Vision-based 3D information modeling and applications

Ni, Yun

2021

Ni, Y. (2021). Vision-based 3D information modeling and applications. Doctoral thesis, Nanyang Technological University, Singapore. <https://hdl.handle.net/10356/145895>

<https://hdl.handle.net/10356/145895>

<https://doi.org/10.32657/10356/145895>

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License (CC BY-NC 4.0).

Downloaded on 26 Apr 2023 19:11:06 SGT

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

**VISION-BASED 3D INFORMATION MODELING
AND APPLICATIONS**

NI YUN

SCHOOL OF ELECTRICAL AND ELECTRONIC ENGINEERING

2021

Vision-Based 3D Information Modeling and Applications

Ni Yun

School of Electrical and Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

2021

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

31/03/2020

.....

Date

.....

Ni Yun

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

31/03/2020

.....

Date

chaup

.....

Prof. Lap-Pui Chau

Authorship Attribution Statement

This thesis contains material from 1 paper published in the following peer-reviewed journal.

Chapter 4 is published as Yun Ni, Jie Chen, and Lap-Pui Chau, "Reflection removal on single light field capture using focus manipulation," *IEEE Transactions on Computational Imaging*, vol. 4, no. 4, pp. 562 - 572, 2018. DOI: [10.1109/TCI.2018.2860465](https://doi.org/10.1109/TCI.2018.2860465).

The contributions of the co-authors are as follows:

- Dr. Jie Chen and A/P Lap-Pui Chau suggested the initial direction of using LF captures to remove reflections.
- The idea to identify background and reflection edges based on their relative depths to camera focus was co-developed by Dr. Jie Chen and me. The means to recover backgrounds and reflections from their partial edges identified was devised by me.
- Experiments were carried out by me. Data used were collected/synthesized by me.
- The manuscript was prepared by me. Dr Jie Chen and A/P Lap-Pui Chau edited the manuscript.

31/03/2020

.....

Date

.....

Ni Yun

Acknowledgements

During my PhD studies, I have received an enormous amount of support from many people, and I would like to express my deepest gratitude to them.

I am grateful to my supervisor, Professor Lap-Pui Chau, for he is always supportive and encouraging. He has given me a great deal of freedom in choosing research topics and constantly encourages me to work on areas that interest me.

I would also like to thank all the staff and students in my research group. Dr. Jie Chen provided valuable guidances for conducting research when I started my PhD. Mr. Alexander Matyasko have answered many of my Linux system questions. Dr. Cheen Hau Tan has been generous with his time and is always ready to help, whenever I encounter a problem. Thanks also to Ms. Xiaoxi Ma, Ms. Shengyu Nan, and Mr. Yi Wang.

In addition, I would like to thank my friends and colleagues at ST Engineering-NTU Corporate Lab for making my postgraduate years fun.

Finally, I would like to thank my parents, especially my mother. She is my biggest supporter and always believes in me. I wouldn't be where I am today without her guidance and encouragement.

Abstract

We can infer the 3D structure of our surroundings simply by looking. It is long hoped that imaging devices can mirror such an ability, which is crucial to many computer vision tasks. This thesis is about our work on developing algorithms, as well as utilizing novel optical devices, in particular light-field cameras, to infer 3D information without active illuminations, and how such information can be used in various practical applications:

- We develop an algorithm to synthesize novel views. When we shift to a different viewpoint, certain scene points not captured in the input image will be revealed. We first infer the colors of these points, based on 3D plane notations, and then use the expanded scene points to generate the target image.
- We propose using depths calculated from light field (LF) data to remove reflections. The depths are used to roughly identify background and reflection scene points. We then reconstruct the background and the reflection layers using scene points identified.
- Finally, we utilize depth information to help identify different kinds of materials. Given images captured using a multi-camera cell phone, we estimate a depth probability map. The estimated depth probability map, together with one of the color images, are then inputted into a trained neural network to determine the material type.

The first method utilizes 3D plane notations, while the rest use depths recovered from LF data or stereo images. 3D information is crucial in accomplishing these tasks.

Table of Contents

Acknowledgements	ix
Abstract	xi
List of Figures	xv
List of Tables	xvii
Symbols and Acronyms	xix
1 Introduction	1
1.1 Motivation	1
1.2 Objectives	4
1.3 Contributions	4
1.4 Outline of the Thesis	6
2 Literature Review	9
2.1 Novel view synthesis	9
2.2 Reflection removal	11
2.3 Material recognition	14
3 Novel View Synthesis with Planar Structure Guidance	17
3.1 Introduction	17
3.2 Proposed Method	19
3.2.1 Planar Structure Guidance	19
3.2.2 Network Architecture	20
3.2.2.1 Contour Completion Network	20
3.2.2.2 Texture Inference Network	21
3.3 Experiments	23
3.3.1 Results	24
3.3.2 Comparisons with baseline methods	25
3.3.3 Failure case studies	27
3.4 Conclusion and Future Work	27

4	Reflection Removal On Single Light Field Capture Using Focus Manipulation	31
4.1	Introduction	31
4.2	Proposed Method	33
4.2.1	Optical Geometry Analysis	34
4.2.2	Focus Manipulation for Background/Reflection Classification	35
4.2.3	Initial Layer Separation	38
4.2.4	EPI Consistency Constraint	38
4.2.5	Final Layer Separation	41
4.2.6	Alternating optimization	43
4.2.6.1	Layers update	44
4.2.6.2	Disparities update	45
4.3	Experiments	45
4.3.1	Evaluations on Synthetic Data	48
4.3.2	Evaluations on Real Data	50
4.3.3	Limitations	58
4.4	Conclusion	58
5	Material Recognition Using a Multi-camera Cellphone	61
5.1	Introduction	61
5.2	Proposed Method	62
5.2.1	Inputs	62
5.2.2	Network architecture	64
5.3	Training the Network	65
5.4	Experiments	68
5.5	Conclusion and Future Work	70
6	Conclusions and Recommendations for Future Work	71
6.1	Conclusions	71
6.2	Recommendations for Future Work	72
	List of Author's Publications	75
	Bibliography	77

List of Figures

1.1	An illustration of stereo matching. Following pinhole camera geometry, the two scene point (shown as the red and the blue triangles respectively) will be projected to different positions on the image plane on the image planes of the left and the right cameras.	2
3.1	Diagram illustrating how a 3D point is projected to different pixel positions $[x_r, y_r]$ and $[x_t, y_t]$ in the reference and the target images. The correspondence between $[x_r, y_r]$ and $[x_t, y_t]$ is stated in Eq. (3.1).	19
3.2	Diagram of the contour completion and the texture inference network. All down convs, up convs, and convs are partial convs described in Section 3.2.2.2	22
3.3	Final results after combining the output results of the contour completion and the texture inference networks.	24
3.4	Contour completion results. Here we show target contours completed based on input ones warped from reference contours.	25
3.5	Texture inference results of various planes. One can see that extra textures inferred are consistent with given parts in (a).	26
3.6	Comparisons. We compare with results of [1] [2] (novel view synthesis methods), as well results obtained after using [3] (an image inpainting method) to complete the warped reference images.	29
3.7	A failure case. For the plane in the yellow polygon, the expanded texture does not fully cover the area inside the estimated contour.	30
4.1	(a) a real scene captured using Lytro. (b) edges labeled by our method (background in red, reflection in cyan). (c) recovered background. (d) recovered reflection.	32
4.2	Illustration of the two plane representation for LF.	34
4.3	Optical diagram on the formation of EPIs in a LF camera, for scene points at different distances around the focused scene plane.	36
4.4	Classified edges of a scene where the background (buildings) is afar and reflection (person taking the picture) is near. In (b), background edges labeled are marked in red, while reflection edges labeled are marked in cyan.	36
4.5	Comparison between an actual input EPI and one constructed from initial layer and disparity estimates. One can see that the two are different (see close ups of two segments in red and cyan).	39

4.6	Disparity interpolation based on superpixel segmentation. (a) initial layer estimates of the scene shown in Fig. 4.1(a). (b) superpixels of initial layer estimates. (c) disparities of edge scene points. (d) initial semi-dense disparities interpolated from (c). Disparities are displayed using scaled colors. Top: background. Bottom: reflection.	40
4.7	Qualitative evaluation on synthetic scenes. Here we show separation results on the scene <i>Car</i> and <i>Orange</i> , along with ground truths. . .	47
4.8	More results of synthetic scenes.	49
4.9	Qualitative evaluation on real scenes containing weak or modest reflection. In scene one, the background is an image on the bulletin board, and the reflected scene depicts the person taking the picture and the site behind. In scene two, the background depicts displays of a jewelry shop, and the reflected scene includes the person taking the picture, railings behind, and shopping mall advertising on the other side.	51
4.10	Qualitative evaluation on real scenes containing strong reflection. .	52
4.11	More qualitative evaluation on real scenes.	53
4.12	Separation results of a scene with complex depths.	54
4.13	Qualitative evaluation on handshake blurred scenes.	55
4.14	Experimental setup.	56
4.15	Evaluation with ground-truth backgrounds. NCC between obtained and ground-truth $\mathbf{I}_b$ is calculated as 0.98, 0.93, and 0.95, for the three scenes.	57
4.16	A failure case.	58
5.1	Sample MPI images. The first column shows the estimated colors, C_7 , C_{16} , and C_{22} of three different depth ranges. The second column shows the estimated projections $C_7 \cdot \bar{\alpha}_7$, $C_{16} \cdot \bar{\alpha}_{16}$, and $C_{22} \cdot \bar{\alpha}_{22}$ of scene points in these depth ranges.	63
5.2	Diagram of the proposed neural network.	65
5.3	Analygraph from rectified input images.	66
5.4	Sample stereo images of the three types of materials collected. . . .	67
5.5	Confusion matrix that shows the numbers of true positives, true negatives, false positives, and false negatives.	68
5.6	Stereo image pairs of actual and printed foliage	69
5.7	Confusion matrices of the full network and the network without the depth branch.	70

List of Tables

4.1	Quantitative evaluation on synthetic scenes. The best performance of each data entry has been highlighted in bold.	46
-----	--	----

Symbols and Acronyms

Symbols

$\mathcal{R}^n$	the n -dimensional Euclidean space
$*$	the convolution
$\circ$	the Hadamard product
$\oslash$	the Hadamard division
$\sum$	the sum of multiple items
$ \cdot $	the 1-norm of a vector or matrix
$\frac{\partial y}{\partial x}$	the drerivative of y with respective to x

Acronyms

LiDAR	Light detection and ranging
ToF	Time-of-flight
SURF	Speeded up robust features
SIFT	Scale-invariant feature transform
LF	Light field
EPIs	Epipolar-plane images
PSFs	Point spread functions
RANSAC	Random sample consensus
VGG	Visual geometry group
BRDF	Bidirectional Reflectance Distribution Function
SVM	Support vector machine
HSH	Hemispherical Harmonic
GANs	Generative adversarial networks
SAI	Sub-aperture-image
IRLS	Iteratively reweighted least squares
NCC	Normalized cross-correlation
MPIs	Multiplane image scene representations

Chapter 1

Introduction

1.1 Motivation

While observing our surroundings, we immediately figure out the spatial layout: whether the chair is located further away than the shelf, how to move our heads to see the box partially blocked in the view by the standing fan, etc. Such an ability to infer 3D information is highly desired in many applications where we try to build systems that can function just like human.

Nevertheless, it is challenging to match human vision systems ability to sense the environment and deduce 3D clues such as depths, 3D planes, and object poses. Researchers have proposed various approaches to infer 3D information, most of which focus on recovering depths. Depending on illumination sources, these approaches can be labeled as either active or passive.

Active approaches. Active illumination based approaches send signals first and resolve depths of scene points by measuring their travel time and/or phase shifts between transmitted and returned signals. Using different light sources, common active illumination systems include Light Detection and Ranging (LiDAR), time-of-flight (ToF) cameras, and structured light scanners. ToF cameras and structured light scanners can generate 3D depth maps at a high frame rate, but the results can be erroneous when strong ambient light is present [4]. On the other hand, LiDAR can calculate depths rather accurately. But the process is time-consuming and only scene points within a limited field of view will be sampled. Software

algorithms and hardware designs have been proposed to mitigate the drawbacks of these systems, in order to infer depths both speedily and accurately [5, 6].

FIGURE 1.1: An illustration of stereo matching. Following pinhole camera geometry, the two scene point (shown as the red and the blue triangles respectively) will be projected to different positions on the image plane on the image planes of the left and the right cameras.

Passive approaches. Not employing additional light sources, passive approaches use conventional cameras. The basic approach to infer depths is to capture scene points from more than one viewpoint and measure the difference in respective pixel positions. As shown in Fig. 1.1, the positional shift between the left and the right projections on respective image planes is greater for the scene point that is nearer (the blue triangle). Furthermore, one can deduce based on geometry that the positional shift is inversely proportional to the depth of a scene point. Hence if one can match pixels in the two images that belong to the same scene points, one can identify their depths. Hand-crafted methods use techniques such as the speeded up robust features (SURF) [7] and the scale-invariant feature transform (SIFT) [8] to extract features and compute the correspondences. Neural networks can also be employed to match pixels of the same point [9–12]. The stereo-matching based approaches will achieve perfect results, if all correspondences are correctly identified. Nevertheless, complications such as occlusions and the lack of texture can cause noticeable estimation errors. Furthermore, depths calculated in this way are relative instead of absolute. That is to say we can tell whether a scene point

is nearer or further away compared with another scene point, but we do not know its exact distance from the camera.

Some researchers propose adding extra optical modulations to the cameras, in order to estimate 3D information more efficiently. One such design results in light field (LF) cameras [13] [14], especially Lytro. An additional micro-lens array is inserted between the main lens and camera sensors, which can refract light rays of a scene point to different sensor locations based on its relative depth. Consequently unlike conventional cameras that only record the radiances of incoming light rays, Lytro cameras record their directions as well. Epipolar-plane images (EPIs), whose linear structures indicate disparities of corresponding scene points, can be derived. Various methods utilize EPIs to estimate depths [15] [16] [17] [18]. It is worth mentioning that unlike multi-view inputs used in stereo-based methods, only single LF captures are needed. Another modulation scheme is to add a coded mask which attenuates incoming light rays based on its pattern [19] [20] [21]. With a patterned mask placed in front of the aperture, light rays passing through the main lens can be modulated to highlight depth-related characteristics. The encoded radiance values captured by the sensor can then be processed to recover the 3D information. [19] utilizes an amplitude mask whose pattern is calculated to emphasize the depth-dependent point spread function (PSF) differences. A statistical model is then used to reconstruct both the depth maps and the all-focus input image. Like [19], [20] and [21] also use patterned masks to distinguish PSFs at various depths. But instead of calculating mask patterns based on a statistical models, they take a data-driven approach and train neural networks to learn the optimal design. Furthermore, [20] and [21] also use trained neural networks to calculate depth maps from modulated images.

The inferred 3D information is extremely useful. In this thesis we present our work on how estimated 3D information is estimated and used to accomplish three different computer vision tasks. Furthermore, the 3D information is estimated from stereo viewpoints (LF can be considered a special case of stereo viewpoints). Another example where 3D information is useful is in robotics. We may wish to figure out enough spatial clues of the environment based on left and right images captured by a stereo camera, in order to control a robotic arm to perform interactive tasks such as picking up and placing objects at designated locations.

