COMPUTATIONAL CAMERA AND ILLUMINATION TECHNIQUES
FOR RECOVERING “INVISIBLE” PHENOMENON

by

Yu Ji

A dissertation submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science

Winter 2016

© 2016 Yu Ji
All Rights Reserved
COMPUTATIONAL CAMERA AND ILLUMINATION TECHNIQUES
FOR RECOVERING “INVISIBLE” PHENOMENON

by

Yu Ji

Approved: 
Kathleen F. McCoy, Ph.D.
Chair of the Department of Computer and Information Sciences

Approved: 
Babatunde A. Ogunnaike, Ph.D.
Dean of the College of Engineering

Approved: 
Ann L. Ardis, Ph.D.
Interim Vice Provost for Graduate and Professional Education
I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: Jingyi Yu, Ph.D.
Professor in charge of dissertation

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: Chandra Kambhamettu, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: Christopher Rasmussen, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: Gonzalo Arce, Ph.D.
Member of dissertation committee
ACKNOWLEDGEMENTS

First of All, I would like to acknowledge the support and help of my research advisor Jingyi Yu. It was him who brought me into this exciting research area of computational photography and computer vision. He not only provides me with the freedom to find my own way in the research but also guides me to stay on track. He is always available to me and willing to spend enormous of time to mentor me. He is the type of advisor that every graduate student wants.

I would also like to thank Professor Chandra Kambhamettu, Professor Christopher Rasmussen, and Professor Gonzalo Arce for serving on my advisory committee members and providing insightful suggestions.

I share the credit of my work with collaborators at the University of Delaware and other research institutes. They are Jinwei Ye, Feng Li, Yuanyuan Ding, Haibin Ling, Sing Bing Kang, Haiting Lin, S. Susan Young, Nianyi Li, Yang Yang and Wei Yang. It has been a great privilege to work with each of them, and this dissertation would have remained a dream had it not been for their constant help and invaluable discussions.

Thanks to all other members of the UD Graphics & Imaging lab (Xuan Yu, Luis D. Lopez, Miao Tang, Yuqi Wang, Christopher Thorpe, Xiaogang Chen, Can Chen, Xinqing Guo, Qiaosong Wang and Mingyuan Zhou) for sharing their thoughts, codes and providing such an enjoyable and supportive environment during my Ph.D. study. SThanks to all of my friends in UD, I enjoyed the time that I spent with everyone of you.
I gratefully acknowledge the funding sources that made my Ph.D work possible. My work is partially supported by National Science Foundation under grants IIS-CAREER-0845268 and IIS-RI-1016395, and by the Air Force Office of Science Research (AFOSR) Young Investigator Award (YIP).

I owe my deepest gratitude to my wife, my daughter and my parents. Thanks to them for being proud of me and loving me, and all the sweetest memories.
DEDICATION

I dedicate this dissertation to my loving wife whose encouragement have meant to me so much during the pursuit of my Ph.D. degree. I dedicate this dissertation to my parents who have given me support throughout my life.
# TABLE OF CONTENTS

LIST OF TABLES ................................................. x
LIST OF FIGURES .............................................. xi
ABSTRACT ...................................................... xvi

Chapter

1 INTRODUCTION .............................................. 1
   1.1 Dissertation Statement ................................. 2
   1.2 Contributions ........................................... 3
   1.3 Blueprint of the Dissertation ......................... 4

2 RELATED WORKS ........................................... 6
   2.1 Invisible Phenomenon ................................. 6
   2.2 Acquisition Hardware ................................. 8
   2.3 Reconstruction Algorithm ......................... 10

3 DYNAMIC FLUID SURFACE ACQUISITION USING A CAMERARRAY ............................................. 13
   3.1 Acquisition Device .................................. 13
   3.2 Fluid Surface Reconstruction ....................... 16
      3.2.1 Correspondence Maps ........................... 17
      3.2.2 Volumetric Reconstruction .................... 19
   3.3 Results and Discussions ............................. 25

4 ANGULAR DOMAIN RECONSTRUCTION OF DYNAMIC 3D FLUID SURFACE ..................................... 30
   4.1 Bokode-based Acquisition System ................. 30
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2</td>
<td>Correspondence Matching</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>4.3</td>
<td>Surface Reconstruction from Angularly Sampled Normals</td>
<td>37</td>
<td></td>
</tr>
<tr>
<td>4.3.1</td>
<td>Normal Integration in Polar Coordinate</td>
<td>39</td>
<td></td>
</tr>
<tr>
<td>4.4</td>
<td>Experiments</td>
<td>43</td>
<td></td>
</tr>
<tr>
<td>4.4.1</td>
<td>Synthetic Scene Simulation</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>4.4.2</td>
<td>Real Scene Experiments</td>
<td>46</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>RECONSTRUCTING GAS FLOWS USING LIGHT-PATH APPROXIMATION</td>
<td>49</td>
<td></td>
</tr>
<tr>
<td>5.1</td>
<td>Acquisition System</td>
<td>49</td>
<td></td>
</tr>
<tr>
<td>5.1.1</td>
<td>Light Field Probe (LF-Probe)</td>
<td>49</td>
<td></td>
</tr>
<tr>
<td>5.1.2</td>
<td>System Setup</td>
<td>51</td>
<td></td>
</tr>
<tr>
<td>5.2</td>
<td>Volumetric Gas Reconstruction</td>
<td>54</td>
<td></td>
</tr>
<tr>
<td>5.2.1</td>
<td>RIF Estimation</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>5.2.2</td>
<td>Light Path Refinement</td>
<td>57</td>
<td></td>
</tr>
<tr>
<td>5.3</td>
<td>Experiments</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>5.3.1</td>
<td>Synthetic Scene Simulations</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>5.3.2</td>
<td>Real Scene Experiment</td>
<td>62</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>MEASURING AMBIENT OCCLUSION BY LIGHT FIELD PROBE</td>
<td>66</td>
<td></td>
</tr>
<tr>
<td>6.1</td>
<td>Problem Definition</td>
<td>66</td>
<td></td>
</tr>
<tr>
<td>6.2</td>
<td>Ambient Occlusion Recovery</td>
<td>69</td>
<td></td>
</tr>
<tr>
<td>6.2.1</td>
<td>Ill-Posedness</td>
<td>69</td>
<td></td>
</tr>
<tr>
<td>6.2.2</td>
<td>AO Estimation Via Compressive Sensing</td>
<td>71</td>
<td></td>
</tr>
<tr>
<td>6.2.3</td>
<td>Directional Coded Light Source</td>
<td>73</td>
<td></td>
</tr>
<tr>
<td>6.3</td>
<td>Experiments</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>6.3.1</td>
<td>Synthetic Experiment</td>
<td>74</td>
<td></td>
</tr>
</tbody>
</table>
6.3.2 Photometric Stereo ........................................... 78
6.3.3 Real Data ...................................................... 80

7 CONCLUSION AND FUTURE WORK ............................ 89

7.1 Conclusions ..................................................... 89

  7.1.1 Dynamic Fluid Surface Acquisition Using a Camera Array .. 89
  7.1.2 Dynamic Fluid Surface Acquisition Using Bokode .......... 90
  7.1.3 Gas Flow Reconstruction Using Light Field Probe .......... 90
  7.1.4 Measuring Ambient Occlusion by Light Field Probe ....... 91

7.2 Future Work .................................................... 91

BIBLIOGRAPHY ....................................................... 94

Appendix

PERMISSION LETTERS ............................................. 101
### LIST OF TABLES

5.1 Reconstruction errors using our approach vs. [1] under different configurations. (APE: Averaged Percentage Error. MPE: Maximum Percentage Error.) ................................. 62
LIST OF FIGURES

1.1 Examples of transparent objects. .......................... 2

3.1 Dynamic Fluid Surface Acquisition. .......................... 14

3.2 (Left) The setup of our fluid surface acquisition system. (Right) We divide the camera array into two groups (gray and black) and interleave the trigger for each group to double the frame rate. .... 15

3.3 A diagram showing the pipeline of our camera-array based fluid surface reconstruction algorithm. .......................... 17

3.4 Classical Lucas-Kanade (L-K) tracker loses track at regions with strong distortions. Our locally constrained L-K tracker is able to robustly track in these regions. .......................... 19

3.5 Our refraction carving framework. (a) Given the incident and the refracted directions, we can solve for the surface normal. (b) Each voxel will give a different normal estimate. (c) Using an array of cameras, we can find the optimal voxel/normal. (d) Finally, we apply space carving using the normal consistency criteria. .......................... 21

3.6 In case a camera $C^*$ loses track of a feature point $P$, we use the recovered surface from the rest of the cameras and backtrace $P$ to $C^*$. 22

3.7 Results on synthetic data. (a-c) show the ground truth height field and normal field. (d-f) show the corresponding raw refraction carving results. Notice that the recovered height field is very noisy but the normal field is highly accurate. (g) We integrate (e-f) using the boundary from (d) to obtain the final reconstruction. (h-i) we compare the ray traced refraction distortions on the ground truth surface (h) and our reconstruction (i). .......................... 24

3.8 Results on a synthetic “drop” wave. .......................... 27
3.9 Results on real data. Top row shows the feature tracking results on two views. Bottom row shows the recovered height field. 28

3.10 Results on real data. From top to bottom: the captured frames from the central camera, the recovered normal field, and the recovered height field. The complete sequence can be found in the supplementary video. 29

4.1 A block diagram that shows the pipeline of our Bokode based fluid surface reconstruction framework. 31

4.2 How Bokode is viewed by a camera. 32

4.3 Bokode-based fluid surface acquisition scheme. 33

4.4 Active appearance model for pattern-pixel matching. 36

4.5 Results on a synthetic sinusoid wave (top rows) and on a Helmholtz wave (bottom two rows). We show the cropped Bokode pattern used in reconstruction. In particular, column 2 are sampled surface normals under spherical coordinate in respect to $\theta$ and $\phi$ and we further compute normal maps in Cartesian coordinate and compare with the ground truth ones to demonstrate the accuracy of our method as shown in column 6 and 7. The complete sequences can be found in the supplementary video. 43

4.6 Our experimental setup. We construct a Bokode using a flashlight, a diffuser, a high-resolution pattern and webcam lens. We also use an auxiliary bi-convex spherical lens to collect lights refracted by the fluid surface (top right). 45

4.7 Results on two sets of real data (a perturbed wave and a ring wave). From left to right: we show the captured image with matched features, the sampled normal under spherical coordinate, the reconstructed surface, and the surface normal field computed from the reconstructed surface. 47

5.1 The LF-probe. Left: The assembly of a LF-probe. Right: Each pattern pixel maps to a unique ray. 50

5.2 An illustration of our gas flow acquisition system. 52
5.3 We use a rotation table (a) to calibrate ray directions (b) by matching observed colors to the color calibration curves (c).

5.4 We use ray-ray correspondences to first recover the light paths and then the refractive index field (RIF) within the gas volume.

5.5 Accelerating light path refinement via volume pruning. (a) We prune voxels outside the bounding volume; (b) We restrict the search direction within a cone.

5.6 A simple Gaussian gas flow. (a) A rendered LF-probe image through the flow; (b) Optical flow results on the green channel; (c) Directional variations of rays; (d) The ground truth RIF; (e) Our estimated RIF after 1 iteration; (f) Our final result.

5.7 The fuel injection dataset. (a) A volume rendering of the ground truth data; (b) Optical flow results on the green channel; (c) Directional variations of rays; (d) Slices of our reconstructed RIF vs. the ground truth.

5.8 Our LF-probe based gas flow acquisition system.

5.9 Reconstruction results on a real gas flow. Each row shows the reconstructed flow at a different time instance. (a) The captured LF-probe images; (b) Measured ray direction variations through the flow; (c)-(e) Three vertical slices of the reconstructed RIFs where (d) is the central slice.

5.10 We compose the reconstructed flows onto a new background.

6.1 a sample Ambient occlusions (AO) map of a sculpture.

6.2 An 1-D illustration of ambient occlusions (AO) for different scene points.

6.3 (a) A simple scene illuminated by one of our directionally encoded light sources. Point P’s AO is affected by the sphere and the cylinder. We use 50 different patterns and (b) shows the P’s intensity with respect to different patterns. (c) shows P’s ground truth visibility distribution over the hemisphere (foreshortened by its normal). (d) shows intermediate visibility estimation results at different iterations by our algorithm.
6.4 different properties of three different types light source. 73
6.5 Comparison between my result and [2] on AO estimation on synthetic datasets. Second row shows the ambient occlusion results on the Tentacle scene. Last row shows the recovery albedo and ambient occlusion with closeup views. 75
6.6 Another comparison between our result and [2] on AO estimation on synthetic datasets. Second row shows the ambient occlusion results on the Tentacle scene. Last row shows the recovery albedo and ambient occlusion with closeup views. 76
6.7 The Root Mean Square Error of AO and albedo estimations with different input sizes. 77
6.8 Ambient occlusion and albedo estimation using our approach on the scholar and frog dataset. 79
6.9 Ambient occlusion and albedo estimation using our approach on the scholar and frog dataset. 80
6.10 Fitting a hexagon grid to the scanned light field probe. 81
6.11 Projector-Light Field Probe Alignment. 82
6.12 Calibrating lighting direction using mirror ball. 83
6.13 Calibrating the lighting direction of a mirror ball. Left: lighting directions represented by points on a uniform sphere. Identical directions are shown in the same color. Right: we detect specular highlights on the mirror ball for calibrating the lighting direction. 84
6.14 Each pixel on the pattern maps to unique ray direction and a binary pattern maps to a coded directional light source. 85
6.15 Directional invariance of light field probes. We project identical color wheel images for every microlenslet so that each lighting direction maps to a unique color. 86
6.16 The structure of our measure system. 87
6.17  Left: our system uses two light field probes to emulate encodable
directional light sources. Right: Closeup views of the light field
probes. 

6.18  The AO, normal and albedo estimation results on a real Donald duck.
The bottom row is the closeup view.

ABSTRACT

The problem of modeling and reconstructing the “invisibles”, e.g., specular or transparent objects such as 3D fluid wavefront and gas flows, has attracted much attention in recent years. Successful solutions can benefit numerous applications in oceanology, fluid mechanism and computer graphics as well as lead to new insights towards shape reconstruction algorithms. The problem, however, is inherently difficult for a number of reasons. First such objects do not have their own image. Instead, they borrow appearance from nearby diffuse objects. Second, determining the light path within these objects for shape reconstruction is non-trivial since refractions or reflections non-linearly alter the light paths. Finally, dynamic specular or transparent objects often exhibit spatially and temporally varying distortions that are hard to correct. To capture the “invisibles”, most previous approaches are based on establishing point-pixel correspondences. It is well-known that point-pixel correspondences are under-constrained even for single reflection or refraction. In this dissertation, I propose to resolve the point-pixel ambiguity by using novel computational imaging devices that encodes illumination directions.

In particular, I first developed a multi-view based solution for robustly capturing fast evolving fluid wavefronts. I constructed a portable camera array system as the main acquisition device. I elaborately designed the system to allow high-resolution and high-speed capture and addressed practical issues such as data streaming and storage and time-divided multiplexing.

Then I exploit using Bokode - a computational optical device that emulates a pinhole projector - for capturing ray-ray correspondences which can then be used to directly recover the dynamic fluid surface normals. I further develop a robust feature
matching algorithm based on the Active Appearance Model (AAM) to robustly establishing ray-ray correspondences. My solution results in an angularly sampled normal field and we derive a new angular-domain surface integration scheme to recover the surface from the normal fields.

I also show another novel computational imaging solution to recover the dynamic gas flow by exploiting the light field probe (LF-Probe). A LF-probe resembles a view-dependent pattern where each pixel on the pattern maps to a unique ray. By observing the LF-probe through the gas flow, I acquire a dense set of ray-ray correspondences and then reconstruct their light paths. To recover the RIF, I use Fermat’s Principle to correlate each light path with the RIF via a Partial Differential Equation (PDE). I then develop an iterative optimization scheme to solve for all light-path PDEs in conjunction.

Finally, I extend my directional light coding approach to recover the ambient occlusion (AO) map of an object. In particular, I adopt a compressive sensing framework that captures the object under strategically coded lighting directions. I show that this incident illumination field exhibits some unique properties suitable for AO recovery. Experiments on synthetic and real scenes show that our approach is both reliable and accurate with significantly reduced size of input.
Chapter 1
INTRODUCTION

The problem of modeling and reconstructing time-varying transparent objects such as dynamic 3D fluid wavefront and gas flows has attracted much attention in recent years [3, 4, 5]. Successful solutions can benefit numerous applications in oceanology [6], fluid mechanism [7] and computer graphics [8] as well as lead to new insights towards shape reconstruction algorithms. The problem, however, is inherently difficult for a number of reasons. First specular surface does not have its own image. Instead, it borrows appearance from nearby diffuse objects. Second, determining the light path for shape reconstruction is non-trivial since refractions or reflections non-linearly alter the light paths. Finally, dynamic specular surfaces often exhibit spatially and temporally varying distortions that are hard to correct. Transparent gas flows are even more difficult to reconstruct since the refractive index field (RIF) within the gas volume is uneven and rapidly evolving, introducing large image distortions. Finally, the acquisition system needs to be 1) nonintrusive to avoid affecting the dynamics and 2) portable to support on-site acquisition. Fig. 1.1 shows some examples of transparent objects.

