Development and Application of a Description-Based Interface for 3D Reconstruction

by

Kai Wu

Bachelor of Engineering, Beijing University of Posts and Telecommunications 2014

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Applied Science

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES
(Electrical and Computer Engineering)

The University of British Columbia (Vancouver)

February 2018

© Kai Wu, 2018
Abstract

Advancements in state-of-the-art 3D reconstruction algorithms have sped ahead of the development of interfaces or application programming interfaces (APIs) for developers, especially to those who are not experts in computer vision.

In this thesis, we have designed a novel interface, specifically for 3D reconstruction techniques, which uses a description (covering the conditions of the problem) to allow a user to reconstruct the shape of an object without knowledge of 3D vision algorithms. The interface hides the details of algorithms by using a description of visual and geometric properties of the object. Our interface interprets the description and chooses from a set of algorithms those that satisfy the description. We show that this description can be interpreted to one appropriate algorithm, which can give a successful reconstruction result.

We evaluate the interface through a proof of concept interpreter, which interprets the description and invokes one of three underlying algorithms for reconstruction. We demonstrate the link between the description set by the user and the result returned using synthetic and real-world datasets where each object has been imaged with the appropriate setup.
Lay Summary

3D modelling of the real world has a wide range of applications, including 3D mapping and navigation, content creation for virtual reality, online shopping, video games, visual effects, cultural heritage archival, and so on. However, existing approaches require vision background to fully take advantage of these algorithms. This thesis is dedicated to developing a description-based interface for 3D reconstruction problems. The interface hides the details of algorithms via a description layer, which describes visual and geometric properties of an object. This description can be interpreted into one appropriate algorithm, which leads to a successful reconstruction result. This allows the application developers to deploy 3D reconstruction functionality in their domain-specific applications without the need to understand the algorithm details.
Preface

The entire work presented here has been done by the author, Kai Wu, with the collaboration and supervision of Dr. Sidney Fels and Dr. Gregor Miller. A manuscript describing the core of our work and our results has been submitted to a vision conference and is under anonymous review at the moment of thesis submission.
# Table of Contents

Abstract ......................................................... ii
Lay Summary ..................................................... iii
Preface ............................................................. iv
Table of Contents ................................................ v
List of Tables ...................................................... ix
List of Figures ..................................................... x
List of Acronyms .................................................. xvi
Acknowledgements ............................................... xvii
Dedication ......................................................... xviii

1 Introduction ....................................................... 1
   1.1 Motivating Scenario ........................................ 3
   1.2 Outline ..................................................... 6
       1.2.1 Related Work ........................................ 6
       1.2.2 A Problem Space of 3D Reconstruction ............ 7
       1.2.3 A Description of 3D Reconstruction ............... 7
       1.2.4 A Mapping of 3D Reconstruction ................. 7
       1.2.5 An Interpretation of 3D Reconstruction .......... 7
1.3 Contributions ............................................. 8
1.4 Organization ............................................. 8

2 Related Work ............................................. 10
  2.1 Toolboxes .............................................. 10
  2.2 3D Reconstruction Techniques ......................... 11
      2.2.1 Stereo ........................................... 12
      2.2.2 Shading .......................................... 17
      2.2.3 Silhouette ....................................... 22

3 A Problem Space of 3D Reconstruction .................. 24
  3.1 Problem Space ......................................... 26
  3.2 Assumptions ........................................... 27
      3.2.1 Discussion and Conclusions ....................... 30

4 A Description of 3D Reconstruction ...................... 32
  4.1 Model .................................................. 34
  4.2 Representation ........................................ 35
      4.2.1 Setup of Capturing System ....................... 35
      4.2.2 Texture .......................................... 35
      4.2.3 Brightness ....................................... 37
      4.2.4 Specularity ...................................... 38
      4.2.5 Roughness ....................................... 40
  4.3 Expression ............................................ 41
      4.3.1 Perception of Properties ......................... 41
  4.4 Discussion and Conclusions ............................ 44

