations (rows 6-9). .................................................. 26

2.8 Qualitative comparison for multi-plane NPP completion. We show three baselines, which are either designed for multi-plane NPP scenes (Huang et al. and BPI) or trained on large datasets (Lama). Some zoom-in boxes are resized for visualization. ................................. 27

2.9 Left: The number of NPP images for each periodicity that has the smallest periodicity errors (among Top-3) with different mask sizes in NRTDB. As the mask size grows, the best periodicity emerges in the Top 2nd and 3rd periodicity, thus utilizing them in NPP-Net is useful. Right: The LPIPS results (lower is better) for different mask sizes in NRTDB. PatchMatch performs the best when masks are very small but NPP-Net outperforms all the baselines for large masks. ................................. 28
3.1 A single panorama of an empty indoor environment (top left) is decomposed into four appearance components on the floor and walls: specular and direct sunlight (bottom left), diffuse (not shown) and diffuse ambient without direct sunlight (bottom right). Although the decomposition is not perfect (e.g. power sockets are removed in the bottom right), we demonstrate its effectiveness in enabling high quality virtual staging applications including furniture insertion\(^1\) (top right), changing flooring and sun direction (Sec. 3.7). Multiple realistic effects are rendered, including shadows under the table, sunlight on the side table, sofa pillows, beanbag, dining table and chaise lounge, occlusion of specular reflection by the sofa and chair. The plant creates soft and hard shadows by blocking the skylight and the sunlight. On the other hand, inserting furniture without appearance decomposition fails to reproduce these effects (Fig. 3.3).  

3.2 Our system consists of three modules: Coarse Effects Localization (Sec. 3.3) coarsely localizes lighting effects using automatically generated semantics; Lighting Effects Detection (Sec. 3.4) separates specular reflection on the floor and direct sunlight on the floor and the wall (effects are brightened for visualization); Lighting Effects Removal (Sec. 3.5) removes the detected specular reflection and direct sunlight, outputting a diffuse image (no specular) and an ambient image (no specular and sunlight). The specular reflection, direct sunlight, diffuse image, and ambient image can be used for various virtual staging applications\(^1\).  

3.3 Object insertion result without appearance decomposition. Note that the direct sunlight on the objects, hard shadows in the sunlight region, and blocked specular reflections are all missing (compare to Fig. 3.1).  

3.4 The geometry of lighting effects considered in this work. (a) For specular reflections, when the camera is upright, the light source point (transparent or open window, door, or indoor lamp), the reflection point, and the camera center lie on a vertical plane (shown as gray), corresponding to a column in the panorama. (b) For direct sunlight, the sun direction establishes the mapping between a window (or door) point and a floor point illuminated by direct sunlight. The sunlit floor area can be back-projected to the window according to the sun direction.  

3.5 A semantic map with 7 classes is computed automatically from a panorama. The map is used to obtain coarse lighting effect masks exploiting the geometric constraints in Fig. 3.4. These coarse masks are used to supervise a GAN-based method to obtain accurate specular and sunlit images.
3.6 Architecture for lighting effects detection. The lighting effects network takes a panorama $I$ as input, and predicts the specular ($I_{\text{spec}}$) and sunlight ($I_{\text{sunl}}$) components. If the prediction is good, a local discriminator trying to locate specular regions should fail on the image with specular component removed ($I - I_{\text{spec}}$). This is the key supervision signal for training the lighting effects network. The local discriminator is supervised by coarse specular masks obtained from semantics. Similar techniques are applied to sunlight regions and regions with overlapping specular and direct sunlight.

3.7 Removing effects by naively subtracting the estimated lighting effects versus our approach. Naive subtraction ($I - I_{\text{spec}}$, $I - I_{\text{spec}} - I_{\text{sunl}}$) leaves lighting effects residues (specular/sunlight region is brighter than the surrounding, with visible boundaries). Our result removes these artifacts.  

3.8 Architecture for specular reflection removal. The diffuse network takes a single panoramic image $I$ as input, and predicts the diffuse component $I_{\text{diff}}$. Similar to Fig. 3.6, a local discriminator trying to locate specular regions should fail on the diffuse image $I_{\text{diff}}$. The discriminator is supervised by the fine specular mask obtained via thresholding and dilating the estimated specular component.

3.9 Ablation study for specular reflection removal by not including adversarial, reconstruction, perceptual or exclusion losses. While quantitative metrics are averaged over the image and over the dataset, the visuals illustrate the subtle but noticeable residual specular reflections when the different losses are removed from our method.

3.10 Qualitative results of specular reflection detection and removal. Specular reflections are brightened for visualization. Note that although the walls are shown, only the floor region is evaluated quantitatively. Our method successfully detects and removes the both window and lamp reflections. (d) removes lamp reflections but (b) and (c) do not. (b) (c) (d) leaves residues of window reflections on the floor.

3.11 High resolution appearance decomposition results. The lighting effects are successfully detected on a variety of floor types. For specular reflection removal, the floor texture in (c) is consistent with that in (a). Sunlight regions are inpainted with correct colors. Some floor textures are imperfect due to the limitation of the inpainting algorithm, e.g., the third row, where the camera tripod is inpainted in the original image.

3.12 Diverse lighting conditions of testing images. Row 1: large sunlight region; Rows 2-3: effects from (partially) occluded light sources (see red boxes in (a)). Our method successfully detects the lighting effects.
3.13 When an albedo estimation model [84] pre-trained on a mixture of synthetic and real perspective images is applied to our data (a), the domain shift causes obvious errors (b). But when applied to our ambient image (c), the same method is able to estimate albedos more accurately (d), without requiring to retrain on our dataset (which would not be possible in any case given the lack of ground truth annotation for supervision).

3.14 Sun direction estimation results. Left: histogram of sun elevation errors for images with visible sunlight; Right: An example of projecting the window mask to the floor and the walls according to estimated sun direction. The fine sun estimations provide more accurate elevation results. 33% of the fine estimations and of the 28% coarse estimations are within 5° error. 53% of the fine estimations and of the 52% coarse estimations are within 10° error. The example shows that the fine estimation produces a reasonable sun direction even with overlapping sunlight and specular reflection. This is an extreme example where coarse estimation completely fails and the fine estimation significantly improves the result.

3.15 Changing floor material. Row 1: original flooring; Row 2: change to (different) wood; Row 3: change to (different) carpet. The specular reflection on row 2 column 2 is rendered using the estimated HDR map (Sec. 3.6.1) and a uniform surface roughness.

3.16 Changing sun direction. Row 1: original sun direction; Rows 2-3: New sun directions. Notice the consistency in the shapes and appearances of the sunlit regions in the perspective insets.

3.17 Appearance decomposition enables several applications such as furniture insertion, and changing flooring and sun direction, making it a useful tool for virtual staging. Top row: a home office and a bedroom staged by inserting chair, table, bed, shelves, plant and other objects. Rows 2, 3, 4: a living room is staged with different furniture, multiple new flooring (dark wood, carpet), and is shown under two different sun directions. Notice the sun light on the surfaces and the blocking of specular reflection that add realism to the scene.

3.18 We insert furniture by applying the inverse rendering approach by Li et al. [74] to perspective crops (a) of our panoramas (b). This method is trained on perspective views with synthetically rendered data. We also insert furniture by using the HDR map and 3D layout as illumination (c). Both approaches are not able to faithfully render the shading interactions of the objects with sunlight or specular reflections (see ours in (d)).
List of Tables

2.1 Comparison with baselines and NPP-Net variants for NPP completion and the metrics are evaluated only in unknown regions. The best and second-best results (excluding variants) are highlighted in bold and underline respectively. NPP-Net outperforms all the baselines on NRTDB and DTD. While Facade has some non-NPP images, NPP-Net can still outperform all other baselines except for Lama. See the supplementary for the results tested on the full images. 23

3.1 Ablation study for lighting effects detection on real data. mIoU (%) for specular and sunlight regions are reported. Our full method performs better than the others, showing the effectiveness of the GAN-based design. 46

3.2 Ablation study for specular removal on synthetic panoramas. Note that specular reflections are not strongly visible in all images of the dataset and they are sparse when visible. Hence, we also report the metrics on a subset of images with strong reflections. Although "no adversarial loss" performs better on "all testdata", it does not reach the level of the full method on strong reflections, which significantly affect the visual quality. 47

3.3 Quantitative analysis for specular reflection detection on real data. mIoU (%) for specular regions are reported. "Ours" is inferred in panoramic domain, while the others are inferred in perspective domain. All methods are evaluated in both domains (cross domain via cropping or merging). JSHDR [29] performs better than the other relevant methods, possibly because of a smaller domain gap. 48

3.4 Quantitative analysis for specular reflection removal on synthetic data. "Ours" and "IBCLN (re-train)" are inferred in panoramic domain, while the other methods are inferred in perspective domain. All methods are evaluated in both domains (cross domain via cropping or merging). To bridge the domain gap, we re-train IBCLN [65] on our data. Although performance is improved, it does not reach the level of our method. 50
Chapter 1

Introduction

The visual appearance of our world can be factorized into several formation components. Perceiving scenes from different perspectives, one can decompose their visual appearances into completely different components.

Considering a building facade in Figure 1.1. We can decompose it based on its structure: structural elements (windows) are arranged based on a typical spatial arrangement. Figure 1.1: A scene of building facade. We can decompose its visual appearance into two components based on its structure: (1) structural elements: a motif (window) that occurs repeatedly in a scene. The yellow box visualizes one example. (2) spatial arrangement: how the motif tiles in a scene. In this scene, it refers to the periodicity, showing how can we tile the content in the yellow box to obtain the one in the purple box.
1. Introduction

arrangement (periodicity). We can also consider a scene in Figure 1.2. This scene can be factorized based on its physical attributes: ambient and shading image (similar but not the same as intrinsic image decomposition [84]).

Inspired by the idea of decomposition, instead of directly handling RGB image [10, 66, 97, 160], in this thesis we take advantage of the appearance decomposition to help solve challenging image synthesis tasks. As we will show in two image synthesis examples, the appearance decomposition can serve as domain knowledge to either help the network training or incorporate traditional graphics pipeline for image synthesis.

In chapter 2, we leverage structural appearance decomposition for synthesizing Near-Periodic Patterns (NPP). In particular, NPP are ubiquitous in man-made scenes and are composed of tiled motifs with appearance differences caused by lighting, defects, or design elements. A good NPP representation is useful for many applications including image completion, segmentation, and geometric remapping. But representing NPP is challenging because it needs to maintain global consistency (tiled motifs layout) while preserving local variations (appearance differences). Methods trained on general scenes using a large dataset [66, 91, 119, 146, 147] or single-image optimization [5, 111, 127] struggle to satisfy these constraints, while methods that explicitly model periodicity are not robust to periodicity detection errors. To address these challenges, we learn a neural implicit representation using a coordinate-based

![Figure 1.2: A scene of an indoor empty home (left). We can decompose its visual appearance into two components based on its physical properties: (1) shading component: an image (top right) contains two commonly presented high-frequency light effects, specular reflection and direct sunlight, in an indoor home. (2) ambient component: an image (bottom right) without these two light effects.](image-url)
MLP with single image optimization. We design an input feature warping module and a periodicity-guided patch loss to handle both global consistency and local variations. To further improve the robustness, we introduce a periodicity proposal module to search and use multiple candidate periodicities in our pipeline. We demonstrate the effectiveness of our method on more than 500 images of building facades, friezes, wallpapers, ground, and Mondrian patterns in single and multi-planar scenes.

In chapter 3, we explore intrinsic appearance decomposition for indoor virtual staging. We describe a novel approach to decompose a single panorama of an empty indoor environment into four appearance components: specular, direct sunlight, diffuse and diffuse ambient without direct sunlight. Our system is weakly supervised by automatically generated semantic maps (with floor, wall, ceiling, lamp, window, and door labels) that have shown success on perspective views and are trained for panoramas using transfer learning without any further annotations. A GAN-based approach supervised by coarse information obtained from the semantic map extracts specular reflection and direct sunlight regions on the floor and walls. These lighting effects are removed via a similar GAN-based approach and a semantic-aware inpainting step. The appearance decomposition enables multiple applications including sun direction estimation, virtual furniture insertion, floor material replacement, and sun direction change, providing an effective tool for virtual home staging. We demonstrate the effectiveness of our approach on a large and recently released dataset of panoramas of empty homes.
1. Introduction
Chapter 2

Learning Continuous Implicit Representation for Near-Periodic Patterns

2.1 Introduction

Patterns are all around us and help us understand our visual world. In the 1990s, a human preattentive vision experiment [101] showed that periodicity is a crucial factor in high-level pattern perception. But most real patterns are not composed of perfectly periodic (tiled) motifs. Consider the commonly occurring real-world building facade scene in Figure 2.1 (a). While the windows are laid out periodically, they vary in their individual appearances. There are several design elements (borders, texture), shading variations or obstructions (tree, car, street lamp) that are not periodic. These factors make it challenging to create a good computational representation for such “Near-Periodic Patterns” (NPP).

A good NPP representation must preserve both global consistency (similar motifs layout) and local variations (different appearances). For global consistency, the distance (periods) and orientations between adjacent motifs should be accurate (e.g., window layout). At the same time, the local details in the scene should be fully encoded (e.g., appearance variations in windows or the horizontal design strips). In this paper, we present a novel method to learn such a representation that can be used for applications such as image completion (our main focus), segmentation of periodic parts, and resolution enhanced scene remapping, e.g., transforming to a fronto-parallel view (see supplementary).
2. Learning Continuous Implicit Representation for Near-Periodic Patterns

![Image](a) A NPP Scene  
(b) Input  
(c) BPI [68]  
(d) Ours

Figure 2.1: Inpainting to remove the tree, street lamp, and car from a near-periodic patterned (NPP) scene in (a). Input image (b) visualizes the mask (white unknown region) and detected (but inaccurate) NPP representation (yellow lattice). Guided by this inaccurate representation, the state of the art method BPI [68] (c) fails to generate windows occluded by the tree (orange arrow) and the white strip across the bottom (green arrow). Our NPP-Net (d) maintains global consistency and local variations, while preserving the known regions.