1.2 Objectives

The overall objectives of this thesis is to estimate and apply of 3D information estimation approaches. The specific topics I have worked on include 1) novel view synthesis with planar structure guidance, 2) reflection removal using single LF capture, and 3) material recognition with a multi-camera cell phone.

Regarding the first topic novel view synthesis, the specific objectives are:

- The method should take general inputs of no particular alignment, and be able to generate images from a wide range of new viewpoints.
- The method should be able to convincingly deduce 3D scene points that are not captured or occluded in the input images, and use the completed 3D structures to synthesize target views.

Regarding the second topic reflection removal, the specific objective is:

- The method ought to be able to accurately recover both background and reflection in a single shot, utilizing the unique features of LF data.

Regarding the third topic material recognition with a multi-camera cell phone, the specific objective is:

- The method should be able to recognize common material types such as foliage, wood, and paper, given input images captured by built-in cameras of a mobile phone.

1.3 Contributions

The contributions of our work described on the three research topics is summarized individually as follows:

The contributions regarding the first topic novel view synthesis are:

- We propose a method to approximate the 3D geometry of a scene as a set of images at various 3D planes. Since scene points belonging to the same plane generally form a patterned texture, predicting appearances of scene points not captured would be much easier, if the planes they belong to are identified.
- We then infer more scene points, for each 3D plane, before computing the target images. When synthesizing the uncaptured part of a plane using neural networks, we adopt partial convolutions that only work with valid input points, since the captured part often has an irregular shape.
- Compared with existing novel view synthesis methods, our method can generate the full target image.
- To sum up, we developed a method to synthesize images of distant views, based on a novel approach to construct scene geometry.

The contributions regarding the application to layer separation using single LF captures are:

- We come up with novel focus manipulation procedures to differentiate background and reflection edges, which are used to generate initial estimates of the two layers.
- We then enforce the sums of background and reflection EPIs to equal input EPIs, and formulate this constraints, together with others enforcing NA image statistics, into an objective function. Using the initial estimates as the starting place, the objective function is then minimized through an alternating scheme to obtain the final separation results.
- The proposed method achieves state-of-the-art results, and can correctly recover the two layers, even when the input is blurred by camera shake.
- Overall, we introduced a reflection removal method that utilizes the unique characteristics of light field data to recover the background layers.

The contributions regarding the application to material recognition with a multi-camera mobile phone are:

- We write a piece of Android Studio code that can obtain images from cameras of a cell phone.
- In addition to appearances, we propose to utilize depths too when determining the material class, as each type of material has its distinct depth distribution. Instead of using depth values calculated from the input images, which can be imprecise due to calibration errors, we input depth probability maps indicating the likelihoods of pixels belonging to particular depths. The outputs of a network trained to estimate the RGBA-D representation of a scene from a set of two views are used to compute these probability maps. We then use a trained neural network to determine the material types.
- During testing, our method can correctly identify the material type 88.6% of the time.
- To conclude, we proposed to include depth information when identifying materials, and developed a neural network based method accordingly.

1.4 Outline of the Thesis

The remainder of this thesis is organized as follows.

Chapter 2 examines existing methods of the three research topics covered in this thesis: 1) novel view synthesis, 2) reflection removal, and 3) material recognition.

Chapter 3 reviews our work done on novel view synthesis. Section 3.1 discusses the pros and cons of existing approaches, as well as how our approach can address some of the problems. Section 3.2 introduces our approach, which is evaluated in section 3.3, in comparison with several baselines. Section 3.4 concludes this chapter.

Chapter 4 reviews our work done on reflection removal using LF data. Section 4.1 presents the unique advantages of using LF captures. Section 4.2 details how the proposed method is derived. Section 4.3 shows experimental results, using both real and synthetic data.

Chapter 5 introduces our method to identify material types. Section 5.1 summarizes how color and depth information is derived from cell phone images in order to determine the material types. Section 5.2 presents our method. Section 5.3

presents the specifics of how the neural network was trained. Section 5.4 reports experimental results.

Chapter 6 draws the overall conclusion and discusses possible future directions.

Chapter 2

Literature Review

2.1 Novel view synthesis

View synthesis is closely related to 3D reconstruction. Using input images, one first builds a model to represent the scene geometry by means such as depth maps, fronto-parallel plane images, and 3D meshes, and then generates the target image based on the geometric model. In addition to guiding subsequent image rendering, scene geometries that have been calculated explicitly also enable applications such as real-time rendering and augment reality.

As mentioned previously, view synthesis methods involve two stages, scene geometry estimation and image rendering. Existing methods differ in their specific approaches to represent scene geometries and render target views. [22] reconstructs the scene geometry from a single 2D image using user marked vanishing point and polygons specifying the foreground objects and the background, and then render the images at designated views. [23] proposed a general method that satisfies various goals identified by the authors such as sustaining epipole consistency, measuring ray differences by angle deviations, and working with both unstructured and light-field inputs. [24] assumes there are only a single ground planes, piece-wise planar objects, and the sky in a scene, and infer the geometry for novel view synthesis accordingly. In [25], Kopf et al. suggested using estimated depths to warp image gradients, and render the novel view image based on the warped gradients. This method is especially effective with scenes containing reflections, since background and reflection gradients mostly do not overlap. [26] uses planar structures captured

in the images to derive plane labeling maps, based on estimated plane parameters. The plane labeling maps produced are then used to obtain proxy meshes for novel view rendering. [27] proposed to segment the input images into superpixels and interpolate more depth values. These superpixels are then warped individually to give raw novel view estimates, which are blended to obtain the final output. [28] aligns the pixels in the input image of an object to its 3D shape template, which is assumed to be known, to obtain the geometry. Based on depth maps of the input images, [29] constructs consensus volumes for these images, while preserve the uncertainty in estimating scene geometry. These volumes are then interpolated to compute geometry for the novel view, whose image can then be obtained through texture mapping.

The rapid increase of computational power and availability of affordable GPUs makes it possible to accomplish many computer vision tasks using neural networks. Compared with traditional approaches based on statistical modeling and optimization, neural network based approaches are data-driven. They have the advantages of short inference time and effective modeling of implicit priors. Many of the recent view synthesis methods utilize the advantages of neural networks to achieve better performances. In [30], Flynn et al. proposed to calculate the plane sweep volumes of various candidate depths for two distinct inputs, which are combined and provided to two neural networks, a color and a selection tower, to generate both the final reprojected images at these depths and their respective pixel-wise probabilities. The target novel view image is obtained as the sum of these images weighted by their probabilities. Given a transformation specifying the target view, [31] uses a trained neural network to warp the input image of an object accordingly and hallucinates the parts invisible in the input image to produce the final output. Nevertheless, the neural network is category specific and needs to be re-trained, if we need to synthesize novel views for a different kind of objects. [32] suggested using a neural network to predict the blending weights to combine ranked input contributions that are computed based on per-view meshes, to obtain the final output. Like [30], [2] also presented a method based on plane sweep volume. But it uses a neural network to calculate the discretized volume representation of the reference image, but not that of the target image. Mildenhall et al. established the relationship between the number of valid depths and the view sampling rate required to successfully reconstruct the scene geometry, and developed a neural network-based novel view synthesis accordingly [33]. Based on

[2], [1] proposed to use generative adversarial networks (GAN) to mitigate artifacts caused by disocclusion of content not visible in the input. Niklaus et al. developed a method to create synthetic 3D Ken Burns effect, where a sequence of novel view images are synthesized based on the start and end of the virtual camera path [34]. Like the proposed method, [35] also noticed that the majority of scene geometry can be considered as planes. But it has not been noticed that points on the same plane form a highly regular texture, a feature that can be used to infer points not captured.

2.2 Reflection removal

Existing reflection removal methods can be categorized according to their input data types: multiple images, videos, and single images.

Multiple images. Multiple-image methods use images taken from shifted viewpoints, from which different motions of reflection and background can be exploited to help guide the separation. Li and Brown observed that after warping the input images to one reference, edges of reflection only occur sparsely, and derived an algorithm accordingly [36]. Yang *et al.* improved upon [36] by exploiting the connection between optical flow estimation and image layer separation [37]. Nevertheless [36] and [37] require that SIFT-flows among input images are dominated by background edges, which is not always true. Authors of [38] stated that after 2D projective transformations, multiple images of the background should be highly correlated to each other, and proposed a method based on this property. But it only works if background is a (nearly) planar surface. In [39], Xue *et al.* applied the random sample consensus (RANSAC) algorithm to classify motion vectors into those of background and reflection, and then reconstructed the two layers based on their motion vectors. However RANSAC algorithm works well only when the two layers are of distinct depths and thus have very unlike motion vectors. Lyu et al. also used multiple images [40]. But instead of images taken from different viewpoints, they use a pair of images from the same viewpoint, taken with and without a polarizer. Plane parameters of the reflective surface are first regressed using a neural network. Based on the image formation model, estimated parameters are then used to calculate initial reflection and background, which are further refined through a

second neural network to generate the final output. Other multiple-image input reflection removal methods include [41–43].

Videos. Meanwhile [44] and [45] used videos instead to separate background and reflection. Video-based methods are often considered as a special case of multiple-image ones, and thus share the same pros and cons. Nandoriya *et al.* proposed using a combination of feature tracks and edge-flow to initialize motion, which is more suitable for video sequences [44]. The objective function is also modified to reflect the input type. The algorithm presented in [45] is especially for in-vehicle black box videos, where unlike the fast forwarding background, the reflection remains relatively stationary.

Single images. Unlike multiple-image and video-based ones, single-image reflection removal methods require just single shots of the scenes. In [46] [47], the authors suggested adopting a user markup that indicates which layer some of the edges belong to in order to guide the separation. However, requiring user inputs limits the applicability and increases the uncertainties of their method. Shih *et al.* noticed that when reflection is caused by double-pane windows, the unique ghosting effect can be used to identify the reflection [48]. However, in most cases, single-pane, instead of double-pane window is the reflecting surface. In [49], Wan *et al.* proposed using an external dataset to search for patches similar to the input, and apply those found as a prior of the background layer. Yet this prior only works if the reflection is relatively weak, since the dictionary coefficients of patches found using the superimposed input will then be close to those of the background. Li and Brown proposed a single-image reflection removal method under the assumption that one layer is smoother than the other [50]. Algorithm presented in [51] is based on similar assumption. Fan *et al.* took a learning-based approach and introduced a deep neural network architecture to set apart the two layers, under the assumption that input edges are mostly of the background [52]. Methods depending on such assumptions usually fall short in presence of strong reflection or camera shakes that make both layers blurry. [53] use neural networks to recover background and reflection layers from single-image inputs as well. During training, three types of losses were used: 1) a feature loss to ensure that synthesized background layers share similar Visual Geometry Group (VGG) features with the ground truth, 2) an adversarial loss to encourage the network to produce more realistic-looking outputs, and 3) an exclusion loss that is based on the observation

background and reflection edges do not overlap. Similar to [46] [47], [54] also depends on user markups to identify edges of the two layers. But unlike [46] [47], [54] uses neural networks to obtain background and reflection images. Taking as input the image to be decomposed and the two rough edge maps, a network outputs initial estimates of the two layers and an attention mask indicating low-confidence areas in the initial background estimate. An additional refinement network then produces the final estimate, based on the initial background estimate, as well as the attention mask previously generated. Since existing real-world dataset do not contain enough paired training data of input images and their ground-truth decompositions, researchers often need to synthesize them using given backgrounds and reflections. Wen et al. proposed to train a neural network to produces per-pixel blending weight masks of the two given layers, which result in input images more similar to real-world ones [55]. Beside coming up with a new way to generate images with reflections, they also propose a network model that recovers background and reflection, as well as the blending weight mask. The network is trained by minimizing individual losses of the output items, as well as a reconstruction loss to make sure that the input images reproduced using these items equal the original ones. Like [55], [56] also seeks to address the problem of not having enough real-world data for training. [56] suggested accepting paired data of input images and their ground-truth background layers captured from slightly different viewpoints for fine-tuning the networks. As such paired data can be easily captured using a handheld camera from numerous angles, it is not hard to collect a large number of them. Nevertheless, since input images and their background layers captured are not aligned, [56] proposed an alignment-invariant loss that measures the difference between their features from the conv5_2 layer of a pre-trained VGG network, which indicates how similar the two images are regarding high-level visual properties.

Most neural network based methods output background and reflection layers concurrently in an end-to-end fashion. However [57] suggested that taking a decoupled and gradual approach will produce better results. The proposed architecture has three sub-networks: 1) one that produces initial background estimates, 2) one that produces initial reflection estimates, and 3) one that produces final background estimates. The estimate from previous sub-network, together with the original image, forms the input of these sub-networks. The idea behind is that since the two layers are of different scenes and thus have distinctive contours, even a rough

estimate of one layer will give a strong indication of how the other layer should look like.

In [58], Johannsen *et al.* proposed a method that uses LF cameras to remove reflection. The method also involves works in [59][60]. Reasonable separation results are shown mostly for synthetic and scenes captured using camera arrays. Highly accurate disparity maps of both layers are required for this method to work. But LF data captured using handheld devices are often of a low resolution and noise corrupted, which can cause strong visual artifacts in layers recovered by [58]. [61] and [62] are other LF-based methods. But [61] does not work with consumer LF cameras. It only works on a portable camera array built by the authors, which severely limits its applicability. Meanwhile the network of [62] was trained using LF captures of almost single depths, which is an oversimplification of real-life scenes.

[63] also utilizes the unique structure of light field data to separate background and reflection. The method presented in [63] is an improvement of our previous work [64], as it is devised to include the case where certain depths of the two layers overlap. In that situation, edges of points inside the shared depths range will not be labelled. Noticing that initial estimates reconstruct these ambiguous edges somewhat accurately, based on neighboring edges labelled, these unlabeled edges are classified accordingly. And the complete edge maps are then used to produce the final outputs. Nevertheless, such a presumption is rather heuristic and does not necessarily hold for all unlabeled edges.

Punnappurah et al. utilize dual-pixel sensors of a digital single-lens reflex (DSLR) camera to obtain left and right views from a single shot, which effectively constitute a two-view LF [65]. Providing that the background is in focus and thus has zero disparity between the two views while reflection is not, gradients that belong to the background can be identified and its image recovered through subsequent optimization. Yet this condition only holds in limited situations.

2.3 Material recognition

Existing material recognition methods can be classified into two categories, depending on the type of input images they use. Some methods work with single view

images. Since single view images are easy to capture, these methods are applicable in nearly all scenarios. Meanwhile every type of material has its unique reflectance that determines its appearance under different viewing conditions. Such reflectance can be represented by the Bidirectional Reflectance Distribution Function (BRDF), which maps intensity values to different viewing and lighting directions. In order to use reflectance as a descriptor, various methods use images of different viewing and/or lighting directions that effectively form a subset of the complete BRDF. Nevertheless, special setups are often needed to obtain these images, which limits the applicability of these methods.