5 A Mapping of 3D Reconstruction .......................... 45
  5.1 Construction of Dataset ............................... 46
      5.1.1 Selected and Baseline Methods .................... 47
      5.1.2 Synthetic Setups ................................ 48
      5.1.3 Quantitative Measures ............................ 48
  5.2 Reduction of Problem Space Dimension ................ 49
      5.2.1 Reduction of Problem Space Dimension: PMVS .... 50
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.2.2 Reduction of Problem Space Dimension: EPS</td>
<td>53</td>
</tr>
<tr>
<td>5.2.3 Reduction of Problem Space Dimension: GSL</td>
<td>57</td>
</tr>
<tr>
<td>5.3 Construction of Mapping</td>
<td>59</td>
</tr>
<tr>
<td>5.4 Discussion and Conclusions</td>
<td>60</td>
</tr>
<tr>
<td>6 An Interpretation of 3D Reconstruction</td>
<td>63</td>
</tr>
<tr>
<td>6.1 Evaluation Methodology</td>
<td>64</td>
</tr>
<tr>
<td>6.1.1 Criteria</td>
<td>65</td>
</tr>
<tr>
<td>6.2 Interpreter</td>
<td>66</td>
</tr>
<tr>
<td>6.3 Overview of Datasets</td>
<td>68</td>
</tr>
<tr>
<td>6.3.1 Calibration</td>
<td>69</td>
</tr>
<tr>
<td>6.3.2 Image Capturing</td>
<td>69</td>
</tr>
<tr>
<td>6.3.3 Synthetic Dataset</td>
<td>70</td>
</tr>
<tr>
<td>6.3.4 Real-World Dataset</td>
<td>71</td>
</tr>
<tr>
<td>6.4 Evaluation of Interpreter</td>
<td>71</td>
</tr>
<tr>
<td>6.4.1 Evaluation 1: Accurate Description, Successful Result</td>
<td>72</td>
</tr>
<tr>
<td>6.4.2 Evaluation 2: Less Accurate Description, Less Successful Result</td>
<td>75</td>
</tr>
<tr>
<td>6.4.3 Evaluation 3: Inaccurate Description, Poor Result</td>
<td>78</td>
</tr>
<tr>
<td>6.5 Discussions and Conclusions</td>
<td>81</td>
</tr>
<tr>
<td>7 Conclusions</td>
<td>85</td>
</tr>
<tr>
<td>7.1 Future Directions</td>
<td>88</td>
</tr>
<tr>
<td>7.2 Closing Words</td>
<td>90</td>
</tr>
<tr>
<td>Bibliography</td>
<td>91</td>
</tr>
<tr>
<td>A Supporting Materials</td>
<td>98</td>
</tr>
<tr>
<td>A.1 Definition of 3D Reconstruction</td>
<td>98</td>
</tr>
<tr>
<td>A.1.1 Basic Notations</td>
<td>98</td>
</tr>
<tr>
<td>A.1.2 Segment and Scell</td>
<td>99</td>
</tr>
<tr>
<td>A.1.3 Consistency</td>
<td>100</td>
</tr>
<tr>
<td>A.1.4 Formal Definition</td>
<td>100</td>
</tr>
<tr>
<td>A.1.5 Applied Definition</td>
<td>101</td>
</tr>
</tbody>
</table>
A.2 Physics-Based Vision . . . . . . . . . . . . . . . . . . . . . . . . . 101
  A.2.1 Radiometric Terms . . . . . . . . . . . . . . . . . . . . . . . . 101
  A.2.2 Reflectance Model: Microfacet Model . . . . . . . . . . . 103
  A.2.3 Image Formation: Radiometric Perspective . . . . . . . . 104
A.3 Results of Real-World Objects . . . . . . . . . . . . . . . . . . . . 106
List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 2.1</td>
<td>Assumptions made by different classes of photometric stereo.</td>
<td>20</td>
</tr>
<tr>
<td>Table 4.1</td>
<td>Model of the 3D reconstruction problem. Visual and geometric properties are selected from the problem space in Chapter 3.</td>
<td>35</td>
</tr>
<tr>
<td>Table 4.2</td>
<td>The representations of the capturing system. The configuration of the current capturing system is label in red.</td>
<td>36</td>
</tr>
<tr>
<td>Table 4.3</td>
<td>A Model and corresponding representations of the 3D reconstruction problem.</td>
<td>40</td>
</tr>
<tr>
<td>Table 4.4</td>
<td>Expression of the reconstruction problem for the four problem conditions proposed in Section 3.</td>
<td>43</td>
</tr>
<tr>
<td>Table 5.1</td>
<td>Summary of the selected and baseline algorithms implemented for the interface.</td>
<td>47</td>
</tr>
<tr>
<td>Table 5.2</td>
<td>Summary of synthetic capturing systems for three classes of algorithms.</td>
<td>48</td>
</tr>
<tr>
<td>Table 5.3</td>
<td>This is a $3 \times 3$ factorial design. Every two properties are selected to test the main effects and interaction, there are in total $\binom{N}{2}$ combinations.</td>
<td>50</td>
</tr>
<tr>
<td>Table 5.4</td>
<td>The problem conditions under which PMVS, EPS, and GSL work successfully in terms of the two metrics accuracy and completeness.</td>
<td>62</td>
</tr>
</tbody>
</table>
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1.1</td>
<td>The three layers of the 3D reconstruction interface.</td>
<td>3</td>
</tr>
<tr>
<td>Figure 1.2</td>
<td>Two target user of the proposed scenario.</td>
<td>4</td>
</tr>
<tr>
<td>Figure 1.3</td>
<td>Scenario of using the traditional approach and the proposed interface for 3D reconstruction tasks.</td>
<td>6</td>
</tr>
<tr>
<td>Figure 2.1</td>
<td>Three classes of algorithms with examples, visual cues used for reconstruction, and potential problems. The bottom left image: Silhouette Cones, ©CC BY-SA 3.0. The bottom right image: Visual Hull, ©CC BY-SA 3.0.</td>
<td>12</td>
</tr>
<tr>
<td>Figure 3.1</td>
<td>A list of properties with labels of the proposed problem space.</td>
<td>26</td>
</tr>
<tr>
<td>Figure 3.2</td>
<td>Embed algorithms into the interface. The process is three-fold: 1) reduce problem space by discovering effective properties denoted by red; 2) discover mapping from the lower-dimensional problem space to algorithm, denoted by green; 3) mapping from problem space to algorithms denoted by blue.</td>
<td>28</td>
</tr>
<tr>
<td>Figure 3.3</td>
<td>Four problem conditions selected based on the definition of problem space and additional assumptions. The description-based interface will be evaluated using objects satisfying these conditions.</td>
<td>30</td>
</tr>
<tr>
<td>Figure 4.1</td>
<td>(a). A <strong>red</strong> diffuse sphere; (b). a <strong>red</strong> specular sphere. The surface reflects light in a mirror-like way, showing a distorted environment. Since no diffuse reflection exists, the colour of the surface is no longer visible.</td>
<td>38</td>
</tr>
</tbody>
</table>
Figure 4.2 Fresnel reflectance for external reflection from a variety of substances. Since copper and aluminum have significant variation in their reflectance over the visible spectrum, their reflectance is shown as three separate curves for R, G, and B. Copper’s R curve is highest, followed by G, and finally B (thus its reddish colour). Aluminum’s B curve is highest, followed by G, and finally R. (Image from “Real-Time Rendering, 3rd edition”, ©2008 A K Peters, by permission).