Existing image completion works applicable for NPP can be classified into two categories. The first category does not explicitly consider knowledge of periodicity. They complete images by training on large datasets [66, 91, 119, 146, 147] or by exploring single image statistics [5, 111, 127]. However, these methods fail to generate good global consistency, especially with large unknown mask inside (interpolation) or outside (extrapolation) the image border, or severe perspective effect. The other category [46, 68, 69, 87] models periodicity as prior for image completion. These works extract explicit NPP representations (e.g., displacement vectors) and use them to guide image completion. These methods can generate good global periodic structure if the estimated periodicity is accurate. However, this is hard to achieve in the presence of strong local variations.

Our work is inspired by the recent progress on implicit neural representations [90] that map image coordinates to RGB values using coordinate-based multi-layer perceptrons (MLP). But, naively using this method fails on our task due to the lack of a good periodicity prior. Thus, we present a periodicity-aware coordinate-based MLP to learn a continuous implicit neural representation, which we call NPP-Net for short. The key idea is to extract periodicity information from a partially observed NPP scene and inject it into both the MLP input and the loss function to help optimize NPP representation.
Three novel steps are proposed for the above idea: (1) The \textit{Periodicity Proposal} step extracts periodicity in the form of a set of candidate periods and orientations that are used together to handle inaccurate detections; (2) The \textit{Periodicity-Aware Input Warping} injects periodicity into the MLP input by warping input coordinates according to the proposed periodicities. This step preserves global consistency and the MLP converges to a good periodic pattern easily; (3) Finally, the \textit{Periodicity-Guided Patch Loss} samples observable patches according to periodicity to optimize the representation. This step preserves local variations, improves extrapolation ability, and removes high-frequency artifacts.

Our approach only requires a single image for optimization. This is important since there are no large dedicated NPP datasets. Thus, we evaluate our approach on a total of 532 NPP subclasses chosen from three datasets [1, 21, 124]. The scenes include building facades, friezes, ground patterns, wallpapers, and Mondrian patterns that are tiled on one or more geometric planes and perspectively warped. Our dataset is larger than those (157 at most) used in previous works [46, 68, 69, 87] that are designed for NPP. We mainly apply NPP-Net for the image completion task, but extend to resolution enhanced NPP remapping and NPP segmentation in the supplementary. We compare NPP-Net with four traditional [5, 26, 46, 68] and five deep learning-based methods [111, 119, 127, 146, 147], and eight variants of NPP-Net. Experiments show that NPP-Net can interpolate and extrapolate images, in-paint large and arbitrarily shaped regions, recover blurry regions when images are remapped, segment periodic and non-periodic regions, in planar and multi-planar scenes. Figure 2.1 shows the effectiveness of NPP-Net, inpainting a complex NPP scene, compared to the state of the art BPI [68]. While our method is not designed for general scenes, it is a useful tool to understand a large class of man-made scenes with near-periodic patterns.

2.2 Related Work

2.2.1 Near-Periodic Patterns Completion

There are two types of image completion methods that can be applied to NPP. The first type of methods [16, 66, 91, 119, 132, 146, 147, 160] do not explicitly consider periodicity as prior for completion. The second stream of methods takes advantage
of periodicity to guide the completion. We focus on reviewing the second type of methods.

The first stage for these methods is to obtain an NPP representation for guidance. Existing methods aim to represent NPP by detecting the global periodicity despite local variations. The types of NPP arrangements vary [46, 77, 96, 99, 100, 139] but commonly, periodic patterns are assumed to form a 2D lattice [40, 63, 69, 78, 81, 94, 95]. The first lattice-based work [40] for periodicity detection without human interaction finds correspondences using visual similarity and geometric consistency. Liu et al. [78] improve this process by incorporating generalized PatchMatch [6] and Markov Random Field. Furthermore, Lettry et al. [63] detect a repeated pattern model by searching in the feature space of a pre-trained CNN. Recently, Li et al. [69] design a compact strategy by searching on deep feature space without any implicit models. But it requires hyperparameter tuning to achieve competitive results. All existing methods describe periodicity using an explicit representation such as keypoints [40, 78, 100, 125], feature-based motifs [99] or displacement vectors [63, 69]. But they do not preserve both global consistency and local variations well.

The second stage is to generate or inpaint an NPP image guided by the NPP representation [39, 46, 68, 69, 80, 82, 83, 87]. One common assumption is that the NPP lie on a single plane. Liu et al. [82] synthesize an NPP image through multi-model deformation fields given an input NPP patch and its representation. Mao et al. [87] propose GAN-based NPP generation. Huang et al. [46] and BPI [68] extend image completion to the multi-plane case. They detect periodicities in these planes [41, 69, 86] and use them to guide image completion, based on [138] and [5]. Unlike earlier work Huang et al., BPI uses the periodicity detection method based on feature maps extracted from a pre-trained network. Also, the state of the art BPI’s image completion step does not use their prior GAN-based method [87].

In summary, the above methods assume that NPP representation is good enough for guidance, which is not guaranteed. By contrast, we merge the two stages by optimizing the implicit representation using image reconstruction error.
2.2.2 Implicit Neural Representations:

Recently, coordinate-based multi-layer perceptron (MLP) has been used to obtain implicit neural representation (INR). It maps coordinates to various signals such as shapes [20, 34, 111], scenes [88, 90] and images [12, 18, 111]. Mildenhall et al. [90] represent a 3D scene from a sparse set of views for novel view synthesis. Siren [111] replaces ReLU by a periodic activation function and designs an initialization scheme for modeling finer details. Chen et al. [18] present a Local Implicit Image Function for the generation of arbitrary resolutions. Skorokhodov et al. [112] design a decoder based on INR with GAN training, for high-quality image generation.

NPP-Net differs from previous methods in two ways: (1) Directly using MLP [90, 111] fails to learn accurate NPP representation without high-level structural understanding. We propose a periodicity-aware MLP. (2) Many works require a large dataset for training, while we optimize on a single image.

2.3 NPP-Net

We aim to build an MLP that maps image coordinates to pixel values, given a partial observation of an NPP image. We will describe NPP-Net using the image completion task. The unknown (masked) region is completed (or inpainted) by training on the remainder of the NPP image. For clarity, we first describe the method for single planar NPP scene and pre-warp the image to be fronto-parallel [154]. Then we will extend NPP-Net to handle multi-planar scenes.

Our key idea is to extract periodicity information from the known NPP region and inject it into the MLP input and the loss function. The initial pipeline of NPP-Net (Figure 2.2) consists of three modules: (1) Periodicity-Aware Input Warping transforms image coordinates using the detected periodicity. (2) Coordinate-based MLP maps the transformed coordinates to the corresponding RGB value. (3) Single Image Optimization provides a periodicity-guided loss function for optimizing the MLP on a single image.
2. Learning Continuous Implicit Representation for Near-Periodic Patterns

Figure 2.2: Initial pipeline of NPP-Net consists of three modules. (1) *Periodicity-Aware Input Warping* (pink) warps input coordinates using detected periodicity. (2) *Coordinate-based MLP* (blue) maps warped and input coordinate features to an RGB value. (3) *Single Image Optimization* (yellow) uses pixel loss and periodicity-guided patch loss on a single NPP image. Final pipeline in Section 2.3.4 shows how multiple periodicities are automatically detected and utilized.

### 2.3.1 Periodicity-Aware Input Warping

A traditional MLP is not good at capturing global periodic structure without additional priors. In fact, previous works [141, 161] have shown that a traditional MLP is unable to extrapolate a 1D periodic signal even with many training samples. The Periodicity-Aware Input Warping module thus explicitly injects periodicity information into the MLP by warping image coordinates \((x, y)\).

Assuming a 2D lattice arrangement, the periodicity is represented as two displacement vectors \(d_1\) and \(d_2\) (orange arrows in Figure 2.2). A perfect infinite periodic pattern is invariant if shifted by \(\alpha d_1 + \beta d_2 (\alpha, \beta \in \mathbb{Z})\). This representation can be transformed into periods and orientations, visualized as the magnitudes and orientations of the red arrows (\(p_1\) and \(p_2\), called *periodicity vectors*). Mathematically, the transformation is obtained by solving \(p_1 \cdot d_2 = d_1 \cdot p_2 = 0\) and \(p_1 \times d_2 = d_1 \times p_2 = d_1 \times d_2\), where the cross product is defined using the corresponding 3D vectors on the plane \(z = 0\). A periodicity is then defined as a vector pair \((p_1, p_2)\).
2. Learning Continuous Implicit Representation for Near-Periodic Patterns

One way to obtain the periodicity for an NPP image is to treat it as a learnable parameter and jointly optimize it with NPP-Net. However, this is hard for two reasons. (1) Good periodicity is not unique (any multiple works). (2) Different parts of an NPP may have different periodicities, leading to a complicated cost function. Thus we adopt an existing periodicity detection method [69] for input warping, which extracts feature maps from a pretrained CNN and searches brute-force to obtain a periodicity. Then, for each periodicity vector $p = (p \cos \theta, p \sin \theta) \in \{p_1, p_2\}$, we define a warp as a bivariate function:

$$f_p(x, y) = (x \cos \theta + y \sin \theta) \mod p.$$  \hspace{1cm} (2.1)

This function generates a warped coordinate value sampled from a periodic pattern with period $|p|$ along direction $\frac{p}{|p|}$, as shown in Figure 2.2. Through this feature engineering, the warped coordinates explicitly encode the periodicity information. The warped coordinates $f_{p_1}(x, y)$ and $f_{p_2}(x, y)$, together with the original coordinates $x$ and $y$, are further normalized to $[-1, 1]$ and passed through positional encoding [90] to allow the network to model high frequency signals [122]. The encoded coordinates are then input to the MLP. We keep the notations of coordinates before and after positional encoding the same for simplicity. The dimension of the features is $4d$, including $d$ frequencies in the positional encoding, and a set of four values for each frequency: $x, y, f_{p_1}(x, y)$, and $f_{p_2}(x, y)$.

### 2.3.2 Coordinate-Based MLP

We adopt coordinate-based MLP to represent NPP images. It is more effective and compact than a CNN to model periodic signals since coordinates are naturally suited for encoding positional (periodic) information. Specifically, we input the warped coordinate features to enforce global consistency, and also input the original coordinate features without warping to help preserve local variations. The output of the MLP is an RGB value corresponding to the input image coordinate. Since ReLU activation function has been proven to be ineffective to extrapolate periodic signals [141], we use the more suitable SNAKE function [161].
2. Learning Continuous Implicit Representation for Near-Periodic Patterns

2.3.3 Single Image Optimization

Pixel Loss:

Pixel loss is the most intuitive way to optimize coordinate-based MLP [90], which compares predicted and ground truth pixel values. For image coordinate $\mathbf{x} = (x, y)$, we adopt the robust loss function $\mathcal{L}_{proj}$ [8], given by:

$$
\mathcal{L}_{pixel}(\mathbf{x}) = \mathcal{L}_{rob}(\hat{C}(\mathbf{x}), C(\mathbf{x})),
$$

where $\hat{C}(\mathbf{x})$ and $C(\mathbf{x})$ are the output RGB values of the MLP and the ground truth RGB values from the input image at position $\mathbf{x}$, respectively. This loss is applied only to the known regions.

But merely adopting pixel loss like NeRF [90] fails to generate a good NPP for two reasons: (1) The high-dimensional input features result in the generation of high-frequency artifacts (Figure 2.3 (b)). See [122, 156] for details about this problem. (2) Pixel loss does not enforce explicit constraints to model the correlation between a coordinate’s features and its neighbors. This constraint is critical for preserving local variations since it helps capture local patch statistics. Thus, pixel loss fails to preserve local variations. For example, in Figure 2.3 (b), pixel loss generates some periodic artifacts in the top non-periodic region\(^1\).

Periodicity-Guided Patch Loss:

To address the limitations of pixel loss, we force the network to learn patch internal statistics by incorporating patch loss, which compares predicted and ground truth patches. The ground truth patches can be sampled at the same position as the predicted patch (for known regions), or sampled according to periodicity (for any region).

**GT Patch at the Same Position:** For a predicted patch in the known region, the ground truth patch at the same position is available. Specifically, for a square patch with size $s$ centered at position $\mathbf{x}$, we input all the pixel coordinates in the patch into MLP to obtain a predicted RGB patch $\hat{I}_s(\mathbf{x})$. Let $I_s(\mathbf{x})$ be the corresponding ground truth at the same position and $M_s(\mathbf{x})$ be the mask of known pixels. We apply

\(^1\)Figure 2.3 is generated based on the final pipeline explained in Section 2.3.4.
Learning Continuous Implicit Representation for Near-Periodic Patterns

perceptual loss \[151\] on masked patches:

$$L_p(x) = L_{pct}(\hat{I}_s(x) \odot M_s(x), I_s(x) \odot M_s(x)),$$ \hspace{1cm} (2.3)

where \(\odot\) is the element-wise product.

**GT Patches Sampled Based on Periodicity:** To train on unknown regions, we propose to sample ground truth patches based on periodicity. This is an effective way to handle the MLP extrapolation problem, which cannot be solved by merely using input warping and SNAKE activation function \[161\]. The input and output images in Figure 2.2 illustrate this sampling strategy.