Existing solutions on transparent object reconstruction can be essentially viewed as a special class of multi-view reconstruction algorithms. Often a known pattern such as a checkerboard is positioned near the object surface and conceptually one can analyze the corresponding feature points in the observed cameras views and then apply stereo [3] or volumetric reconstruction [4] techniques for recovering the surface. In reality, point-pixel correspondences are under-constrained even for single reflection or refraction: to determine the surface normal, it is necessary to know both the incident and the exit ray directions; the pixel location provides the exit direction but the 3D
point does not provide the incident direction, unless the surface height is known in prior. To resolve this ambiguity, additional constraints such as the planarity assumption [9, 10], surface smoothness prior [11] and surface integrability constraints [12] have been proposed.

1.1 Dissertation Statement

In this dissertation, I propose using computational imaging devices to reconstruct these challenging “invisible” phenomena.

I first developed a multi-view based solution for robustly capturing fast evolving fluid wavefronts. I constructed a portable camera array system as the main acquisition device. I elaborately designed the system to allow high-resolution and high-speed capture and addressed practical issues such as data streaming and storage and time-divided multiplexing.

Then I exploit using Bokode - a computational optical device that emulates a pinhole projector - for capturing ray-ray correspondences which can then be used to
directly recover the dynamic fluid surface normals. I further develop a robust feature matching algorithm based on the Active Appearance Model (AAM) to robustly establishing ray-ray correspondences. My solution results in an angularly sampled normal field and we derive a new angular-domain surface integration scheme to recover the surface from the normal fields.

I also show another novel computational imaging solution to recover the dynamic gas flow by exploiting the light field probe (LF-Probe). A LF-probe resembles a view-dependent pattern where each pixel on the pattern maps to a unique ray. By observing the LF-probe through the gas flow, I acquire a dense set of ray-ray correspondences and then reconstruct their light paths. To recover the refractive index field (RIF), I use Fermat’s Principle to correlate each light path with the RIF via a Partial Differential Equation (PDE). I then develop an iterative optimization scheme to solve for all light-path PDEs in conjunction.

Finally, I extend my directional light coding approach to recover the ambient occlusion (AO) map of an object. In particular, I adopt a compressive sensing framework that captures the object under strategically coded lighting directions. I show that this incident illumination field exhibits some unique properties suitable for AO recovery. Experiments on synthetic and real scenes show that our approach is both reliable and accurate with significantly reduced size of input.

1.2 Contributions

This dissertation makes the following contributions in reconstructing “invisible” phenomena using computational imaging devices.

**Dynamic Fluid Wavefront:** I present a multi-view based solution for robustly capturing fast evolving fluid wavefronts. I construct a portable, $3 \times 3$ camera array system as the main acquisition device. I elaborately design the system to allow high-resolution and high-speed capture. My solution can handle the challenging problem of losing track of feature points which can be detrimental or even fatal to single-camera or stereo based methods.
I investigate using Bokode - a computational optical device that emulates a pinhole projector - for capturing ray-ray correspondences which can then be used to directly recover the surface normals. I further develop a robust feature matching algorithm based on the Active Appearance Model to robustly establishing ray-ray correspondences. My solution results in an angularly sampled normal field and I derive a new angular-domain surface integration scheme to recover the surface from the normal fields.

**Dynamic Gas Flow:**

I present a novel computational imaging solution for recovering the refractive index field (RIF) of dynamic gas flow using the light field probe (LFProbe). I use the LF-probe for generating view-dependent features: by coupling the LF-probe with a viewing camera, I establish reliable ray-ray correspondences. To recover the RIF, I use Fermats Principle to correlate each light path with the RIF via a Partial Differential Equation (PDE). I then develop an iterative optimization scheme to solve for all light-path PDEs in conjunction.

**Ambient Occlusion:**

I present a novel computational imaging solution for recovering the ambient occlusion (AO) map of an object using encoded directional light source. I adopt a compressive sensing framework that captures the object under strategically coded lighting directions. I propose a sparsity-prior based solution for iteratively recovering the surface normal, the surface albedo, and the visibility function from a small number of images. To physically implement the scheme, I construct an encodable directional light source using the light field probe.

**1.3 Blueprint of the Dissertation**

This dissertation is organized as follows. Chapter 2 summarizes the related work on recovering transparent objects. Chapter 3 presents a multi-view method for transparent wavefront reconstruction using camera array. Chapter 4 presents a single camera dynamic fluid surface reconstruction approach using Bokode. Chapter 5
presents a computational imaging solution for gas flow reconstruction using light field probe. Chapter 6 presents a compressive sensing framework for Ambient Occlusion reconstruction using directional coded light source. Chapter 7 concludes the thesis and discusses future directions.
Chapter 2
RELATED WORKS

2.1 Invisible Phenomenon

Static Reflective and Refractive Surfaces. Earlier approaches [13] have focused on modeling variations of reflection highlights for recovering surface geometry and reflective properties. Sankaranarayanan et al.[14] use point-pixel correspondences to first estimate the specular flow and then apply quadric approximations to recover mirror-type surfaces. A common issue in point-pixel based solutions is ambiguity: a pixel corresponds to a ray from the camera while the specular surface can lie at any position along the ray. Tremendous efforts have been focused on adding additional constraints [9, 10, 11, 15] for resolving this ambiguity. Bonfort and Sturm [16] use images captured by multiple calibrated cameras to reconstruct specular surface via space carving. Kutulakos and Steger [17] discover that by analyzing the piecewise linear light paths in homogeneous refractive medium, one can view surface reconstruction as a generalized triangulation problem. Their work showcases the usefulness of light paths. In this paper, we demonstrate using non-linear light paths for recovering inhomogeneous refractive media.

Transparent Wavefronts. The problem of acquiring dynamic wavefronts is relatively new to computer vision. Morris and Kutulakos [18] track the corners of a checkerboard pattern over time in a stereo camera setting and then impose the refractive disparity constraint to iteratively solve for surface heights and normals. In a similar vein, Ding et al.[4] construct a camera array system to obtain multi-view point-pixel correspondences. Robustly tracking feature points, however, can be challenging as the observed image can exhibit severe distortions and motion blurs. Ye et al.[19] point out that to
robustly recover light paths, it is important to establish ray-ray correspondences. For example, they propose using Bokode [20], a special pinhole projector, to acquire ray-ray correspondences for directly recovering fluid surface normals. Closest to our approach, Wetzstein et al.[5] replace the conventional checkerboard pattern with a LF-probe with 4D spatial and angular coding to obtain more accurate ray-ray correspondences. We study non-linear light paths using ray-ray correspondences.

Gas Flows. One of the most challenging transparent objects is 3D gas flows. In mechanical engineering, the widely adopted solution is Schlieren photography. Schardin [21] uses a knife edge to partially block rays proportional to their refracted directions to visualize dynamic gas flows, refractive solids, and shock waves. The original Schlieren photography, however, aims to visualize rather than reconstruct flows. Howes [22] modifies the traditional Schlieren to conduct quantitative evaluation of refractive index distribution by encoding the hue. Dalziel et al.[23] propose a much simpler and inexpensive solution called Background Oriented Schlieren (BOS) that images the refractive media via the distortion of a high-frequency background. It then calculates per-pixel deflection vectors using the optical flow. This class of methods [24, 25] can acquire the shifts of pattern positions due to refractions but not directions. The 3D tomography technique by Atcheson et al.[1] captures distorted wavelet noise patterns through the gas volume from multiple viewpoints. It then measures deflections caused by gas refraction to correlate the incident ray (i.e., from the camera) to the exit point (i.e., the feature point on the background pattern). The ray-point correspondences work well for distant patterns (e.g., around 3 meters in their experiments) and is suitable for acquiring large scale gas flow. We develop a low-cost, portable solution for acquiring gas flows of small to medium scales.

Intrinsic Image Recovery. Traditional approaches have focused on intrinsic properties related to reflectance (albedo), normal and geometric occlusion. Many single images based methods [26, 27, 28] aim to perform the reflectance and illumination decomposition. Theoretically, these methods can potentially used to recover AO when the scene is illuminated by a uniform and constant light source. However they rely
on smoothness prior and hence are not reliable in the presence of complex albedo and surface geometry. Recent studies show that image sequences based methods (e.g. [29]) cannot robustly handle soft and persistent shadows produced by AO. Our work is also related to visibility estimation in photometric stereo[30, 31, 32]. Sunkavalli et al.[31] approximate lighting visibility via dimensionality reduction in illumination spaces. Their method, however, is not suitable for handling a large number of varying lighting conditions. Aldrian and Smith [33] conduct inverse-render on simple geometry such as faces under uniform and constant illumination. Hauagge et al.[2] estimate AO from a stack of images captured under a moving point light source. They adopt a simplified visibility model and show impressive results on recovering AO. A major drawback of their technique is the requirement of a large of number input images, which we aim to reduce through computational imaging and compressive reconstruction.

2.2 Acquisition Hardware

Existing image-based solutions for reconstructing real specular surfaces such as mirrors and fluid surfaces can be classified into three categories.

**Single-Camera Solutions.** Solutions in this category use only one camera for recovering the surface. These methods often assume fluid surfaces are piecewise planar and then solve a generalized Structure-from-Motion (SfM) problem [34]. General specular surfaces can also be recovered from distortions [35]. Recent work by Ding et al.[36] first recovers the refraction ray geometry from distorted line patterns (curves) and then approximates surface differential attributes such as curvatures from ray geometry. However, they assume nearly flat surfaces whereas we aim to acquire fast evolving wavefronts. Further, both SfM and shape-from-distortion methods rely on accurately tracking features and are therefore sensitive to image distortions. Our camera array solution resolves this issue via a multiview approach: if one camera loses correspondences, its near cameras can be used to fill in the gap.

**Stereo-Camera Solutions.** One can also use a stereo setup for specular surface reconstruction. Sanderson et al.[37] proposed a stereo camera configuration for resolving
ambiguities commonly observed in single-camera based methods. Morris and Kutulakos [3] introduced the notion of refraction disparity and developed new optimization schemes to simultaneously estimate the height field and the normal field. However, similar to the single-camera solutions, these methods still suffer from refraction distortions and motion blurs: once a camera $C^*$ loses track of the feature points at a frame, it can no longer track them in its consecutive frames. Our solution resolves this problem by using the rest of the cameras to first approximate the surface and then ray-trace the surface to locate the missing feature points in $C^*$ so that we can continue tracking the points.

**Multi-View Solutions.** Finally, one can use multiple cameras for acquiring specular surfaces. Blake [13] measured the variations of specularities from different viewing directions to determine the differential properties of the surface. Bonfort and Sturm [38] used multiple-view geometry to build a volumetric reconstruction of mirror surfaces. Most previous approaches are specifically designed for static specular surfaces. For instance, one can dynamically adjust the calibration pattern to produce reliable correspondences. However, for capturing dynamic fluid surfaces, the pattern needs to be fixed in space and robustly tracking the feature points on the pattern is much more difficult. In addition, the fluid surface is rapidly evolving. Therefore, it is crucial to capture and store the imagery data at high speed to avoid motion blurs.

**Computational Imaging.** The core of our technique is to create a controllable illumination field. Masselus et al.[39] position an object on a rotational table and the illuminate it using a projector. By exploiting the rich shading and geometric information through spatial-angular analysis, they can relight the object with high realism. Later, Cossairt et al.[40] apply computational illumination to produce synthetic illumination between real and synthetic objects. Debevec [41] uses the Light Stage system to exploit the richness in angular and spatial variation of the light field, and the object can be relit with nearly realism. Ezra et al.[42] proposes a novel BRDF measurement device consisting exclusively of LEDs. Their device uses no cameras and is fast and simpler for measuring BRDF.
Our idea resembles the inverted light field camera. Light fields [43] are image-based representations that gather rays sampled both spatially and angularly. Physically, they can be collected through a light field camera array or more recently a light field camera. The concept of light field illumination is relatively new. Wetzstein et al. [44] put coded image patterns behind a microlens array to encode rays spatially and angularly. The device, called the light field probe, can be used to determine light path variations and then to reconstruct transparent objects [45]. In a similar vein, Ji et al. [46] place a color coded pattern behind the light field probe to capture light paths through a 3D gas flow for volumetric heat reconstruction. In this paper, we build a controllable directional illumination field using the light field probe.

2.3 Reconstruction Algorithm

Point-Pixel Correspondences. Most existing solutions for specular surface reconstruction build upon point-pixel correspondences where a special planar pattern such as a checkerboard is placed near the surface and a single or multiple cameras are used to acquire the distorted pattern for shape reconstruction. Murase [10] analyzed the optical flow between the distortion image and the original one and used the center of trajectory to establish point-point correspondences. Blake [13] examined the variation of reflected highlight by changing the viewing position to recover the surface geometry and reflective properties Bonfort and Sturm [38] used images captured by multiple calibrated cameras to reconstruct specular surface with voxels. Recently Sankaranarayanan et al. [14] used standard SIFT algorithm to match point-pixel correspondences resulting from specular flow and further used quadrics approximation to recover mirror-type surfaces from sparse samples. A common issue in point-pixel based solutions is ambiguity: a pixel corresponds to a ray from the camera while the specular surface can lie at any position along the ray. Tremendous efforts have been focused on adding additional constraints [9, 10, 11, 12] for resolving this ambiguity.

The problem of acquiring dynamic specular surfaces is relatively new to computer vision. Morris and Kutulakos [3] tracked the corners of the checkerboard pattern
over time to establish point-pixel correspondences and then imposed the refractive disparity constraint to iteratively solve for surface height and surface normal. However, robustly tracking feature points on dynamic surfaces is challenging as the observed image can exhibit sever distortions and motion blurs. In a similar fashion, Ding et al.\cite{4} recently constructed a camera array system to obtain multi-view point-pixel correspondences. When one of the cameras loses track, the rest of the cameras can still recover the surface and the result can be used to warp the lost-track feature points back to the camera.

Ray-Ray Correspondences. A different class of solutions that can directly resolve the point-pixel ambiguity is to use ray-ray correspondences. The earlier work of Sanderson et al.\cite{37} controlled the illumination direction and coupled it with the observed specular highlights to form ray-ray correspondences. Kutulakos and Steger \cite{47} recovered complex-shaped static specular objects by computing the light paths from the specular object to the camera. By studying indirect projection of 3D points, they formulated the problem of recovering the light path as a general triangulation problem. However their framework by far can only handle static objects as it requires acquiring the object twice whereas we present a simpler solution to directly handle dynamic 3D fluid surfaces.

Closely to our solution, Wetzstein et al.\cite{5} recently to replace the conventional checkerboard pattern with a light field probe which encodes 4D spatial and angular information. In their setup, they used color gradients to code the 2D incident ray direction and 1D (vertical) feature point position. The second (horizontal) dimension of the feature point can be recovered through geometric constraints. Their approach can achieve highly accurate ray-ray correspondences. Our solution differs from theirs in a number of ways. First, we use a much simpler and affordable device, a Bokode that can be easily constructed from a webcam, in place of the light field probes. Our ray-ray correspondences, however, are less accurate than the ones obtained by the light field probe and they cannot be used to directly recover the surface. We therefore only use the recovered normal field and develop an angular-domain normal field integration.
scheme. Finally, dynamic fluid surfaces often cause strong chromatic aberrations and intensity changes due to caustics. Therefore, we choose not to use the color-coded pattern but a special monochromatic pattern and apply Active Appearance Model (AAM) for correspondence matching.

**Compressive Sensing.** Finally, our work is closely related to latest advances on Compressive Sensing (CS) [48] for signal reconstruction. The literature of CS is huge and we refer the readers to the comprehensive survey [49]. In photography, CS allows a reduction in image acquisition energy per image by as much as a factor of 15 at the cost of complex decompression algorithms [50]. There is also emerging interest on applying CS for multi-spectral imaging [51], light field imaging [52] and super-resolution [53], light transport [54], and depth sensing [55]. Our work employs the compressive sensing technique to reduce the number of input images for AO reconstruction.
Chapter 3

DYNAMIC FLUID SURFACE ACQUISITION USING A CAMERA ARRAY

Acquiring dynamic 3D fluid surfaces is a challenging problem in computer vision. Single or stereo camera based solutions are sensitive to refraction distortions, fast fluid motions, and calibration errors.

In this Chapter, I present a complete multi-view based solution for robustly capturing fast evolving fluid wavefronts. I first construct a portable, $3 \times 3$ camera array system as the main acquisition device. The system is controlled by a single workstation and hence is portable. I also address practical issues such as data streaming and storage and time-divided multiplexing for high speed acquisition.