Single-view methods depend on cues such as color, texture, and shape to identify the material class. Based on various hand-crafted features, [66] maps the input images to sets of visual words, which are then classified using a trained augmented latent Dirichlet allocation model. Hu et al. also use hand-crafted features, but they only utilize patch-level kernel descriptors for material recognition [67]. Unlike other existing methods, [68] only models inherent material properties by using small image patches that do not contain any shape information. Binary similarity annotations on a set of such image patches are used to estimate perceptual distances between different material types, which are then utilized to identify attributes defining material types. A regressor is derived to map input images to their likelihoods of exhibiting these attributes, in order to recognize their material types. Having achieved considerable success in image understanding, neural networks are used to recognize material types by many researchers. Instead of hand-crafted features, [69] uses feature maps from convolutional layers, as well as the penultimate fully-connected layer of the VGG model to construct image descriptors. The outputs of the convolutional layers are used to compute a Fisher Vector, an orderless descriptor for local texture features. On the other hand, since the output of the fully-connected layer underlines the spatial ordering of pixels, it is used as an object descriptor directly. Material class of the image can then be identified by applying Support Vector Machine (SVM) classifiers on these descriptors. Xue et al. apply neural networks in such an end-to-end fashion by using a fully connected layer followed by softmax activation to calculate the likelihoods of various material classes [70]. The proposed network includes two modules: one to extract shape features encoded in the spatial relationship of pixels, and the other to extract texture features that are largely local. The two types of features are then combined using a bilinear model, before being inputted to the fully-connected

layers to predict material classes. Like [68], [71] also proposed identifying the material categories based on local attributes. Nevertheless [71] combines the attribute discovery and the material recognition steps using a neural network. [72] utilizes ensemble feature learning to combine multiple types of trained material classifiers and then generates the final predictions. Bell et al. aim to generate labels for images containing multiple types of materials [73]. Neural networks are used to learn the mapping of image patches to their material labels. Given predicted labels at various image coordinates, a fully-connected conditional random field model is then applied to obtain dense material predictions for all pixels.

As it is difficult to deduce reflectance from single-view inputs, various methods suggest including additional images in order to better capture such properties. [74] uses a dome equipped with multiple lamps to capture a scene point under different lighting conditions, which constitute a 2D slice of its BRDF. The brightness values captured are used to estimate Hemispherical Harmonic (HSH) coefficients of the scene point, which characterize its BRDF with only 9 elements. Given a set of labeled pixels with their HSH coefficients, a multi-class SVM classifier is trained to recognize material types based on HSH coefficients calculated. [75] uses a mirror-based camera to obtain samples of the BRDF. But unlike [74] that utilizes various lighting conditions, [75] captures brightness values of a scene point under different viewing angles as a reflectance disk image where each pixel corresponds to a specific direction. Texton features calculated from a set of labelled reflection disk images to reflect angular gradients are then used to identify material types. [76] use images from two different viewing angles, which is considerably fewer than the thousands of angles used in [75]. One of the two images, as well as their pixel-wise differences that reveal reflectance information, is inputted into a neural network to predict the material type. Mobile robot equipped with a camera is needed to obtain input images at specific viewing angles. [77] uses LF data. LF data consist of images from many closely placed viewpoints, which are samples of 4D BRDFs. A neural network is then used to determine the material types. [74], [75], [76], and [77] all achieve high accuracy, but the need to photograph materials using specialized devices limits their applicability.

Chapter 3

Novel View Synthesis with Planar Structure Guidance

3.1 Introduction

Novel view rendering is a major topic in computer vision with many applications in virtual environment simulation. It is closely associated with scene reconstruction in that proxy geometry needs to be estimated first before we can render the novel view. Some of the existing methods achieved this by utilizing the link among projected pixels of the same scene point in different input images. Others infer scene geometry with single-image inputs, using the implicit priors learned by neural networks. These methods all focus on scene points captured as pixels in the input images. While being able to recover the geometry of these scene points correctly, they are not good at predicting geometry beyond the captured part. As displacements increase, fraction of the target image that can be faithfully derived using the estimated geometry decreases. This drawback restricts probable novel views to those with limited displacements from the input.

To address this flaw, we employ the ability of neural networks to generate new data based on the learned statistics, together with a proposed approach to represent scene geometry effectively. Specifically, we approximate an indoor scene as a collection of 3D planar surfaces, and use a set of projected plane images to roughly represent it. A trained neural network is then used to recover scene geometry beyond the part captured, which are represented as described before. This way to

represent proxy geometry is particularly useful to subsequent geometry prediction. [1] has also noticed that scene points not captured in the images are not reconstructed properly. But [1] focuses on using neural networks to manage disocclusions of scene points obscured by those in front, instead of expanding the captured geometry, which likewise only works with novel views of limited displacements.

3D planes are essential to scene reconstruction, as they include important geometric priors. If one takes a look at his/her surroundings, he/she will find a lot of planar surfaces. This is particular true for indoor scenes, where more than half of the geometry can be regarded as 3D planes. Therefore we introduce a new approach to approximate the geometry as a set of major planes. Our approach models scene points of the same plane together. As points belonging to the same plane often form a highly regular appearance, inferring out of sight ones is rather straightforward, if provided with part of the plane image. If scene points are arranged based only on their depths, as in numerous existing methods, these belonging to the same plane will be divided across different depths, making it difficult to predict uncaptured geometry accurately.

Given the image displaying partial scene points of a plane, to infer additional points on that plane is to draw missing areas based on available segments. Our approach involves two steps, contour completion and texture inference. Separating the tasks of structure and appearance inference facilitates the modeling of priors and is shown to achieve superior inpainting results. Our method, in particular, first restores the contours of planes. Plane images can then be completed by simply extending the available textures to areas indicated by the full contours.

In summary, our key contributions are as follows:

- We propose a new way to assemble proxy geometry, which underlines the collective patterns of scene points on the same 3D planes.
- Based on the proposed proxy geometry construction, a method to synthesize novel views is developed. It utilizes the regular shapes and textures of 3D planes to predict uncaptured scene points, and thus can synthesize views that are far from the input.

FIGURE 3.1: Diagram illustrating how a 3D point is projected to different pixel positions $[x_r, y_r]$ and $[x_t, y_t]$ in the reference and the target images. The correspondence between $[x_r, y_r]$ and $[x_t, y_t]$ is stated in Eq. (3.1).

3.2 Proposed Method

3.2.1 Planar Structure Guidance

As presented before, it is advantageous to formulate the 3D structure of an indoor scene as a set of 3D planes, for the novel view synthesis task.

Assume that we are given a reference image that includes projections from k different planes, with areas and parameters specified. The reference image is transformed into a set of k images, $P_1, P_2, \dots, P_k$, with each of them composed of areas that belong to one of the k planes. These images are 2D projections of respective 3D planes onto the input viewpoint.

As shown in Fig. 3.1, R and t denote the rotation and translation from reference to target coordinate system. The correspondence between pixels from reference and target images of a particular plane can then be specified as follows:

$$\begin{bmatrix} x_t \\ y_t \\ 1 \end{bmatrix} \propto I \left(R^T + \frac{R^T t n R^T}{-p - n R^T t} \right) I^{-1} \begin{bmatrix} x_r \\ y_r \\ 1 \end{bmatrix}, \quad (3.1)$$

where $n \in \mathbb{R}^{1 \times 3}$ and $p \in \mathbb{R}^{1 \times 1}$ are parameters of that particular plane in the reference coordinate system, with all of its 3D points $[X_r, Y_r, Z_r]$ defined by the equation $u \cdot [X_r, Y_r, Z_r] = p$, and $I \in \mathbb{R}^{3 \times 3}$ denotes the camera intrinsics. We can then obtain target plane images, $\hat{P}_1, \hat{P}_2, \dots, \hat{P}_k$, by resampling their respective reference counterparts based on Eq. (3.1).

The initial set of target plane images obtained by resampling only includes parts visible from the reference viewpoint. We need to be able to extrapolate them to synthesize parts unobserved in the target image. Our approach includes three steps: contour completion, texture inference, and summation. The first step calculates contours, $M_1, M_2, \dots, M_k$, specifying areas in the plane images that are now visible. The second step generates extended textures, $C_1, C_2, \dots, C_k$, in order to fill areas identified by the new contours. In the final step, M_i and C_i are multiplied to produce the projection in the target image, for each of the planes. The final step sums the computed projections, ground-truth projections of planes not present in the reference image, and ground-truth non-planar pixels to produce the estimated target image. A detailed description of how the first and second step are implemented can be found in section 3.2.2.

3.2.2 Network Architecture

The proposed method performs two individual tasks, contour completion and texture inference, before generating the target image. Below we will discuss in detail how the two tasks are achieved using trained neural networks.

3.2.2.1 Contour Completion Network

The first task is to complete the partial plane contours in the target image. It can be thought of as a classification problem, which is resolved with a trained neural network. To be more precise, initial contours are consolidated into one-hot vectors classifying which 3D planes the pixels are on. Those without a predefined outcome are assigned the vector of all zeros. With the initial classification $P_{initial}$ of only certain areas, we use a neural network to determine that of the entire image P_{final} , which is then converted back into plane contours. As opposed to processing each plane individually, this joint approach utilizes the spatial relationship among

various initial plane contours in order to generate results that are more consistent as a group.

Our network adopts the U-Net architecture [78], with skip connections placed between low and high-level features. A summary of the proposed architecture can be found in Fig. 3.2. In this network, downsamplings are achieved through convolutions with a stride of two. Meanwhile upsamplings are done in two different ways. When a skip connection is involved, the tensor is upsampled by repeating its rows and columns one by one. After being concatenated with an earlier tensor through the skip connection, the upsampled tensor is then passed through a convolution with a stride of one to complete the procedure. When skip connections are not involved, downsamplings are simply implemented by transposed convolutions with a stride of two.

During training, the masked cross-entropy loss between P_{final} and the actual classification P_{target} , which does not include areas of non-planar pixels and areas of planes only present in the target, is used for backpropagations. It is worth mentioning that instead of regular convolutions, partial convolutions introduced in [79] are used, which we will elaborate on in the upcoming section.

3.2.2.2 Texture Inference Network

To synthesize textures not captured by the camera, we input the incomplete texture of a plane $T_{initial}$, the corresponding mask $M_{initial}$, and its Sobel edge map E . We include the Sobel edge map as a part of the input, as it helps generate outputs with fine details [3]. $T_{initial}$ and E are concatenated along the channel dimension and used as the direct input $F_{initial}$. Meanwhile the input mask $M_{initial}$ is used to guide the partial convolutions carried out by the network.

The architecture is very similar to that of the contour completion network, as shown in Fig. 3.2. The difference is in the first layer, since inputs of the texture inference network have a different channel number. The first skip connection is also moved to the next layer, in order to increase the valid receptive field.

Just like in the contour completion network, we use partial convolutions instead of regular ones. A partial convolution operation takes both input tensor F_{in} as well as its respective mask M_{in} identifying elements with valid values, and uses only

FIGURE 3.2: Diagram of the contour completion and the texture inference network. All down convs, up convs, and convs are partial convs described in Section 3.2.2.2

valid ones to compute the output

$$F_{out} = (F_{in} * K \circ W) + b, \quad (3.2)$$

with

$$W = (M_{ones} * K_{ones}) \odot (M_{in} * K_{ones}). \quad (3.3)$$

K and b represent the kernel and bias. M_{ones} and K_{ones} have the same shapes, but with all elements set to one. $*$, $\circ$, and $\oslash$ stand for convolution, Hadamard product, and Hadamard division respectively. W indicates that output values should be weighted, based on the numbers of valid elements conditioned on by the kernel when they are computed. The new mask is then calculated as

$$M_{out} = \begin{cases} 1 & \text{if } M * K_{ones} > 1 \\ 0, & \text{otherwise.} \end{cases} \quad (3.4)$$

The output mask M_{out} is a dilated version of the input mask M_{in} , as more elements are conditioned on valid values after each convolution. And as the input passes through more partial convolutions further into the network, its mask expands gradually during the process.

Regular convolutions, on the contrary, do not distinguish between true elements of the texture, and the other ones. As the available segment generally occupies only a portion of the full image, the output then very much depends on the values we initially assign to the unknown elements. Furthermore, without the weighting scheme, parts of the final output with different receptive fields, e.g. ones consisting mostly of valid elements and ones consisting mostly of ones with the assigned value will require different sets of kernels, in order to generate the targeted output. This conflict will cause subpar training results.

3.3 Experiments

In this chapter, we present and discuss results obtained using the proposed method. In Section 3.3.1, we present the outputs, both the final and the intermediate ones, and discuss how different steps of our method contribute to the final output. We then evaluate the outputs, in comparison with those obtained using other methods, in Section 3.3.2. Finally in Section 3.3.3, we analyze failure cases.

FIGURE 3.3: Final results after combining the output results of the contour completion and the texture inference networks.

3.3.1 Results

The network was trained using the ScanNet dataset [80], and the plane annotations are from [81]. In Fig. 3.3, we show the final synthesized target images along with ground-truth ones. We also show the input images. It is clear that the proposed method can accurately reconstruct images of various novel views, even when respective scene points are not fully captured in the input.

We present results of the intermediate steps, contour completion and texture inference, in Fig. 3.4 and 3.5. In Fig. 3.4, we show how full plane contours are inferred. Initial plane contours, which are calculated based on partial ones warped from the input, are shown in the fourth row. The final plane contours are shown in the third row. The final outputs are quite close to actual ones in the second column. Even the initial outputs are reasonable. In Fig. 3.5, we show results of the second step. Input textures of individual planes are shown in the first column. Extended textures are shown in the third column, which are perceptually very similar to target textures in the second column. Areas of the output and target textures need not agree, as it is the first step that determines which part of the extended textures will be visible. In fact, area of the output texture should be larger than or equal to

FIGURE 3.4: Contour completion results. Here we show target contours completed based on input ones warped from reference contours.

that of the target, where the extra texture is just parts of the plane invisible from the designated viewpoint.

3.3.2 Comparisons with baseline methods

In this section, we further evaluate the performance of our method, by comparing it with two other methods.

FIGURE 3.5: Texture inference results of various planes. One can see that extra textures inferred are consistent with given parts in (a).

We first compare our method against a baseline method. To be more specific, given an incomplete image warped from the reference viewpoint, we simply inpaint it to obtain the final image, without utilizing the geometric cue. And we use a trained neural network [3] to perform inpainting, with the incomplete image and mask specifying areas to be inpainted as the input.

The second method is a state-of-the-art novel view synthesis method presented in [1]. As discussed before in section 2.1, it addresses the issue caused by unobserved scene points, when the viewpoint changes. [1] uses the reference image, as well as images of two additional viewpoints, as the input. The reference image used is the same for both our method and [1]. The two extra images are randomly selected from nearby frames in the sequence.

The implementation of both methods is based on publicly available code uploaded by authors of [3] and [1] on GitHub ^{1 2}. We did not re-train the networks with

¹<https://github.com/knazeri/edge-connect>

²https://github.com/google-research/google-research/tree/master/mpi_extrapolation

our dataset. Network parameters provided were used.

In Fig. 3.6, we show target images obtained using the three methods, as well as the groundtruth. Target images estimated using the proposed method look very similar to the groundtruth. Meanwhile one can see from the estimated image of the second scene that [3] can sometimes colorize uncaptured scene points in a way that is inconsistent with available segments. [1] does not infer uncaptured scene points very well, and thus its output images have visible black areas. To quantitatively evaluate our method, the structural similarity index measure (SSIM) values between the estimated and the ground-truth target images are calculated. Our method achieves the average SSIM value of 0.83.

3.3.3 Failure case studies

Our method failed to generate adequate results in some cases. Below we discuss the two main issues.