Specifically, we sample multiple nearby ground truth patches for supervision by shifting position \(x\) based on the estimated periodicity. The shifted patch center is defined as \(x_{\alpha\beta} = x + \alpha d_1 + \beta d_2 (\alpha, \beta \in \mathbb{Z})\), where \(d_1\) and \(d_2\) are the displacement vectors. Because the predicted and ground truth patches are not necessarily aligned, we adopt contextual loss \(L_{ctx}\) \[89\]:

$$L_{ctx}(x) = \frac{1}{N} \sum_{(\alpha, \beta) \in T_N} L_{ctx}(\hat{I}_s(x_{\alpha\beta}) \odot M_s(x_{\alpha\beta}), I_s(x_{\alpha\beta}) \odot M_s(x_{\alpha\beta})),$$ \hspace{1cm} (2.4)

where \(T_N\) is a set of \((\alpha, \beta)\) pairs corresponding to the \(N\) nearest ground truth patches, since local variations are preserved using nearby patches for supervision.

**Patch Loss:** Our patch loss combines the two sampling strategies: \(L_{patch}(x) = \lambda_p \gamma(x) L_p(x) + \lambda_c L_c(x)\), where \(\lambda_p\) and \(\lambda_c\) are constant weights. \(\gamma(x)\) is a binary function: 1 for \(x\) in known regions and 0 for \(x\) in unknown regions.

**Total Loss:** Our final loss is the combination of patch loss and pixel loss:

$$L = \frac{\lambda_1}{|B_1|} \sum_{x \in B_1} L_{pixel}(x) + \frac{\lambda_2}{|B_2|} \sum_{x \in B_2} L_{patch}(x),$$ \hspace{1cm} (2.5)

where \(\lambda_1\) and \(\lambda_2\) are constant weights. \(B_1\) contains pixel coordinates that are randomly sampled in known areas, and \(B_2\) contains the center coordinates sampled in both known and unknown areas in proportion.

Training with this loss preserves both global consistency and local variations, as shown in Figure 2.3 (e). In fact, only using patch loss cannot ensure global consistency if the detected periodicity is not accurate enough. In Figure 2.3 (c),
2. Learning Continuous Implicit Representation for Near-Periodic Patterns

Figure 2.3: Comparing different losses based on the final pipeline. The red, yellow and cyan dots in (a) visualize the Top-3 periodicities. Zoom-ins of the unknown area (white rectangle) are in (b)-(f). Merely using pixel loss (b) generates high-frequency artifacts across the image and periodic artifacts in the top part. Adopting only patch loss (c) removes the artifacts but has poor global structure. Using pixel loss and patch loss with random sampling (d) cannot preserve global consistency and local variations well since the ground truth patches are not sampled according to periodicity and might be far from the predicted patch. With pixel loss and periodicity-guided patch loss, NPP-Net (e) solves these issues.

The pattern structure is poorly reconstructed because it only focuses on the local structure. We also show the result for the combination of pixel loss and patch loss with random sampling strategy in Figure 2.3 (d). This fails to generate correct periodic patterns and good local details because the output and sampled patches have a large misalignment and are far away from each other.

2.3.4 Periodicity Proposals

Although the above initial pipeline shows good performance, it still fails to handle very inaccurate periodicity detection. To improve the robustness of NPP-Net, we design a Periodicity Proposal module to provide additional periodicity information.
Figure 2.4: Final pipeline of NPP-Net modifies two modules of the initial pipeline. (1) Periodicity Proposal (green) automatically searches and augments the input periodicity to handle inaccurate periodicity detection and encourage the global consistency. (2) Coordinate-based MLP (lavender blue) has two branches: (a) for Top-1 periodicity and original coordinates, and (b) for the rest.

As shown in Figure 2.4, we first search multiple candidate periodicities and then augment the input to MLP to handle inaccurate periodicity detection.

**Periodicity Searching:** Our searching strategy is based on the same periodicity detection method [69] we adopt in the initial pipeline. But the authors’ original implementation requires manual hyperparameter tuning. Instead, we design an automatic tuning method, which evaluates each candidate periodicity (obtained from various hyperparameters) in the context of image completion. Specifically, we first generate $M$ pseudo masks in the known regions and treat them as unknown masks for image completion. Then we execute the initial pipeline for each candidate periodicity, and compute its reconstruction error in pseudo mask regions for periodicity ranking. Since we focus on reconstructing a coarse global structure, we use a lightweight initial NPP-Net without patch loss for efficiency, which takes around 10 seconds for each periodicity in a Titan Xp GPU.

**Periodicity Augmentation:** Like our initial pipeline, existing methods [46, 68, 69, 87] also use one periodicity to guide NPP completion. This cannot guarantee global consistency if the periodicity is inaccurate, especially when the unknown mask is large (see Figure 2.9). So, we augment the pattern periodicity at two levels to improve robustness. At the coarse level, instead of searching the best periodicity, we keep Top-K candidates $\{(p_{i1}, p_{i2}) | i \in \mathbb{Z}^+, i \leq K\}$, to cover multiple possible solutions. This coarse-level augmentation encourages NPP-Net to move towards the most reasonable candidate periodicity. At the fine level, we augment periodicities with small offsets to better handle smaller errors. Specifically, a periodicity vector $p$ is augmented to $A(p) = \{p + \delta \frac{p_i}{|p|} | \delta \in \Delta\}$. We empirically define $\Delta = \{0, \pm0.5, \pm1\}$ (in pixels). Finally, we merge all the augmented periodicity vectors as
\( P = \bigcup_{i \in \mathbb{Z}^+, i \leq K} A(p_{i1}) \cup A(p_{i2}) \). Note that \( |P| = 2K|\Delta| \).

In our final pipeline, \( P \) contains \( K|\Delta| \) periodicities. We perform input warping (Section 2.3.1) for each periodicity and input the transformed coordinate features into the MLP. We add an additional branch to the MLP, as shown in Figure 2.4. Since the Top-1 periodicity is likely the most accurate (see Figure 2.9), we input the coordinate features warped using the Top-1 periodicity (including fine-level augmentation) and original coordinate features, to the first branch. The coordinate features warped using the Top-2 to K periodicities are sent to the second branch. For optimization, we sample patches according to the Top-1 periodicity. All other parts remain the same in our final pipeline. We evaluate these changes in our ablation study.

### 2.3.5 Extensions

#### Non-NPP region segmentation

Parts of a scene may not be near-periodic (e.g., trees in front of a building facade). Thus we aim to segment the non-periodic regions in an NPP image in an unsupervised manner. The key idea is to detect initial non-periodic regions, treat them as the unknown mask, and relabel regions with low reconstruction error as periodic regions.

Specifically, we first apply a traditional image segmentation method [50] to generate an initial segmentation for non-periodic regions, treated as unknown regions. This segmentation method first divides image pixels as superpixels and uses Gaussian Mixture Model for a coarse segmentation, which is further refined by GraphCut. We treat the class that contains the largest number of pixels as the periodic class, and the rest of the classes as non-periodic classes for initialization. One drawback of this method is that it often over-segments the non-periodic regions. Thus we use NPP-Net to refine the initial segmentation. In detail, we use the same pipeline of image completion to complete the unknown (non-periodic) regions. For every pixel \( x \) in non-periodic regions, we can compute the reconstruction error using two metrics since the ground-truth value of \( x \) is known. The first metric is the L1 distance, comparing the difference between output and ground truth pixels, given by:

\[
S_1(x) = |\hat{G}(x) - G(x)|,  \tag{2.6}
\]
where $\hat{G}(x)$ and $G(x)$ are the grayscale value of output and ground truth at $x$, respectively. The second metric is perceptual distance [151] based on pretrained network, given by:

$$S_2(x) = ||P_I(x) - \hat{P}_I(x)||_2,$$

(2.7)

where $P_I$ and $\hat{P}_I$ denote the normalized perceptual activation image (first layer of AlexNet) of the output and ground truth NPP image, respectively. Only $x$ with low reconstruction error is eligible for relabelling, and these $x$ are defined as a set $S$, given by:

$$S = \{x | S_1(x) < \epsilon_1, S_2(x) < \epsilon_2\},$$

(2.8)

where $\epsilon_1$ and $\epsilon_2$ are constants. Finally, we relabel $x \in S$ to periodic class to obtain our segmentation. For implementation details, we set $\epsilon_1 = 0.15$ and $\epsilon_2 = 0.3$. Also, we blur the input NPP image before using our pipeline to remove fine details in the image because we only focus on the global structure.

**NPP remapping**

NPP textures are usually adopted in various applications such as rendering. A perspective camera can be used to obtain real-world NPP texture by rectifying the captured NPP perspective image. However, the potential blurry texture issue prevents us to obtain high-quality texture. Consider an NPP captured by a perspective camera in a tilted angle, as shown in Figure 2.5 (a). The far-away motifs are blurry because the regions are out of focus. This problem becomes severe after rectification due to the remapping issue, shown in Figure 2.5 (b). The goal of NPP remapping is to recover the blurry regions (caused by image remapping errors) in the NPP images. It outputs clear NPP images by preserving local variations and global structure, shown in Figure 2.5 (c).

The key idea is to detect the blurry regions from the input and treat them the same as the unknown regions in the completion task with minor modification. Specifically, we first detect the blurry regions using [115], treated as unknown regions. This prevents the patch loss from sampling ground truth patches in blurry regions for supervision. Simply treating it as a completion problem cannot preserve the local variations (e.g., lighting) in the blurry regions, thus we modify the pixel loss to account for this issue. Instead of computing this loss only in known (clear) regions,
2. Learning Continuous Implicit Representation for Near-Periodic Patterns

(a) Perspective NPP  (b) Rectified NPP  (c) Output

Figure 2.5: Illustration of an NPP Remapping scenario.

we also compute the loss on unknown (blurry) regions, given by:

\[
L_{\text{pixel}}(x) = M(x)L_{\text{rob}}(\hat{C}(x), C(x)) + \sigma(1 - M(x))L_{\text{rob}}(\hat{C}(x), C(x)),
\]

(2.9)

where \(\sigma\) is a constant weight. \(M(x)\) is 1 if \(x\) in clear regions, and 0 in blurry regions.

We keep the remaining part the same as completion to optimize NPP-Net. For implementation details, we set \(\sigma = 0.3\).

**Multi-Plane NPP completion**

NPP-Net can be extended to multi-planar scenes. The key idea is to automatically segment and rectify each plane, apply NPP-Net on each plane for completion, and project the completed image back to the original image.

Given a masked image (Figure 2.6 (a)) with different NPPs on different planes, we first adopt a pre-trained plane segmentation network [145] to obtain a coarse plane segmentation. Since this network is not trained on images with masks, we first use our “No Periodicity” variant to inpaint the masked (unknown) regions. The output is shown in Figure 2.6 (b). This process takes about 30 seconds. Then we can input this inpainted image in (b) to the pre-trained network to generate the coarse plane segmentation in Figure 2.6 (c).
For each segmented plane, we automatically select a bounding box using a similar strategy as the pseudo mask generation in periodicity searching (Sec 2.3.6). This bounding box is utilized as a reference to rectify the plane using TILT [154]. Thus we do not require accurate segmentation since it is only used for bounding box selection. Figure 2.6 (c) visualizes bounding boxes for two plane, and Figure 2.6 (d) shows the rectified planes.

For each rectified plane, we detect the Top-3 periodicity (Figure 2.6 (d)) only using the rectified bounding box regions to remove the influence of other planes. Then we perform NPP segmentation to segment the non-periodic regions (mainly from other planes) and treat them as invalid pixels. In this case, the initial non-periodic regions are defined as all image regions excluding the bounding box regions. The segmentation results are shown in Figure 2.6 (e). After that, we run NPP completion on each plane, and the results are in Figure 2.6 (f).

Finally, we transform all completed planes back to the original image coordinate system and recompose the image. The final inpainted and ground truth images are in Figure 2.6 (g) and (h) respectively.

**Rotated NPP**

NPP-Net can be modified to handle rotated NPP. Define periodicity of 2D rotated NPP by a rotation centroid \( \mathbf{c} = (c_x, c_y) \) and an angular period \( p \) (in radians). The periodicity proposal module can be applied to rotated NPP to estimate \( \mathbf{c} \) and \( p \). In addition, two modifications are needed:

1. Equation 2.1 is rewritten as two functions:

   \[
   f_{c,p}(x, y) = \sqrt{(x - c_x)^2 + (y - c_y)^2},
   \]

   \[
   g_{c,p}(x, y) = \text{atan2}^*(y - c_y, x - c_x) \mod p,
   \]

   where we define \( \text{atan2}^*(y, x) = (\text{atan2}(y, x) + 2\pi) \mod 2\pi \).

2. Periodicity-based patch sampling strategy is modified. Let center of predicted patch be \( \mathbf{x} \). Instead of shifting \( \mathbf{x} \) to obtain centers of known patch \( \mathbf{x}_{\alpha\beta} \) in translated NPP, we directly rotated the input image around \( \mathbf{c} \) based on \( \alpha \cdot p \), where \( \alpha \) is an integer constant. Then we sampled known patch centers in rotated images at position
2.3.6 Implementation Details

We apply the following settings below (for the final pipeline) for all applications. Same as [90], we set the number of frequency $d$ in positional encoding as 10.

**Periodicity Proposal.** In our implementation, the key hyperparameter for periodicity detection method [69] is an integer $q$, which defines the range of possible
displacements in $d_1$ and $d_2$. Specifically, $d_1$ and $d_2$ are in the set $R$, given by:

$$R = \{(x, y)|-\frac{W}{q} < x < \frac{W}{q}, 0 \leq y < \frac{H}{q}\} - \{(x, y)|-\frac{W}{q+1} < x < \frac{W}{q+1}, 0 \leq y < \frac{H}{q+1}\},$$

where $H$ and $W$ are the image height and width. The periodicity detection method outputs the best periodicity in $R$.