To recover fluid surfaces, I place a known pattern beneath the surface and position the camera array on top to observe the pattern. By tracking the distorted feature points over time and across cameras, I obtain spatial-temporal correspondence maps and use them for specular carving to reconstruct the time-varying surface. In case one of the cameras loses track due to distortions or blurs, I use the rest cameras to construct the surface and apply multi-perspective warping to locate the lost-track feature points so that I can continue using the camera in later frames. I apply our system to capture a variety types of fluid motions. Experiments on synthetic and real data demonstrate that our framework is robust and reliable.

3.1 Acquisition Device

I first present a portable camera array system for acquiring the fluid surfaces. In recent year, a number of camera systems have been developed for specific imaging tasks. For example, the Stanford light field camera array [56, 57, 58, 59] is a two dimensional
grid composed of 128 1.3 megapixel firewire cameras which stream live video to a stripped disk array. The MIT light field camera array [60] uses a smaller grid of 64 1.3 megapixel USB webcams for synthesizing dynamic Depth-of-Field effects. These systems require using multiple workstations and their system infrastructure such as the camera grid, interconnects, and workstations are bulky, making them less suitable for onsite tasks.

We have constructed a small-scale camera array controlled by a single workstation. Our system uses an array of 9 Pointgrey Flea2 cameras to capture the dynamic fluid surface. We mount the camera array on a metal grid attached to two tripods so that we can easily adjust the height and the orientation of the camera array. The camera array is connected to a single data server via 5 PCI-E Firewire adaptors. The use of Firewire bus allows us to synchronize cameras through the Pointgrey software solution.

**Data Streaming.** Streaming and storing image data from 9 cameras to a single workstation is another challenge. In our system, a camera captures at 8-bit images of resolution 1024x768 at 30fps. This indicates that we need to stream about 2Gbps data. To store the data, previous solutions either use complex computer farm with
Figure 3.2: (Left) The setup of our fluid surface acquisition system. (Right) We divide the camera array into two groups (gray and black) and interleave the trigger for each group to double the frame rate.

fast ethernet connections or apply compression on the raw imagery data to reduce the amount of data. For fluid surface acquisition, the use of compression scheme is highly undesirable as it may destroy features in the images. We therefore stream and store uncompressed imagery data. To do so, we use an external SATA disk array as the data storage device. The disk array is equipped with 6 Seagate 1TB SATA disks, and connected to the server through a PCI-E x4 card. We configure RAID 0 for disk array to achieve the maximum performance. It is also worth noting that our system is also more affordable: by eliminating multiple workstations, network devices and external camera synchronization units, our system (including 9 cameras) has a total cost under 10,000.

**Time-Divided Multiplexing.** Since the Flea2 cameras can only achieve a maximum frame rate of 30fps, we have adopted a time divided multiplexing scheme to further improve the frame rate of our system. Our solution is similar to the Stanford light field high speed imaging scheme [58] that interleaves the exposure time at each camera. Specifically, we divide the camera array into two groups, four in one group and five in the other. We set the exposure time of each camera to be 10ms to reduce motion blurs. While all cameras still capture at 30fps, we trigger the second camera group with a 1/60 second delay from the first one. We also develop special algorithms for warping
the reconstruction result from the first group to the second (Section 4.2) so that our system is able to perform at 60fps.

**Experiment Setup.** We use a off-the-shelf glass water tank of dimension 30in × 12in × 12in placed firmly on a metal grid to contain dynamic fluids. We print a black-white checkerboard pattern on regular paper, laminate it, and then glue it to a planar plastic plate. We stick this plate onto bottom of the container and use it for both camera calibration and feature tracking.

**Lens Specs.** In our experiments, the choice of camera lenses is also crucial in our acquisition process. For example, a camera’s field-of-view should be large enough to cover the complete fluid surface. In our setup, we choose Rainbow 4.8mm wide angle lens with a focal distance of 12in. Since all cameras are mounted on a reconfigurable rig, we can easily adjust the camera baseline to achieve optimal reconstructions.

**Calibration.** A number of options [56, 60] are available for calibrating the cameras in the array. Since the observable regions of our cameras have large overlaps, we directly use Zhang’s algorithm [61] for calibration by reusing the checkerboard pattern mounted at the bottom of the tank. This approach also has the advantage of automatically calibrating all cameras under the same coordinate system and simplifies our feature warping scheme.

### 3.2 Fluid Surface Reconstruction

To recover dynamic fluid surfaces using our system, we place a known pattern beneath the surface so that each individual camera in the array will observe a distinct time varying distortion pattern. We develop a feature tracking algorithm for robustly tracking checkerboard corners under both distortions and motion blurs. In particular, if a camera $C^*$ in the array loses track, we use the rest cameras to reconstruct the surface and then apply multi-perspective ray tracing to locate the feature points in $C^*$ so that we can still use the camera in later frames. We call this process feature warping.
The tracking results provide pixel-point correspondences in each camera. We then generate a dense (per-pixel-based) correspondence map and apply specular carving to reconstruct the normal field of the fluid surface. We then integrate the normal field to obtain the actual fluid surface. We further apply feature warping to processing time-divided multiplexing image array data to double the frame rate of our acquisition. Fig. 3.3 illustrates the processing pipeline of our framework.

### 3.2.1 Correspondence Maps

A crucial step in our fluid surface reconstruction algorithm is to establish feature correspondences. Notice, although each single corner may be strongly distorted and shifted, the local checkerboard structure (e.g., the relative position of neighboring corners) generally remains stable. We therefore introduce the local affine constraints on the square grid. We assume one-to-one correspondence mapping and each feature point (square corner) can be expressed as affine combinations of its neighboring feature points, and the affine coefficients need to be within a certain range. Let $I_0$ and $I_1$ be the two consecutive frames in a camera in the array, $(V,E)$ represents the lattice...
graph of the square grid where $V$ denotes the checker-board corners and $E$ denotes the edges connecting the corners. Assume that $p_i = [x_i, y_i]^T \in V$ denotes the detected or previously tracked corners in $I_0$ and $m(p_i)$ are their correspondences in $I_1$. Our goal is to establish correspondences between $p_i$ and $m(p_i)$. We compute the matching cost as the combination of the geometric error and the appearance error:

$$
\sum_{i=1}^{\vert V \vert} \{ \| I_0(N_{p_i} - W(I(N_{p_i}))) \| + \lambda \cdot g(p_i, N_{p_i}; m(p_i), N_{m(p_i)}) \} \quad (3.1)
$$

where $W$ is an affine warping function defined by the neighboring corner correspondences. $\| I_0(N_{p_i} - W(I(N_{p_i}))) \|$ is the appearance matching cost for the local patch around $p_i$. $\lambda$ is a constant controlling the relative weight between the appearance cost and the geometric costs. $g(\cdot)$ is the geometric cost that measures the geometric dissimilarity between the original checker-board corner set $N_{p_i}$ and the candidate corner set $N_{m(p_i)}$. Assume $p_i$ as the affine combination of $N_{p_i} : p_i = A \cdot N_{p_i}$. $A$ is the affine coefficients, and similarly, $A_m$ represent the affine coefficients for $m(p_i)$. We define the geometric cost as:

$$
g(p_i, N_{p_i}; m(p_i), N_{m(p_i)}) = \| A - A_m \| \quad (3.2)
$$

We incorporate the matching cost in Eqn. 3.1 and 3.2 into the inverse-compositional framework of the Lucas-Kanade (L-K) tracker [62] and use the modified L-K tracker for tracking checker-board corners under refraction distortions. Figure 3.4 compares the tracking results on one viewing camera using the classical L-K tracker and our approach. Our method is able to robustly track the feature points in presence of strong refraction distortions whereas L-K loses track at these places.

Recall that the tracked feature correspondences are rather sparse. Since our goal is to conduct dense multi-view reconstruction, we further construct dense correspondence maps. Given a correspondence map in each camera at every frame, we first compute a distance map using Chamfer Distance Transform, then generate a local Delaunay triangulation using the 8-neighbors for every sample, and finally interpolate per-pixel correspondence (in terms of the horizontal and vertical shifts) using the
Figure 3.4: Classical Lucas-Kanade (L-K) tracker loses track at regions with strong distortions. Our locally constrained L-K tracker is able to robustly track in these regions.

triangulation. It is important to note that our tracking algorithm may still fail in presence of strong distortions. While this is fatal to most previous single-camera or stereo-camera based approaches [35, 3], our multi-view approach naturally resolves this problem. Specifically, we benefit from the analysis that distortions on specular surfaces are non-generic [63]. Therefore, if one camera $C^*$ in the array loses track on a surface patch due to strong distortions, it is unlikely that the rest cameras will observe the same level of distortions. This indicates that can use the rest cameras to construct the patch. Further, we can apply multi-perspective ray-tracing to locate the lost-track feature points in $C^*$ so that we can continue using $C^*$ in later frames instead of completely discarding $C^*$.

3.2.2 Volumetric Reconstruction

In the previous subsection, we discussed how to track feature points over time in each viewing camera. At the beginning of the acquisition, we further use the camera
array to capture a nearly flat (undisturbed) fluid surface. This process allows us to establish spatial correspondence across the views. Combined with the temporal tracking results, we hence establish spatial-temporal correspondences. Therefore, we can apply specular surface carving [38] for reconstructing the surface.

Specifically, given features correspondences between N views, our goal is to produce a volumetric reconstruction of the fluid surface. To do so, we first discretize the bounding volume of the fluid surface into 3D voxels and then measure the consistency of light paths for each voxel.

**Refraction Light Path Consistency.** Assume $m$ is the refractive index of the surface. The incident ray $\vec{i}$, the exit ray $\vec{r}$, and the normal $\vec{n}$ should be coplanar, and the incident angle $\phi_1$ and exit angle $\phi_2$ satisfy Snells Law, as shown in Figure 3.5(a), i.e.,

$$\frac{\sin \phi_1}{\sin \phi_2} = m \quad (3.3)$$

For every voxel inside the volume, we first project it to each of the observing cameras to get its corresponding pixel coordinate. We then look up the correspondence maps to find its corresponding 3D feature points. As a result, every voxel defines a set of light paths from each of the viewing cameras to the pattern plane. Further, we can solve for the unique normal given by each light path as follows. Assume the voxel is at $\hat{O}[o^x, o^y, o^z]$, the observing cameras COP is $\hat{Q}[q^x, q^y, q^z]$ and the checker board corner position $\hat{P}[p^x, p^y, p^z]$, the normalized incident ray direction $\vec{i}$ and refraction ray direction $\vec{r}$ hence are:

$$\vec{i} = \frac{\hat{O} - \hat{Q}}{||\hat{O} - \hat{Q}||} , \quad \vec{r} = \frac{\hat{P} - \hat{O}}{||\hat{P} - \hat{O}||} \quad (3.4)$$

Let $\alpha$ denote the angle between $\vec{r}$ and $\vec{i}$, we have:

$$\phi_2 + \alpha = \phi_1 , \quad \cos \alpha = \vec{i} \cdot \vec{r} \quad (3.5)$$

Combining it with Eqn. (3), we have:

$$m = \frac{\sin(\alpha + \phi_2)}{\sin \phi_2} \Rightarrow \tan \phi_2 = \frac{\sin \alpha}{m - \cos \alpha} \quad (3.6)$$
Figure 3.5: Our refraction carving framework. (a) Given the incident and the refracted directions, we can solve for the surface normal. (b) Each voxel will give a different normal estimate. (c) Using an array of cameras, we can find the optimal voxel/normal. (d) Finally, we apply space carving using the normal consistency criteria.
Figure 3.6: In case a camera $C^*$ loses track of a feature point $P$, we use the recovered surface from the rest of the cameras and backtrace $P$ to $C^*$.

We thus can decompose $\vec{i}$ to $\vec{r}_i$ along $\vec{r}$ and its orthogonal direction $\vec{h}$, and compute the orthogonal component as:

$$\vec{h} = \vec{i} - \vec{r} \cos \alpha$$  \hspace{1cm} (3.7)

Finally, we can compute the normal direction as:

$$\vec{n} = \sqrt{1 - (\vec{i} \cdot \vec{r}^2)} \left( \vec{i} - (\vec{i} \cdot \vec{r}) \vec{r} \right) - \vec{r}$$  \hspace{1cm} (3.8)

Once we obtain the normal estimates for each light path, we can then measure the consistency as:

$$D = \frac{1}{W} \sum_{1 \leq i, j \leq N, i \neq j} ||\vec{n}_i - \vec{n}_j||$$  \hspace{1cm} (3.9)

where cameras $i$ and $j$ observe “valid” correspondences at the pixel (we exclude the cameras that lose track due to distortions) and $W$ is the total number of light path pairs.

**Restricted Search.** To reduce the computational cost and improve the robustness, we can use a coarse-to-fine scheme. Conceptually, we can first discretize the volume using...
large voxels (lower resolution) to obtain an initial estimation of the surface. Based on the rough estimation, we can then refine the voxels to improve the resolution of the reconstruction. Although faster, this simple scheme is also more sensitive to noise and hence less robust. Our solution is to effectively use temporal coherence of the surface. Assume we have finished reconstructing the fluid surface as a height field $z_k(x, y)$ for frame $k$, where $x$ and $y$ correspond to the two discretization dimensions. For every $[x, y]$, we define a search range in $z$ as: $[z_k(x, y) + L_1 m_k(x, y) - L_2, z(x, y) + L_1 m_k(x, y) + L_2]$ where $m_k(x, y)$ represents the motion vector in $z$ direction of voxel $[x, y, z_k(x, y)]$ in frame $k$, and $L_1$ and $L_2$ are two constants to tolerate the motion estimation errors, as shown in Figure 3.5(d). This significantly reduces the search range and improve the robustness of the carving scheme. Surface Integration. Our specular carving algorithm outputs both the height map and the normal map of the surface. However, the height map is often much more noisy than the normal map, as shown in Figure 3.7. We therefore choose to use the normal map and integrate it to recover the surface. Surface integration from a gradient field has been well studied and can be formulated as to solve the Poisson equation. In our implementation, we adopt a similar approach to [2] for recovering the surface. To reduce noise, we further smooth the height map to obtain more reliable boundary conditions. Figure 3.7 compares our reconstruction result with the ground truth on synthetic fluid surfaces.

CUDA Implementation. Our volumetric reconstruction algorithm needs to be applied to a large number of high-resolution frames. Since the operation conducted at each voxel is nearly the same (estimating the normal and measuring the consistency), we have implemented a GPU version of the algorithm using NVidias CUDA. Compared with the un-optimized CPU solution, our GPU implementation achieves over 100 times speedup on an Nvidia GeForce 9600 Graphics Card. We also use CUDA to implement the surface integration algorithm to form a unified processing pipeline.

Feature Warping. Compared with the single or stereo camera based solutions, our multi-view approach can use the rest cameras to recover the surface even if one camera loses track due to distortions or blurs. Further, our implementation interleaves the
Figure 3.7: Results on synthetic data. (a-c) show the ground truth height field and normal field. (d-f) show the corresponding raw refraction carving results. Notice that the recovered height field is very noisy but the normal field is highly accurate. (g) We integrate (e-f) using the boundary from (d) to obtain the final reconstruction. (h-i) we compare the ray traced refraction distortions on the ground truth surface (h) and our reconstruction (i).
exposure of two camera groups in order to double the frame rate in our acquisition. In both case, it is essential to “recover” the lost-track feature points in a single or a group of cameras in order to continue the tracking tasks in later frames.

Specifically, given the reconstructed surface as a triangular mesh, our goal is to find where the feature points (corners on the checkerboard) will be located in a viewing camera $C^*$, where $C^*$ can be the one that loses track or in a different camera group. Since refraction is non-linear phenomenon, this is a classical inverse ray tracing problem and generally does not have a closed-form solution. Our solution is to apply multi-perspective warping using the recently proposed General Linear Cameras [64]. We first apply forward tracing: for each triangle $\hat{P}_{1,2,3}$ on the mesh, we forward trace the three rays at the vertices from the camera to checkerboard as $\vec{r}_{1,2,3}$ which also defines a local GLC. We intersect the rays with the checker-board plane at $\hat{P}_0$. We then find checkerboard corners inside $\Delta \hat{P}_1\hat{P}_2\hat{P}_3$ and using the local GLC to compute its refraction ray $\vec{r}'$ and trace the ray back to $C^*$ and find its corresponding pixel coordinate.

### 3.3 Results and Discussions

We have validated our framework on both synthetic and real fluid surfaces. For synthetic surfaces, we use the PovRay Ray Tracer to render 9 image sequences of fluid surfaces from a virtual 3x3 camera array. For real surfaces, we capture an array of video streams of the surface using our acquisition device.