One issue is that our texture inference network may not be able to expand the existing texture of a plane large enough to fill all of its projected area in the target image. To be more specific, as a result of the U-Net architecture, output features of the earliest layer in the skip connections decide which elements in the final image are filled. However the ratios of valid elements in earlier features only increase moderately from that of the input. If the final area of a plane is significantly larger than the initial one, it will not be fully inpainted. One such case is shown in Fig. 3.7. The contour predicted is mostly correct, but it includes areas outside the expanded texture. Consequently the estimated plane image, as well as the final output, will be incomplete.

Another issue is with data labeling. We noticed imprecise plane parameters and segmentations in the dataset, which can cause artifacts in the final output.

3.4 Conclusion and Future Work

In this chapter, we present our method to synthesize images of novel viewpoints. Since indoor scenes are mainly composed of 3D planes, we reformulate the problem

of estimating target images into one of estimating new contours and textures for these planes. The target images are then constructed using the estimated plane contours and textures. Scene points of different planes form different textures. By grouping scene points based on the 3D planes they belong to, we are able to infer scene points not captured in the input images effectively and generate more realistic-looking target images.

The bottleneck of our method is the receptive field in the texture inference network. With regular convolutional operations, we can only expand the existing plane textures so far. In some cases, the results may not be large enough. In the future, we plan to explore non-local operations such as the one presented in [82] and [83] to address this problem.

(a): input images (reference and additional viewpoints)

(b): target images (groundtruth)

(c): estimated target images (proposed method)

(d): estimated target images ([3])

(e): estimated target images ([1])

(f): estimated target images ([2])

FIGURE 3.6: Comparisons. We compare with results of [1] [2] (novel view synthesis methods), as well results obtained after using [3] (an image inpainting method) to complete the warped reference images.

FIGURE 3.7: A failure case. For the plane in the yellow polygon, the expanded texture does not fully cover the area inside the estimated contour.

Chapter 4

Reflection Removal On Single Light Field Capture Using Focus Manipulation

4.1 Introduction

A large portion of this chapter is extracted from a paper, "Reflection removal on single light field capture using focus manipulation", published in the IEEE Transactions on Computational Imaging [84] co-authored by the author of this thesis.

When one takes pictures of a scene through reflective surfaces, such as glass windows, glass panel covered sign boards, and so on, captured images often contain reflection. Sometimes this causes difficulties to recognize details of background. Besides obscuring details of background, reflection also introduces additional features not belonging to background. Therefore such images are difficult for subsequent computer vision applications such as stereo reconstruction, object recognition, or tracking.

Reflection removal is part of a bigger effort to tackle problems caused by layer superimposition [85][86]. There are twice as many unknowns as observations, as we need to infer two layers out of one captured image. In other words, the problem is ill-posed, with no definitive solution. Moreover, the two layers could be of a

FIGURE 4.1: (a) a real scene captured using Lytro. (b) edges labeled by our method (background in red, reflection in cyan). (c) recovered background. (d) recovered reflection.

similar nature, which makes it even more difficult to narrow down the possible candidates.

Existing reflection removal methods can be divided into two groups, multiple-image and single-image ones, based on their inputs. Methods of the first group use images taken from shifted positions [36–39, 41–45] or with and without a polarizer [40]; while the other group use single shots as their input [46–57]. In general, multiple-image methods achieve more satisfactory separation results, as there is more information to be explored. However, they do not work with dynamic scenes containing moving objects, which would have changed a lot over the duration of multiple acquisitions. Meanwhile single-image methods often have to rely upon manual inputs to generate comparable results [46] [47] [54], or only work in limited scenarios.

We propose to separate the two layers using single captures taken with a consumer LF camera. Unlike conventional cameras, they record radiances of incoming

rays, as well as their directions, which are shown to be useful in many applications such as depth estimation [15, 16, 18, 58, 87–90], motion deblurring[91], view synthesis[92], super-resolution [93], post-capture refocusing [14][94] and intrinsic image decomposition[95]. Using LF cameras gives us the advantages of both multiple-image and single-image methods. We are able to analyze the scene from different views, while only a single shot is needed. Baseline of LF data is much narrower though, and consequently perspective differences to be explored are much vaguer, compared to multiple-image input. Still we believe the benefit of requiring just single shots makes it favorable to use LF data. [58, 61–63, 96] also utilize LF data to separate background and reflection. Meanwhile [65] uses data with a similar structure captured using dual-pixel sensors, which are basically two-view LF.

An earlier publication of our work is able to generate rough estimates of the layers [64]. This paper gives greater insight into the problem and can produce much better results by specifying detailed focus manipulation procedures, enforcing the EPI consistency constraint, and implementing an alternating optimization scheme. Furthermore, an alternating scheme that takes turns estimating the layers and their disparities is also introduced. The proposed algorithm improves separation results of [64] significantly, as demonstrated by experiments. See Fig. 4.1 for example results obtained using our method.¹

4.2 Proposed Method

In this section, we first analyze optical geometry of how scene points at different distances are projected on EPIs, in section 4.2.1. We then introduce the way to label edge points of two layers accordingly in section 4.2.2. In section 4.2.3, we show how initial layer estimates are obtained, given labeled edges scene points. In section 4.2.4, we present the EPI consistency constraint. Finally, section 4.2.5 and 4.2.6 detail how it is used to generate final improved separation results.

FIGURE 4.2: Illustration of the two plane representation for LF.

4.2.1 Optical Geometry Analysis

LF is a vector function that describes the amount of light propagating in every direction through every point in space [97]. Conventional cameras use converging lenses to record a 2D projection of high dimensional LF data. This inevitably loses a lot of information, especially each light ray’s directional information. LF cameras are designed to resolve the integration process and capture the extra directional information.

As illustrated in Fig. 4.2(a), the light field is usually represented as four-dimensional data: $\mathbf{L}(\mathbf{x}, \mathbf{y}, \mathbf{s}, \mathbf{t})$, in which (x, y) represents the spatial plane, and (s, t) represents the angular (sub-aperture) plane. Fig. 4.2(b) shows the camera sensor of a micro-lens array based LF camera. Each micro-lens covers a group of pixels inside the dotted gray circle. Collocated pixels under each micro-lens (e.g., the ones lighted in red) can be extracted and converted into a sub-aperture-image (SAI), which represents the vision from a specific viewing angle at the sub-aperture plane. Similarly, the blue pixels can form another SAI. In this paper, we denote a SAI by $\mathbf{I}^{s_c+p, t_c+q}$, where s_c+p, t_c+q in the superscript denotes its view index offset from the central view s_c, t_c , with $p \in \{-N, -N+1, \dots, N\}$ and $q \in \{-N, -N+1, \dots, N\}$. That is to say we have a total of $(2N+1)$ -by- $(2N+1)$ SAIs. We used Light Field Toolbox for Matlab v0.4 to decode raw LF data in our experiments, which gave us 15-by-15 SAIs with $N = 7$ [98]. For simplicity, we denote the central view SAI $\mathbf{I}^{s_c, t_c}$ by $\mathbf{I}^c$.

¹In this paper, we enhanced contrast of images for better visibility.

For the 4D light field data $\mathbf{L}(\mathbf{x}, \mathbf{y}, \mathbf{s}, t)$, when we fix coordinates in both spatial and angular domain, i.e., $y = y'$, and $t = t'$, the light ray will be restricted to a subspace $\mathbf{L}(\mathbf{x}, \mathbf{y}', \mathbf{s}, t')$, and the resulted image is called EPI, just like these shown in bottom row of Fig. 4.3. Slopes of linear structures on EPI correspond to spatial disparities of scene points across different viewing angles, as have been analyzed in [99]. By identifying slope of each tilting line on EPI, a disparity map, and further, depth map can be generated [15] [16] [17]. In this paper, we use EPI slope and disparity interchangeably.

For scene points at different distances, their EPI slopes show different patterns. We look into it by analyzing the optical geometry inside a LF camera. Camera focus can be adjusted by changing the separation between the sensor plane and the main lens. As shown in Fig. 4.3(a), when the sensor plane (x plane) is fixed, the focused scene plane within which sharp image forms is also determined. Furthermore, in Fig. 4.3(a), the cone of red rays from the focused scene plane, with different ray directions spanning the angular domain, is focused at the sensor plane, and thus produces a vertical line on the EPI. Note that for simplicity, the angular plane (s plane) is overlaid with the main lens plane. The micro-lens array has been ignored since they do not alter the relationship among the SAIs. As shown in Fig. 4.3(b), for scene points nearer than the focused scene plane, its image plane goes over the sensor plane. And the cone of green rays forms the green strip with positive slope on the EPI. Similarly, scene points further than the scene focused plane will produce negative EPI slope, as shown in Fig. 4.3(c). The pos/negativity pattern of the EPI slope is not only useful to provide scene distance information with respect to camera focus, but also provides an unique means to distinguish background from reflection via camera focus adjustment.

4.2.2 Focus Manipulation for Background/Reflection Classification

According to our analysis in Sec 4.2.1, when camera is focused somewhere in between background and reflection, opposite EPI slope signs of the two layers allow us to identify their scene points accordingly. We list below focus manipulation procedures to ensure that the LF camera is focused at the right place:

FIGURE 4.3: Optical diagram on the formation of EPIs in a LF camera, for scene points at different distances around the focused scene plane.

FIGURE 4.4: Classified edges of a scene where the background (buildings) is afar and reflection (person taking the picture) is near. In (b), background edges labeled are marked in red, while reflection edges labeled are marked in cyan.

- Estimate depths of background by marking the sharpest point while twisting the main lens dial or utilizing Lytros depth features [100].
- Estimate depths of reflection in the same way. Note that depths of reflection refer not to those of scene points to be reflected, but to depths of their virtual image.

- Set the camera focus somewhere in between depths of the two layers, and take a capture. Specifically, since both layers span a range of depths, the camera focus should be somewhere between boundaries of the two, e.g., between the upper-bound of background depths and lower-bound of reflection depths, if background is near while reflection is afar.

Following the above procedures, EPI slopes of background and reflection scene points will be of opposite signs, and we can label them accordingly.

Specifically, we first adopt the method introduced in [17], to generate rough EPI slope estimates for all scene points. We calculate variances of pixels, along different EPI slopes, and choose ones with the lowest variances. That is to say for each EPI slope candidate d , we first sample the corresponding set of radiance values across the angular s or t -dimension, for the $2N + 1$ views. Then we calculate the variance among this set of $2N + 1$ values. Given variances of all EPI slope candidates, one with the lowest value is identified as the scene point's initial disparity. A detailed analysis can be found in [17].

Regarding the EPI slope candidates, we used those separated by a uniform spacing of 0.05 between -1.0 and 1.0. The rough EPI slope estimates are not always credible, especially those of heavily superimposed or textureless regions. Therefore we only consider the edge scene points whose summed differences between their horizontal and vertical 1D 9-neighbors are above a certain threshold, as their EPI slopes are clearer and thus easier to identify. The threshold was set to 0.05 in our experiments.

Then based on pos/negativity of their EPI slopes, these edges scene points are classified into the background set S_1 or the reflection set S_2 . In order to reduce noises, isolated elements of both sets are discarded. Specifically, those with less than or equal to 12 thresholded edge points within 7-by-7 neighborhoods are removed in our experiment. Since edge points of the two layers commonly do not overlap, elements of S_1 that are 1 pixel away from those of S_2 are discarded, and vice versa. Finally, we notice artifacts around the top and sides boundaries. Hence elements of S_1 and S_2 that are no more than 6 pixels away from these boundaries are discarded as well.

Fig. 4.4 shows S_1 and S_2 of a scene where background is far while reflection is near. One can see that most of the background and reflection edges are correctly identified by our method.

4.2.3 Initial Layer Separation

Given labeled edge scene points of both layers, initial background and reflection, $\hat{\mathbf{I}}_b$ and $\hat{\mathbf{I}}_r$, of the central view SAI $\mathbf{I}^c$ can be obtained, after solving a constrained optimization problem. As presented in [47], a mix of two Laplacian distributions approximates edge distribution of natural images well. Furthermore, this prior, together with labeled background and reflection edge scene points, forms a proper objective function with respect to the two layers. Therefore we adopt its objective function,

$$\begin{aligned} \min_{\hat{\mathbf{I}}_b, \hat{\mathbf{I}}_r} = & \sum_{i,k} \rho(\mathbf{F}_{i,k} * \hat{\mathbf{I}}_b) + \sum_{i,k} \rho(\mathbf{F}_{i,k} * \hat{\mathbf{I}}_r) \\ & + \beta \sum_{i \in S_1, k} \rho(\mathbf{F}_{i,k} * \mathbf{I}^c - \mathbf{F}_{i,k} * \hat{\mathbf{I}}_b) \\ & + \beta \sum_{i \in S_2, k} \rho(\mathbf{F}_{i,k} * \mathbf{I}^c - \mathbf{F}_{i,k} * \hat{\mathbf{I}}_r), \end{aligned} \quad (4.1)$$

where ρ calculates the log probability of a layer under the Laplacian mixture edge distribution assumption, $\mathbf{F}_{i,k}$ is the k^{th} order linear derivative filter at pixel location i , and $*$ denotes a 2D convolution operator. In our experiments, first and second order linear derivative filters were used. The first two terms make sure that edges of the output background and reflection will follow the Laplacian mixture distribution and thus look natural, as low-probability image derivatives will be punished. Meanwhile the last two terms enforce the initial edge classification results and make sure gradients of $\hat{\mathbf{I}}_b$ and $\hat{\mathbf{I}}_r$ at their labeled edge scene points equal those of the input.

Eq. (4.1) is then minimized using iteratively reweighted least squares (IRLS) algorithm to generate initial layer estimates. It is worth noting that [47] required user markup to label background and reflection scene points. In our method though, edge scene points are labeled automatically, based on signs of their EPI slopes, as has been explained in Sec. 4.2.2.