In the periodicity searching, we evaluate different $q$ in $\{i|i \in \mathbb{Z}^+, i < 10\}$ for multiple periodicities, which are ranked based on reconstruction errors. Specifically, we generate Top-M (M=3) pseudo square masks whose centers are far away from the image boundary or unknown regions. The mask size for each mask is empirically set to $\frac{5L}{\sqrt{2}}$, where $L$ is the distance from the center pixel to the nearest invalid pixel. Then we run the initial pipeline for evaluation of each $q$ based on its reconstruction error in these masked regions.

**Network Architecture.** The first MLP contains 9 fully-connected Snake layers [161] with 512 channels. It also includes a skip connection that concatenates the Top-1 coordinate and original features to the fifth layer’s activation. The outputs of this MLP are concatenated with the Top 2nd to K-th coordinate features and then fed to the second MLP. The second MLP has 4 fully-connected Snake layers with 512, 512, 256, and 3 channels. The output features of the first MLP are also concatenated with the second layer activation of the second MLP for the skip connection.

**Single-Image Optimization.** In the patch loss, we sample square ground truth patches for supervision. We set the patch size $s$ in $\{64, 96, 128, 160\}$ based on the Top-1 periodicity, where larger periods are matched with a larger patch size. For the patch center $x$, we filter out the shifted patch centers $x_{\alpha\beta}$ with their known patch area smaller than 70% of the whole patch area, and we use $N = 3$ (size of $T_N$). We set $\lambda_p$ in $L_{patch}$ to 0.4, and $\lambda_c$ to 1, 10, 5 in completion, remapping and segmentation (explained later), respectively. In final Loss $L$, we set $\lambda_1 = 1$, $\lambda_2 = 0.001$, $|B_1| = 8192$, and $|B_2| = 2$. Further, $B_2$ contains half of sampled centers in unknown regions and half of those in known regions.

For training details, we use the Adam optimizer with the learning rate that starts with $5 \times 10^{-4}$ and decays every 500 epochs. Furthermore, for every 2000 epoch, we shrink the patch size by twice and increase the number of the sampled patches by twice to encourage the network to focus on finer-level details.
2. Learning Continuous Implicit Representation for Near-Periodic Patterns

**Runtime.** The periodicity proposal takes 100 seconds (for 10 candidate periodic- ities). The optimization takes around 8 minutes, 5 minutes, and 3 minutes for NPP completion, NPP remapping, and NPP segmentation to converge on a single NVIDIA Titan XP GPU, respectively.

2.4 Experiments

2.4.1 Dataset

We evaluate NPP-Net on 532 images selected from three relevant datasets for NPP completion: PSU Near-Regular Texture Database (NRTDB) [1], Describable Textures Dataset (DTD) [21], and Facade Dataset [124]. There are 165 NPP images in the NRTDB dataset including facades, friezes, bricks, fences, grounds, Mondrian images, wallpapers, and carpets. Similarly, there are 258 NPP images in the DTD Dataset including honeycombs, grids, meshes and dots. The Facade Dataset has 109 rectified images of facades. Some of these facades are strictly not NPP because often the windows are not arranged periodically. But nonetheless we include these to evaluate our approach when the NPP assumption is not strictly satisfied. Finally, we also collect a small dataset with 11 NPP images for real-world applications (e.g., removing trees in the scene). In general, scenes in NRTDB are more challenging than DTD since they contain more non-periodic regions (boundaries, trees, sky, etc.), complex illuminations and backgrounds, and multiple periodicities across an image (Figure 2.7 row 3). We use TILT [154] to rectify all the images to be fronto-parallel if needed.

2.4.2 Metrics

No single metric can evaluate NPP image completion comprehensively. So we adopt three metrics to cover different scales, including LPIPS (perceptual distance) [151], SSIM [134], and PSNR. Lower LPIPS, higher SSIM, and higher PSNR mean better performance. A known limitation for SSIM and PSNR is that blurry images also tend to receive high scores in these metrics [62], while LPIPS handles this issue better.
### 2.3.3 Ablation Study

We perform three studies. First, we compare to a “No Periodicity” variant, which uses a standard coordinated-based MLP without a periodicity prior. Results in Table 2.1 show that it fails to understand the arrangement of tiled motifs. Note that the Facade dataset may have different performances because it has some non-NPP images. NPP-Net outperforms all the baselines on NRTDB and DTD. While Facade has some non-NPP images, NPP-Net can still outperform all other baselines except for Lama. See the supplementary for the results tested on the full images.

### Table 2.1: Comparison with baselines and NPP-Net variants for NPP completion and the metrics are evaluated only in unknown regions. The best and second-best results (excluding variants) are highlighted in bold and underline respectively. NPP-Net outperforms all the baselines on NRTDB and DTD. While Facade has some non-NPP images, NPP-Net can still outperform all other baselines except for Lama. See the supplementary for the results tested on the full images.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>LPIPS ↓</td>
<td>SSIM ↑</td>
<td>PSNR ↑</td>
</tr>
<tr>
<td>Large Datasets</td>
<td>PEN-Net [146]</td>
<td>0.497</td>
<td>0.452</td>
<td>17.97</td>
</tr>
<tr>
<td></td>
<td>ProFill [147]</td>
<td>0.401</td>
<td>0.300</td>
<td>16.35</td>
</tr>
<tr>
<td></td>
<td>Lama [119]</td>
<td>0.196</td>
<td>0.551</td>
<td>18.64</td>
</tr>
<tr>
<td>Single Image</td>
<td>Image Quilting [26]</td>
<td>0.428</td>
<td>0.674</td>
<td>13.25</td>
</tr>
<tr>
<td></td>
<td>PatchMatch [5]</td>
<td>0.263</td>
<td>0.542</td>
<td>18.14</td>
</tr>
<tr>
<td></td>
<td>DIP [127]</td>
<td>0.554</td>
<td>0.292</td>
<td>16.46</td>
</tr>
<tr>
<td></td>
<td>Siren [111]</td>
<td>0.636</td>
<td>0.084</td>
<td>14.38</td>
</tr>
<tr>
<td></td>
<td>Huang et al. [46]</td>
<td>0.287</td>
<td>0.410</td>
<td>16.99</td>
</tr>
<tr>
<td></td>
<td>BPI [68]</td>
<td>0.254</td>
<td>0.442</td>
<td>16.86</td>
</tr>
<tr>
<td>NPP-Net Variants</td>
<td>No Periodicity</td>
<td>0.429</td>
<td>0.449</td>
<td>18.22</td>
</tr>
<tr>
<td></td>
<td>Pixel Only</td>
<td>0.308</td>
<td>0.618</td>
<td>20.20</td>
</tr>
<tr>
<td></td>
<td>Patch Only</td>
<td>0.322</td>
<td>0.340</td>
<td>15.83</td>
</tr>
<tr>
<td></td>
<td>Pixel + Random</td>
<td>0.216</td>
<td>0.670</td>
<td>20.74</td>
</tr>
<tr>
<td></td>
<td>Initial Pipeline</td>
<td>0.213</td>
<td>0.647</td>
<td>20.38</td>
</tr>
<tr>
<td></td>
<td>Top1 + Offsets</td>
<td>0.205</td>
<td>0.656</td>
<td>20.60</td>
</tr>
<tr>
<td></td>
<td>Top5 + Offsets</td>
<td>0.197</td>
<td>0.661</td>
<td>20.73</td>
</tr>
<tr>
<td></td>
<td>Top3 w/o Offsets</td>
<td>0.210</td>
<td>0.648</td>
<td>20.44</td>
</tr>
<tr>
<td></td>
<td>Top3 + Offsets</td>
<td>0.188</td>
<td>0.679</td>
<td>21.01</td>
</tr>
</tbody>
</table>

NPP-Net | Top3 + Offsets | 0.188 | 0.679 | 21.01 | 0.249 | 0.504 | 18.32 | 0.263 | 0.485 | 15.93

23
without offsets. Table 2.1 shows that the initial pipeline performs the worst. Larger \( K \) (Top-5) may hurt the performance as more inaccurate periodicities may be included. But, smaller \( K \) (Top-1) also performs badly because the correct periodicity may not be included. A suitable \( K \) (Top-3) without offsets performs worse since offsets can help better handle smaller errors. With the appropriate \( K \) and offsets, NPP-Net generates the best results.

### 2.4.4 Baselines

We compare against non-periodicity-guided and periodicity-guided baselines. For the former, we select two traditional methods, Image Quilting [26] and PatchMatch [5], that can handle pattern structure locally for some scenes with properly selected patch size. We then consider two learning-based methods DIP [127] (CNN-based) and Siren [111] (MLP-based) trained on a single image for inpainting. We also choose several learning-based methods: PEN-Net [146], ProFill [147] and Lama [119] that are trained on large real-world datasets [21, 50, 52, 158] since they show competitive NPP completion examples in their work.

For periodicity-guided methods, we choose two baselines - Huang et al. [46] and BPI [68]. Both works were designed for multi-planar scenes, but can be used for single-plane completion as well. BPI first segments and rectifies planes, then performs periodicity detection [69] on each plane, and inpaints each plane independently. For fair comparison in single-plane NPP image, we only compare with BPI’s completion step to remove potential inconsistency introduced from other steps (e.g., plane rectification). For Huang et al., we use their pipeline without modification as their method works directly for a single plane and the completion step cannot be easily separated out. We will also compare with these two methods in multi-plane NPP images.

### 2.4.5 Comparison with Baselines:

Table 2.1 shows the quantitative results for all methods. For NRTDB and DTD datasets, among the single-image baselines, BPI obtains the almost best LPIPS because it generates a more reasonable global structure guided by periodicity. PatchMatch obtains better SSIM and PSNR even if it generates blurred results for large
masks. Lama achieves the best results among the baselines since it adopts fast Fourier convolution for a larger image receptive field, which allows it to implicitly learn the underlying periodicity from large datasets. Our NPP-Net outperforms all baselines on these two datasets by optimizing only on a single image. For the Facade dataset, even if some non-NPP images are included, NPP-Net performs better than other baselines (except for Lama). Lama effectively learns scene prior from large datasets and thus works well for non-NPP images, leading to the best performance in this dataset.

Qualitative results are shown in Figure 2.7. Large rectangle masks are challenging since there is less information from which to estimate the representation. Perceptually, PatchMatch works well when the motifs are small (row 3) but results in blur with large masks (row 2 and 6). Although BPI and Huang et al. perform better than non-periodicity-guided baselines, they generate artifacts since the NPP representation (periodicity) has poor global consistency (row 1-6) or lacks local variations (row 6-9). Note that the Top-1 periodicity in row 1 (red dots) is inaccurate - the actual periods are half of the one shown. We show that NPP-Net can extrapolate NPP images well (row 5), generalize to irregular masks (row 6), and work for scenes that contain non-periodic regions (row 4). We show that NPP-Net can be extended to handle multi-planar scenes in Figure 2.8. Among the baselines, Lama (trained on large datasets) can better handle local variations (row 2 purple box). Although it captures some global structure when the mask is not large, Lama performs worse than BPI when the mask is outside the image border for extrapolation (all rows) and perspective effect is severe (row 2 cyan box). Learning from the Top-K periodicities, NPP-Net produces the best images, maintaining global consistency and local variations.

2.4.6 Influence of Mask Size:

We study the influence of different mask sizes for image completion in NRTDB. First, for each image, we compute the error of Top-3 periodicities and pick the one with the smallest error. Then for each proposed periodicity, we show the number of images picked across the dataset with different mask sizes in Figure 2.9 (left). To compute the periodicity error, we manually annotate the periodicity with the smallest period as ground truth periodicity for the dataset. For each periodicity, we generate a 2D
2. Learning Continuous Implicit Representation for Near-Periodic Patterns

Figure 2.7: Qualitative results for NPP completion. We show four baselines that operate on a single image. The red, yellow, and cyan dots in input images show the first, second, and third periodicity from periodicity searching module, respectively. For visualization, all periods are scaled by 2. Our NPP-Net outperforms all baselines for global consistency (rows 1-6) and local variations (rows 6-9).
2. Learning Continuous Implicit Representation for Near-Periodic Patterns

Figure 2.8: Qualitative comparison for multi-plane NPP completion. We show three baselines, which are either designed for multi-plane NPP scenes (Huang et al. and BPI) or trained on large datasets (Lama). Some zoom-in boxes are resized for visualization.

point cloud, defined as \( \{ \alpha d_1 + \beta d_2 | \alpha, \beta \in \mathbb{Z} \} \). We also filter points that are out of image range. The periodicity error is calculated using the average L2 distance between every point in proposed point clouds and its nearest neighbor in the ground truth point cloud (one-directional chamfer distance). In Figure 2.9 (left), while the Top 1st periodicity is the best one for most of the images with small mask size, the number decreases when the mask is large (64% of the image). This illustrates that the other two periodicities contain better periodicity and leveraging them by our periodicity augmentation strategy can be helpful for learning NPP representation, especially when the mask is large. Second, we show the LPIPS performances for different mask sizes in Figure 2.9 (right). In particular, we filter out images that contain large non-periodic regions. When the mask area is small (4% of the image), PatchMatch slightly outperforms NPP-Net because the unknown regions may not contain pattern structure, and simply sampling nearby patches is sufficient to produce good results. Among single-image methods, Huang et al., BPI and NPP-Net perform better when the mask size increases since they are guided by periodicity. Taking better advantage of periodicity information, NPP-Net is more robust to various mask sizes, especially for larger masks.
2. Learning Continuous Implicit Representation for Near-Periodic Patterns

Figure 2.9: **Left:** The number of NPP images for each periodicity that has the smallest periodicity errors (among Top-3) with different mask sizes in NRTDB. As the mask size grows, the best periodicity emerges in the Top 2nd and 3rd periodicity, thus utilizing them in NPP-Net is useful. **Right:** The LPIPS results (lower is better) for different mask sizes in NRTDB. PatchMatch performs the best when masks are very small but NPP-Net outperforms all the baselines for large masks.