Figure 4.3 The UI for determining the property settings, including albedo, specular, and roughness of the surface. As shown in (a), the problem condition in this case is: texture (0.8), albedo (0.8), specular (0.2), roughness(0.2). (b) demonstrates the effect of the property settings on a sphere, (c) on a teapot, and (d) shows the real-world object.

Figure 5.1 Example synthetic images. The value of each property ranges from 0 to 1.

Figure 5.2 Performance of PMVS under six conditions. For instance, (a) shows the performance under the condition of changing texture and albedo levels, while the others are fixed. The main effect of a property is illustrated by the colour variation along the corresponding axis. The monotonic colour variation diagonally indicates no interaction between the two properties, otherwise, there is an interaction effect. Thus in (a), we observe a main effect of texture on completeness, no other main effects and interaction effects are present.

Figure 5.3 (a) shows the reflection of light off a specular surface. $V_1$ received the diffuse component while $V_2$ receives the specular component. (b), (c) shows the images observed from these two views. The specular area (red circle) observed in $V_2$ is visible in $V_1$. 
Figure 5.4 Performance of Example-based PS under six problem conditions. For instance, (a) shows the performance under the condition of changing texture and albedo levels, while the others are fixed. The main effect of a property is illustrated by the colour variation along the corresponding axis. The monotonic colour variation diagonally indicates no interaction between the two properties, otherwise, there is an interaction effect. Thus in (a), we observe a main effect of albedo on mean angular error, no other main effects and interaction effects are present.

Figure 5.5 An example illustrates the effect of roughness on PS. Albedo is set as 0.8, and specular is set as 0.8. The first column shows the input images, the second column shows the estimated normal map, the third column shows the integrated surface, and last column shows the angular error. We can see from the qualitative results (normal map and height map), and quantitative result (angular error) that a medium level roughness would lead to a worse normal estimation since it blurs the specular lobe.

Figure 5.6 Performance of Gray-coded SL under six problem conditions. For instance, (a) shows the performance under the condition of changing texture and albedo levels, while the others are fixed. The main effect of a property is illustrated by the colour variation along the corresponding axis. The monotonic colour variation diagonally indicates no interaction between the two properties, otherwise, there is an interaction effect. Thus in (a), we observe a main effect of albedo on completeness, no other main effects and interaction effects are present.

Figure 5.7 Performance of PMVS, EPS, and GSL under all problem conditions. These are look-up tables that provide information regarding the performance of selected algorithms compared to baseline methods. Once a threshold value ε is specified, these look-up tables can be used as mapping from problem condition to algorithms, i.e., return successful algorithms given a problem condition.
Figure 6.1 Visual phenomena that indicate poor quality of reconstruction results. (a) has rough surface which indicates poor accuracy, (b) has incomplete holes which indicates poor completeness, and (c) has spikes which indicates poor normal estimation. 66

Figure 6.2 Representative objects of the four problem condition discussed in Chapter 3. (a)-(d): Synthetic objects, and (e)-(h) real-world objects. 68

Figure 6.3 Image capturing systems for MVS/VH, PS, and SL. 70

Figure 6.4 The representatives of the four classes of objects used for evaluation. Example images and problem conditions of synthetic objects are in the first two rows. The correct description of the problem condition of corresponding object is presented in third row. The last row shows the algorithms returned by the mapping, from which the interpreter selects one successful algorithm, which is coloured in red. 71

Figure 6.5 The representatives of the four classes of objects used for evaluation. Example images and problem conditions of real-world objects are in the first two rows. The correct description of the problem condition of corresponding object is presented in third row. The last row shows the algorithms returned by the mapping, from which the interpreter selects one successful algorithm, which is coloured in red. 72

Figure 6.6 (a) shows the problem condition of object ‘bust’ and ‘statue’ specified in the 3D software. (b) and (c) shows the synthetic object rendered using the problem condition in (a). 73

Figure 6.7 The interpreter selects an appropriate algorithm based on description. Based on the discovered mapping from Chapter 5, EPS and GSL can achieve a successful reconstruction, which is labelled in green while PMVS would fail to give successful result, which is labelled in red. 73

Figure 6.8 (a), (c) show the reconstruction results of the interpreted algorithm, GSL. (b), (d) show the results of the baseline method. 74
Figure 6.9 Evaluation 1: correct description leads to successful reconstruction result. The baseline results are provided so that we can determine the quality of result returned by the algorithm chosen by the interpreter.

Figure 6.10 (a) shows the reconstruction using the incorrect description while (b) shows the reconstruction result using the correct description. The text labelled in red represents incorrectly estimated property.

Figure 6.11 (a)-(d) show the reconstruction results given a less accurate description while (e) shows the result given an accurate description. The text labelled in red represents the property estimated incorrectly.

Figure 6.12 Evaluation 2: less accurate description may lead to poor result. Desc\textsubscript{i} represents inaccurate descriptions. For each object, the first row represent the description, with the incorrectly estimated property coloured in red while the correct ones in black. The algorithms determined by mapping are below the description with the algorithm selected by interpreter coloured in red (BL: baseline). The last row shows the corresponding reconstruction results.

Figure 6.13 Evaluation 2: less accurate description may lead to poor reconstruction results. Desc\textsubscript{i} represents inaccurate descriptions. For each object, the first row represent the description, with the incorrectly estimated property coloured in red while the correct ones in black. The algorithms determined by mapping are below the description with the algorithm selected by interpreter coloured in red (BL: baseline). The last row shows the corresponding reconstruction results.