4.2.4 EPI Consistency Constraint

Assuming disparities, $\mathbf{D}_b$ and $\mathbf{D}_r$, and the two layers, $\mathbf{I}_b$ and $\mathbf{I}_r$, are known (we will explain how initial disparities are obtained in Section 4.2.5), we can assemble

FIGURE 4.5: Comparison between an actual input EPI and one constructed from initial layer and disparity estimates. One can see that the two are different (see close ups of two segments in red and cyan).

their horizontal EPIs,

$$\begin{aligned}\tilde{L}_b(\mathbf{x}, \mathbf{y}', \mathbf{s}, \mathbf{t}_c') &= \mathcal{W}_h(\mathbf{D}_b(\mathbf{x}, \mathbf{y}'), \mathbf{I}_b(\mathbf{x}, \mathbf{y}')), \\ \tilde{L}_r(\mathbf{x}, \mathbf{y}', \mathbf{s}, \mathbf{t}_c') &= \mathcal{W}_h(\mathbf{D}_r(\mathbf{x}, \mathbf{y}'), \mathbf{I}_r(\mathbf{x}, \mathbf{y}')), \end{aligned} \quad (4.2)$$

where $\mathcal{W}_h: \mathbb{R}^{n \times 3} \rightarrow \mathbb{R}^{(2N+1) \times n \times 3}$ represents the transformation of a row, given the disparities, to its horizontal EPI. And the sum of assembled background and reflection horizontal EPIs should give us assembled horizontal EPIs of the input,

$$\tilde{L}(\mathbf{x}, \mathbf{y}', \mathbf{s}, \mathbf{t}_c') = \tilde{L}_b(\mathbf{x}, \mathbf{y}', \mathbf{s}, \mathbf{t}_c') + \tilde{L}_r(\mathbf{x}, \mathbf{y}', \mathbf{s}, \mathbf{t}_c'). \quad (4.3)$$

Likewise we assemble vertical EPIs of the two layers,

$$\begin{aligned}\tilde{L}_b(\mathbf{x}', \mathbf{y}, \mathbf{s}_c', \mathbf{t}) &= \mathcal{W}_v(\mathbf{D}_b(\mathbf{x}', \mathbf{y}), \mathbf{I}_b(\mathbf{x}', \mathbf{y})), \\ \tilde{L}_r(\mathbf{x}', \mathbf{y}, \mathbf{s}_c', \mathbf{t}) &= \mathcal{W}_v(\mathbf{D}_r(\mathbf{x}', \mathbf{y}), \mathbf{I}_r(\mathbf{x}', \mathbf{y})), \end{aligned} \quad (4.4)$$

where $\mathcal{W}_v: \mathbb{R}^{m \times 3} \rightarrow \mathbb{R}^{m \times (2N+1) \times 3}$ represents the transformation of a column, given the disparities, to its vertical EPI. And their sum should give us assembled vertical EPIs of the input,

$$\tilde{L}(\mathbf{x}', \mathbf{y}, \mathbf{s}_c', \mathbf{t}) = \tilde{L}_b(\mathbf{x}', \mathbf{y}, \mathbf{s}_c', \mathbf{t}) + \tilde{L}_r(\mathbf{x}', \mathbf{y}, \mathbf{s}_c', \mathbf{t}). \quad (4.5)$$

If layers and their disparities are accurate, assembled EPIs, $\tilde{L}(\mathbf{x}, \mathbf{y}', \mathbf{s}, \mathbf{t}_c')$ and $\tilde{L}(\mathbf{x}', \mathbf{y}, \mathbf{s}_c', \mathbf{t})$, should equal actual input EPIs, $L(\mathbf{x}, \mathbf{y}', \mathbf{s}, \mathbf{t}_c')$ and $L(\mathbf{x}', \mathbf{y}, \mathbf{s}_c', \mathbf{t})$. However, as one example shown in Fig. 4.5, input EPIs obtained using initial layer and disparity estimates do not agree with the actual input EPIs. This indicates errors in initial estimates. To improve upon them, we propose to include a EPI consistency constraint that penalizes the difference between assembled input EPIs

FIGURE 4.6: Disparity interpolation based on superpixel segmentation. (a) initial layer estimates of the scene shown in Fig. 4.1(a). (b) superpixels of initial layer estimates. (c) disparities of edge scene points. (d) initial semi-dense disparities interpolated from (c). Disparities are displayed using scaled colors. Top: background. Bottom: reflection.

and actual ones. The assumption that the sum of background and reflection EPIs equals input EPI is noted in [60] as well.

4.2.5 Final Layer Separation

Based on Eq. (4.2) to (4.5), the final objective function with EPI consistency constraint is:

$$\begin{aligned} \min_{\mathbf{I}_b, \mathbf{I}_r, \mathbf{D}_b, \mathbf{D}_r} = & \lambda_1 K_1 + \lambda_2 K_2 + \lambda_3 K_3 + \lambda_4 K_4 \\ & + \lambda_5 K_5 + \lambda_6 K_6, \end{aligned} \quad (4.6)$$

where

$$\begin{aligned} K_1 = & \sum_{y'=1}^m |L(\mathbf{x}, \mathbf{y}', \mathbf{s}, t_c') - \tilde{L}(\mathbf{x}, \mathbf{y}', \mathbf{s}, t_c')| \\ & + \sum_{x'=1}^n |L(\mathbf{x}', \mathbf{y}, \mathbf{s}_c', t) - \tilde{L}(\mathbf{x}', \mathbf{y}, \mathbf{s}_c', t)|, \end{aligned} \quad (4.7)$$

$$K_2 = |\mathbf{I}^c - \mathbf{I}_b - \mathbf{I}_r|, \quad (4.8)$$

$$\begin{aligned} K_3 = & \sum_{i \in S_{1,k}} |\mathbf{F}_{i,k} * \mathbf{I}^c - \mathbf{F}_{i,k} * \mathbf{I}_b| \\ & + \sum_{i \in S_{2,k}} |\mathbf{F}_{i,k} * \mathbf{I}^c - \mathbf{F}_{i,k} * \mathbf{I}_r|, \end{aligned} \quad (4.9)$$

$$K_4 = \sum_{i,k} |(\mathbf{F}_{i,k} * \mathbf{I}_b)(\mathbf{F}_{i,k} * \mathbf{I}_r)|, \quad (4.10)$$

$$K_5 = \sum_{i,k} |\mathbf{F}_{i,k} * \mathbf{I}_b| + \sum_{i,k} |\mathbf{F}_{i,k} * \mathbf{I}_r|, \quad (4.11)$$

$$K_6 = \sum_{i,k} |\mathbf{F}_{i,k} * \mathbf{D}_b| + \sum_{i,k} |\mathbf{F}_{i,k} * \mathbf{D}_r|. \quad (4.12)$$

K_1 enforces assembled EPIs to equal input EPIs, for m rows and n columns of central view SAI. K_2 penalizes the difference between the central view SAI and sum of obtained background and reflection. K_3 is inherited from previous objective function Eq. (4.1), to enforce edge scene points classification results. Background and reflection represent different scenes, and their edges generally do not overlap. K_4 ensures this mutual exclusiveness of background and reflection edges. Like K_3 , it will prevent residuals of one layer in another, and ensure a clean separation between the background and reflection. K_5 and K_6 are commonly used priors. For K_5 , it is well known that gradient fields of natural images tend to be sparse [46]. Hence it is used to enforce sparsity on gradients of recovered backgrounds

and reflections. Meanwhile K_6 is the smoothness prior that is generally used as a regularization term in estimating disparity/depth maps [101][102][103]. These two terms, K_5 and K_6 , will further constrain the recovered layers and disparities to be reasonable.

In terms of weights, λ_1 to λ_5 are set to 1, and λ_6 is set to 0.1 for all separation results shown in this paper. λ_6 is significantly smaller than the other weights, because most disparities of LF data are well within the plus-or-minus one pixel range. Larger weights can lead to over-smoothing and erase important details.

Disparity interpolation based on superpixel segmentation. Proper initial guesses of variables $\mathbf{I}_b$, $\mathbf{I}_r$, $\mathbf{D}_b$, and $\mathbf{D}_r$ are crucial to recover the two layers accurately. For the layers, we initialize with $\hat{\mathbf{I}}_b$ and $\hat{\mathbf{I}}_r$ from Sec. 4.2.3. For their disparities though, we only have sparse estimates of edge scene points in the background set S_1 and reflection set S_2 that are credible. It is difficult to put EPI consistency constraint to use effectively with few disparities available. Therefore we propose to interpolate more dense initial disparities.

Specifically, we first segment initial layer estimates into small clusters called superpixels, using structured edge detection box [104] [105]. We used the default settings in its demo code. Fig. 4.6(b) shows resulted pixel clusters of initial background and reflection. It is generally assumed that pixels inside each cluster share the same depth/disparity [106]. Thus we average disparities of edge scene points enclosed inside a region, and propagate that value to all pixels. When there are no or very few edge scene points inside a region, disparity of its pixels is set to be zero. Fig. 4.6(d) shows initial disparities, $\hat{\mathbf{D}}_b$ and $\hat{\mathbf{D}}_r$, interpolated from those of background and reflection edge scene points respectively.

The semi-dense background and reflection disparities obtained are then used as initial guesses of $\mathbf{D}_b$ and $\mathbf{D}_r$ in Eq. (4.6). By extending relatively sparse estimates to nearby scene points of similar texture and color, they are closer to actual disparities, and thus are better starting points (compare Fig. 4.6(c) and 4.6(d)). Meanwhile unlike dense disparities that assign an estimate to every scene point, there are still some regions with no disparities. It is difficult to obtain accurate disparity estimates of all background and reflection scene points, especially with noisy real-life LF data. Therefore we purposely do not assign ambiguous regions

in the semi-dense initial disparities, but derive their values subsequently during optimization, when more precise layer estimates are available.

4.2.6 Alternating optimization

It is difficult to minimize the objective function of Eq. (4.6) with respect to the two layers and their disparities at the same time. Thus we propose to separate it into two steps, **layers update** and **disparities update**, and alternate between them. In the first step, energy is minimized with respect to the two layers, $\mathbf{I}_b$ and $\mathbf{I}_r$, to generate new separation results, while their disparities, $\mathbf{D}_b$ and $\mathbf{D}_r$, are treated as constants,

$$\min_{\mathbf{I}_b, \mathbf{I}_r} = \lambda_1 K_1 + \lambda_2 K_2 + \lambda_3 K_3 + \lambda_4 K_4 + \lambda_5 K_5. \quad (4.13)$$

In the second step, energy with respect to the disparities, $\mathbf{D}_b$ and $\mathbf{D}_r$, is minimized, while the newly updated layers are treated as constants:

$$\min_{\mathbf{D}_b, \mathbf{D}_r} = \lambda_1 K_1 + \lambda_6 K_6. \quad (4.14)$$

The alternation between two steps continues until stable separation results are acquired. It takes no more than three alternations to recover the backgrounds and reflections in our experiments. In **disparities update** step, we also discard any estimate that is outside the plus-minus one pixel range. For both steps, we stopped updating its variables, when change in mean absolute error is less than one percent.

The objective function of both steps need to be rewritten in the form of $|\mathbf{A}\mathbf{x} - \mathbf{y}|$, with $\mathbf{x} \in \mathbb{R}^{2mn \times 1}$ being its variables unrolled and concatenated as a vector (in **layers update** step, we process three color channels separately), before they can be minimized using IRLS algorithm. However, the mappings of row/column to horizontal/vertical EPI, $\mathcal{W}_h$ and $\mathcal{W}_v$, are not linear with respect to the layers or their disparities. In addition, K_4 also does not fit the form of $|\mathbf{A}\mathbf{x} - \mathbf{y}|$. Below we discuss how they need to be adjusted, before we can minimize objective functions of the two steps:

4.2.6.1 Layers update

In this step, $\mathcal{W}_h$ and $\mathcal{W}_v$ of K_1 need to be linearized with respect to $\mathbf{I}_b$ and $\mathbf{I}_r$. For horizontal EPI, employing first-order Taylor series expansion, row (x, y') of SAI $\mathbf{I}^{s_c+p, t_c}$ roughly equals:

$$\begin{aligned} \mathbf{I}^{s_c+p, t_c}(\mathbf{x}, \mathbf{y}') &\approx \mathbf{I}^c(\mathbf{x}, \mathbf{y}') + p\mathbf{D}(\mathbf{x}, \mathbf{y}') \cdot \left(\frac{\partial \mathbf{I}^c(\mathbf{x}, \mathbf{y}')}{\partial x} \right) \\ &\approx (\mathbf{E} + p\mathbf{D}(\mathbf{x}, \mathbf{y}') \cdot \mathbf{H}_x) \mathbf{I}^c(\mathbf{x}, \mathbf{y}'), \end{aligned} \quad (4.15)$$

where $\mathbf{E}$ is the identity matrix, $\mathbf{H}_x$ denotes 1D linear derivative filter that calculates horizontal derivatives, and $\cdot$ is element-wise multiplication. Transformation matrix $\mathbf{E} + p\mathbf{D}(\mathbf{x}, \mathbf{y}') \cdot \mathbf{H}_x$ maps row (x, y') of central view SAI $\mathbf{I}^c$ to the same row of $\mathbf{I}^{s_c+p, t_c}$ linearly.

In regard to vertical EPI, column (x', y) of SAI $\mathbf{I}^{s_c, t_c+q}$ is written as,

$$\begin{aligned} \mathbf{I}^{s_c, t_c+q}(\mathbf{x}', \mathbf{y}) &\approx \mathbf{I}^c(\mathbf{x}', \mathbf{y}) + q\mathbf{D}(\mathbf{x}', \mathbf{y}) \cdot \left(\frac{\partial \mathbf{I}^c(\mathbf{x}', \mathbf{y})}{\partial y} \right) \\ &\approx (\mathbf{E} + q\mathbf{D}(\mathbf{x}', \mathbf{y}) \cdot \mathbf{H}_y) \mathbf{I}^c(\mathbf{x}', \mathbf{y}). \end{aligned} \quad (4.16)$$

Transformation matrix $\mathbf{E} + q\mathbf{D}(\mathbf{x}', \mathbf{y}) \cdot \mathbf{H}_y$ maps column (x', y) of central view SAI $\mathbf{I}^c$ to the same column of $\mathbf{I}^{s_c+q, t_c}$ linearly.

Horizontal EPI is rows (x, y') of SAI $\mathbf{I}^{s_c+p, t_c}$ placed on top of each other, for $p = -N$ to N , while vertical EPI is columns (x', y) of SAIs $\mathbf{I}^{s_c, t_c+q}$ placed next to each other, for $q = -N$ to N . Therefore diagonal combination of individual transformation matrices followed by subsequent operations will give us approximations of $\mathcal{W}_h$ and $\mathcal{W}_v$ that are linear with respect to $\mathbf{I}_b$ and $\mathbf{I}_r$.

As for K_4 , it is rewritten as $\sum_{i,k} |(\mathbf{F}_{i,k} * \bar{\mathbf{I}}_r)(\mathbf{F}_{i,k} * \mathbf{I}_b)| + |(\mathbf{F}_{i,k} * \bar{\mathbf{I}}_b)(\mathbf{F}_{i,k} * \mathbf{I}_r)|$, where $\mathbf{I}_b$ and $\mathbf{I}_r$ are approximate by layer estimates from previous alternation, $\bar{\mathbf{I}}_b$ and $\bar{\mathbf{I}}_r$.

4.2.6.2 Disparities update

In this step, $\mathcal{W}_v$ and $\mathcal{W}_h$ need to be linearized with respect to $\mathbf{D}_b$ and $\mathbf{D}_r$. For horizontal EPI, row (x, y') of SAI $\mathbf{I}^{s_c+p, t_c}$ is approximated as:

$$\mathbf{I}^{s_c+p, t_c}(x, y') \approx p\mathbf{H}_x \mathbf{I}^c(x, y') \cdot \mathbf{D}(x, y') + \mathbf{I}^c(x, y'), \quad (4.17)$$

where transformation matrix $p\mathbf{H}_x \mathbf{I}^c(x, y')$ connects $\mathbf{D}(x, y')$ to $\mathbf{I}^{s_c+p, t_c}(x, y')$ linearly.

For vertical EPIs, column (x', y) of SAI $\mathbf{I}^{s_c, t_c+q}$ roughly equals:

$$\mathbf{I}^{s_c, t_c+q}(x', y) \approx q\mathbf{H}_y \mathbf{I}^c(x', y) \cdot \mathbf{D}(x', y) + \mathbf{I}^c(x', y), \quad (4.18)$$

where transformation matrix $q\mathbf{H}_y \mathbf{I}^c(x', y)$ connects $\mathbf{D}(x', y)$ to $\mathbf{I}^{s_c, t_c+q}(x', y)$ linearly.