### 2.5 Limitation

NPP-Net has two limitations: (1) The periodicity proposals cannot be too erroneous, allowing tolerance of about 10%. (2) It assumes a multi-planar scene with translated, circular, and potentially other types of symmetrical NPP that can be modeled.

### 2.6 Conclusion

In conclusion, we show how to learn an effective implicit neural representation of Near-Periodic Patterns. We design the periodicity proposal, periodicity-aware input warping, periodicity-guided patch loss to maintain global consistency and local variations. We compare NPP-Net with nine baselines and eight variants on three datasets to demonstrate its effectiveness. We believe that NPP-Net is a strong tool to understand a large class of man-made scenes.
Chapter 3

Semantically Supervised Appearance Decomposition for Virtual Staging from a Single Panorama

3.1 Introduction

With the current pandemic-related restrictions and many working from home, there is a significant uptick in home sales. Remote home shopping is becoming more popular and effective tools to facilitate virtual home tours are much needed. One such tool is virtual staging: how would furniture fit in a home and what does it look like if certain settings (e.g., sun direction, flooring) are changed? To provide effective visualization for staging, panoramas are increasingly being used to showcase homes. Panoramas provide surround information but methods designed for limited-FOV perspective photos cannot be directly applied.

However, inserting virtual furniture into a single panoramic image of a room in a photo-realistic manner is hard. Multiple complex shading effects should be taken into consideration. Such complex interactions are shown in Fig. 3.1; the insertion induces effects such as occlusion, sunlight cast on objects, partially shadowed specular reflections, and soft and hard shadows. This task is challenging due to the lack of ground truth training data. The ground truth annotations for appearance decomposition tasks either require significant human labor [11], or specialized devices in a controlled environment [37, 106], which is hard to be extended to a large scale. Previous approaches [28, 74, 76] rely on synthetic data for supervised training. However, there is the issue of domain shift and the costs of designing and rendering
3. Semantically Supervised Appearance Decomposition for Virtual Staging from a Single Panorama

Figure 3.1: A single panorama of an empty indoor environment (top left) is decomposed into four appearance components on the floor and walls: specular and direct sunlight (bottom left), diffuse (not shown) and diffuse ambient without direct sunlight (bottom right). Although the decomposition is not perfect (e.g. power sockets are removed in the bottom right), we demonstrate its effectiveness in enabling high quality virtual staging applications including furniture insertion\(^1\) (top right), changing flooring and sun direction (Sec. 3.7). Multiple realistic effects are rendered, including shadows under the table, sunlight on the side table, sofa pillows, beanbag, dining table and chaise lounge, occlusion of specular reflection by the sofa and chair. The plant creates soft and hard shadows by blocking the skylight and the sunlight. On the other hand, inserting furniture without appearance decomposition fails to reproduce these effects (Fig. 3.3).

We present a novel approach to insert virtual objects into a single panoramic image of an empty room in a near-photo-realistic way. Instead of solving a general inverse rendering problem from a single image, we identify and focus on two ubiquitous shading effects that are important for visual realism: interactions of inserted objects with (1) specular reflections and (2) direct sunlight (see Fig. 3.1). This is still a challenging problem that has no effective solutions because of the same reasons mentioned above. We observe that while ground truth real data for these effects is hard to obtain, by contrast, it is easier to obtain semantics automatically, given recent advances on layout estimation \([98, 118, 162]\) and semantic segmentation \([17, 85, 131]\). An automatically generated semantic map with ceiling, wall, floor, window, door, and lamp classes is used to localize the effects coarsely. These coarse localizations are used to supervise a GAN based approach to decompose the input image into four appearance components:

\[\text{specular and direct sunlight, diffuse, diffuse ambient without direct sunlight}\]
(1) diffuse, (2) specular (on floor), (3) direct sunlight (on floor and walls), and (4) diffuse ambient without sunlight. The appearance decompositions can then be used for several applications including insertion of furniture, changing flooring, albedo estimation and estimating and changing sun direction.

We evaluate our method on ZInD [22], a large dataset of panoramas of empty real homes. The homes include a variety of floorings (tile, carpet, wood), diverse configurations of doors, windows, indoor light source types and positions, and outdoor illumination (cloudy, sunny). We analyze our approach by comparing against ground truth specular and sunlight locations and sun directions, and by conducting ablation studies. Most previous approaches for diffuse-specular separation and inverse rendering were not designed for the setting used in this work to enable direct apples-to-apples comparisons; they require perspective views or supervised training of large scale real or rendered data. But we nonetheless show performances of such methods as empirical observations and not necessarily to prove that our approach is better in their settings.

Our work also has several limitations: (1) we assume the two shading effects occur either on the planar floor or walls, (2) our methods detect for mid-to-high frequency shading effects but not subtle low-frequency effects or specular/sunlight interreflections, and (3) our methods can induce artifacts if the computed semantic map is erroneous. Despite these limitations, our results suggest that the approach can be an effective and useful tool for virtual staging. Extending the semantics to furniture can enable appearance decomposition of already-furnished homes.

To sum up, our key idea is using easier-to-collect and annotate discrete signals (layout/windows/lamps) to estimate harder-to-collect continuous signals (specular/sunlight). This is a general idea that can be useful for any appearance estimation task. Our contributions include: (1) A semantically and automatically supervised framework for locating specular and direct sunlight effects (Sec. 3.3). (2) A GAN-based appearance separation method for diffuse, specular, ambient, and direct sunlight component estimation (Sec. 3.4 and 3.5). (3) Demonstration of multiple virtual staging applications including furniture insertion and changing flooring and sun direction (Sec. 3.7). To our knowledge, we are the first to estimate direct sunlight and sun direction from indoor images. The overall pipeline is illustrated in Fig. 3.2. Our code is released at: https://github.com/tiancheng-zhi/pano_decomp.
Figure 3.2: Our system consists of three modules: *Coarse Effects Localization* (Sec. 3.3) coarsely localizes lighting effects using automatically generated semantics; *Lighting Effects Detection* (Sec. 3.4) separates specular reflection on the floor and direct sunlight on the floor and the wall (effects are brightened for visualization); *Lighting Effects Removal* (Sec. 3.5) removes the detected specular reflection and direct sunlight, outputting a diffuse image (no specular) and an ambient image (no specular and sunlight). The specular reflection, direct sunlight, diffuse image, and ambient image can be used for various virtual staging applications.

### 3.2 Related Work

**Inverse Rendering.** The goal of inverse rendering is to estimate various physical attributes of a scene (*e.g.* geometry, material properties and illumination) given one or more images. Intrinsic image decomposition estimates reflectance and shading layers [9, 48, 71, 72, 84, 123]. Other methods attempt to recover scene attributes with simplified assumptions. Methods [7, 13, 57, 72] for a single object use priors like depth-normal consistency and shape continuity. Some methods [103, 109] use priors of a particular category (*e.g.*, no occlusion for faces). Some works assume near-planar surfaces [2, 23, 24, 30, 45, 73]. In addition, human assistance with calibration or annotation is studied for general scenes [53, 144].

Data-driven methods require large amounts of annotated data, usually synthetic images [67, 74, 75, 135]. The domain gap can be reduced by fine-tuning on real images using self-supervised training via differentiable rendering [104]. The differentiable rendering can be used to optimize a single object [55, 150]. Recent works [14, 15] extend NeRF [90] to appearance decomposition. The combination of Bayesian framework and deep networks is explored by [19] for reflectance and illumination.
3. Semantically Supervised Appearance Decomposition for Virtual Staging from a Single Panorama

Figure 3.3: Object insertion result without appearance decomposition. Note that the direct sunlight on the objects, hard shadows in the sunlight region, and blocked specular reflections are all missing (compare to Fig. 3.1).

estimation. In our work, we model complex illumination effects on real 360° panoramas of empty homes. Similar to [53], we believe that discrete semantic elements (like layout, windows, lamps, etc.) are easier to collect and train good models for. By contrast, diffuse and specular annotations are continuous spatially varying signals that are harder to label.

Illumination Estimation. Many approaches represent indoor lighting using HDR maps (or its spherical harmonics). Some estimate lighting from a single LDR panorama [27, 36], a perspective image [31, 113], a stereo pair [114], or object appearance [35, 93, 136]. Recent approaches [33, 74, 135] extend this representation to multiple positions, enabling spatially-varying estimation. Others [32, 49, 54] estimate parametric lighting by modeling the position, shape, and intensity of light sources. Zhang et al. [148] combine both representations and estimate a HDR map together with parametric light sources. However, windows are treated as the source of diffuse skylight without considering directional sunlight. We handle the spatially-varying high-frequency sun illumination effects, which is usually a challenging case for most methods.

Some techniques estimate outdoor lighting from outdoor images. Early methods [59, 60] use analytical models to describe the sun and sky. Liu et al. [79] estimates sun direction using 3D object models. Recently, deep learning methods [42, 43, 149]
regress the sun/sky model parameters or outdoor HDR maps by training on large scale datasets. A recent work [120] estimates high-frequency illumination from shadows. However, they use outdoor images as input, where the occlusion of the sunlight by interior walls is not as significant as that for indoor scenes.

**Specular Reflection Removal.** There are two main classes of specular reflection removal techniques. One removes specular highlights on objects. Non-learning based approaches usually exploit appearance or statistical priors to separate specular reflection, including chromaticity-based models [3, 105, 121, 142], low-rank model [38], and dark channel prior [56]. Recently, data-driven methods [29, 107, 140] train deep networks in a supervised manner. Shi et al. [107] train a CNN model using their proposed object-centric synthetic dataset. Fu et al. [29] present a large real-world dataset for highlight removal, and introduce a multi-task network to detect and remove specular reflection. However, the reflection on floors is more complex than highlights, because it may reflect window textures and occupy a large region.

The second class removes reflections from a glass surface in front of the scene. Classical methods use image priors to solve this ill-posed problem, including gradient sparsity [4, 64], smoothness priors [70, 128], and ghosting cues [108]. Recently, deep learning has been used for this task [25, 28, 44, 65, 129, 137, 153] and achieved significant improvements by carefully designing network architectures. Li et al. [65] develop an iterative boost convolutional LSTM with a residual reconstruction loss for single image reflection removal. Also, Hong et al. [44] propose a two-stage network to explicitly address the content ambiguity for reflection removal in panoramas. However, they mostly use supervised training, requiring large amounts of data with ground truth. Most previous works of both classes do not specifically consider the properties of panoramic floor images.

Our task is in-between these classes, because the reflections are on the floor rather than on glass surfaces but the appearance is similar to that of glass reflection because the floor is flat. To our knowledge, this scenario has not been well studied.
Virtual Staging Services. Some companies provide virtual staging services including Styldod\textsuperscript{3}, Stuccco\textsuperscript{4}, and PadStyler\textsuperscript{5}. Users can upload photos of their rooms and the company will furnish them and provide the rendering results. However, most of them are not free and we are not able to know whether the specular reflections and direct sunlight are handled automatically. Certain applications like changing sun direction are usually not supported. By contrast, our approach is released for free use.

3.3 Semantics for Coarse Localization of Lighting Effects

We focus on two ubiquitous shading effects that are important for virtual staging: (1) specular reflections and (2) direct sunlight. These effects are illustrated in Fig. 3.4. Dominant specular reflections in empty indoor environments are somewhat sparse and typically due to sources such as lamps, and open or transparent windows and doors. The sunlight streaming through windows and doors causes bright and often high-frequency shading on the floor and walls. These effects must be located and removed before rendering their interactions with new objects in the environment. However, these effects are hard to annotate in large datasets to provide direct supervision for training.

Our key observation is that these effects can be localized (at least coarsely) using the semantics of the scene with simple geometrical reasoning. For example, the locations of windows and other light sources (like lamps) constrain where specular reflections occur. The sunlit areas are constrained by the sun direction and locations of windows. Our key insight is that semantic segmentation provides an easier, discrete supervisory signal for which there are already substantial human annotations and good pre-trained models \cite{61, 131}. In this section, we describe how to automatically compute semantic panoramic maps and use them to localize these effects coarsely (Fig. 3.5(b) and (e)). These coarse estimates are used to supervise a GAN-based approach to refine the locations and extract these effects (Fig. 3.5 (c) and (f)).

\textsuperscript{3}https://www.styldod.com/
\textsuperscript{4}https://stuccco.com/
\textsuperscript{5}https://www.padstyler.com/
3. Semantically Supervised Appearance Decomposition for Virtual Staging from a Single Panorama

![Diagram showing lighting effects](image)

(a) Specular Reflection  
(b) Direct Sunlight

Figure 3.4: The geometry of lighting effects considered in this work. (a) For specular reflections, when the camera is upright, the light source point (transparent or open window, door, or indoor lamp), the reflection point, and the camera center lie on a vertical plane (shown as gray), corresponding to a column in the panorama. (b) For direct sunlight, the sun direction establishes the mapping between a window (or door) point and a floor point illuminated by direct sunlight. The sunlit floor area can be back-projected to the window according to the sun direction.

### 3.3.1 Transfer Learning for Semantic Segmentation

We define seven semantic classes that are of interest to our applications: floor, ceiling, wall, window, door, lamp, and other (see Fig. 3.5(b) for an example). Most works for semantic segmentation are designed for perspective views. Merging perspective semantics to panorama is non-trivial with problems of view selection, inconsistent overlapping regions, and limited FoV. A few networks [116, 117] are designed for panoramic images but building and training such networks for our dataset requires annotations and significant engineering effort, which we wish to avoid. Thus, we propose to use a perspective-to-panoramic transfer learning technique, as follows: we obtain an HRNet model [131] pre-trained on a perspective image dataset ADE20K [159] (with labels adjusted to our setting) and treat it as the “Teacher Model”. Then we use the same model and weights to initialize a “Student Model”, and adapt it for panoramic image segmentation.