Figure 6.14 (a) shows the reconstruction result given an incorrect description whereas (b) shows the result given a correct description.
Figure 6.15 Evaluation 3: inaccurate description may lead to poor result. For each description, the first row represents the settings of properties. The algorithms determined by mapping are shown below with the algorithm selected by interpreter coloured in red (BL: baseline). The last row shows the corresponding reconstruction results. We can see that the results of inaccurate descriptions are poorer than those of accurate descriptions.

Figure 6.16 The evaluation of interpreter using synthetic objects. The first column presents the description provided to the interpreter. Description \( i \) matches with the problem condition of synthetic object in column \( (i + 1) \), which is labelled in green rectangle. The last column is the algorithm selected by the interpreter. Since the interpreter would return a successful reconstruction given a description that matches the problem condition, the quality of reconstruction of the labelled objects indicates success/failure of the interpreter.

Figure A.1 Relation between a cell and a segment

Figure A.2 Illustration of light-matter interaction

Figure A.3 The light-matter interaction. Scene radiance is linear related to incident radiance

Figure A.4 The light-sensor interaction. Image irradiance is linearly related to scene radiance

Figure A.5 The camera response function is typically non-linear. Thus pixel intensity is non-linear with respect to irradiance. However, this can be corrected by radiometric calibration. More often, it is assumed that pixel intensity is linearly related to image irradiance in many vision algorithms

Figure A.6 Reconstruction results of MVS, PS, SL, and the baseline method VH

Figure A.7 Reconstruction results of MVS, PS, SL, and the baseline method VH (continued)
List of Acronyms

• **3D**: 3-dimensional
• **BRDF**: Bi-directional Reflectance Distribution Function
• **CAD**: Computer Aided Design
• **DoF**: Degree of Freedom
• **EPS**: Example-based Photometric Stereo
• **GSL**: Gray-coded Structured Light
• **MVS**: Multi-View Stereo
• **PMVS**: Patch-based Multi-View Stereo
• **PS**: Photometric Stereo
• **SfS**: Shape from Shading
• **SL**: Structured Light
• **VH**: Visual Hull
Acknowledgements

I want to thank my supervisor, Professor Sidney Fels, for offering me this opportunity to pursue this research direction, and provide invaluable guidance in the supervision of this thesis. I’m also deeply thankful to Dr. Gregor Miller for his mentorship and constant words of encouragement to keep me sane.

I want to thank my lab mates for making it such a delight to come to work everyday. I’m thankful for all the white board discussion, coding and experimenting, and late night grinding before deadlines. Those are the memories I will cherish for the rest of my life.

Last but not least, thanks to my family for their unconditional support, especially in times of stress.
Dedication

献给我爷爷吴国利先生
Chapter 1

Introduction

Modelling of the 3D world has been an active research topic in computer vision for decades and has a wide range of applications including 3D mapping and navigation, online shopping, 3D printing, computational photography, video games, visual effects, and cultural heritage archival. The goal of 3D modelling is to reconstruct a 3D model represented by point cloud, voxel grid, depth maps, or surface mesh, from RGB or range sensors, optionally incorporating the material of the surface.

Achieving this goal is an extremely challenging task, as it involves the reverse process of image formation, which is highly likely to result in a variety of possible results and solutions. To overcome this challenge, some assumptions must be made in terms of materials, viewpoints, and lighting conditions. In turn, a solid understanding of the interaction of light with surface geometry and material is a prerequisite to fully take advantage of the existing techniques. In the past decades, we have witnessed a variety of tools and approaches to 3D modelling applied successfully to an assortment of sub-domains, such as Computer Aided Design (CAD) tools [1], arm-mounted probes, active methods [2, 3, 11, 39] and passive image-based methods [17, 23, 25, 38]. Among the existing approaches, active techniques such as laser scanners [39], Structured Light (SL) systems [11], and Photometric Stereo (PS) [69], as well as passive methods such as Multi-View Stereo (MVS) [59], have been the most successful. Laser scanners and structured light techniques are seen to generate the most accurate results, but are generally complicated to calibrate, time
consuming to scan, and demanding in terms of storing and processing. Photometric Stereo is able to achieve highly detailed reconstruction comparable to that of laser scanners, but the true depth information is lost due to the use of a single viewpoint. Further, MVS requires minimal setup and can work in both controlled, small scale lab settings as well as outdoor, medium to large scale environments. However, the quality of reconstruction is generally noisier, and is susceptible to texture and material property of the surface. All of the aforementioned techniques requires an understanding of calibration, stereo correspondence, physics-based vision, and so on, which are not easy tasks to master.

Regardless of past success and strong demands across various areas, we have not yet witnessed any substantial progress in terms of making the mentioned techniques accessible to application developers or system designers (termed users for the rest of the thesis), who generally have little or no computer vision expertise. We’ve made two key observations about computer vision algorithms: 1) few of these methods work well under all circumstances, nor do they share the same setup or inputs/outputs, making it difficult for developers to choose an optimal method for their particular application; 2) expertise knowledge is a prerequisite to fully exploit the potentials of existing vision techniques. These observations lead us to the following question which we address in this thesis: is it achievable to create an interface that can return a reliable reconstruction by one of the best possible algorithms based on the descriptions of the object or scene to be reconstructed?