As in layers update step, diagonal combination of individual transformation matrices followed by subsequent operations give us approximations of $\mathcal{W}_h$ and $\mathcal{W}_v$ that are linear with respect to $\mathbf{D}_b$ and $\mathbf{D}_r$.

To summarize, algorithm 1 shows the main steps of proposed reflection removal method.

4.3 Experiments

In this paper, LF captures were taken using a Lytro Illum LF camera and decoded using Light Field Toolbox for Matlab v0.4 [98]. Our method was implemented in Matlab, on a PC with 16G RAM and 3.7 GHz CPU. Due to border vignetting, only central SAIs were used. Processing time varies among scenes. For example, it took 19 minutes to process the first scene in Table. 4.1, using the central 7-by-7 SAIs.

TABLE 4.1: Quantitative evaluation on synthetic scenes. The best performance of each data entry has been highlighted in bold.

		Ours	Ours (initial) [64]	Johann- sen <i>et al.</i> [58]	Li and brown [36]	Fan <i>et al.</i> [52]	Arvanit- opoulos <i>et. al</i> [51]
<i>Lock</i>	PSNR I_b	35.16	28.73	25.70	29.84	22.93	29.72
	PSNR I_r	34.86	28.70	19.53	29.65	N.A.	N.A.
	NCC I_b	0.96	0.79	0.55	0.86	0.78	0.86
	NCC I_r	0.88	0.57	0.44	0.55	N.A.	N.A.
<i>Orange</i>	PSNR I_b	32.90	24.36	27.66	23.98	20.27	25.21
	PSNR I_r	32.94	24.62	20.50	23.96	N.A.	N.A.
	NCC I_b	0.99	0.92	0.97	0.91	0.93	0.95
	NCC I_r	0.96	0.64	0.81	0.17	N.A.	N.A.
<i>Car</i>	PSNR I_b	24.83	22.90	32.76	26.04	19.29	23.87
	PSNR I_r	24.76	22.86	20.63	26.08	N.A.	N.A.
	NCC I_b	0.88	0.54	0.95	0.89	0.77	0.82
	NCC I_r	0.88	0.80	0.97	0.60	N.A.	N.A.
<i>Mug</i>	PSNR I_b	30.58	28.20	27.95	26.72	21.02	26.87
	PSNR I_r	30.82	28.32	22.24	26.74	N.A.	N.A.
	NCC I_b	0.96	0.90	0.89	0.87	0.76	0.87
	NCC I_r	0.84	0.67	0.63	0.30	N.A.	N.A.
<i>Ladder</i>	PSNR I_b	34.47	28.69	30.14	29.12	24.28	28.87
	PSNR I_r	34.10	28.63	21.88	29.03	N.A.	N.A.
	NCC I_b	0.92	0.54	0.74	0.77	0.83	0.80
	NCC I_r	0.92	0.79	0.82	0.55	N.A.	N.A.
<i>Palm</i>	PSNR I_b	34.41	30.40	31.01	28.56	20.53	28.57
	PSNR I_r	34.71	30.09	20.71	28.51	N.A.	N.A.
	NCC I_b	0.91	0.71	0.82	0.90	0.89	0.94
	NCC I_r	0.98	0.94	0.95	0.48	N.A.	N.A.
<i>Avg.</i>	PSNR I_b	32.06	27.21	29.20	27.38	21.39	27.19
	PSNR I_r	32.03	27.20	20.91	27.33	N.A.	N.A.
	NCC I_b	0.94	0.73	0.82	0.87	0.83	0.87
	NCC I_r	0.91	0.74	0.77	0.44	N.A.	N.A.

FIGURE 4.7: Qualitative evaluation on synthetic scenes. Here we show separation results on the scene *Car* and *Orange*, along with ground truths.

Algorithm 1 Reflection Removal using LF Cameras

Input: Single LF capture of a scene with reflection $\mathbf{L}$.

Output: Background image $\mathbf{I}_b$, and reflection image $\mathbf{I}_r$.

Initial layer separation:

1. Estimate disparities of edge scene points by voting;
2. Classify them into the background set S_1 and reflection set S_2 , based on signs of their disparity estimates;
2. Improve edge scene points identifications, S_1 and S_2 , by applying the steps described in Section 4.2.2;
3. Generate initial background and reflection $\hat{\mathbf{I}}_b$ and $\hat{\mathbf{I}}_r$ via Eq. (4.1);

Final layer separation:

1. Interpolate semi-dense initial disparities $\hat{\mathbf{D}}_b$ and $\hat{\mathbf{D}}_r$, based on segmentations of $\hat{\mathbf{I}}_b$ and $\hat{\mathbf{I}}_r$;
 2. Alternating optimization:
 Initialization: $\hat{\mathbf{I}}_b$, $\hat{\mathbf{I}}_r$, $\hat{\mathbf{D}}_b$, and $\hat{\mathbf{D}}_r$.
while separation unstable **do**
 Update $\mathbf{I}_b$ and $\mathbf{I}_r$, via Eq. (4.13).
 Update $\mathbf{D}_b$ and $\mathbf{D}_r$, via Eq. (4.14).
end while
-

4.3.1 Evaluations on Synthetic Data

In this section, we evaluate the performance of our method using synthetic data. We first synthesized LF with superimposed layers by overlaying weighted reflection and background. More specifically, two LF captures, $\mathbf{L}_1$ and $\mathbf{L}_2$, were taken with a particular focus, such that content of $\mathbf{L}_1$ is further away from the focus, and content of $\mathbf{L}_2$ is nearer than the focus. Note that two component LF were captured with the same focus. We multiplied them with weights that sum to one and then overlaid them to synthesize LF with reflection $\mathbf{L}_s$. We make the common assumption that weights of the two LF are constant. Thus

$$\mathbf{L}_s = \alpha_1 \mathbf{L}_1 + \alpha_2 \mathbf{L}_2, \quad (4.19)$$

where α_1 and α_2 are the mixing coefficients, and they add to one. α_1 and α_2 were randomly generated in the interval $[0.3, 0.7]$ so that both layers are discernible to some extent.

FIGURE 4.8: More results of synthetic scenes.

Comparisons were carried out against two single-image methods, Arvanitopoulos *et al.* [51] and Fan *et al.* [52], one multiple-image method, Li and Brown [36], and one LF based method [58]. For single-image methods [51][52], central view SAIs were used as the input. For multiple-image method [36], nine different SAIs were used as the input. [51][52][36] were implemented using source codes that are publicly available. For [58], separation results were provided by its authors, using default parameters of the method.

We used two metrics to evaluate the separation results: PSNR and normalized cross-correlation (NCC). NCC is effective in measuring similarity between ground truths and recovered layers [39]. Considering attenuation effects, we adjusted mean intensity of the recovered layer to equal that of ground truth, before calculating PSNR and NCC values. In addition, since [51] [52] focus mostly on recovering

background correctly, we only included PSNR and NCC of background obtained using those two methods for comparisons.

Evaluation on six synthetic scenes is reported in Table. 4.1. Our method achieves the highest PSNR and NCC, for almost all testing scenes. It is worth mentioning that [36] uses SIFT flow, which can only calculate pixel-wise correspondences. Therefore it does not work with the small baseline of LF data, and generated unsatisfactory results. In addition, one can see that our final results after enforcing EPI consistency constraint improves upon initial ones significantly.

In Fig. 4.7, we display recovered background and reflection, along with ground truths of scene *Car* and *Orange*. As can be seen, layers recovered by our method are visually closest to the ground truths. More results can be found in Fig. 4.8.

4.3.2 Evaluations on Real Data

We then carried out evaluations on real data and compared with [51] [36] [58] [107]. [51] does not explicitly generate reflection. Therefore we calculated the difference between input and the background, and used the difference as its recovered reflection in qualitative comparisons. Compared with synthetic scenes, it is more challenging to work with real scenes, where factors like imperfect reflecting surfaces introduce errors and uncertainty.

Weak to modest reflection. In Fig. 4.9, we show layers obtained over real data. One can see that results of our method are the most visually appealing ones. Details that cannot be perceived in the input are recovered, such as the person walking by near left end of recovered reflection in scene two. [58] restores such details even more, such as arm of the person in scene one. However, the color distortions in backgrounds and reflections obtained using [58] are rather obvious in scene two.

Strong reflection. Scenes shown in Fig. 4.9 involve weak or modest reflection. In that case, background is the dominant layer, while reflection can be seen as details to be suppressed or smoothed. Methods like [51] [52] are based on this presumption, and they focus more on background. Meanwhile our method makes no such presumption, and intends to recover both layers. Therefore it can handle scenes with strong reflection as well. In fact, it works best when edges of both layers

FIGURE 4.9: Qualitative evaluation on real scenes containing weak or modest reflection. In scene one, the background is an image on the bulletin board, and the reflected scene depicts the person taking the picture and the site behind. In scene two, the background depicts displays of a jewelry shop, and the reflected scene includes the person taking the picture, railings behind, and shopping mall advertising on the other side.

FIGURE 4.10: Qualitative evaluation on real scenes containing strong reflection.

FIGURE 4.11: More qualitative evaluation on real scenes.

are well-defined. In Fig. 4.10, we show separation results of scenes with strong reflections. One can see that our method recovered background and reflection truthfully, even though the two layers overlay severely. This is especially obvious in scene three, where one can see noticeable residues in backgrounds generated using [36][58] [107], while the floor plan obtained using the proposed method is clean.

More results can be found in Fig. 4.11 and 4.12. In Fig. 4.11, we include cases with

(a) LF input

(b) Recovered background

(c) Recovered reflection

FIGURE 4.12: Separation results of a scene with complex depths.

FIGURE 4.13: Qualitative evaluation on handshake blurred scenes.

FIGURE 4.14: Experimental setup.

weak reflections, as well as cases with strong reflections. In Fig. 4.12, we show an example where both layers, especially the background, have complex depths. The foliage has considerable occlusions and depth variations. The proposed method can still recover the two layers faithfully.

Handshake blur. Camera shake causes blurring to both layers. In such event, presumption made by methods such as [50] that reflection is smoother than background does not hold, and the two layers cannot be separated correctly.

Meanwhile our method can handle moderately blurred input caused by handshake well, as shown in Fig. 4.13, while results of [50] and [58] are not as accurate. The sparse dictionary used in [58] was learned from a training set of relatively noise-free and sharp LF, which may be the reason for strong visual artifacts, especially of the top blurred scene.

Controlled experiments. We conducted controlled experiments on three scenes, where we also captured ground-truth backgrounds. The experimental set-up is shown in Fig. 4.14. We used a tripod, two poster stands of similar heights, and a small glass panel. In Fig. 4.15, we compare backgrounds obtained using the proposed method with ground-truth ones. One can see that visually the two are very similar. In the second scene, even specular highlight of the background is

FIGURE 4.15: Evaluation with ground-truth backgrounds. NCC between obtained and ground-truth I_b is calculated as 0.98, 0.93, and 0.95, for the three scenes.

recovered successfully. To quantitatively assess the performance of our method with real data, NCC was calculated for these three scenes, which is around 0.95. We would like to point out that even though we used a tripod to stabilize shots, there is still a small but noticeable motion parallax between inputs including reflections and ground-truth backgrounds. Therefore unlike NCCs for synthetic data in Table I, we report the highest values within plus/minus 5 pixels of relative shift, for the scenes in Fig. 4.15

FIGURE 4.16: A failure case.

4.3.3 Limitations

In particular, the proposed method will fail to distinguish between background and reflection edges, when the two layers have overlapping depths. In Fig. 4.16, we show a scene where a part of the reflection (the person who is taking the picture) has a similar depth as the background. Its edges are wrongly classified into the background set, and the person is restored as a part of the background instead. In addition, unlike neural network based methods, our method cannot reconstruct the background and reflection layers in real time.

4.4 Conclusion

In this chapter, we present a reflection removal method, using LF data. It can recover background and reflection properly, in both static and moving scenes. Specifically, we first label edges of the two layers by novel focus manipulation procedures, and generate initial rough layer estimates accordingly. We then enforce the input

EPIs to equal the sums of background and reflection EPIs, which underlies the inherent structure of LF data. Much better separation results are obtained through subsequent alternating optimization.

As has been demonstrated in experiments, our method outperforms prior methods by a large margin, over both synthetic and real-world data. Furthermore, it can handle challenging cases such as those with strong reflection or handshake blur well.

Chapter 5

Material Recognition Using a Multi-camera Cellphone

5.1 Introduction

Material recognition is a very useful technique. Our work in particular is about how to identify material types, given images captured using a multi-camera cellphone. There are two reasons why a multi-camera cellphone is used. First of all, it can be used and carried around easily. Second, these cameras capture images from various viewpoints, which include depth cues. To be more specific, we compute a depth probability map based on the captured images. One of the images, as well as the computed depth probability map, is then inputted into a neural network to determine the type.

The input image records two types of information, the appearance and the shape, which are important indications of the material type. The appearance of a material captured in the image underlines its texture, which are a unique characteristic. Regarding the shape, materials are often associated with certain shapes, as each of them are generally used to make specific objects. For example, wood is often used to make furniture pieces that have well-defined corners. Meanwhile as leather is used because it is pliable, in general objects made of leather have no edges. Both the appearance and object shape cues can be extracted from the captured image or large patches of the captured image.

Besides appearances and shapes, we believe depths are useful as well. For example, depths of metal and furry surfaces exhibit very different characteristics. The first ones change gradually along the surfaces, while the later ones fluctuate much more from points to points. Such a difference will help us better identify the material type. To incorporate depth information, our method takes depth probability maps as the other input. Such maps indicate likelihoods of scene points having certain candidate depths. Meanwhile depth values computed using existing methods have significant noises, due to the small baseline among images captured by different cellphone cameras.

5.2 Proposed Method

5.2.1 Inputs

We first need to calculate the two types of inputs to be used, a color image and a depth probability map. For the color image, we just use I_1 , which is one of the two images captured by the cameras. Obtaining the depth probability map is more complex, which will be discussed in detail below.

We utilize a neural network trained to estimate multiplane image scene representations (MPIs) of specific viewpoints to compute the depth probability map [2]. A MPI scene representation consists of a set of rgba images, and each of them deals with scene points within a particular depth range. The rgb channels of these images, $C_1, C_2, \dots, C_n \in \mathbb{R}^{h \times w \times 3}$, reflect the colors. Meanwhile the alpha values, $\alpha_1, \alpha_2, \dots, \alpha_n \in \mathbb{R}^{h \times w \times 1}$, record the occlusion ordering, when compositing the full image. To be more specific, we first alpha composite the two furthest images, C_n and C_{n-1} :

$$I_{comp_{n-1}} = C_{n-1} + (1 - \alpha_{n-1}) \cdot C_n. \quad (5.1)$$

The initial composition $I_{comp_{n-1}}$ is then combined with the next rgb image C_{n-2} :

$$I_{comp_{n-2}} = \alpha_{n-2} \cdot C_{n-2} + (1 - \alpha_{n-2}) \cdot I_{comp_{n-1}}. \quad (5.2)$$

The step described in Eq. 5.2 continues until all rgba images are included. And the final composition I_{comp_1} is the image one will see from that particular viewpoint.

FIGURE 5.1: Sample MPI images. The first column shows the estimated colors, C_7 , C_{16} , and C_{22} of three different depth ranges. The second column shows the estimated projections $C_7 \cdot \bar{\alpha}_7$, $C_{16} \cdot \bar{\alpha}_{16}$, and $C_{22} \cdot \bar{\alpha}_{22}$ of scene points in these depth ranges.