To supervise the Student Model, we sample perspective views from the panorama. Let $I_{pano}$ be the original panorama, $\Phi$ be the sampling operator, $f_{tea}$ be the Teacher
Figure 3.5: A semantic map with 7 classes is computed automatically from a panorama. The map is used to obtain coarse lighting effect masks exploiting the geometric constraints in Fig. 3.4. These coarse masks are used to supervise a GAN-based method to obtain accurate specular and sunlit images.

3.3.2 Coarse Localization of Lighting Effects

Coarse Specular Mask Generation. The panoramic geometry dictates that the possible specular area, on the floor, is in the same columns as the light sources. This assumes that the camera is upright and that the floor is a mirror. See Fig. 3.4 for an
3. Semantically Supervised Appearance Decomposition for Virtual Staging from a Single Panorama

illustration. Thus, we simply treat floor pixels, that are in the same columns as the light sources (windows and lamps in the semantic map), as the coarse specular mask. To handle rough floors, we dilate the mask by a few pixels. See Fig. 3.5(c) for an example coarse specular mask.

**Coarse Sunlight Mask Generation.** We first estimate a coarse sun direction and then project the window mask to the floor and walls according to this direction. Since sunlight enters the room through windows (or open doors), the floor texture can be projected back onto the window region based on the sun direction. This projected floor area must be bright and a score is estimated as the intensity difference between window region and projected floor. The projection requires room geometry which can be obtained by calculating wall-floor boundary from the semantic map. Basically, we assume an upright camera at height=1 and the floor plane at Z=-1. Then, the 3D location (XYZ) of a wall boundary point is computed using known $\theta - \phi$ from the panorama, Z=-1, and cartesian-to-spherical coordinate transform, which can be used to calculate the wall planes.

For a conservative estimate of the floor mask, we use the K-best sun directions. Specifically, the sun direction is given by $d = (\theta, \phi)$, where $\theta$ is the elevation angle and $\phi$ is the azimuth angle. We do a brute-force search for $\theta$ and $\phi$ with step size $\theta = 5^\circ$. Let $s(\theta, \phi)$ be the matching score. First, top-K elevation angles $\theta_1, ..., \theta_K$ are selected based on score $s'(\theta) = \max_\phi s(\theta, \phi)$. Let $\phi^*(\theta) = \arg\max_\phi s(\theta, \phi)$ be the optimal azimuth angle for elevation $\theta$. Then the selected sun directions are $(\theta_1, \phi^*(\theta_1)), ..., (\theta_K, \phi^*(\theta_K))$. In practice, we set $K = 5$. Fig. 3.5 (e) for an example coarse direct sunlight mask.

3.4 GAN-Based Lighting Effects Detection

Both the coarse masks estimated above are rough conservative estimates. In this section, we will demonstrate that this inaccurate information can still be used as a supervision signal that can effectively constrain and guide an accurate estimation of specular reflection and sunlit regions. We sketch the key idea in Sec. 3.4.1, and explain the detailed loss function in Sec 3.4.2.
3.4.1 Key Idea and Overall Architecture

Simply training a lighting effects network with the coarse masks will result in the network predicting only the coarse masks. How do we design a network that is able to produce more accurate masks? Fig. 3.6 illustrates our architecture that augments the lighting effects network with three local discriminators. The lighting effects network takes a single panoramic image \( I \) as an input, and predicts the specular \( (I_{\text{spec}}) \) and sunlight \( (I_{\text{sunl}}) \) components. The first local discriminator takes the image with specular component removed \( (I - I_{\text{spec}}) \) as an input and tries to locate the specular reflection via per-pixel binary classification. Its output is denoted by \( D_{\text{spec}} \). It is supervised by a coarse specular mask \( M_{\text{spec}} \) obtained from semantics described in Sec. 3.3.2. If the specular prediction \( I_{\text{spec}} \) is good, the discriminator should fail to locate specular reflection on image \( I - I_{\text{spec}} \). Hence, the lighting effects networks should try to fool the discriminator. This idea can be seen as a per-pixel GAN treating the pixels inside the coarse mask as fake examples and the pixels outside the mask as real examples. This approach is also applied to sunlight regions, where the sunlight discriminator output is denoted by \( D_{\text{sunl}} \) and the coarse sunlight mask is \( M_{\text{sunl}} \).

Because artifacts could appear when specular reflection and direct sunlight overlap with each other if the intensities are not predicted accurately, we include a third discriminator trained to detect overlapping specular reflection and direct sunlight on the image with both effects removed \( (I - I_{\text{spec}} - I_{\text{sunl}}) \). Its prediction is denoted by \( D_{\text{over}} \). This discriminator is supervised by the intersection of the coarse specular mask and the coarse sunlight mask, represented as \( M_{\text{spec}} \odot M_{\text{sunl}} \), where \( \odot \) is the element-wise product.

3.4.2 Loss Function

Sparsity Loss. We observe that, in an empty room, specular reflection usually appears on the floor, and direct sunlight usually appears on the floor and the walls. Thus, we define the region of interest (ROI) for specular reflection to be the floor and the ROI for direct sunlight to be the floor and the walls. Different regions should have different levels of sparsity. Lighting effects should be sparser outside the ROI than inside the ROI. They should also be sparser outside the coarse mask than inside...
3. Semantically Supervised Appearance Decomposition for Virtual Staging from a Single Panorama

Figure 3.6: Architecture for lighting effects detection. The lighting effects network takes a panorama $I$ as input, and predicts the specular ($I_{\text{spec}}$) and sunlight ($I_{\text{sunl}}$) components. If the prediction is good, a local discriminator trying to locate specular regions should fail on the image with specular component removed ($I - I_{\text{spec}}$). This is the key supervision signal for training the lighting effects network. The local discriminator is supervised by coarse specular masks obtained from semantics. Similar techniques are applied to sunlight regions and regions with overlapping specular and direct sunlight.

the coarse mask. Thus, we apply different levels of sparsity to different regions. Let $I$ be the input image, $R$ be the binary ROI mask, and $M$ be the coarse mask inside the ROI. The sparsity loss is:

$$L_s = \lambda_{s1} || I \odot R ||_1 + \lambda_{s2} || I \odot R \odot (1 - M) ||_1 + \lambda_{s3} || I \odot M ||_1,$$

(3.1)

where, $\lambda_{s1} \geq \lambda_{s2} \geq \lambda_{s3}$ are constant weights, and $\odot$ is the element-wise product. We use such sparsity loss for specular reflection and direct sunlight separately, denoted by $L_{sp,s}$ and $L_{sl,s}$.

Adversarial Loss. As mentioned in Sec. 3.4.1, after removing the lighting effects, a discriminator should not be able to locate the effects. Consider specular reflection as an example. Let $I$ be the input image, and $I_{\text{spec}}$ be the estimated specular
reflection. We use a local discriminator to find reflection areas from \( I - I_{\text{spec}} \). The discriminator is supervised by coarse specular mask \( M_{\text{spec}} \). The discriminator loss uses pixel-wise binary cross entropy, \( L_{sp,d} = f_{ce}(D_{\text{spec}}, M_{\text{spec}}) \). To fool the discriminator, an adversarial loss:

\[
L_{sp,a} = f_{ce}(D_{\text{spec}}, 1 - M_{\text{spec}})
\]  

(3.2)
is applied to region \( M_{\text{spec}} \). Similarly, we build adversarial losses for sunlight regions and regions with overlapping specular reflection and direct sunlight, denoted by \( L_{sl,a} \) and \( L_{ol,a} \), respectively.

**Total Loss.** The total loss \( L \) is the sum of the two sparsity losses and the three adversarial losses:

\[
L = \lambda_{sp,s} L_{sp,s} + \lambda_{sl,s} L_{sl,s} + \lambda_{sp,a} L_{sp,a} + \lambda_{sl,a} L_{sl,a} + \lambda_{ol,a} L_{ol,a},
\]  

(3.3)
where \( \lambda_{sp,s}, \lambda_{sl,s}, \lambda_{sp,a}, \lambda_{sl,a}, \lambda_{ol,a} \) are constant weights.

### 3.5 Lighting Effects Removal

The previous architecture and method focused on estimating the specular and direct sunlight regions. But, naively subtracting the predicted regions from the original image, i.e. \( I - I_{\text{spec}}, I - I_{\text{spec}} - I_{\text{sunl}} \), may still produce artifacts as shown in Fig. 3.7. In this section, we instead directly estimate the diffuse image (without specular reflection) and the ambient image (without specular reflection or direct sunlight).

#### 3.5.1 GAN-Based Specular Reflection Removal

As shown in Fig. 3.8, a deep network is used to predict a diffuse image. To supervise the training of the network, we adopt the same GAN strategy as Sec. 3.4.2. Instead of using the coarse mask generated by Sec. 3.3.2, we obtain a fine specular mask by thresholding the estimated specular reflection for discriminator supervision.

Let \( I \) be the original image, \( I_{\text{spec}} \) be the specular image, \( I_{\text{diff}} \) be the diffuse image, and \( I_{\text{recon}} = I_{\text{diff}} + I_{\text{spec}} \) be the reconstructed image. Inspired by [153], our loss \( L_{\text{diff}} \) consists of an adversarial loss \( L_{\text{adv}} \), a reconstruction loss \( L_{\text{recon}} \), a perceptual loss
Figure 3.7: Removing effects by naively subtracting the estimated lighting effects versus our approach. Naive subtraction \((I - I_{\text{spec}}, I - I_{\text{spec}} - I_{\text{sunl}})\) leaves lighting effects residues (specular/sunlight region is brighter than the surrounding, with visible boundaries). Our result removes these artifacts.

Figure 3.8: Architecture for specular reflection removal. The diffuse network takes a single panoramic image \(I\) as input, and predicts the diffuse component \(I_{\text{diff}}\). Similar to Fig. 3.6, a local discriminator trying to locate specular regions should fail on the diffuse image \(I_{\text{diff}}\). The discriminator is supervised by the fine specular mask obtained via thresholding and dilating the estimated specular component.
$L_{\text{perc}},$ and an exclusion loss $L_{\text{excl}}$:

$$L_{\text{diff}} = \lambda_a L_{\text{adv}} + \lambda_r L_{\text{recon}} + \lambda_p L_{\text{perc}} + \lambda_e L_{\text{excl}}, \quad (3.4)$$

where $\lambda_a, \lambda_r, \lambda_p, \lambda_e$ are constant weights.

**Adversarial Loss.** $L_{\text{adv}}$ is the same as the loss defined in Sec. 3.4.2, replacing the coarse specular mask by the fine specular mask.

**Reconstruction Loss.** $L_{\text{recon}} = \|I_{\text{recon}} - I\|_1$ calculates the L1 loss between the reconstructed image and the original image.

**Perceptual Loss.** Following [51] and [153], $L_{\text{perc}} = \|\Phi(I_{\text{recon}}) - \Phi(I)\|_1$ calculates the L1 distance between VGG features [110], where $\Phi$ is the VGG feature function. It is for enhancing perceptual quality.

**Exclusion Loss.** $L_{\text{excl}} = \|\tanh(|\nabla I_{\text{diff}}|) \odot \tanh(|\nabla I_{\text{spec}}|)\|_F,$ where $\|\cdot\|_F$ is the Frobenius norm, and $\odot$ is the element-wise product. This loss minimizes the correlation between diffuse and specular components in the gradient domain to prevent edges from appearing in both diffuse and specular images, following [153].

### 3.5.2 Semantic-Aware Inpainting for Sunlight Removal

A process similar to that used for diffuse image estimation can be used for predicting the ambient image without sunlight. In practice, however, bright sunlit regions are the highly dominant brightness component and are often saturated and the network does not predict the ambient image well. Thus, we use an inpainting approach called SESAME [92] that is semantic-aware and preserved the boundaries between different classes (wall versus floor, etc).
3.6 Experimental Analysis

3.6.1 Implementation Details

**Improving Lamp Semantics with HDR Estimation.** The ADE20K definition of lamps includes the whole lamp, while we care about the bright bulb. Thus, we estimate a HDR map via supervised learning to obtain a more accurate bulb segmentation. To obtain a HDR image $I_{hdr}$, we train U-Net [102] to predict the HDR-LDR residue in log space $I_{res} = \ln(1 + I_{hdr} - I_{ldr})$, using Laval [31] and HDRI Heaven \(^6\) datasets. A weighted L2 loss is adopted. We set batch size 32, learning rate $10^{-4}$ and epochs 300. Pixels in the lamp segmentation with $I_{hdr} > 2$ are kept.

**Improving Room Semantics with Layout Estimation.** The floor-wall and ceiling-wall boundaries predicted by semantic segmentation might not be straight in perspective views, causing visual artifacts. Thus, we adopt a layout estimation method LED\(^2\)-Net [130] to augment the semantic map. The layout estimation predicts floor, wall and ceiling regions for the current room, which is used to improve the floor-wall and ceiling-wall boundaries. We adopt batch size 6, learning rate $3 \times 10^{-4}$, and epochs 14.

**Super-Resolution in Post-Processing.** Due to limited computing, we run our methods at low-resolution (256 × 512). We enhance the resolution for applications requiring high resolution results with post-processing. We enhance the resolution of the specular image to 1024 × 2048 via a pre-trained super-resolution model [133]. To ensure the consistency between the diffuse image and the original image, we adopt Deep Guided Decoder [126] for super-resolution with the original image as guidance. The final resolution is 1024 × 2048. For the ambient image, we simply use super-resolution [133] and copy-paste the inpainted region on to the high resolution diffuse image. To preserve high-frequency details, we calculate the difference between the diffuse and ambient components as the high-resolution sunlight image.