The interface consists of the following three layers, see Figure 1.1: the description layer sits on top and acts as the medium between the user and the lower layers. It is through this that the user provides a description of the 3D reconstruction problem. The description is passed to the interpreter layer, which chooses appropriate algorithms given the description, and then configures each algorithm’s parameters. The interpreter can also define any necessary pre- or post-processing operations (such as noise removal or image scaling). The lowest layer of the three is where the algorithms sit.
1.1 Motivating Scenario

We propose a scenario to further emphasize and justify the motivation: Ben is an engineer for 3DSense, a company designing and manufacturing 3D capturing systems and software. It is Ben’s job to design 3D camera rigs and related software for 3D capturing tasks. Ben has a background on software engineering while very limited computer vision knowledge. Daisy, a visual effects artist, wants a 3D modelling system that is able to reconstruct real world objects for her projects. She has little programming experience, but is highly skilled at 3D modelling tools. See Figure 1.2 for more details of their backgrounds respectively.

Ben starts by looking into general computer vision libraries and existing 3D reconstruction techniques, and comes across a general vision library called OpenCV, several multi-view stereo software (PMVS, MVE), a photometric stereo technique called example-based PS, and a structured light technique called Gray-coded SL. Since he fails to find any out-of-box 3D reconstruction algorithms implemented in OpenCV, he decides to use the existing techniques instead. He develops a software that integrates multiple algorithms, and provides a window to tune algorithm-specific parameters, as shown in Figure 1.3 (a). Ben builds a multi-purpose camera rig with camera-projector pairs and light sources, that is able to capture appropriate images for algorithms across varied categories, including MVS, PS, SL, and VH techniques. More specifically, the rig consists of three rings of camera-projector pairs, each positioned at a zenith angle of 15°, 45°, and 75°. The azimuth angle between two neighbouring cameras is 30°. The light sources are positioned so that no camera views are blocked. He first tests the system with a white porcelain cup, and selects PMVS as the reconstruction algorithm. The ‘parameter’ window displays a
list of parameters to tune this specific algorithm, including window size, cell size, photo-consistency measure, and seven other parameters, the effects of which are largely unclear. He runs the software using the default parameter settings since he is unaware of the effects of these parameters. The reconstructed result turns out to be very poor in the sense that the surface is not smooth and contains many holes. He suspects that this is due to the settings of the parameters. He tries to increase the cell size while fixing the other parameters, which leads to a even sparser result. After multiple ‘trial-and-error’, the software still fails to achieve a successful result. Ben decides to try out another library called OpenVL, which is able to produce a successful solution by selecting one of the underlying algorithms given a description of appearance of the object. He writes a software that allows the users to simulate the appearance of an object by tuning visual and geometric parameters using sliders, including texture, albedo, specularity, and roughness, as shown in
Figure 1.3 (b). The ‘visualization’ window displays a synthetic object, showing the effects of these parameters. The software then invokes the API to run reconstruction once an algorithm is selected by the interpreter. Ben places the *white porcelain cup* inside the camera rig and moves the sliders until the synthetic object resembles that of the target object. The parameter settings are as follows: 0.2 (texture), 0.8 (albedo), 0.8 (specularity), 0.2 (roughness). He then presses the ‘reconstruction’ button, the algorithm EPS is selected by the underlying interpreter, and a successful result is obtained. After some refinements of the camera rig and the software, Ben sends the prototype to the testing team for further testing.

Daisy the 3D artist, is invited to a user testing of the 3D modelling system by 3DSense. She starts by placing a *white porcelain vase* inside the camera rig and opens the software. She sees a parameter configuration interface with several sliders labelled with visual and geometric properties, including ‘texture’, ‘albedo’, ‘specularity’, and ‘roughness’. She moves the slides and observes that the synthetic object displays the effect of the property immediately. After some practice, she starts to tune the visual parameters of the target object: she moves each slider until the synthetic object appears to have the same appearance as the target object. The settings are as follows: 0.2 (texture), 0.8 (albedo), 0.8 (specularity), and 0.8 (roughness). The interpreter then selects GSL as the reconstruction algorithm, and proceeds to capture the input images using the camera rig for reconstruction. However, there are noticeable holes on the reconstructed model. She observes that this ceramic vase is highly smooth and glossy, thus decides to set ‘roughness’ to 0.2. The interpreted algorithm becomes EPS, and the reconstruction result turns out to be smooth and with no surface holes. Daisy is curious to see how sensitive the interpreter is with respect to less accurate descriptions. She sets the parameters to the complete opposite, i.e., 0.8 (texture), 0.2 (albedo), 0.2 (specularity), and 0.8 (roughness). This time, PMVS is selected by the interpreter, and a noisy reconstructed model is obtained. She then proceeds to do the same with all her objects including a bust, statue tuning each of the properties before doing the reconstruction.
1.2 Outline

The problem addressed in this thesis can be described as follows: construct an interface for 3D reconstruction that can return a reliable reconstruction result by one of the best-suited algorithms, which is determined by the description of the problem condition. More specifically, a problem space is proposed that transforms the 3D reconstruction problem from one requiring knowledge of algorithm details to one that is based on the relation between the problem condition and algorithms. Next, a well defined model and representations are developed to describe the problem space definitively. Lastly, mapping between the problem space and the algorithms is discovered, from which a proof of concept interpreter is proposed. An evaluation is then carried out to verify the robustness of the interpreter.

1.2.1 Related Work

We discuss the existing software and toolboxes for 3D reconstruction, and present the required vision background needed to fully take advantage of these toolboxes. A review of the 3D acquisition techniques is provided, organized by the visual and geometric cues used for reconstruction.
1.2.2 A Problem Space of 3D Reconstruction

Existing approaches to 3D reconstruction focus more on providing algorithm solutions to problems, which we call an *algorithm-centred approach*. This approach provides little insight to the problem conditions that a specific algorithm is applicable to. We proposed a *problem-centred approach* that gives a well-defined problem space, which allows further investigation of the relation between problem conditions and algorithms (termed *mapping* in this thesis). This mapping can be used to choose a best possible algorithm based on a described problem condition. The problem condition consists of a variety of visual and geometric properties of objects. The aggregate problem conditions is called *problem space*.