Combining Eq. 5.1 and 5.2, the final composition I_{comp1} equals:

$$\bar{\alpha}_1 \cdot C_1 + \bar{\alpha}_2 \cdot C_2 + \dots + \bar{\alpha}_n \cdot C_n, \quad (5.3)$$

where

$$\bar{\alpha}_i = \begin{cases} \alpha_i, & \text{if } i = 1 \\ \alpha_i \sum_{j=1}^{i-1} (1 - \alpha_j), & \text{if } i = 2, 3, \dots, n-1 \\ \sum_{j=1}^{i-1} (1 - \alpha_j), & \text{if } i = n. \end{cases} \quad (5.4)$$

The compounded alpha values, $\bar{\alpha}_1, \bar{\alpha}_2, \dots, \bar{\alpha}_n$, give precedence to nearer scene points, as scene points further away in the same line of sight will be occluded.

As shown in Fig. 5.1, the MPIs estimated by [2] generally have the same rgb images for all MPI depths. That is to say $C_1 \approx C_2 \approx \dots \approx C_n$. Hence the concatenation of $\bar{\alpha}_1, \bar{\alpha}_2, \dots, \bar{\alpha}_n$, $\bar{\alpha} \in \mathbb{R}^{h \times w \times n}$, represents the likelihoods of scene points having particular depths, and thus are used as depth probability maps.

The method in [2] uses images from two different viewpoints. Assuming that camera parameters are known, one of them is then reprojected at a set of depths onto the other. A combination of such images and the image being reprojected onto forms the input of the neural network. Therefore we reproject I_2 to the reference viewpoint of I_1 at multiple depths as well, using camera matrices obtained through calibrations. The depths used do not necessarily need to be the same as with those used in [2], as long as their respective inverses (disparities) are evenly spaced within the range. The reprojected images, together with I_2 , are fed into the network, which produces the depth possibility map of the reference viewpoint.

5.2.2 Network architecture

As shown in Fig. 5.2(a), our neural network includes a color branch and a MPI branch, which extract features from the two types of inputs separately. A combination of the features extracted is then inputted into a classifier, which is a fully-connected layer, to determine the material class.

Details of the color and the mpi branch are presented in Fig. 5.2(b). Both use the architecture of ResNet-50 [108]. Layers conv1, conv2_x, conv3_x, conv4_x, conv5_x, and an average pooling layer are used to extract features from the color images. The same layers are used to extract features from the depth probability maps. The number of filters in conv1 is increased though, since depth probability maps have more channels than images. The flattened color and depth features are then added

FIGURE 5.2: Diagram of the proposed neural network.

and passed on to the classifier, which calculates the likelihoods of possible material classes.

5.3 Training the Network

In this section, we discuss details of how the network is trained to identify material classes.

Android camera app. The Android phone we use has a wide angle lens camera, a ultra wide angle lens camera, and telephoto lens camera. However, it only stores post-processed images computed using the captured ones. And there is no officially supported developer API to access these cameras. Thus based on Android studio

FIGURE 5.3: Anaglyph from rectified input images.

code developed for a different kind of Android phone¹, we wrote our own app. We use the wide angle lens camera and the telephoto lens camera, as the field of view of the ultra wide angle lens camera is substantially larger than those of the other two.

We noticed that the focus distances will be adjusted automatically by default. Meanwhile since the camera intrinsic matrices are not provided and need to be obtained through calibrations, we will make the focus distances constant so that the intrinsic matrices will be the same from all scenes captured. The focus distance ranges of the two cameras we use do not overlap. We selected two values that are close.

Camera calibration. The manufacturer of our phone does not provide camera matrices needed to transform between two different camera viewpoints. Therefore we decide to estimate them using the MATLAB Stereo Camera Calibrator app². In Fig. 5.3, we show anaglyphs of the two input images, I_1 and I_2 , after we apply rectification based on the estimated camera matrices. One can see that pixels of

¹<https://github.com/google/basicbokeh>

²<https://www.mathworks.com/help/vision/ug/stereo-camera-calibrator-app.html>

FIGURE 5.4: Sample stereo images of the three types of materials collected.

the same scene points mostly lie on the same lines. This indicates the estimated camera matrices are rather accurate.

Material classes. As shown in Fig. 5.4, three types of materials are collected: foliage, tree bark, and wooden surface. For each class, we collect approximately 20 pairs of images. Since 1) the number of images are limited and 2) information used to identify the material class is sufficient in image patches, we input patches of images and depth probability maps instead of full ones into the network. For training, we use patches randomly cropped from the images. For testing, we use patches in the center of the images.

Training loss. The cross entropy loss between the estimated and the groundtruth likelihoods is calculated and used during training.

FIGURE 5.5: Confusion matrix that shows the numbers of true positives, true negatives, false positives, and false negatives.

Learning rate. The overall learning rate decays from epoch to epoch. Furthermore, since pretrained ResNet can already extract meaningful image features, we initialize the weights in the image branch with those pre-trained on ImageNet, and set their learning rates to be 10% of the overall learning rate. Meanwhile the MPI branch and the classifier will be trained using the full learning rate.

5.4 Experiments

We trained our network for 25 epochs, when the training precision no longer increases. The testing precision is 88.3%. We show the confusion matrix in Fig. 5.5. One can see that all cases, except one where wood is misidentified as foliage, are correctly identified.

Actual materials versus paper. We conducted one additional experiment where we add the category paper. It consists of the stereo image pairs of paper printed with material textures. These image pairs are obtained by re-projecting images

FIGURE 5.6: Stereo image pairs of actual and printed foliage

of the wide angle lens to obtain images of the telephoto lens, assuming that scene points captured in a wide angle lens image lie on the same horizontal 3D plane. As shown in Fig. 5.6, such image pairs will look like those of actual materials. It is difficult to correctly identify their category, based on the image features alone. Nevertheless, their alpha values will have a different distribution, compared with those of the actual materials. For the paper category, the peak alpha values will occur around the same depth for all pixel positions. But for the actual materials, the peak alpha values will occur at different depths for different pixel positions, according to the depths of the respective scene points.

we trained a network with only the image branch, which can be considered a generalization of most image feature based material recognition methods, with the actual material and paper categories. In Fig. 5.7 a, we show the confusion matrices on the test results, which indicates that the network with only the image branch is unsuccessful at identifying the paper. Meanwhile, the network with both branches can produce better results, as shown in Fig. 5.7 b.

FIGURE 5.7: Confusion matrices of the full network and the network without the depth branch.

5.5 Conclusion and Future Work

We have presented a material classification method that utilizes both color and depth information in images captured using a multi-camera cell phone. Our method first calculates depth probability maps from the images. The color features extracted from the images and the depth features extracted from the depth probability maps are then combined and inputted into a classifier to identify the material classes.

One interesting extension is to work on distinguishing between materials and images of materials. Many existing material recognition methods rely only on color features. As images of the two groups look very similar, it would be difficult to tell them apart, using these methods. Meanwhile images of the two groups have very different depth properties. Thus we plan to examine how our depth branch can be modified to emphasize such a difference and correctly classify materials and images of materials.

Chapter 6

Conclusions and Recommendations for Future Work

6.1 Conclusions

In this thesis, we present our works done on various topics about recovering and utilizing 3D information. One work is about synthesizing images of novel viewpoints based on 3D plane notations. The other two are about how to use 3D information in two specific applications: reflection removal and material recognition.

In chapter 3, we present a method to synthesize novel views. Unlike existing ones, our method is able to infer the colors of scene points not captured in the input image, based on 3D plane notations. To be more specific, 3D plane notations make it possible to identify scene points belonging to the same planes. Since scene points belonging to the same planes form patterned textures, we can easily estimate the colors of additional scene points. As a result, we can generate the full target image without having to crop out areas of scene points not captured in the input.

We discuss in chapter 4 how to remove reflections using LF data. LF data include images captured from evenly spaced viewpoints. The arrangement of these viewpoints makes it possible to accurately compute depths with single captures. The computed depths are then used to separate reflections from backgrounds.

Finally in chapter 5, we introduce our method to identify material classes using images captured by a multi-camera cell phone. Our method utilizes both traditional color features as well as depth features calculated from the input images. It is able to correctly identify three common material classes: foliage, tree bark, and wood surface. In addition, since our method already takes depth information into consideration, it can be potentially extended to distinguish between materials and images of materials.

As an overall conclusion, works in this thesis pertain to how 3D cues, in particular depths and 3D plane structures, can be used to achieve different computer vision tasks. We demonstrate how 3D information leads to better results in both specific and more abstract tasks.

6.2 Recommendations for Future Work

Future directions to be looked into are listed below:

First of all, how to generate a video giving its virtual camera path will be explored. It can be viewed as an extension of the novel view synthesis problem addressed in chapter 3. The difference is that we need to synthesize multiple instead of single views. Meanwhile since video clips contain frames captured from hundreds of linked viewpoints, substantial adjustments need to be made to the single-view approach, in order to generate the video efficiently.

Second, means to identify and segment 3D planes captured in images will be researched. Planes contain important geometric and semantic cues of the scene. For example in chapter 3, such information is used to group scene points belonging to the same texture together. Yet existing methods often fail to produce exact segmentations, and plane parameters regressed are often imprecise. Therefore the novel view synthesis method in chapter 3 use images with given planar segments and parameters. It is hoped that an improved plane detection method will be developed, which can be further integrated into our novel view synthesis method to make it end-to-end.

Finally, active 3D imaging devices will be explored. All works presented in this thesis use passive 3D imaging devices. Meanwhile active devices carry out 3D sensing in a different way that works better with textureless regions but only generates reliable measurements sparsely. It would be helpful to research how they can be utilized, whether alone or in combination with passive devices, to better infer 3D information. In particular, we will look into ToF cameras, as they are relatively affordable and compact than other active 3D imaging devices.

List of Author's Publications

Journal Articles

- **Yun Ni** and Lap-Pui Chau, “Novel view synthesis with planar structure guidance,” to be submitted.
- **Yun Ni**, Jie Chen, and Lap-Pui Chau, “Reflection removal on single light field capture using focus manipulation,” *IEEE Transactions on Computational Imaging*, vol. 4, no. 4, pp. 562 - 572, 2018.
- Jie Chen, Junhui Hou, **Yun Ni**, and Lap-Pui Chau, “Accurate light field depth estimation with superpixel regularization over partially occluded regions,” *IEEE Transactions on Image Processing*, vol. 27, no. 10, pp. 4889 - 4900, 2018.

Conference Proceedings

- **Yun Ni**, Jie Chen, and Lap-Pui Chau, “Reflection removal based on single light field capture,” in *Proceedings of the IEEE International Symposium on Circuits and Systems*, 2017, pp. 1–4.
- Cheen-Hau Tan, Jie Chen, **Yun Ni**, Lap-Pui Chau, Ling Min Soh, “Vision-Based rain gauge for dynamic scenes,” in *Proceedings of the International Conference on Digital Signal Processing*, 2018, pp. 1–4.
- Jie Chen, **Yun Ni**, Junhui Hou, and Lap-Pui Chau, “Light field scene flow with occlusion regularization,” in *Proceedings of the Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, 2017, pp. 1–4.

Bibliography

- [1] Pratul P Srinivasan, Richard Tucker, Jonathan T Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 175–184, 2019. [xv](#), [11](#), [18](#), [26](#), [27](#), [29](#)
- [2] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018. [xv](#), [10](#), [11](#), [29](#), [62](#), [64](#)
- [3] Kamyar Nazeri, Eric Ng, Tony Joseph, Faisal Qureshi, and Mehran Ebrahimi. Edgeconnect: Generative image inpainting with adversarial edge learning. 2019. [xv](#), [21](#), [26](#), [27](#), [29](#)
- [4] Sergi Foix, Guillem Alenya, and Carme Torras. Lock-in time-of-flight (tof) cameras: A survey. *IEEE Sensors Journal*, 11(9):1917–1926, 2011. [1](#)
- [5] Mohit Gupta, Amit Agrawal, Ashok Veeraraghavan, and Srinivasa G Narasimhan. Structured light 3d scanning in the presence of global illumination. In *CVPR 2011*, pages 713–720. IEEE, 2011. [2](#)
- [6] Supreeth Achar, Joseph R Bartels, William L’Red’ Whittaker, Kiriakos N Kutulakos, and Srinivasa G Narasimhan. Epipolar time-of-flight imaging. *ACM Transactions on Graphics (ToG)*, 36(4):1–8, 2017. [2](#)
- [7] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006. [2](#)
- [8] Ce Liu, Jenny Yuen, and Antonio Torralba. Sift flow: Dense correspondence across scenes and its applications. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):978–994, 2010. [2](#)
- [9] Jure Žbontar and Yann LeCun. Stereo matching by training a convolutional neural network to compare image patches. *The journal of machine learning research*, 17(1):2287–2318, 2016. [2](#)
- [10] Wenjie Luo, Alexander G Schwing, and Raquel Urtasun. Efficient deep learning for stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5695–5703, 2016.