**Deep Architecture and Hyper-parameters.** For semantic transfer learning, we use $w_{trans} = w_{reg} = 1$, batch size 4, learning rate $10^{-5}$, epochs 5, and random

\(^6\)https://hdrihaven.com/hdris/?c=indoor
perspective crops with FoV 80° to 120°, elevation -60° to 60°, azimuth 0° to 360°. We use U-Net [102] for Lighting Effects Network and Diffuse Network, and FCN [85] for discriminators, optimized with Adam [58]. See supplementary materials for network architectures. The networks use learning rate $10^{-4}$, weight decay $10^{-5}$, and batch size 16. For effects detection, $\lambda_{sp,s} = \lambda_{sl,s} = \lambda_{sp,a} = \lambda_{sl,a} = \lambda_{ol,a} = 1$. For specular sparsity, $\lambda_{s1} = 10, \lambda_{s2} = 3, \lambda_{s3} = 1$. For sunlight sparsity, $\lambda_{s1} = 5, \lambda_{s2} = \lambda_{s3} = 1$. For specular reflection removal, $\lambda_r = \lambda_p = \lambda_e = 1, \lambda_a = 10$. The networks take $\sim$2 days for training on 4 TITAN Xp GPUs and 0.05s for inference. Resolution enhancement using Guided Deep Decoder [126] converges in $\sim$30 min for a single image.

**Dataset.** We evaluate our approach on the ZInD [22] dataset that includes 54,034 panoramas for training and 13,414 panoramas for testing (we merge their validation and test sets). For better visualization, we inpaint the tripod using SESAME [92]. Sec. 3.6.2 and 3.6.3 evaluate our approach at $256 \times 512$ resolution. Sec. 3.6.4 visualizes results at resolution $1024 \times 2048$.

### 3.6.2 Ablation Study

**Perspective Merging vs. Panoramic Semantics.** We compare with merging perspective semantics to panorama, by sampling 14 perspective views and merging the semantics via per-pixel voting. On Structured3D [155] test set (mapped to our labels), the merging method achieves mIoU=44.8%, while ours achieves 68.1%. This shows developing methods directly is meaningful. We also test our method on different emptiness levels: No furniture mIoU: 68.1%; Some furniture: 66.6%; Full furniture: 60.6%. Since ZInD does not contain much furniture, this mIoU reduction is expected.

**Lighting Effects Detection.** We manually annotate specular and sunlight regions on 1,000 test panoramas sampled from the test set for quantitative analysis. The annotation is coarse because there is no clear boundary of specular regions. For a fair comparison, we report the mean IoU with respect to the best threshold for binarization for each method. The quantitative evaluation is performed on the floor region only.

To show that the GAN-based method introduces priors for refining the coarse
Table 3.1: Ablation study for lighting effects detection on real data. mIoU (%) for specular and sunlight regions are reported. Our full method performs better than the others, showing the effectiveness of the GAN-based design.

<table>
<thead>
<tr>
<th>Method</th>
<th>Specular ↑</th>
<th>Sunlight ↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coarse mask</td>
<td>6.7</td>
<td>3.0</td>
</tr>
<tr>
<td>Supervised by coarse mask w/o GAN</td>
<td>8.7</td>
<td>19.9</td>
</tr>
<tr>
<td>No overlap effects discriminator</td>
<td>34.6</td>
<td>30.0</td>
</tr>
<tr>
<td>Our full method</td>
<td><strong>38.9</strong></td>
<td><strong>47.2</strong></td>
</tr>
</tbody>
</table>

mask, we train a network with the same architecture as Lighting Effects Network in a standard supervised manner (without GAN), using the coarse mask as labels. We also compare with the coarse mask itself and the method without the third discriminator for overlap regions. Tab. 3.1 shows that the full method outperforms the others, demonstrating the effectiveness of our GAN-based approach with three discriminators.

Specular Reflection Removal. We use synthetic data for evaluating specular reflection removal. We render 1,000 images based on microfaucet reflectance model using the Mitsuba [47] renderer with floor textures from AdobeStock. We also define images with peak specular intensity ranking top 5% in the test data as “strong reflection”. PSNR (higher is better) and LPIPS [152] (lower is better) are reported.

We compare with the naive method by subtracting the specular component from the original image directly, and methods without each loss components. In Tab. 3.2, the full method outperforms the other variants on all metrics except for the ones without adversarial loss. Although the methods without adversarial loss achieves a high score on “all testdata”, it performs worse than the full method on “strong reflection”. Besides, as shown in Fig. 3.9, adversarial and exclusion losses help remove perceivable visual artifacts or specular residues although average metrics don’t reflect clearly. Thus, we adopt the full method with all four losses for the applications.

3.6.3 Performances of Other Relevant Approaches

Most previous approaches for diffuse-specular separation and inverse rendering were not designed for the setting used in this work to enable direct apples-to-apples
Table 3.2: Ablation study for specular removal on synthetic panoramas. Note that specular reflections are not strongly visible in all images of the dataset and they are sparse when visible. Hence, we also report the metrics on a subset of images with strong reflections. Although "no adversarial loss" performs better on "all testdata", it does not reach the level of the full method on strong reflections, which significantly affect the visual quality.

<table>
<thead>
<tr>
<th>Method</th>
<th>All Testdata</th>
<th>Strong Reflection</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR ↑</td>
<td>LPIPS ↓</td>
</tr>
<tr>
<td>Input subtracts specular image</td>
<td>33.2</td>
<td>0.0247</td>
</tr>
<tr>
<td>No adversarial loss</td>
<td>33.4</td>
<td>0.0233</td>
</tr>
<tr>
<td>No reconstruction loss</td>
<td>33.2</td>
<td>0.0248</td>
</tr>
<tr>
<td>No perceptual loss</td>
<td>31.3</td>
<td>0.0483</td>
</tr>
<tr>
<td>No exclusion loss</td>
<td>33.3</td>
<td>0.0242</td>
</tr>
<tr>
<td>No adversarial/reconstruction</td>
<td>33.4</td>
<td>0.0232</td>
</tr>
<tr>
<td>No adversarial/perceptual</td>
<td>33.4</td>
<td>0.0270</td>
</tr>
<tr>
<td>No adversarial/exclusion</td>
<td>33.3</td>
<td>0.0233</td>
</tr>
<tr>
<td>No reconstruction/perceptual</td>
<td>14.4</td>
<td>0.3595</td>
</tr>
<tr>
<td>No reconstruction/exclusion</td>
<td>33.2</td>
<td>0.0242</td>
</tr>
<tr>
<td>No perceptual/exclusion</td>
<td>32.4</td>
<td>0.0333</td>
</tr>
<tr>
<td>Our full method</td>
<td>33.3</td>
<td>0.0241</td>
</tr>
</tbody>
</table>

comparisons; they require perspective views or supervised (ground truth) training of large scale real or rendered data or are used for a related task. But accurately annotating large data for specular and sunlight effects is non-trivial, because of blurred boundaries or complex texture. The visual and quantitative evaluations below shows that previous approaches do not easily bridge the domain gap. These methods have strong value for the domains they were designed for but our approach provides an effective tool for virtual staging with real-world panoramas.

**Specular Reflection Detection.** We evaluate a bilateral filtering based method BF [142], two deep reflection removal methods IBCLN [65] and LASIRR [25], and a deep highlight removal method JSHDR [29]. Since these methods are designed for perspective images, we randomly sample 5,000 perspective crops from the 1,000 annotated panoramic images and report the performance in Tab. 3.3 left column. In the right column, we also evaluate in panoramic domain by converting the panorama to
3. Semantically Supervised Appearance Decomposition for Virtual Staging from a Single Panorama

Figure 3.9: Ablation study for specular reflection removal by not including adversarial, reconstruction, perceptual or exclusion losses. While quantitative metrics are averaged over the image and over the dataset, the visuals illustrate the subtle but noticeable residual specular reflections when the different losses are removed from our method.

Table 3.3: Quantitative analysis for specular reflection detection on real data. mIoU (%) for specular regions are reported. "Ours" is inferred in panoramic domain, while the others are inferred in perspective domain. All methods are evaluated in both domains (cross domain via cropping or merging). JSHDR [29] performs better than the other relevant methods, possibly because of a smaller domain gap.

<table>
<thead>
<tr>
<th>Method</th>
<th>Perspective mIoU ↑</th>
<th>Panoramic mIoU ↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>BF [142]</td>
<td>11.3</td>
<td>8.9</td>
</tr>
<tr>
<td>IBCLN [65]</td>
<td>15.7</td>
<td>15.0</td>
</tr>
<tr>
<td>LASIRR [25]</td>
<td>20.5</td>
<td>17.5</td>
</tr>
<tr>
<td>JSHDR [29]</td>
<td>21.4</td>
<td>14.1</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>41.8</strong></td>
<td><strong>38.9</strong></td>
</tr>
</tbody>
</table>

a cubemap, running the baseline methods, and merging the results back to panorama. Among the evaluated relevant methods, JSHDR [29] achieves the best performance, likely because it is designed for specular highlight removal, which is closer to our setting than glass reflection removal methods.
Direct Sunlight Detection. We are unaware of any work estimating direct sunlight for our setting. Thus, we evaluate an intrinsic image decomposition method USI3D [84]. We assume that the sunlight region should have a strong shading intensity and threshold the output shading image as the sunlight prediction. This achieves mean IoU 7.4% on perspective crops and 7.9% on panoramas from cubemap merging while our method achieves 47.2% on perspective crops and 47.2% on panoramas. We conclude that using the semantics and sun direction is important for sunlight estimation.

Specular Reflection Removal. Similar to lighting effect detection, we report performances based on 5,000 perspective crops sampled from the 1,000 synthetic panoramic images. Fig. 3.10 and the first four rows in Tab. 3.4 show the performances of pre-trained models on perspective images. Similar to specular reflection detection, JSHDR [29] achieves the best performance, likely due to smaller domain differences, since it is trained on a large-scale real dataset. A similar conclusion can be drawn from the evaluation in panoramic domain. We also attempted to bridge the domain gap for a more reasonable quantitative evaluation. For JSHDR, only the compiled code is available online, making it hard to adapt to our domain. Thus, we train IBCLN on our panoramic data. Specifically, let \( \{ I \} \) be the set of original images and \( \{ I_{\text{spec}} \} \) be the set of estimated specular components. We treat \( \{ I - I_{\text{spec}} \} \) and \( \{ I_{\text{spec}} \} \) as the transmission image set and the reflection image set, respectively. These two sets are then used for rendering the synthetic images for training. Tab. 3.4 Rows 5 and 11 show that by re-training using the synthetic data based on our specular reflection estimation, the performance of IBCLN improves but still does not reach the level of our method.

3.6.4 Diverse Lighting and High Resolution Results

Fig. 3.12 visualizes effects detection results in diverse lighting conditions, including large sunlight area and (partially) occluded light sources. The results show that our detection can handle such cases.

Fig. 3.11 visualizes high-resolution appearance decomposition results. The specular reflections are well removed and the floor texture is consistent with the original image.
Table 3.4: Quantitative analysis for specular reflection removal on synthetic data. "Ours" and "IBCLN (re-train)" are inferred in panoramic domain, while the other methods are inferred in perspective domain. All methods are evaluated in both domains (cross domain via cropping or merging). To bridge the domain gap, we re-train IBCLN [65] on our data. Although performance is improved, it does not reach the level of our method.

<table>
<thead>
<tr>
<th>Evaluation Domain</th>
<th>Method</th>
<th>All Testdata</th>
<th>Strong Reflection</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>PSNR ↑</td>
<td>LPIPS ↓</td>
</tr>
<tr>
<td>Perspective</td>
<td>BF [142]</td>
<td>23.4</td>
<td>0.1632</td>
</tr>
<tr>
<td></td>
<td>IBCLN [65]</td>
<td>27.5</td>
<td>0.1013</td>
</tr>
<tr>
<td></td>
<td>LASIR [25]</td>
<td>20.8</td>
<td>0.1984</td>
</tr>
<tr>
<td></td>
<td>JSHDR [29]</td>
<td>32.0</td>
<td>0.0354</td>
</tr>
<tr>
<td></td>
<td>IBCLN [65] (re-train)</td>
<td>33.4</td>
<td>0.0279</td>
</tr>
<tr>
<td></td>
<td><strong>Ours</strong></td>
<td><strong>34.5</strong></td>
<td><strong>0.0271</strong></td>
</tr>
<tr>
<td>Panoramic</td>
<td>BF [142]</td>
<td>22.1</td>
<td>0.1526</td>
</tr>
<tr>
<td></td>
<td>IBCLN [65]</td>
<td>28.8</td>
<td>0.0890</td>
</tr>
<tr>
<td></td>
<td>LASIR [25]</td>
<td>23.0</td>
<td>0.1425</td>
</tr>
<tr>
<td></td>
<td>JSHDR [29]</td>
<td>30.6</td>
<td>0.0698</td>
</tr>
<tr>
<td></td>
<td>IBCLN [65] (re-train)</td>
<td>32.3</td>
<td>0.0250</td>
</tr>
<tr>
<td></td>
<td><strong>Ours</strong></td>
<td><strong>33.3</strong></td>
<td><strong>0.0241</strong></td>
</tr>
</tbody>
</table>

The sunlight region is inpainted with the correct colors. Due to the limitation of the inpainting algorithm, the texture is not perfect. As inpainting algorithms improve, we can plug-and-play those within our pipeline.

### 3.7 Applications

**Improvement of Albedo Estimation.** Albedo estimation via intrinsic image decomposition suffers from the existence of specular reflection and direct sunlight. As shown in Fig. 3.13 (b), a pre-trained USI3D [84] incorrectly bakes specular reflection and direct sunlight into the albedo component. When we run the same algorithm on the ambient image after removing the lighting effects, the albedo estimation (c) is significantly better.