1.2.3 A Description of 3D Reconstruction

In previous cases, the mapping from a problem condition to an algorithm has been ambiguous due to the problem space that is poorly defined. Here, we set out to provide a rigorous description of the problem condition itself. First, a model consisting of key object properties is developed. Second, the representations of the problem space are proposed. Lastly, common 3D reconstruction tasks are expressed using the proposed model and representations.

1.2.4 A Mapping of 3D Reconstruction

To derive a more precise mapping from problem space to algorithms, we need to evaluate the performance of selected algorithms under varied properties and their combinations. We use synthetic datasets to achieve this goal. The main challenge in conducting such a comprehensive evaluation is the large variations of shape and material properties. To overcome this issue, we first discover properties that have main and interaction effect on algorithm performance (termed *effective property*). Then we evaluate the performance of each algorithm under the conditions of these *effective properties*, which serves as the basis of the mapping.

1.2.5 An Interpretation of 3D Reconstruction

We conduct the evaluation of the interface around three key evaluation questions: 1) can the proof of concept interpreter return one of the best-suited algorithms
that achieves a successful reconstruction given the correct description; 2) will a less accurate description give a poorer reconstruction result; 3) will an inaccurate description give a poor reconstruction result. To answer these questions, we carry out three separate experiments, use synthetic and real-world datasets to evaluate the interpreter in Section 6.4.

1.3 Contributions

The main contribution of this thesis is the development and application of an interface for a subset of 3D reconstruction problem, which hides algorithm details and allows users to describe conditions surrounding the problem. We focus on a subset of problem, which is defined in Chapter 3, to approach this problem in a tractable manner. This described condition, which consists of varied visual and geometric properties, can be interpreted so that an appropriate algorithm is chosen to reconstruct a successful result. This endeavour is non-trivial for two reasons: 1) currently, most approaches can only achieve satisfactory results on a limited set of categories of objects; 2) a solid understanding of reconstruction algorithm details is a prerequisite to fully take advantage of the existing techniques, which is difficult for application developers to obtain. To some extent, our interface attempts to address a broader problem space by incorporating multiple algorithms. Though it covers a wider range of problem space than a single algorithm, it is still confined within the space covered by the existing techniques within the interface. Thus, our evaluation is carried out within the problem space covered by the selected algorithms.

1.4 Organization

We organize this thesis as follows. Chapter 2 briefly introduces 3D reconstruction toolboxes and gives an overview of current landscape of 3D reconstruction field. In Chapter 3, we propose a simplified problem space of 3D reconstruction problems and propose four problem conditions that will be investigated in depth. In Chapter 4, we provide a formal description of problem condition of a 3D reconstruction problem. In Chapter 5, we develop the relation from problem condition to algorithms by evaluating the performance of a selection of algorithms under varied
problem conditions. In Chapter 6, we use both synthetic and real-world datasets to
demonstrate the interpretation of the user specified description and the robustness
of the proof of concept interpreter.
Chapter 2

Related Work

In this chapter, we review the existing software and algorithms in the field of 3D computer vision. Section 2.1 discusses the existing software and toolboxes for 3D computer vision. Section 2.2 presents a comprehensive review of the field of image-based 3D reconstruction algorithms based on varied visual and geometric cues, which include stereo correspondence, shading, silhouette, texture distortion, and focus.

2.1 Toolboxes

There have been many attempts in developing computer vision or image processing frameworks that support rapid development of vision applications. There are multiple general vision libraries in this field including OpenCV [14], VLFeat [65], VXL [4] and multiple Matlab libraries [37, 45]. These libraries often provide tools for multiple image processing and computer vision problems, including low-vision tasks such as feature detection and matching, middle-level vision tasks such as segmentation and tracking, and high-level vision problems such as classification and recognition. All of these software frameworks and libraries provide vision components and algorithms without any context of how and when they should be applied. As a result, they often require expert vision knowledge for effective use. For example, many feature detectors and descriptors are provided by OpenCV but with no indication of under what conditions each works most effectively.
We have witnessed many successful software packages in the field of image-based reconstruction, which is a sub-field of 3D reconstruction. One of the most widely used open source software is PMVS developed by Furukawa [23], which is used not only by computer vision and graphics engineers, but also production companies like Industrial Light & Magic, and Google, etc. It’s often used together with Bundler, which is a Structure from Motion software that estimate camera parameters from images developed by Noah Snavely [63], and Poisson Surface Reconstruction developed by Michael Misha Kazhdan, which is a surface mesh software that estimate the triangulated surface from oriented point cloud [36]. Some other notable open source software include VisualSim [70], CMP-MVS [28], MVE [22], and openMVG [48]. However, effective use of those software requires a basic understanding of the relevant domain, including feature detection, matching, camera calibration, dense correspondence search, etc.

This current situation motivates us to provide an description-based interface for non-vision users to access the state-of-the-art techniques in their own applications.

### 2.2 3D Reconstruction Techniques

Image-based 3D reconstruction attempts to recover the geometry and material (optional) of the object from images under different viewpoints or illuminations. The end goal here can be described as “given a set of images of an object or a scene, estimate the most likely 3D shape that explains those images, under the assumption of known materials, viewpoints, and lighting conditions”. This definition reveals that if these assumptions are violated, this becomes an ill-posed problem since multiple combinations of geometry, viewpoint and illumination can produce exactly the same images [52]. Thus this makes for an extremely challenging task.