**Synthetic Surfaces.** We first conduct experiments using our method on simulated fluid motions. In Figure 3.8 column 2, we show sample frames of a “drop” wave sequence, where the initial wavefront is a Gaussian function $z(x, y) = 0.1e^{-(x-w/2)^2-(y-h/2)^2}$, where $w = h = 128$. We assume that the fluid dynamics follows the Navier-Stokes (NS) equation and propagate the wave via a discrete NS solver [16]. We use a checkerboard with 23x35 grids of unit squares and position it at $z = 0$. We further set the refraction index of the fluid as $1.33$ to emulate water. Using the PovRay, we render images from a 3x3 camera array positioned at $z = 30$. The camera plane is set to be
parallel to the xy plane and the FOV of each camera set to match the real ones in our camera array. We start with detecting all feature points (corners) at the first frame and correlate them with the ones on the checker-board. As the wavefront propagates, we apply our modified L-K algorithm to track these features over time. Since all frames are rendered without motion blurs or noise, we found our algorithm highly robust for establishing spatial temporal correspondence maps. Next, we combine the tracking results with the ground truth camera parameters for volumetric reconstruction. We apply our specular carving scheme at a resolution of 100x100 in xy dimension. The volumetric reconstruction provides a height map and a normal map at each time instance. We observe that, by using the normal consistency constraint, the recovered normal maps are much more accurate than the height map, as shown in Figure 3.7. Finally, we integrate the normal map, where the boundary of the surface is extracted from the recovered height field. The bottom row of Figure 3.7(g) shows the recovered fluid surface at a specific frame.

To illustrate the robustness of our approach over time, we show the reconstruction results at different time frames in Figure 3.8. Compared with the ground truth, our method faithfully captures shape deformations of the fluid surface over time. The complete recovered sequences can be found in the supplementary materials. We have further computed the actual reconstruction errors. Specifically, we sample the surface at a 100x100 resolution and compute the error in the height at each grid. The average error is $1.47 \times 10^{-4}$ for the Gaussian wave sequence and less than $2.63 \times 10^{-4}$ for the other sequences.

**Real Surfaces.** To capture real fluid surfaces, we set up the camera array system as shown in Figure 3.2(Left). We align the cameras so that they lie approximately on the same plane parallel to the checkerboard plane. We first calibrate the cameras in the array using Zhang’s algorithm [65] with a duplicated checkerboard. We then take an image of the undisturbed water surface and detect feature correspondences for the 0th frame. Notice that the translation of the pattern due to refractions provide an initial height of the surface which we will use in our specular carving algorithm.
We separate 9 cameras into two groups, as shown in Figure 3.2(Right). The capture time is interleaved by 16.67ms. To generate fluid motions without disturbing the camera setup, we use a hair dryer to blow air onto the surface. We start with reconstructing the first frame using cameras in group A. Similar to the synthetic case, we detect feature correspondences and apply specular carving to recover the normal field and then the height field. Figure 3.9 shows the tracked features from two views and reconstructed surface. Given the recovered surface, we apply multi-perspective ray tracing to locate the feature points in cameras in group B. We then reapply the tracking algorithm on images captured by group B, with the warped feature points as their previous frames. Using the correspondence results, we recover the surface at the second frame. We repeat this process by iteratively warping the reconstructed surface between group A and B to improve the robustness in tracking. When parts of a camera’s frame lose track due to distortions or blurs, we simply discard the corresponding portion and use the rest of the cameras for reconstruction. We then apply multi-perspective ray tracing to locate the actual feature points in the camera so that we can continue to use the camera in its later frames. Figure 10 shows four acquired frames in the central camera in the array and their corresponding reconstruction results. Notice that several patches on the surface exhibit severe distortions or blurs, our algorithm is still able to reconstruct reasonable surfaces and continue tracking the missing feature points using the warped results.
Figure 3.9: Results on real data. Top row shows the feature tracking results on two views. Bottom row shows the recovered height field.
Figure 3.10: Results on real data. From top to bottom: the captured frames from the central camera, the recovered normal field, and the recovered height field. The complete sequence can be found in the supplementary video.
Chapter 4

ANGULAR DOMAIN RECONSTRUCTION OF DYNAMIC 3D FLUID SURFACE

In this Chapter, I present a novel and simple computational imaging solution to robustly and accurately recover 3D dynamic fluid surfaces. Traditional specular surface reconstruction schemes place special patterns (checkerboard or color patterns) beneath the fluid surface to establish point-pixel correspondences. However, point-pixel correspondences alone are insufficient to recover surface normal.

To resolve the ambiguity, I exploit using Bokode - a computational optical device that emulates a pinhole projector - for capturing ray-ray correspondences which can then be used to directly recover the surface normals.

I further develop a robust feature matching algorithm based on the Active-Appearance Model to robustly establishing ray-ray correspondences. Experiments results demonstrate that our approach is robust and accurate, and is easier to implement than state-of-the-art multi-camera based approaches.

4.1 Bokode-based Acquisition System

Fig. 4.1 shows the algorithm flow my proposed Bokode-based fluid surface reconstruction framework. I use Bokode projecting out pattern towards fluid surface and capture the distorted pattern. By associating the distorted pattern with the projected one using AAM matching, I then obtain the incident-exit ray correspondences for computing surface normals. Since the normals are sampled in angular-domain, we therefore reconstruct the surface with our new spherical coordinate based surface integration algorithm.
Figure 4.1: A block diagram that shows the pipeline of our Bokode based fluid surface reconstruction framework.

Bokode is an optical device that resembles a pinhole projector [66]. In essence, a Bokode emits lights originating from the common 3D point over different angles as shown in Fig.4.2(a). When capturing a Bokode using a camera with a small aperture, the Bokode would appear as a single dot as the camera only captures a specific angle of rays. In contrast, if a Bokode is captured by a camera with a large aperture, the pattern emitted the Bokode can be partially captured as shown in Fig.4.2(b). Using this unique feature, Mohan et al.[66] proposed to use Bokode as an invisible identification tag. In the similar vein, we explore using Bokode as a special active illumination device for fluid surface reconstruction.

To physically implement a Bokode, the simplest approach is to construct a pinhole type of device that only allows the lights to pass through the hole. However, similar to pinhole cameras constructed as such, this design suffers from blurry images and insufficient lights. In reality, a Bokode can be approximated using a small lens projector with the projection pattern positioned at the focal length. The viewing camera is then position relatively faraway from the Bokode and focuses at infinity to effectively sample the angular rays emitted from the Bokode as shown in Fig.4.2(a).

It is important to note that a commodity projector cannot be directly used as a Bokode. Although both Bokode and commodity projectors use back light to illuminate the projection pattern, the Bokode requires the pattern be placed at the depth of the focal length whereas the projector places the pattern much farther away from the lens for magnifying the pattern. Further, the Bokode uses a much smaller aperture to effectively emulate a pinhole system while the aperture of a commodity projector is set ultra large to ensure the brightness of projection. In our setup, we construct a
lens-based Bokode that contains four layers: lens, pattern, diffuser and light source. We use the lens of a commodity web camera as the Bokode lens. The lens has a diameter of 2mm and a focal length of 8mm. We print a special monochrome pattern on a transparency at a resolution of 5080 dpi. We further use the diffuser to ensure that lights emits towards all directions. Finally, to increase the brightness, we use an ultra-bright LED flash light of 400 lumen as the back light to the Bokode.

Fig.5.8 shows our Bokode-based fluid surface acquisition system. We place the Bokode underneath a water tank to project lights towards the fluid surface. Since the bottom of water tank is flat and thin, we ignore the refraction effects and view the Bokode as if it sits directly at the bottom of the tank. We assume each feature on the projection pattern corresponds to a thin beam of parallel light rays and the top (wavefront) fluid surface interacting with each light beam is nearly flat. Therefore, their corresponding exit rays remain approximately parallel.

Let $P_i(x_i, y_i)$ be a point on the projection pattern on the Bokode, where $(x_i, y_i)$ are the relative coordinates of $P_i$ to the lens’ optical center, and $f_b$ be the focal length of the Bokode lens. $P_i$ then maps to a light beam with direction $\alpha_i = \ldots$
Figure 4.3: Each point on the Bokode pattern maps to a beam of parallel rays. These rays are refracted by fluid surface and gathered by the viewing camera at a pixel.

\[
\arctan\sqrt{x_i^2 + y_i^2} / f_b. \quad \text{We call the ray direction from the Bokode towards the fluid surface the incident ray direction.}
\]

On the camera side, we use a calibrated camera focusing at infinity to capture the light rays after being refracted by the fluid surface. For each pixel \( P'_i(x'_i, y'_i) \) on the captured image, we can use the camera parameters to obtain its corresponding ray direction as \( \beta_i = \arctan\sqrt{x'_i^2 + y'_i^2} / f_c \) where \( f_c \) is the focal length of the camera lens. We call \( \beta_i \) the exit ray direction. Notice that once we obtain \( P_i \) and \( P'_i \) correspondences, we can simply intersect the rays to obtain the fluid surface position and normal, although in reality only the directions of the rays are useful. In this paper, we call \( P_i \) and \( P'_i \) correspondences the Incident-Exit-Ray (IER) correspondences.

To calibrate our system, we first calibrate the camera and align its optical axis with the Bokode’s main axis. We then capture a Bokode image without adding any fluid to the tank. In this case, the exit ray directions captured by the camera are identical to the incident ray directions. In our implementation, the Bokode projects a special pattern with many feature points and the calibration process associates the feature points with the incident ray directions from the Bokode. Once the tank is filled...
with fluid, we then find the matching feature points (see Sec.5.2) which would directly provide IER correspondences.

4.2 Correspondence Matching

The choice of the projection pattern is important for reliable correspondence matching and hence surface reconstruction. Most previous approaches use a checkerboard pattern and track the feature points (corners). For our Bokode-based solution, checkerboard pattern is less suitable. This is because existing two-view or multi-view based approaches assume that the cameras can view the complete checkerboard, which greatly helps tracking the pattern over time and across views. In our case, the Bokode has a much wider field of view with $130^\circ$ compared to the viewing camera with merely $20^\circ$. Therefore, the camera can only view part of the pattern and since checkerboard patterns are highly symmetric, tracking the features consistently is challenging. Another option is to use the color-based pattern. For example, Wetzstein et al.\cite{5} red, blue, and green gradient to encode the 2D directions and 1D vertical positions. The color-based pattern is suitable for static surface but can cast challenge to dynamic fluid surfaces due to chromatic abberations and caustics. Chromatic abberations destroys color calibration results and caustics changes the intensity, making it difficult to match colors. Further, the color patterns are usually of a much lower resolution. The common resolution of color printer is 1440 dpi while our Bokode pattern is printed at 5080 dpi.

We choose to use an irregular monochrome patterns as shown in Fig.4.2(b) and apply Active Appearance Model for correspondence matching. Our pattern consists of an array of tiled asymmetric symbols. Each symbol has the size of 80$\mu$m and the entire pattern is of dimension $1cm \times 1cm$. The symbol has sharp corners and edges to provide effective feature points. Specifically, we use the corners as the major feature points and interpolate along the edges between every pair of major features to generate secondary feature points. Further, we place a checkerboard square marker at the center of the pattern calibrating its position and add a dot to the top-corner of the square to identify its orientation.
To track the feature points, we apply the Active Appearance Model (AAM) [67, 68] that was originally developed for pattern recognition. An AAM is composed of two components: a shape model and an appearance model. The shape model is described using as a set of \( N \) feature points \((x_1, y_1; x_2, y_2; ..., x_N, y_N)\) and is represented as a mean shape \( s_0 \) with a linear combination \( p_i \) of variations on \( n \) shape basis \( \{s_i\} \):

\[
s(p) = s_0 + \sum_{i=1}^{n} p_i s_i \quad (4.1)
\]

The appearance model is defined as the intensity of image patches surrounding the mean shape. Similar to the shape model, we model the appearance model with a mean appearance \( A_0 \) plus a linear combination \( \lambda_i \) of variations on \( m \) appearance basis \( \{A_i\} \):

\[
A(p) = A_0 + \sum_{i=1}^{m} \lambda_i A_i \quad (4.2)
\]

Same as classical AAM-based recognition techniques [67, 68], we obtain the mean shape \( s_0 \), the mean appearance \( A_0 \), the shape variations \( \{p_i\} \) and the appearance variations \( \{\lambda_i\} \) by applying the Principal Component Analysis (PCA) to our training data. Fig.4.4(a) shows the mean shape and some shape variations; Fig.4.4(b) shows the mean appearance and two appearance bases. To match the shape to a specific image, the AAM technique finds the optimal shape parameters and appearance parameters that minimize the appearance variations:

\[
E_a(x) = \sum_{x \in s_0} [A_0(x) + \sum_{i=1}^{m} \lambda_i A_i(x) - I(W(x; p))]^2 \quad (4.3)
\]

where \( W(x; p) \) is the affine warping defined by a shape model \( s(p) \) and the mean shape \( s_0 \) that maps every pixel from the model coordinate to the corresponding image coordinate.

We generate our training data by rendering a large set of distorted patterns using varying fluid surface normals, height, and orientations. To match a new distorted pattern, we first segment the captured image to small patches, each containing a single symbol. We then perform AAM search on each symbol to robustly handle non-uniform intensity caused by the caustics. In the first frame of the video sequence, we initialize
Figure 4.4: (a) Shape model: the shape in blue is the mean shape $s_0$ and the ones in red are shape variations estimated by PCA; (b) Appearance model: the left shows the mean appearance $A_0$ and the first two eigen appearance; (c) An example of AAM matched pattern.

The match by aligning the model and the capture symbols at their centroid. For the consecutive frames, the matched shape from the previous frame is used to initialize the matching process. Since our symbols are have high contrast to the background, we further generate a distance map $M_d$ based on contour of the shape as additional cost constraint to guide the AAM search:

$$E_d(x) = \sum_{x \in s_0} (M_d(W(x;p)))^2$$  \hspace{1cm} (4.4)

Therefore, the total cost function for shape matching hence becomes $E = E_a + wE_d$, where $w$ is the weighting factor to the distance map constraint. Fig.4.4(c) shows an AAM matched result of captured image.

Once we match the captured symbol with the projected one, we instantly obtain IER correspondences. Recall that the incident ray directions $d_{in}$ are encoded in the projected Bokode image and are precomputed in the calibration step and the exit ray directions $d_{exit}$ can be calculated with the viewing camera parameters. We assume that the refraction indices of air and the fluid $n_1$ and $n_2$ are known in prior respectively and
we can solve for the surface normal using the Snell’s law as:

\[ \mathbf{n} = n_2 \mathbf{d}_{in} - n_1 \mathbf{d}_{exit} \quad (4.5) \]

Since we only sample a relatively sparse set of feature points on the pattern, we obtain a sparsely and irregularly sampled normal field. We then apply the Radial-Basis Function (RBF) function to interpolate the normal field.

### 4.3 Surface Reconstruction from Angularly Sampled Normals

Next, we show how to reconstruct the fluid surface from the angularly sampled normal field recovered from Bokode discussed in Sec. 5.2. Recall that the key advantage of our approach is that it directly recovers the normal direction from the IER correspondences. The resulting normal field, however, is very different from the classical height-field based one. To elaborate, if we model the surface as a height field \(z(x, y)\), the scaled normal vector at each point \((x, y)\) is simply \((z_x, z_y, -1)\); when given the boundary condition (Neumann or Dirichlet) and the height-field based normal field, the problem of integrating the normal field to recover the same can be formulated to find the optimal surface \(f\) where:

\[
\min_f \iint ((f_x - z_x)^2 + (f_y - z_y)^2) \, dx \, dy \quad (4.6)
\]

Previous approaches [69, 70] have shown that solving this optimization problem is equivalent Poisson equation: \(\Delta f = z_{xx} + z_{yy}\), where \(\Delta\) is the Laplacian operator: \(\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\). In the discrete case, one can linearize the Laplacian and the derivative operator and form an linear system in \(f\) and directly solve for \(f\) as shown in the supplementary material.

Our angular normal sampling scheme, in contrast, measures normals at discrete inclination and azimuth angles. These angular domain normals cannot be directly mapped to spatial domain normals, i.e., we cannot perform ray-surface intersection as the surface is unknown. Further, by using a wide aperture viewing camera, we can only recover the exit ray direction rather than the ray itself, i.e., we cannot perform ray-ray intersections [71]. We therefore derive a new angular-domain surface integration
scheme. We parameterize the surface in spherical coordinates as $r(\theta, \phi)$ where the origin coincides with the Bokode's pinhole, $r$ is the radial distance from the origin, and $\theta$ and $\phi$ correspond to the azimuthal and polar angles respectively. Our sampling essential recovers the normals at discrete samples of $\theta$ and $\phi$ and our goal is to recover the radius $r$ from the sampled normal field.

At each surface point $r(\theta, \phi)$, we can compute its gradients under spherical coordinate as $(\partial r/\partial \theta, \partial r/\partial \phi)$. In reality, we only have sampled normal directions measured in the Cartesian coordinate. We therefore need to further convert the gradients in Cartesian coordinate to spherical coordinate.