- [11] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5410–5418, 2018.
- [12] Lidong Yu, Yucheng Wang, Yuwei Wu, and Yunde Jia. Deep stereo matching with explicit cost aggregation sub-architecture. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. [2](#)
- [13] Ren Ng, Marc Levoy, Mathieu Brédif, Gene Duval, Mark Horowitz, Pat Hanrahan, et al. Light field photography with a hand-held plenoptic camera. *Computer Science Technical Report CSTR*, 2(11):1–11, 2005. [3](#)
- [14] Ren Ng. Fourier slice photography. In *ACM SIGGRAPH 2005 Papers*, pages 735–744. 2005. [3](#), [33](#)
- [15] Sven Wanner and Bastian Goldluecke. Globally consistent depth labeling of 4d light fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 41–48, 2012. [3](#), [33](#), [35](#)
- [16] Sven Wanner and Bastian Goldluecke. Variational light field analysis for disparity estimation and super-resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 36(3):606–619, 2014. [3](#), [33](#), [35](#)
- [17] Changil Kim, Henning Zimmer, Yael Pritch, Alexander Sorkine-Hornung, and Markus Gross. Scene reconstruction from high spatio-angular resolution light fields. *ACM Transactions on Graphics (TOG)*, 2013. [3](#), [35](#), [37](#)
- [18] Changha Shin, Hae-Gon Jeon, Youngjin Yoon, In So Kweon, and Seon Joo Kim. Epinet: A fully-convolutional neural network using epipolar geometry for depth from light field images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4748–4757, 2018. [3](#), [33](#)
- [19] Anat Levin, Rob Fergus, Frédo Durand, and William T Freeman. Image and depth from a conventional camera with a coded aperture. *ACM transactions on graphics (TOG)*, 26(3):70, 2007. [3](#)
- [20] Harel Haim, Shay Elmaleh, Raja Giryes, Alex M Bronstein, and Emanuel Marom. Depth estimation from a single image using deep learned phase coded mask. *IEEE Transactions on Computational Imaging*, 4(3):298–310, 2018. [3](#)
- [21] Yicheng Wu, Vivek Boominathan, Huaijin Chen, Aswin Sankaranarayanan, and Ashok Veeraraghavan. Phasecam3dlearning phase masks for passive single view depth estimation. In *2019 IEEE International Conference on Computational Photography (ICCP)*, pages 1–12. IEEE, 2019. [3](#)
- [22] Youichi Horry, Ken-Ichi Anjyo, and Kiyoshi Arai. Tour into the picture: using a spidery mesh interface to make animation from a single image. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 225–232, 1997. [9](#)

- [23] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 425–432. ACM, 2001. [9](#)
- [24] Derek Hoiem, Alexei A Efros, and Martial Hebert. Automatic photo pop-up. In *ACM SIGGRAPH 2005 Papers*, pages 577–584. 2005. [9](#)
- [25] Johannes Kopf, Fabian Langguth, Daniel Scharstein, Richard Szeliski, and Michael Goesele. Image-based rendering in the gradient domain. *ACM Transactions on Graphics (TOG)*, 32(6):199, 2013. [9](#)
- [26] Sudipta N Sinha, Drew Steedly, and Richard Szeliski. Piecewise planar stereo for image-based rendering. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1881–1888. IEEE. [9](#)
- [27] Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Transactions on Graphics (TOG)*, 32(3):30, 2013. [10](#)
- [28] Konstantinos Rematas, Chuong H Nguyen, Tobias Ritschel, Mario Fritz, and Tinne Tuytelaars. Novel views of objects from a single image. *IEEE transactions on pattern analysis and machine intelligence*, 39(8):1576–1590, 2016. [10](#)
- [29] Eric Penner and Li Zhang. Soft 3d reconstruction for view synthesis. *ACM Transactions on Graphics (TOG)*, 36(6):235, 2017. [10](#)
- [30] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world’s imagery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5515–5524, 2016. [10](#)
- [31] Eunbyung Park, Jimei Yang, Ersin Yumer, Duygu Ceylan, and Alexander C Berg. Transformation-grounded image generation network for novel 3d view synthesis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3500–3509, 2017. [10](#)
- [32] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (TOG)*, 37(6):1–15, 2018. [10](#)
- [33] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 2019. [10](#)
- [34] Simon Niklaus, Long Mai, Jimei Yang, and Feng Liu. 3d ken burns effect from a single image. *ACM Transactions on Graphics (TOG)*, 38(6):1–15, 2019. [11](#)

- [35] Miaomiao Liu, Xuming He, and Mathieu Salzmann. Geometry-aware deep network for single-image novel view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4616–4624, 2018. [11](#)
- [36] Yu Li and Michael S Brown. Exploiting reflection change for automatic reflection removal. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2432–2439, 2013. [11](#), [32](#), [46](#), [49](#), [50](#), [53](#)
- [37] Jiaolong Yang, Hongdong Li, Yuchao Dai, and Robby T Tan. Robust optical flow estimation of double-layer images under transparency or reflection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1410–1419, 2016. [11](#)
- [38] Xiaojie Guo, Xiaochun Cao, and Yi Ma. Robust separation of reflection from multiple images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2187–2194, 2014. [11](#)
- [39] Tianfan Xue, Michael Rubinstein, Ce Liu, and William T Freeman. A computational approach for obstruction-free photography. *ACM Transactions on Graphics (TOG)*, 34(4):79, 2015. [11](#), [32](#), [49](#)
- [40] Youwei Lyu, Zhaopeng Cui, Si Li, Marc Pollefeys, and Boxin Shi. Reflection separation using a pair of unpolarized and polarized images. In *Advances in Neural Information Processing Systems 32*, pages 14532–14542, 2019. [11](#), [32](#)
- [41] Kun Gai, Zhenwei Shi, and Changshui Zhang. Blind separation of superimposed moving images using image statistics. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(1):19–32, 2012. [12](#), [32](#)
- [42] Tharatch Sirinukulwattana, Gyeongmin Choe, and In So Kweon. Reflection removal using disparity and gradient-sparsity via smoothing algorithm. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 1940–1944, 2015.
- [43] Richard Szeliski, Shai Avidan, and P Anandan. Layer extraction from multiple images containing reflections and transparency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 246–253, 2000. [12](#)
- [44] Ajay Nandoriya, Mohamed Elgharib, Changil Kim, Mohamed Hefeeda, and Wojciech Matusik. Video reflection removal through spatio-temporal optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2411–2419, 2017. [12](#)
- [45] Jun Young Cheong, Christian Simon, Chang-Su Kim, and In Kyu Park. Reflection removal under fast forward camera motion. *IEEE Transactions on Image Processing (TIP)*, 26(12):6061–6073, 2017. [12](#), [32](#)

- [46] Anat Levin, Assaf Zomet, and Yair Weiss. Learning to perceive transparency from the statistics of natural scenes. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, pages 1271–1278, 2003. [12](#), [13](#), [32](#), [41](#)
- [47] Anat Levin and Yair Weiss. User assisted separation of reflections from a single image using a sparsity prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 29(9):1647, 2007. [12](#), [13](#), [32](#), [38](#)
- [48] YiChang Shih, Dilip Krishnan, Frédo Durand, and William T Freeman. Reflection removal using ghosting cues. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3193–3201, 2015. [12](#)
- [49] Renjie Wan, Boxin Shi, Ah-Hwee Tan, and Alex C Kot. Sparsity based reflection removal using external patch search. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 1500–1505, 2017. [12](#)
- [50] Yu Li and Michael S Brown. Single image layer separation using relative smoothness. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2752–2759, 2014. [12](#), [56](#)
- [51] Nikolaos Arvanitopoulos Darginis, Radhakrishna Achanta, and Sabine Süsstrunk. Single image reflection suppression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4498–4506, 2017. [12](#), [46](#), [49](#), [50](#)
- [52] Qingnan Fan, Jiaolong Yang, Gang Hua, Baoquan Chen, and David Wipf. A generic deep architecture for single image reflection removal and image smoothing. In *Proceedings of the The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. [12](#), [46](#), [49](#), [50](#)
- [53] 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. [12](#)
- [54] Huaidong Zhang, Xuemiao Xu, Hai He, Shengfeng He, Guoqiang Han, Jing Qin, and Dapeng Wu. Fast user-guided single image reflection removal via edge-aware cascaded networks. *IEEE Transactions on Multimedia*, 2019. [13](#), [32](#)
- [55] Qiang Wen, Yinjie Tan, Jing Qin, Wenxi Liu, Guoqiang Han, and Shengfeng He. Single image reflection removal beyond linearity. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3771–3779, 2019. [13](#)
- [56] Kaixuan Wei, Jiaolong Yang, Ying Fu, David Wipf, and Hua Huang. Single image reflection removal exploiting misaligned training data and network

- enhancements. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8178–8187, 2019. [13](#)
- [57] Jie Yang, Dong Gong, Lingqiao Liu, and Qinfeng Shi. Seeing deeply and bidirectionally: A deep learning approach for single image reflection removal. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 654–669, 2018. [13](#), [32](#)
- [58] Ole Johannsen, Antonin Sulc, and Bastian Goldluecke. What sparse light field coding reveals about scene structure. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3262–3270, 2016. [14](#), [33](#), [46](#), [49](#), [50](#), [53](#), [56](#)
- [59] Ole Johannsen, Antonin Sulc, and Bastian Goldluecke. Occlusion-aware depth estimation using sparse light field coding. In *Proceedings of the German Conference on Pattern Recognition*, pages 207–218, 2016. [14](#)
- [60] Ole Johannsen, Antonin Sulc, and Bastian Goldluecke. Variational separation of light field layers. In *Proceedings of the Vision, Modeling and Visualization*, pages 135–142, 2015. [14](#), [40](#)
- [61] Qiaosong Wang, Haiting Lin, Yi Ma, Sing Bing Kang, and Jingyi Yu. Automatic layer separation using light field imaging. *arXiv preprint arXiv:1506.04721*, 2015. [14](#), [33](#)
- [62] Paramanand Chandramouli, Mehdi Noroozi, and Paolo Favaro. Convnet-based depth estimation, reflection separation and deblurring of plenoptic images. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, pages 129–144, 2016. [14](#)
- [63] Tingtian Li, Daniel PK Lun, Yuk-Hee Chan, et al. Robust reflection removal based on light field imaging. *IEEE Transactions on Image Processing*, 28(4): 1798–1812, 2018. [14](#), [33](#)
- [64] Yun Ni, Jie Chen, and Lap-Pui Chau. Reflection removal based on single light field capture. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–4, 2017. [14](#), [33](#), [46](#)
- [65] Abhijith Punnappurath and Michael S Brown. Reflection removal using a dual-pixel sensor. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1556–1565, 2019. [14](#), [33](#)
- [66] Ce Liu, Lavanya Sharan, Edward H Adelson, and Ruth Rosenholtz. Exploring features in a bayesian framework for material recognition. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 239–246. IEEE, 2010. [15](#)
- [67] Diane Hu, Liefeng Bo, and Xiaofeng Ren. Toward robust material recognition for everyday objects. In *BMVC*, volume 2, page 6. Citeseer, 2011. [15](#)

- [68] Gabriel Schwartz and Ko Nishino. Automatically discovering local visual material attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3565–3573, 2015. [15](#), [16](#)
- [69] Mircea Cimpoi, Subhransu Maji, and Andrea Vedaldi. Deep filter banks for texture recognition and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3828–3836, 2015. [15](#)
- [70] Jia Xue, Hang Zhang, and Kristin Dana. Deep texture manifold for ground terrain recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 558–567, 2018. [15](#)
- [71] Gabriel Schwartz and Ko Nishino. Recognizing material properties from images. *IEEE transactions on pattern analysis and machine intelligence*, 2019. [16](#)
- [72] Peng Bian, Wanwan Li, Yi Jin, and Ruicong Zhi. Ensemble feature learning for material recognition with convolutional neural networks. *EURASIP Journal on Image and Video Processing*, 2018(1):64, 2018. [16](#)
- [73] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. Material recognition in the wild with the materials in context database. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3479–3487, 2015. [16](#)
- [74] Oliver Wang, Prabath Gunawardane, Steve Scher, and James Davis. Material classification using brdf slices. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2805–2811. IEEE, 2009. [16](#)
- [75] Hang Zhang, Kristin Dana, and Ko Nishino. Reflectance hashing for material recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3071–3080, 2015. [16](#)
- [76] Jia Xue, Hang Zhang, Kristin Dana, and Ko Nishino. Differential angular imaging for material recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 764–773, 2017. [16](#)
- [77] Ting-Chun Wang, Jun-Yan Zhu, Ebi Hiroaki, Manmohan Chandraker, Alexei A Efros, and Ravi Ramamoorthi. A 4d light-field dataset and cnn architectures for material recognition. In *European Conference on Computer Vision*, pages 121–138. Springer, 2016. [16](#)
- [78] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. [21](#)
- [79] Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 85–100, 2018. [21](#)

- [80] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5828–5839, 2017. [24](#)
- [81] Chen Liu, Kihwan Kim, Jinwei Gu, Yasutaka Furukawa, and Jan Kautz. Planercnn: 3d plane detection and reconstruction from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4450–4459, 2019. [24](#)
- [82] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018. [28](#)
- [83] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018. [28](#)
- [84] Yun Ni, Jie Chen, and Lap-Pui Chau. Reflection removal on single light field capture using focus manipulation. *IEEE Transactions on Computational Imaging*, 4(4):562–572, 2018. [31](#)
- [85] Yanghai Tsin, Sing Bing Kang, and Richard Szeliski. Stereo matching with linear superposition of layers. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(2):290–301, 2006. [31](#)
- [86] Sudipta N Sinha, Johannes Kopf, Michael Goesele, Daniel Scharstein, and Richard Szeliski. Image-based rendering for scenes with reflections. *ACM Transactions on Graphics (TOG)*, 31(4):100–1, 2012. [31](#)
- [87] Michael W Tao, Sunil Hadap, Jitendra Malik, and Ravi Ramamoorthi. Depth from combining defocus and correspondence using light-field cameras. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 673–680, 2013. [33](#)
- [88] Michael W Tao, Pratul P Srinivasan, Jitendra Malik, Szymon Rusinkiewicz, and Ravi Ramamoorthi. Depth from shading, defocus, and correspondence using light-field angular coherence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1940–1948, 2015.
- [89] Stefan Heber and Thomas Pock. Convolutional networks for shape from light field. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3746–3754, 2016.
- [90] Matthieu Hog, Neus Sabater, Benoît Vandame, and Valter Drazic. An image rendering pipeline for focused plenoptic cameras. *IEEE Transactions on Computational Imaging*, 3(4):811–821, 2017. [33](#)
- [91] Pratul P Srinivasan, Ren Ng, and Ravi Ramamoorthi. Light field blind motion deblurring. *arXiv preprint arXiv:1704.05416*, 2017. [33](#)

- [92] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. *ACM Transactions on Graphics (TOG)*, 35(6):193, 2016. 33
- [93] Reuben A Farrugia, Christian Galea, and Christine Guillemot. Super resolution of light field images using linear subspace projection of patch-volumes. *IEEE Journal of Selected Topics in Signal Processing*, 11(7):1058–1071, 2017. 33
- [94] Yingqian Wang, Jungang Yang, Yulan Guo, Chao Xiao, and Wei An. Selective light field refocusing for camera arrays using bokeh rendering and superresolution. *IEEE Signal Processing Letters*, 26(1):204–208, 2018. 33
- [95] Ting-Chun Wang, Manmohan Chandraker, Alexei A Efros, and Ravi Ramamoorthi. Svbrdf-invariant shape and reflectance estimation from light-field cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5451–5459, 2016. 33
- [96] Tingtian Li and Daniel PK Lun. A novel reflection removal algorithm using the light field camera. In *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE, 2018. 33
- [97] Gabriel Lippmann. La photographie intégrale. *Academie des Sciences*, 146: 446–451, 1908. 34
- [98] Donald G Dansereau, Oscar Pizarro, and Stefan B Williams. Decoding, calibration and rectification for lenselet-based plenoptic cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1027–1034, 2013. 34, 45
- [99] Robert C Bolles, H Harlyn Baker, and David H Marimont. Epipolar-plane image analysis: An approach to determining structure from motion. *International Journal of Computer Vision (IJCV)*, 1(1):7–55, 1987. 35
- [100] Lytro Illum Depth Composition Features. <http://manuals.lytro.com/illum/depth-composition-features/>. Accessed: 2018-02-04. 36
- [101] Kevin Karsch, Ce Liu, and Sing Bing Kang. Depth extraction from video using non-parametric sampling. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 775–788, 2012. 42
- [102] Guofeng Zhang, Jiaya Jia, Tien-Tsin Wong, and Hujun Bao. Consistent depth maps recovery from a video sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 31(6):974–988, 2009. 42
- [103] Jian Sun, Nan-Ning Zheng, and Heung-Yeung Shum. Stereo matching using belief propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 25(7):787–800, 2003. 42

- [104] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 391–405, 2014. [42](#)
- [105] Piotr Dollár and C Lawrence Zitnick. Fast edge detection using structured forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 37(8):1558–1570, 2015. [42](#)
- [106] Miaomiao Liu, Mathieu Salzmann, and Xuming He. Discrete-continuous depth estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 716–723, 2014. [42](#)
- [107] Chao Li, Yixiao Yang, Kun He, Stephen Lin, and John E Hopcroft. Single image reflection removal through cascaded refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3565–3574, 2020. [50](#), [53](#)
- [108] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. [64](#)