The 3D reconstruction technique exploits a variety of visual and geometric cues to extract geometry from images: stereo correspondence, shading, contour, texture, focus, etc. This review of algorithms is structured based on these reconstruction cues. Please refer to Figure 2.1 for an overview, where the algorithms are organized based on the cue used for reconstruction.
Table 2.1: Three classes of algorithms with examples, visual cues used for reconstruction, and potential problems. The bottom left image: Silhouette Cones, ©CC BY-SA 3.0. The bottom right image: Visual Hull, ©CC BY-SA 3.0.

2.2.1 Stereo

Stereo correspondence is one of the most widely used visual cues in 3D vision. Passive methods, including stereoscopy, trinocular stereo, and MVS, identify correspondences across different views, and estimate the 3D point by triangulation. However, these passive approaches suffer from uniform or periodic surfaces. Active techniques attempt to overcome the correspondence problem by replacing one of the cameras with a controllable illumination source, e.g., single-point laser, slit laser scanner, temporal or spatially modulated Structured Light (SL), etc. Here we refer readers to the survey article by Blais for recent developments of active methods. We classify one of the most widely used passive methods, MVS algorithms, based on the taxonomy proposed in [59], which divides the field into four classes based on reconstruction method, and categorize one of the active methods, Structured Light technique, by projection patterns.
Passive method: Multi-View Stereo

Volumetric stereo algorithms compute a cost function in a 3D volume, then extract a surface from this volume. One successful example is voxel colouring, which traverses a sampled 3D space in “depth-order” to identify voxels that have a unique colouring, constant across all possible interpretations of the scene [58]. Another thread of work formulates the problem in the Markov Random Field (MRF) framework and extracts the optimal surface by Graph-Cut algorithms [54, 66, 67].

Surface Evolution algorithms work by iteratively evolving a volume or surface to minimize a cost function. This includes methods based on voxels, level set, and surface meshes. The Space Carving technique achieves a least-commitment shape [46] by iteratively removing inconsistent voxels from the scene [38]. Level-set techniques cast the problem as a variational problem, and use a set of PDE’s as cost functions, which are deformed from an initial set of surfaces towards the detected objects [17]. Other approaches use a deformable model and represent the scene as surface meshes that moves as a function of internal and external forces [16]. Hiep et al. presented a visibility-based method that transforms a dense point cloud into a surface mesh, which is fed into a mesh-based variational refinement that captures small details, smartly handling photo-consistency, regularization and adaptive resolution.

Region Growing algorithms start with a sparse set of scene points, then propagate these points to spatial neighbours, and refine the cost function with respect to position and orientation of the points. Otto and Chau proposed one of the first work on region growing stereo search [51]. The idea of this algorithm is as follows: start with an approximate match between a point in one image and a point in another, use an adaptive least-squares correlation algorithm to produce a more accurate match, and use this to predict approximate matches for points in the neighbourhood of the first match. Lhuillier and Quan proposed a two-view quasi-dense approach, which first sorts a list of point correspondences into a list of seed points by correlation score. Next, at each step of the propagation, a ‘best’ seed point is chosen. Lastly, in the immediate spatial neighbourhood of this seed point, new potential matches are checked and the best points are added to the current list of seed points [40, 41]. This “best-first” strategy guarantees convergence by choosing
only new matches that have not yet been selected. Further, a patch based approach is proposed that undergoes multiple iterations of matching, propagation, and filtering [23]. A stereoscopic approach called PatchMatch Stereo, which is inspired by an approximate nearest neighbour matching algorithm called PatchMatch [7], starts by randomly assigning an oriented plane to each pixel in two views. Next, each pixel is taken through three iterations of propagation and refinement. The plane is propagated to spatial neighbours, the corresponding pixel from another view, and across time. It can achieve sub-pixel accuracy, but is computationally heavy and challenging for parallelism. There has been some efforts to apply PatchMatch Stereo to multi-view scenarios [24, 64, 72], and develop new propagation schemes to increase the computational efficiency [24].

**Depthmap Merging** algorithms work by computing a per-view depthmap. By treating a depthmap as a 2D array of 3D points, multiple depthmaps can be considered as a merged 3D point cloud. A ‘winner-takes-all’ approach uses a set of sampled depth values and picks the value with the highest photo-consistency score for each pixel independently. Uniform depth sampling may suffice for simple and compact objects. However, for complex and large scenes, a proper sampling scheme is crucial to achieve high speed and quality. More sophisticated cost functions are derived to account for occlusion or non-Lambertian effects which may add noise to the photo-consistency score [25, 67]. In the case of severe occlusion, spatial consistency can be enforced under the assumption that neighbouring pixels have similar depth values. This can be formulated under the Markov Random Field (MRF) framework, where the problem becomes minimizing the sum of a unary $\Phi(\cdot)$ and pairwise term $\Psi(\cdot, \cdot)$. The unary term reflects the photo-consistency score of assigning a depth value $d_p$ from a set of depth value to the pixel $p$, whereas the pairwise term enforces the spatial regularization, and assigns the cost of setting depth label $k_p, k_q$ to a pair of neighbouring pixels $p$ and $q$, respectively.

$$E(\{k_p\}) = \sum_p \Phi(k_p) + \sum_{(p,q) \in \mathcal{N}} \Psi(k_p, k_q)$$

MVS algorithms generally have a less strict requirement on the setup, and work relatively reliably in unconstrained environments. However, it suffers under the following two conditions:
Lack of texture: Multi-View Stereo algorithms take advantage of textural information to establish point correspondences across different views. Thus homogeneous surfaces pose great challenges to MVS algorithms. We have witnessed surprisingly good results on a textureless object “Dino” in the Middlebury MVS benchmark [59]. It turns out that MVS algorithms are able to exploit very weak and intricate image textures, most of which come from shading or shadowing effects. However, these texture are so weak that images have to have very high quality.