Recall that $x = r \sin \phi \cos \theta$, $y = r \sin \phi \sin \theta$, and $z = r \cos \phi$, we have:

$$\frac{\partial r}{\partial \theta} = r \cdot \frac{\sin \phi (\frac{\partial z}{\partial x} \sin \theta - \frac{\partial z}{\partial y} \cos \theta) + \sin \phi (\frac{\partial z}{\partial x} \cos \theta + \frac{\partial z}{\partial y} \sin \theta) - \cos \phi}{\sin \phi (\frac{\partial z}{\partial x} \sin \theta - \frac{\partial z}{\partial y} \cos \theta).}$$  
$$\frac{\partial r}{\partial \phi} = r \cdot \frac{\sin + \cos \phi (\frac{\partial z}{\partial x} \cos \theta + \frac{\partial z}{\partial y} \sin \theta) - \sin \phi (\frac{\partial z}{\partial x} \cos \theta + \frac{\partial z}{\partial y} \sin \theta)}{\sin \phi (\frac{\partial z}{\partial x} \sin \theta - \frac{\partial z}{\partial y} \cos \theta).}$$

(4.7)

Eq.(4.7) illustrates the constraint between radius gradient to surface normal. Our goal is to find the optimal surface $r$ (in spherical coordinate) under the constraints.

We first discretize the surface $r$ in discrete $(\theta_i, \phi_j)$. We can then approximate Eq.(4.7) using finite difference as:

$$\begin{cases} 
  r_{i+1,j} - r_{ij} = r_{ij} \cdot P_{ij} \\
  r_{i,j+1} - r_{ij} = r_{ij} \cdot Q_{ij}
\end{cases}$$

(4.8)

where

$$\begin{cases} 
  P_{ij} = \Delta \theta \cdot \frac{\sin \phi_j (\frac{\partial z}{\partial x}(i, j) \sin \theta_i - \frac{\partial z}{\partial y}(i, j) \cos \theta_i) + \sin \phi_j (\frac{\partial z}{\partial x}(i, j) \cos \theta_i + \frac{\partial z}{\partial y}(i, j) \sin \theta_i) - \cos \phi_j}{\sin \phi_j (\frac{\partial z}{\partial x}(i, j) \sin \theta_i - \frac{\partial z}{\partial y}(i, j) \cos \theta_i) + \sin \phi_j (\frac{\partial z}{\partial x}(i, j) \cos \theta_i + \frac{\partial z}{\partial y}(i, j) \sin \theta_i) - \cos \phi_j} \\
  Q_{ij} = \Delta \phi \cdot \frac{\sin \phi_j + \cos \phi_j (\frac{\partial z}{\partial x}(i, j) \cos \theta_i + \frac{\partial z}{\partial y}(i, j) \sin \theta_i) - \sin \phi_j (\frac{\partial z}{\partial x}(i, j) \cos \theta_i + \frac{\partial z}{\partial y}(i, j) \sin \theta_i)}{\sin \phi_j + \cos \phi_j (\frac{\partial z}{\partial x}(i, j) \cos \theta_i + \frac{\partial z}{\partial y}(i, j) \sin \theta_i) - \sin \phi_j (\frac{\partial z}{\partial x}(i, j) \cos \theta_i + \frac{\partial z}{\partial y}(i, j) \sin \theta_i)}
\end{cases}$$
In \( \{P_{ij}\} \) and \( \{Q_{ij}\} \), \( \frac{\partial z}{\partial x}(i,j) \) and \( \frac{\partial z}{\partial y}(i,j) \) are observed normals in Cartesian coordinate corresponding to \((\theta_i, \phi_j)\). Assume we have discretized \( \theta \) and \( \phi \) into a \( m \times n \) grid, we then form an over-constrained linear system from Eq.(4.8): we have two equations for each radius \( r_{ij} \) except for the boundary. For the \( \theta \) boundaries, we have \( r(0, \phi_j) = r(2\pi, \phi_j) \), i.e., \( r_{0j} = r_{mj} \). However, for the \( \phi \) boundaries, it is not easy to acquire radius at those points. We solve this problem by using the current frame’s reconstruction result to predict the boundary of later frames. In all we have \( mn \) unknowns and \( 2mn \) linear equations. By stacking these together, we obtain a linear system with equations \( A\Theta = 0 \), where \( A \) is the coefficient matrix formed by \( \{P_{ij}, Q_{ij}, i = 1, ..., m; j = 1, ..., n\} \) and \( \Theta = \{r_{ij}, i = 1, ..., m; j = 1, ..., n\} \). Then we apply Singular Value Decomposition (SVD) on our linear system to obtain the least square solution of surface radiuses.

In the supplementary material, we prove that this is a valid approach as traditional spatial-domain surface completion can also be formulated and solved using a similar over-constrained linear system.

### 4.3.1 Normal Integration in Polar Coordinate

Recall that the key advantage of our Bokode-based acquisition technique is that it directly recovers the surface normal without requiring knowing the surface height. The downside, however, is that we obtain a different sampling of the surface: it is sampled by a bunch of rays originated from the center of the bokode lenslet. This fact indicates that the projection from the bokode pattern to the surface is not orthographical. Although the recent light field method used similar sampling scheme, they also extracted surface point positions together with normals from their color-coded light field probe image. Hence combining with the point position information, they can estimate the surface by integrating the normal field. However, we only have the normal field. Thus the surface is reconstructed over grid on the bokode pattern plane. That’s why in our framework, the surface reconstruction problem cannot be solved by the traditional normal field integration approach.
To recover the surface from the normal field, we present a novel polar-coordinate based surface integration scheme. Traditional surface integration methods assume that the surface is a height field $Z(x, y)$, where the un-normalized surface normal can be computed as $n = n_1 d_{in} - n_2 d_{out}$, where $d_{in}$ and $d_{out}$ are the unit vectors in incident and exiting ray directions and $n_1, n_2$ are the corresponding refractive indices. To recover the surface from the normal field, one can use Poisson surface integration. Let $\{p = -n_x/n_z, q = -n_y/n_z\}$ denotes the estimated gradient field from image. The surface height field $Z$ can be obtained by solving the following minimization problem:

$$\min_Z \int \int ((Z_x - p)^2 + (Z_y - q)^2) dxdy$$

(4.9)

It yields to the Poisson equation: $\Delta Z = div(p, q)$ given the Euler-Lagrange equation, where $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ is the Laplacian operator and $div(p, q) = \frac{\partial p}{\partial x} + \frac{\partial q}{\partial y}$ is the divergence operator. Furthermore, the Poisson equation can be discretized and yield to the linear system $\tilde{A}X = \tilde{b}$, where $\tilde{A}$ is the 2D Laplacian matrix and $\tilde{b}$ is the divergence of gradient field. Since $\tilde{A}$ is a square matrix, the surface $X$ can then be obtained by $X = \tilde{A}^{-1}\tilde{b}$.
Next we solve the surface reconstruction problem in polar coordinate in a similar approach as solving poisson equation in Euclidian coordinate. Let’s first consider a point \( x_{i,j} \) on the image plane, and its four neighbors, \( x_{i-1,j}, x_{i,j-1}, x_{i+1,j}, x_{i,j+1} \), the gradients are: \( p_{x}(i, j) = x_{i+1,j} - x_{i,j} \), \( p_{y}(i, j) = p_{y}(i, j+1) - p_{y}(i, j) \) if we only consider the first order derivative, we have:

\[
A = \begin{bmatrix}
... & c_{i-1,j} & ... & c_{i,j} & ... & c_{i,j+1} & ... & c_{i+1,j} & ...
... & -1 & ... & 0 & 1 & ... & 0 & ... & ...
... & 0 & ... & -1 & 1 & ... & 0 & ... & ...
... & 0 & ... & 0 & -1 & 1 & ... & 0 & ...
... & 0 & ... & 0 & -1 & 0 & ... & 1 & ...
... & ...
\end{bmatrix}
\]

\[
X = \begin{bmatrix}
x_{i-1,j} \\
... \\
x_{i,j} \\
... \\
x_{i,j+1} \\
... \\
x_{i+1,j} \\
... \\
\end{bmatrix}
\]

\[
b = \begin{bmatrix}
p_{x}(i-1, j) \\
p_{y}(i, j-1) \\
p_{x}(i, j) \\
p_{y}(i, j) \\
\end{bmatrix}
\]

However, \( A \) is not a square matrix. We need to apply svd to solve it, that is, \( A^{T}AX = A^{T}b \). In order to prove that the linear system obtained by our svd method is the same as the poisson solution, we need to show that: \( A^{T}A = \tilde{A} \) and \( A^{T}b = \tilde{b} \). The new equation for \( x_{i,j} \) is obtained by multiplying \( r_{i,j} \) in \( A^{T} \) with every columns of \( A \) and with \( b \) at the other side of the equation.
\[
\begin{bmatrix}
\vdots & \vdots & \vdots & \vdots & \vdots \\
r_{i-1,j} & -1 & 0 & 0 & 0 \\
r_{i,j-1} & \vdots & \vdots & \vdots & \vdots \\
r_{i,j} & 0 & -1 & 0 & 0 \\
r_{i,j+1} & 0 & 0 & 1 & 0 \\
r_{i+1,j} & 0 & 0 & 0 & 1 \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
\end{bmatrix}
\]

\[
A^T = \begin{bmatrix}
\vdots & \vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
A^Tb = \begin{bmatrix}
\vdots \\
r_{i,j} \begin{pmatrix}
(p_x(i-1,j) - p_x(i,j)) \\
\vdots \\
(p_y(i, j-1) - p_y(i, j)) \\
\vdots \\
\end{pmatrix}
\end{bmatrix}
\end{bmatrix}
\]

Hence we have:
\[
A^T A = \begin{bmatrix}
\vdots & \vdots & \vdots & \vdots & \vdots \\
r_{i,j} \begin{pmatrix}
1 & \ldots & 1 & -4 & 1 & \ldots & 1 & \ldots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\end{pmatrix}
\end{bmatrix}
\]

which is the exactly same linear system obtained by poisson method.

We, in contrast, parameterize the surface under the polar-coordinate as \( r(\theta, \phi) \). The bokode lenslet center is set as the origin. \( \theta \) is the angle measured form the x-axis in the xy-plane and \( \phi \) is the angle measured from the z-axis. Radius \( r \) is the distance measured from an end point to the origin. At every point \((\theta, \phi)\), we convert the acquired normal field from Cartesian coordinate to gradients \((\tilde{A}, \tilde{B})\) in polar coordinates. We define the gradients in polar coordinate as \( \frac{\partial r}{\partial \theta} \) and \( \frac{\partial r}{\partial \phi} \). Our goal is to recover the surface
Figure 4.5: Results on a synthetic sinusoid wave (top rows) and on a Helmholtz wave (bottom two rows). We show the cropped Bokode pattern used in reconstruction. In particular, column 2 are sampled surface normals under spherical coordinate in respect to $\theta$ and $\phi$ and we further compute normal maps in Cartesian coordinate and compare with the ground truth ones to demonstrate the accuracy of our method as shown in column 6 and 7. The complete sequences can be found in the supplementary video.

most consistent with the sampled gradient field, that is to minimize the following energy function:

$$ \min_r \int \int ((\frac{\partial r}{\partial \theta} - \tilde{A})^2 + (\frac{\partial r}{\partial \phi} - \tilde{B})^2) d\theta d\phi $$

(4.10)

Unlike Poisson surface reconstruction, finding the optimal surface is highly non-linear. We therefore present an approximation scheme based on a similar SVD linearization.

4.4 Experiments

We have validated our approach on both synthetic and real fluid surfaces. For synthetic surfaces, we have implemented a Ray-tracer that back-traces feature points from the Bokode pattern to pixels in the viewing camera. For real surfaces, we capture video streams of dynamic fluid surfaces using our acquisition system (see Sec.4.1).
4.4.1 Synthetic Scene Simulation

We first conduct experiments on a synthetic sinusoidal wave: \( z(x,y,t) = 20 + \cos(\pi t \sqrt{(x - w/2)^2 + (y - h/2)^2/200}) \), where \( w = h = 300 \). On the Bokode side, we use a pattern of physical size 300 \( \times \) 300 with 48 symbols on it. In our setup, the viewing camera captures 32 symbols. We also use the Helmholtz Equation to propagate the of the same wavefront at \( t = 0 \), to synthesize more realistic fluid effects and test the robustness of our algorithm. In the Helmholtz wave case, we use a higher resolution pattern of 600 \( \times \) 600 with 220 symbols to improve the angular resolution. The refraction index of the fluid is set to be 1.33 to emulate water.

To generate the AAM training data, we render 100 distorted pattern image on randomly sampled normals. Since we use ray-tracing, we obtain ground-truth feature points at both the corners and along edges. In our experiment, we use 8 corner points and 10 edge points in between each pair of neighboring corners on the symbol. We apply AAM matching to the synthesize fluid images using the training data and obtain the ray-ray correspondences. We then compute the surface normal at each feature point and interpolate a dense normal field. For the sinusoidal wave, the angularly resolution of our sampled normal map is 0.33° and for the Helmholtz wave, we generate a higher angular resolution of 0.2° to recover fine details. Finally, we apply our spherical coordinate surface integration scheme. In both cases, we use the ground truth wave boundary for integration.

Fig. 4.5 shows our recovered wavefronts at different time instances. The video sequence of the results can be found in supplementary material. We further compute the reconstruction error to illustrate the accuracy of our method. The amplitude of sinusoidal wave is in range of [19.9, 20.1] and our average reconstruction error is 9.782 \( \times \) 10^{-4}. Helmholtz wave is in a similar range and our reconstruction error is 7.773 \( \times \) 10^{-4}. This implies that using a denser pattern would improve accuracy. In reality, however, producing a dense pattern for the Bokode is challenging due to the resolution limit on commodity printer.
Figure 4.6: Our experimental setup. We construct a Bokode using a flashlight, a diffuser, a high-resolution pattern and webcam lens. We also use an auxiliary bi-convex spherical lens to collect lights refracted by the fluid surface (top right).
4.4.2 Real Scene Experiments

To capture real fluid surfaces, we set up our system for capturing real fluid surface as shown in Fig.5.2. We construct a Bokode that consists of a bright flash light of 400 lumen, a micro-pattern and a lenslet with 2mm aperture and 8mm focal length dissembled from a cheap conventional web camera. We print a monochrome Bokode pattern at a resolution of 5080 dpi on a 1cm × 1cm transparency using the professional printing service provided by PageWorks (http://www.pageworks.com). On the viewing side, we couple a high resolution DSLR camera (Canon 60D, lens 85/1.8) with an auxiliary bi-convex spherical lens of 100mm with focal length 170mm to capture a wider angular range of rays. We adjust the viewing camera to focus at the focal plane of the auxiliary lens to record an HD video at a resolution of 960 × 720 at 30 fps. Our water tank is of size 24cm × 18cm × 36cm and the viewing camera under our lens and aperture setting can observe an area of around 600mm × 600mm. We pre-calibrate the viewing camera using Zhang’s algorithm[61] and then capture the image of the Bokode pattern with water for obtaining the incident ray direction with respect to each feature point on the pattern.

We experiment our method on two types of wavefront. The first one is created by randomly perturbing the fluid at one end of the water tank to propagate the wave towards the other end; The second is a “ring-type” wave that is created by blowing air into towards the fluid. One major challenge that we observe in the real fluid surface case but not in the synthetic one is the effects of caustics which changes the intensity of the observed patterns. To reuse the training data, we use only the binary gradient map of the rendered images as the appearance model and then apply AAM matching. In some cases, the acquired image can exhibit motion blurs and we need to apply manual alignments. To integrate the surface, we assume that the fluid boundary is flat in the first frame and then apply the Navier Stokes (NS) model to propagate the boundary [72]. Fig.5.9 shows our acquired raw data, the AAM tracked results, and our reconstructions a number of frames of real fluid surfaces. In the “ring-type” wavefronts, several acquired patches exhibit strong distortions. Our technique is able to reasonably
Figure 4.7: Results on two sets of real data (a perturbed wave and a ring wave). From left to right: we show the captured image with matched features, the sampled normal under spherical coordinate, the reconstructed surface, and the surface normal field computed from the reconstructed surface.
align the distorted pattern using AAM and our reconstruction results are consistent with the observed distortions. In fact, the quality of our reconstruction can be further improved by using more training samples in AAM. We refer the reviewers to the supplementary videos for the completely reconstructed sequences.
Chapter 5
RECONSTRUCTING GAS FLOWS USING LIGHT-PATH APPROXIMATION

Transparent gas flows are difficult to reconstruct: the refractive index field (RIF) within the gas volume is uneven and rapidly evolving, and correspondence matching under distortions is challenging.

In this Chapter, I present a novel computational imaging solution by exploiting the light field probe (LF-Probe). A LF-probe resembles a view-dependent pattern where each pixel on the pattern maps to a unique ray. By observing the LF-probe through the gas flow, I acquire a dense set of ray-ray correspondences and then reconstruct their light paths.