**Improvement of Sun Direction Estimation.** With the detected direct sunlight, we can improve the sun direction estimation proposed in Sec. 3.3.2. There are two modifications: (1) Instead of using the RGB image, we use the direct sunlight image
Figure 3.10: Qualitative results of specular reflection detection and removal. Specular reflections are brightened for visualization. Note that although the walls are shown, only the floor region is evaluated quantitatively. Our method successfully detects and removes the both window and lamp reflections. (d) removes lamp reflections but (b) and (c) do not. (b) (c) (d) leaves residues of window reflections on the floor.
3. Semantically Supervised Appearance Decomposition for Virtual Staging from a Single Panorama

Figure 3.11: High resolution appearance decomposition results. The lighting effects are successfully detected on a variety of floor types. For specular reflection removal, the floor texture in (c) is consistent with that in (a). Sunlight regions are inpainted with correct colors. Some floor textures are imperfect due to the limitation of the inpainting algorithm, e.g., the third row, where the camera tripod is inpainted in the original image.

walls for coarse sun direction estimation because walls are usually white, which may mislead the matching score.

To evaluate the sun direction, we obtained sun elevation angles for part of the dataset from the authors of ZInD [22]. The elevation angles are calculated based on timestamps and geolocations. The ground truth azimuth angles are not available because of unknown camera orientations. The elevation angles provided may not be always accurate because the timestamps are based on the time the data were uploaded to the cloud. Thus, we manually filter obviously incorrect values and select images with visible sunlight for evaluation. A total of 257 images are evaluated.

We search for the top-1 direction at step size 1°. Fig. 3.14 plots the histogram of
Figure 3.12: Diverse lighting conditions of testing images. Row 1: large sunlight region; Rows 2-3: effects from (partially) occluded light sources (see red boxes in (a)). Our method successfully detects the lighting effects.

the angular estimation error. More than half of the estimations are within 10° error. Fig. 3.14 also visualizes an example of projecting the window mask according to the estimated sun direction. Compared with the coarse sun direction, the improved fine estimation provides more accurate estimations.

Changing Floor Material. The decomposition is used to change flooring with texture and BRDF parameters. Examples include switching between wood (specular), carpet (diffuse) or tile (specular). Fig. 3.15 shows four examples, including wood-to-wood (keep specularity), wood-to-carpet (remove specularity), carpet-to-wood (render specularity via HDR map), and carpet-to-carpet changes. Direct sunlight are
3. Semantically Supervised Appearance Decomposition for Virtual Staging from a Single Panorama

![Original Image](image1)
![Albedo estimated from (a)](image2)
![Ambient Image](image3)
![Albedo estimated from (c)](image4)

Figure 3.13: When an albedo estimation model [84] pre-trained on a mixture of synthetic and real perspective images is applied to our data (a), the domain shift causes obvious errors (b). But when applied to our ambient image (c), the same method is able to estimate albedos more accurately (d), without requiring to retrain on our dataset (which would not be possible in any case given the lack of ground truth annotation for supervision).

rendered on the new material by scaling the floor intensity according to the estimated sunlight scale.

**Changing Sun Direction.** Using the estimated sun direction, we project the sunlit region back to the window position. Then, we re-project sunlight onto the floor and the walls using a new sun direction. Fig. 3.16 visualizes two examples of changing sun direction.

**Furniture Insertion.** With the decomposition result, sun direction and the estimated HDR map (Sec. 3.6.1), we can insert virtual objects into an empty room, by calculating the occluded specular reflection and direct sunlight via ray tracing. Using the computed layout (scale set manually), we render ambient, sunlight and specular effects separately and combine them. Fig. 3.17 shows several insertion examples,
Figure 3.14: Sun direction estimation results. Left: histogram of sun elevation errors for images with visible sunlight; Right: An example of projecting the window mask to the floor and the walls according to estimated sun direction. The fine sun estimations provide more accurate elevation results. 33% of the fine estimations and of the 28% coarse estimations are within 5° error. 53% of the fine estimations and of the 52% coarse estimations are within 10° error. The example shows that the fine estimation produces a reasonable sun direction even with overlapping sunlight and specular reflection. This is an extreme example where coarse estimation completely fails and the fine estimation significantly improves the result.

Figure 3.15: Changing floor material. Row 1: original flooring; Row 2: change to (different) wood; Row 3: change to (different) carpet. The specular reflection on row 2 column 2 is rendered using the estimated HDR map (Sec. 3.6.1) and a uniform surface roughness.

where sunlight is cast on the desk (a) and the bed (b&c), specular reflection is blocked by the chair (a) and bed-side table (c), and sunlight is blocked by the clothing hanger
Figure 3.16: Changing sun direction. Row 1: original sun direction; Rows 2-3: New sun directions. Notice the consistency in the shapes and appearances of the sunlit regions in the perspective insets.

(b) and bed (c).

Fig. 3.18 shows scenes rendered with Li et al. [74], rendered with the estimated HDR map without appearance decomposition, and rendered with our full method. Li et al. [74] trains on synthetic data with ground truth supervision and predicts illumination and scene properties for a perspective view. The method estimates spatially-varying lighting for each pixel rather than each 3D voxel. A selected pixel and its lighting is used for rendering. This fails when the object parts are far away from the selected pixel in 3D (e.g., first row in Fig. 3.18). Besides, the method
Figure 3.17: Appearance decomposition enables several applications such as furniture insertion, and changing flooring and sun direction, making it a useful tool for virtual staging. Top row: a home office and a bedroom staged by inserting chair, table, bed, shelves, plant and other objects. Rows 2, 3, 4: a living room is staged with different furniture, multiple new flooring (dark wood, carpet), and is shown under two different sun directions. Notice the sun light on the surfaces and the blocking of specular reflection that add realism to the scene.

renders shadows in the specular region by scaling the original floor intensity without explicitly removing blocked specular reflection, leading to insufficient darkening of
the floor (*e.g.*, second row).

![Figure 3.18](image)

Figure 3.18: We insert furniture by applying the inverse rendering approach by Li *et al.* [74] to perspective crops (a) of our panoramas (b). This method is trained on perspective views with synthetically rendered data. We also insert furniture by using the HDR map and 3D layout as illumination (c). Both approaches are not able to faithfully render the shading interactions of the objects with sunlight or specular reflections (see ours in (d)).

**Combination of Multiple Applications.** The applications can be combined to allow more flexible virtual staging effects. Fig. 3.17 shows high-quality visualizations that include two or more of changing floor material, sun direction and object insertion.
3.8 Limitation

There are three main limitations. First, we focus on direct lighting effects (first light bounce, no global illumination) on the floor and the walls. However, the specular reflection and direct sunlight could also appear on objects like kitchen countertops or appliances. This could potentially be solved by extending our semantics and geometry to include furniture or other objects as long as some coarse mask and geometry can be obtained automatically to allow for accurate ray-tracing. Second, we assume the window producing the sunlight is visible for sun direction estimation. Using multi-view information from different panoramas in a single home can help overcome this limitation. Third, our approach is tailored to panoramas. It is possible to extend our ideas to narrower perspective field of views but there are several alternate methods in that space and we lose the advantages provided by a panorama. Given the increased availability of panoramas of homes, we believe the method is timely and can be an effective tool for virtual staging.

3.9 Conclusion

In summary, we present an appearance decomposition method for empty indoor panoramic scenes. Relying on supervision from semantics, our method detects and removes the specular reflection on the floor and the direct sunlight on the floor and the walls. The decomposition result can be applied to multiple virtual staging tasks, including albedo and sun direction estimation, furniture insertion, changing sun direction and floor material.
3. Semantically Supervised Appearance Decomposition for Virtual Staging from a Single Panorama
Chapter 4

Conclusion

We introduce two applications for image synthesis by incorporating appearance decomposition.

The first application focuses on structural appearance decomposition. Our proposed method can complete, remap (deblur), and segment near-periodic scenes by decomposing them into motifs and periodicities. The key takeaway for this application is that: our proposed method can be extended to any kind of spatially describable pattern.

The second application focuses on intrinsic appearance decomposition. The proposed method decomposes a indoor empty panorama into specular, direct sunlight, and ambient components. This decomposition enables multiple image synthesis applications such as furniture insertion, changing sunlight directions, and changing floor materials. The key takeaway for this application is that: our proposed method can be applied to obtain continuous signals from discrete signals that are easy to collect.

We believe the idea of leveraging appearance decomposition for image synthesis can also be helpful in more real-world applications such as night-to-day image translation [157] and scene relighting [143].
4. Conclusion
Chapter 5

Acknowledgements

The work presented in Chapter 2 will appear in ECCV 2022, co-authored with Tiancheng Zhi, Martial Hebert, Srinivasa Narasimhan. The work presented in Chapter 3 will appear in SIGGRAPH 2022, co-authored with Tiancheng Zhi, Ivaylo Boyadzhiev, Sing Bing Kang, Martial Hebert, and Srinivasa Narasimhan.

Both of the papers are supported by a gift from Zillow Group, USA, and NSF Grants #CNS-2038612, #IIS-1900821.
5. Acknowledgements
Bibliography

[1] PSU Near-Regular Texture Database. http://vivid.cse.psu.edu/. 2.1, 2.4.1, ??


and Pattern Recognition (CVPR), 2014. 2.1, 2.4.1, ??, 2.4.4


67


[42] Yannick Hold-Geoffroy, Kalyan Sunkavalli, Sunil Hadap, Emiliano Gambaretto,


[46] Jia-Bin Huang, Sing Bing Kang, Narendra Ahuja, and Johannes Kopf. Image completion using planar structure guidance. ACM Transactions on graphics (TOG), pages 1–10, 2014. 2.1, 2.2.1, 2.3.4, ??, 2.4.4, ??, ??


[50] Borovec Jiri, Svihlik Jan, Kybic Jan, and David Habart. Supervised and unsupervised segmentation using superpixels, model estimation, and graph cut. Journal of Electronic Imaging, 2017. 2.3.5, 2.4.4


[66] Jingyuan Li, Ning Wang, Lefei Zhang, Bo Du, and Dacheng Tao. Recurrent feature reasoning for image inpainting. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2020. 1, 2.1, 2.2.1


[68] Yikai Li, Jiayuan Mao, Xiuming Zhang, William T Freeman, Joshua B Tenenbaum, Noah Snavely, and Jiajun Wu. Multi-plane program induction with 3d box priors. In Neural Information Processing Systems (NeurIPS), 2020. (document), 2.1c, 2.1, 2.2.1, 2.3.4, ??, 2.4.4, ??, ??, ??

[69] Yikai Li, Jiayuan Mao, Xiuming Zhang, William T Freeman, Joshua B Tenenbaum, and Jiajun Wu. Perspective plane program induction from a single image. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 4434–4443. IEEE, 2020. 2.1, 2.2.1, 2.3.1, 2.3.4, 2.3.4, 2.3.6, 2.4.4


[75] Zhengqin Li, Ting-Wei Yu, Shen Sang, Sarah Wang, Sai Bi, Zexiang Xu, Hong-Xing Yu, Kalyan Sunkavalli, Miloš Hašan, Ravi Ramamoorthi, and Manmohan
Bibliography

Chandraker. Openrooms: An end-to-end open framework for photorealistic indoor scene datasets, 2021. 3.1, 3.2


[87] Jiayuan Mao, Xiuming Zhang, Yikai Li, William T Freeman, Joshua B Tenenbaum, and Jiajun Wu. Program-guided image manipulators. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 4030–4039. IEEE, 2019. 2.1, 2.2.1, 2.3.4


[90] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In European conference on computer vision (ECCV), pages 405–421. Springer, 2020. 2.1, 2.2.2, 2.3.1, 2.3.3, 2.3.6, 3.2


[97] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic


[130] Fu-En Wang, Yu-Hsuan Yeh, Min Sun, Wei-Chen Chiu, and Yi-Hsuan Tsai.


[146] Yanhong Zeng, Jianlong Fu, Hongyang Chao, and Baining Guo. Learning pyramid-context encoder network for high-quality image inpainting. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1486–1494, 2019. 1, 2.1, 2.2.1, ??, 2.4.4

[147] Yu Zeng, Zhe Lin, Jimei Yang, Jianming Zhang, Eli Shechtman, and Huchuan Lu. High-resolution image inpainting with iterative confidence feedback and guided upsampling. In European Conference on Computer Vision (ECCV), pages 1–17. Springer, 2020. 1, 2.1, 2.2.1, ??, 2.4.4


Bibliography

Conference on Computer Vision and Pattern Recognition (CVPR), 2018. 2.3.3, 2.3.5, 2.4.2


[153] Xuaner Zhang, Ren Ng, and Qifeng Chen. Single image reflection separation with perceptual losses. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4786–4794, 2018. 3.2, 3.5.1, 3.5.1

[154] Zhengdong Zhang, Arvind Ganesh, Xiao Liang, and Yi Ma. Tilt: Transform invariant low-rank textures. International journal of computer vision (IJCV), pages 1–24, 2012. 2.3, 2.3.5, 2.4.1


[157] Ziqiang Zheng, Yang Wu, Xinran Han, and Jianbo Shi. Forkgan: Seeing into the rainy night. In The IEEE European Conference on Computer Vision (ECCV), August 2020. 4


[161] Liu Ziyin, Tilman Hartwig, and Masahito Ueda. Neural networks fail to learn periodic functions and how to fix it. In Neural Information Processing Systems (NeurIPS), pages 1583–1594, 2020. 2.3.1, 2.3.2, 2.3.3, 2.3.6

[162] Chuuhang Zou, Alex Colburn, Qi Shan, and Derek Hoiem. Layoutnet: Reconstructing the 3D room layout from a single rgb image. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2051–2059, 2018. 3.1