To recover the RIF, I use Fermat’s Principle to correlate each light path with the RIF via a Partial Differential Equation (PDE). I then develop an iterative optimization scheme to solve for all light path PDEs as a group. Specifically, I initialize the light paths by fitting Hermite splines to ray-ray correspondences, discretize their PDEs onto voxels, and solve a large, over-determined PDE system for the RIF. The RIF can then be used to refine the light paths. Finally, I alternate the RIF and light path estimations to improve the reconstruction. Comprehensive experiments show that my approach can accurately and robustly reconstruct small to medium scale gas flows. In particular, the use of ray-ray correspondences greatly improves reconstruction quality.

5.1 Acquisition System

5.1.1 Light Field Probe (LF-Probe)

The core of our approach is to acquire ray-ray correspondences using the LF-probe [73, 5]. A LF-probe, as shown in Fig. 5.1, can be viewed as an “inverted” light
field camera Lytro (www.lytro.com). In Lytro, a microlenslet array is placed in front of the camera sensor to acquire the 4D light field, where the sensor-lens distance is identical to the microlens’ focal length. Each microlens serves as a virtual pinhole camera and the lenslet array serves as a camera array. The LF-probe keeps the same design except replacing the sensor with a specially designed color pattern. Uniform backlight is then used to illuminate the pattern and each pixel on the pattern maps to a unique ray.

To obtain a dense set of correspondences, similar to [73, 5], we use color-coded pattern to encrypt the 4D ray positions and directions emitted by the probe. In particular, we use a combination of horizontal red gradient and vertical blue gradient behind each microlens to discriminate rays of different directions. The red/blue gradients are identical for each microlens unit. To determine 2D positions, we use the variation of green channel. We first discretize the position to blocks based on micro lenses by assuming the block behind each microlens has the same position as the center of the lens. We then randomly choose the green intensity for each block to form a random noise pattern as a whole. To find out the position shifted due to refraction, we perform
optical flow between the refracted pattern and the original one on the green channel to estimate the deflection vectors. In this way, by matching color in the captured image, we can directly acquire the positions and directions of rays emitting from the LF-probe.

5.1.2 System Setup

Fig. 5.2 shows our gas flow acquisition system. We place 3 LF-probes to surround the target gas flow and 3 synchronized cameras to capture the corresponding LF-probe through the gas volume. Our goal is to first acquire a dense set of correspondences between rays entering the gas volume and the ones exiting the volume and then use these incident-exit ray pairs for estimating the light paths and the RIF.

Calibration. In order to reliably correlate the incident-exit ray pairs, we conduct two calibration procedures, one between each LF-probe and its viewing camera and the second between cameras. As discussed in Sec. 5.1.1, we determine ray-ray correspondences via color matching. Therefore we first calibrate colors between the printed LF-probe pattern and the observed image. To do so, we mount the LF-probe on a rotation table and then capture images of the probe at different viewing angles, as shown in Fig. 5.3(a). An additional checkerboard pattern is placed next to the probe for measuring the orientation of the probe w.r.t. the camera. Once we determine the direction $\beta$ of a ray collected by the camera and the angle $\alpha$ between the probe’s normal and the camera’s principal axis, we can then compute the ray’s direction as $\gamma = \alpha + \beta$, as shown in Fig. 5.3(b). For each lenslet within the probe, we obtain a curve between each RGB channel and the directions (decomposed into vertical and horizontal components). A sample curve from a specific lenslet is shown in Fig. 5.3(c). By matching colors of red/blue channels for each lenslet, we can map each observed pixel to an incident ray direction. To determine the incident ray’s origin, we use a random noise pattern on the green channel so that we can associate a patch of pixels with a location. We obtain the ground truth by acquiring the LF-probe without any gas flows. When capturing the gas flows, we then use the optical flow for tracking the pattern.
Figure 5.2: An illustration of our gas flow acquisition system.
Figure 5.3: We use a rotation table (a) to calibrate ray directions (b) by matching observed colors to the color calibration curves (c).
Figure 5.4: We use ray-ray correspondences to first recover the light paths and then the refractive index field (RIF) within the gas volume.

Calibrating the exit rays is equivalent to calibrating the camera intrinsics and we directly apply Zhang’s algorithm [65]. Calibration between cameras is more challenging. Notice each viewing camera has a narrow Field-of-View (FoV) in order to capture a high resolution image of the LF-Probe. Therefore, the view frustums of the three cameras barely overlap. To resolve this issue, we place three additional cameras between the viewing cameras and conduct pair-wise camera calibrations. Once we finish the calibration process, each camera is able to acquire a dense set of ray-ray correspondences w.r.t. the LF-probe. For the rest of paper, we use \((P^{\text{in}}, d^{\text{in}})\) to represent rays emitting from the LF-Probes (the incident rays) and \((P^{\text{out}}, d^{\text{out}})\) for the rays entering the camera (the exit rays) as shown in Fig. 5.4, where \(P\) and \(d\) are the origin and direction of a ray respectively.

5.2 Volumetric Gas Reconstruction

Given a dense set of ray-ray correspondences across the gas volume, our goal is to recover the RIF that best matches these correspondences. We first derive how RIF is correlated with the light path using Fermat’s principle: the light always travels along the path with the shortest Optical Path length (OPL) [74]. Assuming an arbitrary path
c, the OPL $S$ of $c$ is computed as the weighted path length w.r.t. the refractive index $n$ at every point $p(x, y, z)$ (or voxel in the discrete case) on the path $c$:

$$S = \int_c n(p) ds$$  \hspace{1cm} (5.1)

We can further parameterize $p(x, y, z)$ as function of the time $t$ that light reaches $(x, y, z)$ as $p(x(t), y(t), z(t))$, then we have:

$$S = \int_c L dt, \; L = n(p) \sqrt{x_t^2 + y_t^2 + z_t^2}$$ \hspace{1cm} (5.2)

where $x_t = \frac{\partial x}{\partial t}$, $y_t = \frac{\partial y}{\partial t}$, $z_t = \frac{\partial z}{\partial t}$ and $n(p)$ is the refractive index at $p$. $L$ is often referred to as the optical Lagrangian [75].

Our goal is to solve for both the light path $c$ and the RIF $n$. Base on the Fermat’s Principle, each light path $c$ is with the shortest OPL, i.e., $\frac{ds}{dt} = 0$, substituting $S$ with Eqn. 5.2, we have:

$$\frac{d}{dt} \int_c L dt = 0$$  \hspace{1cm} (5.3)

**5.2.1 RIF Estimation**

If we have the light paths, we can then estimate the RIF. By Eqn. 5.3, $L$ should satisfy the Euler-Lagrange equation:

$$\left( \begin{array}{ccc} \partial L & \partial L & \partial L \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \end{array} \right) = \frac{d}{dt} \left( \begin{array}{ccc} \partial L & \partial L & \partial L \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \end{array} \right)$$ \hspace{1cm} (5.4)

Let us only consider the $x$ component. By substituting $L$ of Eqn. 5.2 into Eqn. 5.4, we have:

$$\frac{\partial n(p) \sqrt{x_t^2 + y_t^2 + z_t^2}}{\partial x} = \frac{d}{dt} \left( \frac{\partial n(p) \sqrt{x_t^2 + y_t^2 + z_t^2}}{\partial x_t} \right)$$ \hspace{1cm} (5.5)

Changing the order of derivatives, we can rewrite the Eqn. 5.5 as:

$$\frac{\partial n(p)}{\partial x} \sqrt{x_t^2 + y_t^2 + z_t^2} = \frac{d}{dt} \left( a(p) \frac{x_t}{\sqrt{x_t^2 + y_t^2 + z_t^2}} \right)$$ \hspace{1cm} (5.6)

Recall that $x_t/\sqrt{x_t^2 + y_t^2 + z_t^2}$ is the $x$ component normalized direction at position $p$. 

55
Assuming the light ray reaches the gas volume at \( t = t_0 \) and leaves at \( t = t_1 \). Since at each time instance, the shortest OPL constraint should be satisfied, we can integrate Eqn. 5.6 from \( t_0 \) to \( t_1 \):

\[
\int_{t_0}^{t_1} \frac{\partial n(p)}{\partial x} \sqrt{x^2 + y^2 + z^2} \, dt = n(p) \left( \frac{x_t}{\sqrt{x_t^2 + y_t^2 + z_t^2}} \right)_{t_0}^{t_1} \tag{5.7}
\]

Substituting the incident-exit ray pair: \((P_{in}, \vec{d}_{in}; P_{out}, \vec{d}_{out})\) into Eqn. 5.7, we have:

\[
\int_{P_{in}}^{P_{out}} \frac{\partial n(p)}{\partial x} \, ds = n(P_{out})\vec{d}_{out} - n(P_{in})\vec{d}_{in} \tag{5.8}
\]

Similar derivations hold for the \( y \) and \( z \) components of \( p \). To recover the RIF, we discretize 3-D space into voxels and estimate the refraction index at each voxel. Specifically, we can predefine the gas volume and discretize Eqn. 5.8 for each light path \( c \) as:

\[
\sum_c l(p) \begin{pmatrix}
  n(p_{x+1,y,z}) - n(p_{x,y,z}) \\
  n(p_{x+1,y,z}) - n(p_{x,y,z}) \\
  n(p_{x,y+1,z}) - n(p_{x,y,z}) \\
  n(p_{x,y,z+1}) - n(p_{x,y,z})
\end{pmatrix} = n_{air} \begin{pmatrix}
  \vec{d}_{out} - \vec{d}_{in} \\
  \vec{d}_{out} - \vec{d}_{in} \\
  \vec{d}_{out} - \vec{d}_{in} \\
  \vec{d}_{out} - \vec{d}_{in}
\end{pmatrix} \tag{5.9}
\]

where \( l(p) \) is the Euclidean distance of a ray inside the voxel \( p \). For voxels at the bounding faces of the gas volume, we assume their refractive indices are equal to \( n_{air} = 1.000293 \), i.e., the boundary condition for the PDEs. Using all light paths, we form a PDE system of the RIF. We further impose an additional constraint that \( \forall p, n(p) > 1 \), and solve this over-determined PDE system using the Preconditioned Conjugate Gradient (PCG) method.

The RIF estimation method presented above requires knowing the light paths. However, we only have ray-ray correspondences at the first iteration. Therefore, we initialize light paths by fitting a Hermite spline [76] to each ray-ray correspondence \((P_{in}, \vec{d}_{in}; P_{out}, \vec{d}_{out})\). We choose the Hermite spline because it directly accounts for the locations and the tangent directions at two control points that are directly provided by the ray-ray correspondence.
Figure 5.5: Accelerating light path refinement via volume pruning. (a) We prune voxels outside the bounding volume; (b) We restrict the search direction within a cone.

### 5.2.2 Light Path Refinement

Once we obtain the initial estimation of the RIF, we refine the light paths within the gas volume using Fermat’s Principle. Specifically, we set out to find the shortest OPL from $P_{\text{in}}$ to $P_{\text{out}}$. This is a classical shortest path problem on polyhedra and NP-hard [77]. Our approach is to approximate the solution by mapping the problem to a planar graph so that existing efficient algorithms such as the Dijkstra’s [78] can be directly used. It is important to note that constructing the graph is non-trivial. A brute-force construction is to directly treat the centroid of each voxel as a node and impose six-direction connectivity. However, such graph approximation does not consider how much the distance light travels inside each voxel. We therefore include two types of additional nodes: 1) the corners of voxels and 2) the Steiner points [79] on the faces and edges of voxels. In our experiments, we generate $8 \times 8 = 64$ Steiner points on each face and 8 on each edge. The weight between two nodes is computed as: $w = n \cdot l$, where $n$ is the estimated refractive index of the voxel and $l$ is the Euclidean distance between the nodes.

The resulting graph, however, contains too many links and directly applying the Dijkstra’s algorithm is computational expensive. For example, if we discretize the volume to $32 \times 32 \times 32$ voxels, we obtain a graph of $10^6$ nodes. To speed up shortest
path computation, we aggressively prune the nodes based on the observation that light paths only slightly deviate from a linear path. Specifically, we impose a “bounding volume” along the previously estimated light path as shown in Fig. 5.5(a) and only conduct searching within the bounding volume. Furthermore, inside each voxel, we prune a large amount of corners/Steiner points from the search by assuming that the light direction will not change drastically after the refinement. Specifically, we fit a cone of $10^\circ$ with the incident direction into the voxel as its central axis and the start point as its apex and only consider those nodes falling within the cone, as shown in Fig. 5.5(b).

5.3 Experiments

We have validated our approach on both synthetic and real data. All experiments are conducted on a PC with an Intel i7-3930K CPU (3.20GHz 6-core) and 16G memory.

5.3.1 Synthetic Scene Simulations

We first test our solution on a static gas volume whose RIF follows Gaussian distribution:

$$n(x, y, z) = n_{air} - (n_{air} - 1)e^{-(x-x_0)^2+(y-y_0)^2+(z-z_0)^2)/2},$$

where $(x_0, y_0, z_0)$ is the center of flow. We discretize the gas volume to $23 \times 23 \times 23$ voxels. The resulting RIF has $12,167$ unknowns. We emulate a LF-probe with a lenslet array of $60 \times 60$ microlenses and color pattern as described in Sec. 5.1.1. To capture the LF-probe image, we have implemented a voxel-based Ray-tracer that can trace along non-linear light paths within the volume. Fig. 5.6(a) shows a sample of rendered LF-probe image. If we assume that each microlens is seen once by the viewing camera, each LF-probe provides at least $3,600$ ray-ray correspondences. In this synthetic scene, we use three LF-probes surrounding the gas volume to mimic the real setup. This leads to a total of $10,800$ ray-ray correspondences. By Eqn. 5.9, we form three equations (for the x, y, and z components respectively) for each correspondence and we have $32,400$ equations in total for $12,167$ unknowns.
Figure 5.6: A simple Gaussian gas flow. (a) A rendered LF-probe image through the flow; (b) Optical flow results on the green channel; (c) Directional variations of rays; (d) The ground truth RIF; (e) Our estimated RIF after 1 iteration; (f) Our final result.
Figure 5.7: The fuel injection dataset. (a) A volume rendering of the ground truth data; (b) Optical flow results on the green channel; (c) Directional variations of rays; (d) Slices of our reconstructed RIF vs. the ground truth.
Since our synthesized images are noise-free, we directly map the red/blue channels to incident ray directions and apply optical flow on the green channel to determine incident ray origins. The measured optical flows and angular variations are shown in Fig. 5.6(b) and (c). Once we acquire the ray-ray correspondences, we apply the reconstruction algorithm described in Sec. 5.2 to iteratively estimate the light paths and the RIF. We stop the iteration when the change from the previous iteration is below a threshold. In this example, our estimated RIF converges after 3 iterations and each iteration takes around 5 minutes. To visualize our results, we show the central slice of the reconstructed RIF in Fig. 5.6(f). To evaluate accuracy, we use two error measures: the Averaged Percentage Error (APE) and the Maximum Percentage Error (MPE) over all voxels. Our reconstruction achieves high accuracy: APE = 1.19%, MPE = 1.74%.

Next, we test on a more realistic fuel injection dataset from DFG SFB 382 (www.volvis.org). Since this volume data has more complex geometric structures, we discretize the volume into $32 \times 32 \times 32$ voxels. We also increase the number of microlenses in the lenslet array to $160 \times 160$ in order to acquire dense enough ray-ray correspondences. We apply the same approach to iteratively recover the RIF and the light paths. In this example, the RIF converges after 5 iterations. Fig. 5.7(d) compares our reconstruction results with the ground truth data. Our reconstruction produces low error throughout the volume (APE = 1.84%, MPE = 5.29%).

Finally, we compare our ray-ray vs. traditional point-pixel correspondence based solutions. Specifically, we have implemented the multi-camera BOS gas flow reconstruction algorithm [1] and conducted comparisons under various configurations: w.r.t the number of camera-pattern pairs and gas-pattern distances. When testing different numbers of cameras, we fix the gas-pattern distance to $5 \times$ the gas volume size; when studying the impact of gas-pattern distance, we use 4 camera-pattern pairs. The results are reported in Table 1. With even a small number of camera-LF-Probe pairs, our approach produces accurate reconstruction. In contrast, BOS works well when using a large number of camera-pattern pairs and distant patterns. Its performance drops
Errors w.r.t the Number of Camera-Pattern Pairs

<table>
<thead>
<tr>
<th># of cameras</th>
<th>3</th>
<th>4</th>
<th>8</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Atcheson et al.[1]</td>
<td>APE</td>
<td>5.09%</td>
<td>4.61%</td>
<td>3.67%</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>17.52%</td>
<td>13.78%</td>
<td>12.78%</td>
</tr>
<tr>
<td>Ours</td>
<td>APE</td>
<td>1.84%</td>
<td>1.13%</td>
<td>0.93%</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>5.29%</td>
<td>3.20%</td>
<td>2.57%</td>
</tr>
</tbody>
</table>

Errors w.r.t the Pattern-Gas Distance

<table>
<thead>
<tr>
<th>Distance †</th>
<th>2×</th>
<th>4×</th>
<th>8×</th>
<th>16×</th>
</tr>
</thead>
<tbody>
<tr>
<td>Atcheson et al.[1]</td>
<td>APE</td>
<td>10.34%</td>
<td>4.82%</td>
<td>2.08%</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>25.73%</td>
<td>14.51%</td>
<td>10.49%</td>
</tr>
<tr>
<td>Ours</td>
<td>APE</td>
<td>1.28%</td>
<td>1.09%</td>
<td>1.12%</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>2.93%</td>
<td>2.77%</td>
<td>3.04%</td>
</tr>
</tbody>
</table>

† refers to the pattern to gas distance in units of the gas volume size.

Table 5.1: Reconstruction errors using our approach vs. [1] under different configurations. (APE: Averaged Percentage Error. MPE: Maximum Percentage Error.)

with either fewer camera-pattern pairs or closer patterns. This is because point-pixel correspondences are insufficient for determining the light paths unless the pattern is placed far away. However, distant patterns will cover smaller FoV and therefore more cameras would be needed.

5.3.2 Real Scene Experiment

Fig. 5.8 shows our acquisition system. We use 3 Pointgrey Flea2 cameras to capture the LF-probes through the gas. To assemble a LF-probe, we use a FresnelTech (www.fresneltech.com) hexagonal lenslet array sheet with 100 × 90 microlenses, each having a diameter 0.09" and focal length 0.12". Our color pattern is printed onto a transparency using a Canon PIXMA Pro9000 inkjet photo printer with 1200 dpi. Our probe has a FoV of 18.92° and an angular resolution of 18.92°/48 = 0.39° under 48 color tones. All cameras are synchronized and capture at a resolution of 1024 × 768 at 30fps. Each camera uses a lens with focal length 16mm to cover a FoV of 22° horizontally and 18° vertically. Since the gas flow is fast-evolving, we use a fast shutter of 1/320 s. To reduce image noise due to fast shutter, we use ultra bright illuminations. Specifically, we place a lamp of 250W (6500 lumen) covered by a diffuser behind the pattern. To
generate real 3D gas flows, we use an alcohol lamp whose flame temperature can reach around $600^\circ\text{C}$. We recover a cube right above the flame of size $10\text{cm} \times 10\text{cm} \times 10\text{cm}$ and discretize the volume into $32 \times 32 \times 32$ voxels.

Fig. 5.9 shows our reconstructed gas flows at three different time instances. For each time instance, we also show one of the three LF-probe images and its corresponding optical flow and angular variations (Fig. 5.9(a) and (b)). The gas flow inside the volume is highly inhomogeneous. We then show 3 vertical slices for each time instance (Fig. 5.9(c)-(e)) to illustrate the RIF distribution inside the volume. Our reconstructed RIF indicates that the central portion of the volume has a lower refractive index, i.e., higher temperature, which is consistent with the physical model. We refer the reader to our supplementary videos for the complete sequence of reconstructed RIF.

An important application of our solution is to synthesize gas motions and gas
Figure 5.9: Reconstruction results on a real gas flow. Each row shows the reconstructed flow at a different time instance. (a) The captured LF-probe images; (b) Measured ray direction variations through the flow; (c)-(e) Three vertical slices of the reconstructed RIFs where (d) is the central slice.
appearance for computer animation. Conventional physical-based fluid animation techniques are generally complex and computationally intensive. Our acquired results are real and can be directly composed into still images or film footage to achieve realistic visual effects. In Fig. 5.10, we compose the reconstructed flows from Fig. 5.9 onto a new background of a pool scene. The results are generated by ray-tracing through the RIFs. Our approach vividly synthesizes how heat distorts the background. The entire sequence can be found in our supplementary videos.

Figure 5.10: We compose the reconstructed flows onto a new background.
Chapter 6
MEASURING AMBIENT OCCLUSION BY LIGHT FIELD PROBE

There has been emerging interest on recovering traditionally challenging intrinsic scene properties.

In this Chapter, I present a novel computational imaging solution for recovering the ambient occlusion (AO) map of an object. AO measures how much light from all different directions can reach a surface point without being blocked by self-occlusion. Previous approaches either require obtaining highly accurate surface geometry or acquiring a large number of images.

I adopt a compressive sensing framework that captures the object under strategically coded lighting directions. I show that this incident illumination field exhibits some unique properties suitable for AO recovery: every rays contribution to the visibility function is binary while their distribution for AO measurement is sparse. This enables a sparsity-prior based solution for iteratively recovering the surface normal, the surface albedo, and the visibility function from a small number of images.

To physically implement the scheme, we construct an encodable directional light source using the light field probe. Experiments on synthetic and real scenes show that my approach is both reliable and accurate with significantly reduced size of input.

6.1 Problem Definition

The problem of recovering intrinsic properties of a scene/object from images has attracted much attention in the past decade. Tremendous efforts have been focused on intrinsic properties related to shading and reflectance [80, 81]. The problem is inherently challenging as it is under-constrained: we have multiple unknowns but a smaller set of constraints (images). Additional constraints such as priors are needed.
Figure 6.1: a sample Ambient occlusions (AO) map of a sculpture.

to make the problem trackable. For example, the earlier work by Weiss [29] used a maximum likelihood (ML) method to recover illumination invariant intrinsic properties from a sequence of images captured at a fixed viewpoint but under significantly different lightings. More recent work has focused on recovering intrinsic properties related to surface reflectance and visibility [82, 83].

In this Chapter, I explore a challenging type of intrinsic properties called the ambient occlusion map. Ambient Occlusion (AO) characterizes the visibility of a surface point due to local geometry occlusions. Given a scene point \( x \), its AO measures
the occlusion of ambient light caused by local surface geometry:

\[
A(x) = \frac{1}{\pi} \int_{\Omega} v(x, \hat{\omega}) \langle \hat{n}, \hat{\omega} \rangle d\omega
\]  

(6.1)

where \( \hat{\omega} \) is the direction of incident light; \( \hat{n} \) is the normal of \( x \); and \( \langle \cdot, \cdot \rangle \) refers to the dot product. \( v(x, \hat{\omega}) \) is the local visibility function and is equal to 0 if the light ray from \( \hat{\omega} \) is occluded from \( x \). Fig. 6.1 shows a sample of AO map.

AO reveals local visibility of illumination and hence affects the appearance of objects under shading. Applications are numerous in both computer vision and graphics, ranging from inverse rendering [33], image based relighting [84] to photometric stereo [30, 31, 32]. Recovering AO from images, however, is highly ill-posed (See details in Sec. 6.2.1). A brute-force approach is to first capture the 3D geometry of object and then compute AO using ray casting [85]. This task, however, is challenging since accurate 3D reconstruction remains difficult. The seminal work by Hauagge et al. [2] analyzes a stack of images acquired under a moving point light source. To make the problem trackable, they adopt a parametric visibility model, a cone-shaped function centered at the normal. A simple, per-pixel statistics was then used to cancel out surface albedo and to estimate visibility. Although highly effectively, their approach requires using a large number of images (350 images) and the parametric visibility model is not always valid (e.g., point A and D in Fig. 6.2).
6.2 Ambient Occlusion Recovery

In this section, we first show why AO recovery from images is an ill-posed problem and then discuss how to solve it using CS. Before proceeding, we clarify our assumptions. Similar to previous work [2], we assume Lambertian reflectance and fixed camera location. The AO map is measured w.r.t. the camera’s viewpoint.

6.2.1 Ill-Posedness

By Eqn. 6.1, we can discretize the lighting directions and approximate AO as:

$$\tilde{A} = \frac{1}{c} \sum_{i=1}^{N} v_i (\hat{w}_i \cdot \hat{n})$$ (6.2)

where $c$ is the normalization factor that regularizes $\tilde{A}$ into the range $[0, 1]$, $\hat{w}_i$ is the discretized lighting direction and $v_i$ is the local visibility function.

To measure $\tilde{A}$, we can illuminate the object using a dense set of uniform directional lights $\hat{w}_i$ and measure its corresponding intensity as:

$$I_i = \rho v_i (\hat{w}_i \cdot \hat{n})$$ (6.3)

where $\rho$ is the surface albedo.

Summing up images captured from all directions, we have

$$\sum_{i=1}^{N} I_i = \rho \sum_{i=1}^{N} v_i (\hat{w}_i \cdot \hat{n}) = \rho \tilde{A} c$$ (6.4)

Notice that the AO term $\tilde{A}$ in Eqn. 6.2 cannot be resolved based on the intensity of images since the albedo $\rho$ is also unknown. To address this issue, Hauagge et al.[2] adopt a simplified parametric visibility model. Specifically, they assume that the visibility function follows cone-shaped distribution centered at the normal as $A = \pi \sin^2 \alpha$, where $\alpha$ is the cone’s half angle. Computing the visibility function is to estimate $\alpha$. Under uniformly distributed lighting directions, they have shown that computing $\kappa = E[I^2] / E[I^2]$ ($E[.]$ stands for expectation) directly cancels the albedo and can be used to directly solve for $\alpha$. For their assumption to work, densely distributed light sources will be needed and an image needs to captured for each lighting direction.
Figure 6.3: (a) A simple scene illuminated by one of our directionally encoded light sources. Point P’s AO is affected by the sphere and the cylinder. We use 50 different patterns and (b) shows the P’s intensity with respect to different patterns. (c) shows P’s ground truth visibility distribution over the hemisphere (foreshortened by its normal). (d) shows intermediate visibility estimation results at different iterations by our algorithm.
6.2.2 AO Estimation Via Compressive Sensing

Our approach is motivated by recent compressive sensing schemes. Instead of capturing one lighting direction at a time, we aim to enable multiple lighting directions in one shot. A downside though is that we can no longer use the $\kappa$ statistics mentioned above for canceling out the albedo for visibility estimation. Instead, we build our solution on compressive signal reconstruction.

We use a binary vector $b = [l_1, \ldots, l_N]$ to represent the status of $N$ lighting directions, where $l_i = 1$ or 0 corresponds to if the lighting direction $\hat{w}_i$ is enabled or disabled. Under this formation, we have:

$$I = \rho \sum_{i=1}^{N} l_i v_i \langle \hat{w}_i \cdot \hat{n} \rangle$$

(6.5)

We can now use a set of $M$ strategically directional lighting patterns. For each pattern $b^j$, $j = 1 \ldots M$, we capture an image $I^j$. This results in an $M \times N$ measurement matrix $B = [b^1, b^2, \ldots, b^M]^T$ and we can rewrite Eqn. 6.5 as

$$I = \rho B [V * W(\hat{n})]$$

(6.6)

where $W(\hat{n}) = [\langle \hat{w}_1 \cdot \hat{n} \rangle, \langle \hat{w}_2 \cdot \hat{n} \rangle, \ldots, \langle \hat{w}_N \cdot \hat{n} \rangle]$ and $[*]$ refers to the pairwise element-wise product.

Given the measurements, for each pixel, we aim to solve for $\rho$ (a scalar), $V$ (a $N \times 1$ vector) and $\hat{n}$ (a unit vector with two degrees of freedoms). Hence the number of unknowns is $N + 3$. Again, we want to use fewer constraints (number of input images), i.e., $M < N + 3$. Our solution is to reduce the problem to two sub-problems and solve them using iterative optimization.

Visibility Recovery Sub-problem.

The simplest initialization of the surface normal is to directly use the camera’s viewing direction $\hat{n}_0 = [0, 0, 1]$ for all normals. This works well in our scheme but incurs slower convergence. A better scheme is to use the lighting pattern that incurs highest
intensity and use their averaged direction. This results in the initial assignment of $W_0$. We then set out to find the optimal $\rho$ and $V$ by optimizing the following objective function:

$$
\rho, V \leftarrow \arg \min_{\rho, V} \|\rho B(W_0 * V) - I\|_2
$$

Subject to: $V = [0, 1]$

(6.7)

Notice that $V$ is a binary pattern and solving $V$ in this optimization is NP-hard.

Recent works in compressive sensing [48] and signal representation [86, 87] show that this problem can be reduced to an $\ell_\infty$ regularized $\ell_1$ minimization. Specifically, we transform the optimization objective function to:

$$
\hat{\rho}, \hat{V} \leftarrow \arg \min_{\rho, V} \left\{ \|\rho B(W_0 * V) - I\|_2^2 + \lambda_1 \|V\|_1 + \lambda_2 (\|V - 0.5\|_\infty) + \lambda_3 \|\nabla V\|_1 \right\}
$$

(6.8)

where $\lambda_1, \lambda_2$ and $\lambda_3$ are weighting factors. The new objective function consists of four terms: 1) $\|\rho B(W_0 * V) - I\|_2^2$ corresponds to the fidelity term where the estimated $V$ should be consistent with the observed pixel intensities $I$; 2) $\|V\|_1$ is the sparse prior term that forces the visibility of negligible light directions, i.e., those do not affect the observation should be zero. With this term, the solution would favor a sparse set of visible light directions; 3) $\|V - 0.5\|_\infty$ is the binary prior term. It is used to clamp the elements of $V$ with high values to 1 and lows values to 0. Combining $\|V\|_1$ and $\|V - 0.5\|_\infty$ with weighting factors allows us to obtain an approximate binary solution; and 4) $\|\nabla V\|_1$ is the total variation term, i.e., to bias towards a solution with compact visible areas. Under this formulation, Eqn. 6.8 can be modeled as a second order cone problem (SOCP), we use the CVX optimization toolbox [88] to obtain the solution.

We have validated our approach on both synthetic and real experiments.

**Normal Recovery Sub-problem**

Recall that under our formulation, the per-direction visibility vector $\hat{V}$ obtained by solving Eqn. 6.8 is not binary. We need to regularize $\hat{V}$ to binary using a predefined
threshold $k$, we set $k = 0.35$ in our experiment. This results in a binary visibility vector $\tilde{V}$. Now that we have both the visibility vector and albedo, we can refine the estimation of normal $\hat{n}$ by solving for the following least square problem:

$$
\hat{\rho}, \hat{\hat{n}} \leftarrow \arg \min_{\rho, \hat{n}} \| \hat{\rho}B[W(\hat{n}) * \tilde{V}] - I \|_2
$$

Subject to $\|\hat{n}\|_2 = 1$

(6.9)

Specifically, we relax the constraint to $\|\hat{n}\| \leq 1$ and directly solve it via constrained least square minimization (e.g., the lsqmin function in Matlab). The results are normalized to have unit length $\tilde{n}$. Next, we use $\tilde{n}$ to update $W$. We repeat the process to iteratively improve the visibility and normal estimation. In our implementation, we preset the maximum number of iterations (15 in most cases). In order to perform CS acquisition, it is critical that the sampling basis are incoherent. We choose to use the Hadamard transform (HT) [89, 41] to generate the sampling pattern.

### 6.2.3 Directional Coded Light Source

Our technique in sec 6.2.2 requires using an array of light sources. The brute-force approach is to construct a dome such as Lightstage [41] and D [42]. However, constructing such apparatus is expensive and require elaborate calibrations.
We introduce the directional coded light source (DCLS) to tackle this problem. A DCLS can be viewed as the inverse of the light field camera and it emits an illumination field that can be strategically coded.

In this work, we use the projector and the light field probe to approximate a DCLS. A light field probe is a micro lenslet array with the thickness is identical to the microlens focal length. We attach a diffuser onto the light field probe. We can dynamically change spatial/angular coding by directly controlling the projection image.

A DCLS can be viewed as the inverse of the light field camera, it emits an illumination field that can be spatially and angularly coded. Unlike the point light source only possess one illumination direction, the DCLS has many illumination directions at a single position. We instead use the light field probe to emulate a light field light source (DCLS). An DCLS can be viewed as the inverse of the light field camera and it emits an illumination field that can be strategically coded. Fig. 6.4 shows different properties of three different types light source.

6.3 Experiments

6.3.1 Synthetic Experiment

For synthetic experiments, we render the scene using the POVRay ray tracer (www.povray.org). We place an object at the origin of the coordinate system and the camera on z-axis viewing towards the negative z direction. We create two 4D light field sources, each covers nearly a hemisphere. Each light field source consists of 88 uniformly distributed lighting directions. We generate 50 Hadamard patterns and render the corresponding images.

We test our technique on two scenes: the turtle scene and the tentacle scene. We apply our optimization scheme Eqn. 6.8 and 6.9 to iteratively compute the visibility term and surface normal. For Eqn. 6.8, we use the gradient of $V$ on the grid to compute the total variation term. We set the control coefficients as $\lambda_1 = 0.25$, $\lambda_2 = 10.5$ and $\lambda_3 = 0.25$ and the maximum number of iterations as 15.
Figure 6.5: Comparison between my result and [2] on AO estimation on synthetic datasets. Second row shows the ambient occlusion results on the Tentacle scene. Last row shows the recovery albedo and ambient occlusion with closeup views.
Figure 6.6: Another comparison between our result and [2] on AO estimation on synthetic datasets. Second row shows the ambient occlusion results on the Tentacle scene. Last row shows the recovery albedo and ambient occlusion with closeup views.
Figure 6.7: The Root Mean Square Error of AO and albedo estimations with different input sizes.
Next, we compare our technique with [2]. [2] requires uniformly distributed lighting directions. We emulate the sampling by using uniformly partitioning a geosphere and map the position of the vertices to direction. We render 161 images, each with only one lighting direction on. The complete results with the ground truth ambient occlusion and albedo is shown in Fig. 6.5 and Fig. 6.6. Overall, our technique produces comparable results as [2]. Furthermore, for points lying in a valley, our method produces better estimations since [2] assumes cone-shaped visibility model which is no longer valid in this case. Also note that [2] requires all 3 color channels in their optimization algorithm. If one color has zero albedo, it will fail to produce accurate AOs, (e.g., the tentacle and tail of the turtle). Finally their estimated AO tends to be smoother since they use statistical behavior of visibility within a patch of pixels.

Further, our algorithm can significantly reduce the size of input. Fig. 6.7 plots the rooted means squared error of AO and albedo w.r.t the number of input images. Our results show that with a small number of inputs (e.g., 40), we can produce comparable results as [2].

6.3.2 Photometric Stereo

We also show that our technique can improve photometric stereo by exploring two public datasets SCHOLAR and FROG which provide the images captured under calibrated directional lightings. We emulate our coded lighting process by summing the multiple images that correspond to the code pattern. In our experiment, we synthesize 10 images from the Scholar dataset and 6 for the Frog dataset and test our algorithm for normal recovery. The first image in Fig. 6.8 and Fig. 6.9 show two sample image.

Recall that both datasets use a sparse set of directional lights: 12 for the SCHOLAR and 8 for the FROG. To robustly apply total variation in our $\ell_\infty$ regularized $\ell_1$ optimization, we find, for each direction, its nearest lighting direction and compute the corresponding difference in visibility term. For both datasets, we set the control coefficients as $\lambda_1 = 0.25$, $\lambda_2 = 8.5$ and $\lambda_3 = 0.05$ and the maximum number of iterations as 15. The second column of Fig. 6.8 and Fig. 6.9 show our estimated lighting
Figure 6.8: Ambient occlusion and albedo estimation using our approach on the scholar and frog dataset.
Figure 6.9: Ambient occlusion and albedo estimation using our approach on the scholar and frog dataset.

visibility and the middle two columns show our estimated normal and albedo maps. The last two columns shows the computed normal and its error map w.r.t the ground truth normal. Our results are comparable to state-of-the-art solutions. However, we not only obtain the normal map but also the AO map, which is largely missing using classical photometric stereo techniques.

6.3.3 Real Data

Light Field Probe.

To produce encodable directional light sources, we use the light field probes that consists of a microlenslet array. To cover both sides of the object, we use two Epson HC8350 projectors and two Fresnel 300 micro lenslet arrays to create the left and right light field source. The projected resolution is 1920 x 1080. We attach a diffuser onto the back of the micro lenslet array. To expand over the full hemisphere, the left micro lenslet arrays is set to have an included angle about 75° with the horizontal plane, and the right micro lenslet arrays is 45°. The object is placed on a rotation table. When we rotate the object one cycle, the illumination field will spread the whole hemisphere. A
Figure 6.10: Fitting a hexagon grid to the scanned light field probe.

PointGray Flee 3 camera is on the rotation axis and viewing the object. The captured image has a resolution of 4096 x 2160. The camera is aligned with the rotation axis of the turning table. We also use the telephoto lenses to minimize view-dependent occlusions.

Procedure for Constructing Our Encodable Directional Light Source

To construct an encodable directional light source, the first step is to align the projectors with the micro lenslet array. To do so, we first scan the microlenslet using a digital scanner and fit a regular hexagonal grid plot with the scanned microlenslet image.

In the second step, we print the hexagonal grid plot onto a diffuser. On the plot, we also print at each corners a coding pattern for alignment with the projector. We attached the printed hexagonal grid onto the microlenslets to form the light field probe where each hexagon is aligned with a microlens. The projector projects the same
corner coding pattern onto the light field probe and we capture an image. Finally, we compute the homography between the captured image and the printed grid plot by aligning the four corner patterns. Once we obtain the homography, we can then locate the center of each microlens under the projector’s coordinate system.

To establish correspondences between the projection pattern and the direction of rays, we project the same pattern for all microlenses. Next we position a mirror ball of 2 inch diameter at the center of the rotation table. Each time we only turn on one pixel at the identical location with respect to their microlens (so that the resulting incident lights have an identical direction). We then capture the resulting image of the mirror sphere and use the location of the specular highlight to compute the incident direction.

Specifically, we use a telephoto lens and place the camera far away from the mirror ball. Basically we can reuse the environment map model to estimate the incident lighting direction. The reflected direction $\vec{R}$ can be approximate as $[0, 0, 1]$ and the normal direction $\vec{N}$ at the reflected point on the mirror ball surface can be computed from the specular highlight in the image as: $\vec{N} = [p_x/r, p_y/r, \sqrt{1 - p_x^2 - p_y^2}/r]$, $p_x, p_y$ is the specular highlight position, $r$ is the radius of the mirror ball in the image. Therefore
Figure 6.12: Calibrating lighting direction using mirror ball.
Figure 6.13: Calibrating the lighting direction of a mirror ball. Left: lighting directions represented by points on a uniform sphere. Identical directions are shown in the same color. Right: we detect specular highlights on the mirror ball for calibrating the lighting direction.
the incident light direction can be computed via the reflection law as:

\[
\vec{L} = 2(\vec{N} \cdot \vec{R})\vec{N} - \vec{R}
\]  

(6.10)

Now that we have calibrated our encodable directional light source, we can generate an arbitrary directional lighting field. Recall that each pixel behind the microlens in light field probe maps to a ray direction, as shown in Fig. 6.14. Therefore, we can encode the projection pattern for the microlens. Specifically, we use the binary pattern: which pixel is on corresponds to which direction is enabled. This allows us to simultaneously enable multiple lighting directions and dynamically change the combination of directions using our coding scheme.

Fig. 6.15 shows our light field probe setup with angularly coded by a color wheel pattern. The emitted rays with the same directions have the same color.

Due to low angular resolution of the micro lenslets, we need to rotate the object about 60 degree each time and we capture 12 images at each position with different lighting patterns. At each position, we solve for the visibility and normal independently. The visibility measures for the same patch are stitched over the hemisphere. We choose
Figure 6.15: Directional invariance of light field probes. We project identical color wheel images for every micro-lenslet so that each lighting direction maps to a unique color.

The one that incurs least error (with respect to input) as the estimated normal and albedo. The control coefficients we use in our experiments are: $\lambda_1 = 0.1, \lambda_2 = 10.5$ and $\lambda_3 = 0.2$.

Fig. 6.18 shows the reconstruction results on a toy duck. In our experiment, a total of 72 images were captured at a resolution of 4096 x 2160. To reduce image noise and computational cost, we downsample the images and only process regions of interest (about 500x400 image resolution). To calibrate the camera response curves, we use the color calibration toolbox provided [90].

We use the Hadmard pattern with 75% of the lighting directions enabled at each capture. This is more advantageous than acquiring an image under a single light direction since it provides more lights to reduce noise. Fig. 6.18 shows our recovered AO, normal map and albedo map. Our results are comparable to the synthetic ones. The recovered albedo map, however, exhibits some color bleeding artifacts (e.g., the head and belly region of the duck), which might be caused by the error when we align
the rotated images together. Also notice that our approach is per-pixel based and does not impose smoothness priors to adjacent pixels and therefore our normal map appears noisier than traditional photometric stereo.
Figure 6.17: Left: our system uses two light field probes to emulate encodable directional light sources. Right: Closeup views of the light field probes.

Figure 6.18: The AO, normal and albedo estimation results on a real Donald duck. The bottom row is the closeup view.
Chapter 7

CONCLUSION AND FUTURE WORK

In this dissertation, I have presented a unified framework of transparent object/phenomenon acquisition using novel computational camera and lighting devices. On the capture side, I have shown computational illumination device can simplify the capture system and directly resolve the ambiguity. On the algorithms side, I have also developed a class of companion computational photography algorithms for transparent object/phenomenon reconstruction.

7.1 Conclusions

This dissertation makes the following contributions in computational photography and computer vision.

7.1.1 Dynamic Fluid Surface Acquisition Using a Camera Array

In Chapter 3, I have presented a new framework for reconstructing dynamic fluid surfaces by using a camera array. It is the first multi-view (≥ 3) solution for fluid surface acquisition. I have addressed many practical issues, ranging from hardware designs, to data streaming and storage, and to robust reconstruction. I have validated my approach on both synthetic and real world specular surfaces. Experiments have demonstrated that my framework is robust and accurate. In particular, my solution can handle the challenging problem of losing track of feature points which can be detrimental or even fatal to single-camera or stereo based methods. Further, by using time-divided multiplexing, my method is capable of capturing fast evolving fluid wavefronts.
7.1.2 Dynamic Fluid Surface Acquisition Using Bokode

In Chapter 4, I have presented a novel and affordable solution for reconstructing dynamic fluid surfaces by using a special optical device called the Bokode to emulate a pinhole projector. By associating the projection pattern with the observed pixels, I directly obtain ray-ray correspondences that can be used to recover the surface normal field. My method hence is one of the few that directly resolves the point-pixel ambiguity in single-view based solution. Another unique feature of my approach is that it provides an angular reconstruction of the normal field whereas most, if not all, previous approaches recover a spatial (height-field) sampling. I have hence developed a tailored surface integration algorithm for integrating the normal field.

My technique has a number of limitations. First, I rely on the AAM technique for feature alignment. I chose not to use color patterns as chromatic aberrations caused by refraction can greatly affect color registration. The quality of AAM, however, depends heavily on the training data. In my first few trials on acquiring real surfaces, I was unable to match many symbols due to distortions and I had to render a much larger set of training data and occasionally need to conduct manual alignment.

7.1.3 Gas Flow Reconstruction Using Light Field Probe

In Chapter 5, I have presented a new computational imaging solution for reconstructing dynamic 3D gas flows. I use the LF-probe for generating view-dependent features: by coupling the LF-probe with a viewing camera, I establish reliable ray-ray correspondences. I have then used these correspondences to iteratively estimate the light paths and the refractive index field of the gas volume. Experiments on synthetic and real data have shown that my approach provides a portable, reliable, and accurate solution for reconstructing dynamic inhomogeneous gas volumes of small to medium scales.

One limitation of my approach is its sensitivity to noise. I use a fast shutter to capture fast evolving flows. To reduce image noise, I illuminate the LF-probes with ultra-bright light sources. In my experiments, I still observe color noise in the acquired
LF-probe images and have to apply denoising filters. An alternative is to use error-correction patterns. The LF-probe I use has a rather small FoV. As a result, I can only detect ray direction variations within a small range (e.g., ±15°). This has limited my approach to small to medium scale flows. The issue can be alleviated by using microlens arrays with a wider FoV. On the reconstruction front, my light path refinement step is still computational expensive despite aggressive node pruning. A GPU-based solution may significantly accelerate my scheme for time-sensitive applications.

7.1.4 Measuring Ambient Occlusion by Light Field Probe

In Chapter 6, I have presented a novel computational imaging solution for recovering the ambient occlusion (AO) map of an object. Our technique borrows the compressive sensing framework by capturing the object under strategically encoded lighting directions. I have developed a sparsity-prior based solution for robustly recover the surface normal, the surface albedo, and the visibility function. To physically implement our scheme, I have constructed an encodable light source by exploiting the recent light field probe designs. Experiments on synthetic and real scenes have shown that our approach is reliable and accurate and can greatly reduce the input size.

7.2 Future Work

For the camera array based reconstruction, An important future direction is to integrate the fluid dynamics model with the acquired fluid surfaces. Like most previous approaches, my framework focuses on reconstructing individual frames without considering temporal coherence. On one hand, one future directions is to explore how to find the optimal surface that obeys the fluid dynamics and has the minimal distance to my initial reconstruction. On the other, another potential direction is to investigate how to use the recovered dynamic fluid surfaces to infer or validate existing fluid dynamics models.
my system can be extended to acquiring wavefront of more complex geometry such as folding or breaking waves. The challenge there is that the refraction path consistency measure becomes obsolete. Since my camera array simultaneously capture the scene from multiple viewpoints, a potential solution is to use light-path triangulation [17]. However, previous solutions rely on varying the lighting/viewing directions for capturing static objects and hence are not directly applicable to acquiring fast evolving fluid surfaces. In the future, I will investigate combining my camera-array with coded illuminations or with a projector array for conducting light-path triangulation at every frame.

For the bokode-based reconstruction, developing a more robust feature matching algorithm should be an immediateness future work. For example, one possible solution is to use temporally coded patterns[91], which would provide a reliable and much denser set of feature correspondences. The challenge there, however, would be the frame rate as I aim to acquire dynamic surfaces.

Another important future direction is surface reconstruction from angularly sampled normals. In my implementation, I only approximate the boundary condition for integrating the surface. It will be useful to investigate how to acquire the ground truth boundary, e.g., by using auxiliary cameras or other types of sensors. Further, it is important to note that my surface integration scheme only provides an approximation. Previous spatial-domain surface completion scheme finds the global optimal surface that best matches the normal field (in $L_2$ normal). Ours finds the local optimal by discretizing the constraints into piecewise linear ones, although my results show that this approximation is highly effective and accurate. In the future, I plan to conduct a more comprehensive study on angular-domain surface completion by using the Variational method in a similar fashion to Poisson surface/image completion.

My current setup only allows us to capture a small area of the fluid surface. The Bokode itself has a very wide field-of-view of up to 160° (depending on the pattern size and the lens’ focal length) and can a single Bokode can cover a large surface. The limitation is on the viewing camera side whose aperture is usually much smaller. My
current solution is to an auxiliary lens to refocus the rays toward the camera. The range of acquisition hence is restricted by the size of the auxiliary lens. A simple solution is to use an ultra-large lens but at a much higher cost. A more practical solution is to construct an auxiliary lens array to emulate the large lens. Compare with light field probe, My ray-ray correspondences from bokode are less accurate and cannot be directly used for recovering the surface height. In the future, it is possible to use light field probe instead of bokode to reconstruct the fluid surface, and compare the reconstruction results on a number of benchmark surfaces and explore possible integrations of the two systems.

For the gas flow reconstruction, There are a number of future directions I plan to explore. My solution can benefit from denser ray-ray correspondences. It is possible to use the light field camera such as Lytro (www.lytro.com) or Raytrix (www.raytrix.de) for acquiring the LF-probe. Compared with my current setting that each microlens on the LF-probe generates a single ray-ray correspondence, the new setup will map each pixel of the microlens pattern to a ray-ray correspondence. My solution by far has not considered fluid dynamics: I reconstruct each frame individually. another future direction is using fluid dynamic models as the Navier-Stokes to enforce temporal coherence. At the same time, I can use the acquired gas flows to improve fluid simulations for producing more realistic animations.

For the ambient occlusion, Compared with [2], my technique uses much fewer images but requires known lighting directions, which is a major limitation. One future direction is to explore eliminating this requirement, e.g., by correlating the 3D geometry of the object with coding patterns. My solution is also per-pixel based and does not consider coherence between neighboring pixels. A straightforward extension is to incorporate the Markov Random Field work to enforce smoothness priors on normal and albedo estimations. Finally, same as most existing photometric stereo techniques, my method cannot handle subsurface scattering and inter-reflection. It is a important future direction to include light transport analysis [92] and acquisition [93] in my AO measurement.


Appendix

PERMISSION LETTERS

The following is the permission letter from Mr. Yuanyuan Ding, Mrs. Jinwei Ye and Mr. Wei Yang for using their publications in this dissertation.
Sub: Permission Letter

To whom it may concern:

I’m writing this letter to give permission to my labmate Yu Ji to use our co-authored paper “Dynamic 3D Fluid Surface Acquisition Using a Camera Array”, published in International Conference on Computer Vision (ICCV) 2011, in his dissertation. I do not, however, give permission for any other use or for any other use or for any re-disclosure of this information.

Your Faithfully,

Yuanyuan Ding

[Signature]

2/1/2016
Sub: Permission Letter

To whom it may concern:

I’m writing this letter to give permission to my labmate Yu Ji to use our co-authored paper “Angular Domain Reconstruction of Dynamic 3D Fluid Surfaces”, published in IEEE Computer Vision and Pattern Recognition (CVPR) 2012, in his dissertation. We contribute equally for this publication. I do not, however, give permission for any other use or for any other use or for any re-disclosure of this information.

Your Faithfully,

Jinwei Ye

2/1/2016
Sub: Permission Letter

To whom it may concern:

I'm writing this letter to give permission to my labmate Yu Ji to use our co-authored paper “Ambient Occlusion via Compressive Visibility Estimation”, published in IEEE Computer Vision and Pattern Recognition (CVPR) 2015, in his dissertation. I do not, however, give permission for any other use or for any other use or for any re-disclosure of this information.

Your Faithfully,
Wel Yang