Non-Lambertian surface: MVS algorithms require to observe the same surface patch from different angles in order to establish correspondences across views. Thus, the same surface patch needs to have similar or same appearance from different perspectives, and hence, most of the algorithms assume Lambertian reflectance. Pure Lambertian surfaces are rare in reality, but it is empirically verified that most MVS algorithms perform reasonably well on non-Lambertian surfaces. As long as the cameras can capture the diffuse reflectance component, then the photo-consistency function is able to identify and ignore images whose non-diffuse effects (e.g., specular highlights) are strong, then utilize the diffuse component in the remaining images. Further, there are some attempts to overcome this limitation, a pure passive methods was proposed that directly model and analyze non-Lambertian effects for MVS algorithms [34, 35].

Active method: Structured Light

To overcome the problem of lack of texture, one of the cameras in stereoscopy can be replaced by an illumination source, e.g., a projector, which is called Structured Light technique. It is based on projecting a temporally or spatially modulated pattern onto a surface and viewing the illuminated surface from one or more points of view. The correspondence is easily detected from the projected and imaged pattern, which is triangulated to obtain the a 3D point. Each pixel in the pattern is assigned a unique codeword, and the codeword is encoded by using grey level, colour or geometric representations. Structured Light is classified based on the following coding strategy: temporal, spatial and direct codification [55]. Temporal techniques generate the codeword by projecting a sequence of patterns. Spatial codification represents each codeword in a unique spatial pattern. Direct codifica-
tion techniques define a codeword for every pixel, which is equal to its grey level or colour.

**Temporally encoded** SL projects a sequence of patterns successively onto the surface. The codeword for a given pixel is formed by a sequence of illumination values for that pixel across the projected patterns. This kind of pattern can achieve high accuracy due to two factors: 1) the codeword basis is small (e.g., two for binary pattern), therefore, each bit is easily distinguishable; 2) a coarse-to-fine strategy is used, and the position of the pixel becomes more precise as the patterns are successively projected. This technique can be further classified by the way pattern is encoded temporally: 1) binary codeword; 2) \( n \)-ary codeword; 3) Gray-code combined with phase shifting; 4) hybrid techniques. More details are available in this review [55].

**Spatially encoded** techniques concentrate all coding into a unique pattern. The codeword that labels a certain pixel is obtained from the neighbourhood of pixels around it. Normally, the visual features gathered in a neighbourhood are the intensity or colour of the pixels or groups of pixels around it.

**Directly encoded** methods represent the codeword in each pixel directly. To achieve this, we need to use either a large range of colour values or introduce periodicity. However, this kind of pattern is highly sensitive to noise because the “distance” between codewords is nearly zero. Moreover, the perceived colour depends not only on the projected colour, but also the intrinsic colour of the surface. Therefore, reference images must be taken to eliminate the effect of surface colour. This kind of coding can be further classified as: 1). codification based on grey levels; 2). codification based on colour. More details are available in review [55].

Structured Light techniques overcomes the lack of texture problem by actively projecting a pattern onto the surface. However, it still suffers under the following conditions:

**Low surface albedo** poses a great challenge to active methods, such as SL, which utilize reflected light to establish correspondences across different views. Regardless of which projection pattern is used, the most critical component of any SL system is the decoding process, which retrieves per-pixel codeword from the imaged projection pattern. Thus, the surface albedo needs to be strong enough so that sufficient amount of reflected light can reach the camera sensor.
Non-Lambertian surfaces exhibit strong reflection in the specular direction. Images of such surfaces are challenging to interpret due to the bright points or highlights, which makes the projected pattern indistinguishable in these areas. Thus, it is impossible to decode the pixels exhibiting specular effects.

Concavity is the cause of global light transport, such as inter-reflection, which results in surface patches receiving light from sources other than the projector. Thus, the intensity value or colour of a pixel becomes noisier, which seriously affects the accuracy of decoding process.

2.2.2 Shading

Shading variation is an effective visual cue for retrieving shape of a surface. Shading variation depends on surface geometry (surface orientation), reflectance (material), and lighting (illumination). This is generally an ill-posed problem because different shapes illuminated under different light conditions may produce the same image. It becomes possible to estimate surface orientation once the reflectance property and illumination are known. This technique of estimating surface shape by shading variation is called Shape from Shading. However, this technique requires strict constraints on surface geometry since only one input image is used, which leads to a novel technique called Photometric Stereo in which surface orientation is determined from two or more images. The idea of Photometric Stereo is to vary the direction of the incident illumination between successive views while holding the viewing direction constant. This provides enough information to determine surface orientation at each pixel [68]. This technique can produce a surface normal map with the same resolution of the input image, i.e., to produce the pixel-wise surface normal map. Since the coefficients of the normal map are continuous, the integrated height map can reach an accuracy that cannot be achieved by any triangulation methods. Therefore, the Photometric Stereo technique is more desirable if the intrinsic geometric details are of great importance.

Shape from Shading

The problem of recovering the shape of a surface from intensity variation is first proposed by Horn [31]. It assumes that the surface under consideration is of a
uniform albedo and reflectance, and that the direction of the single distant light source is either known or can be calibrated by the use of a reference object. Thus the intensity $I(x, y)$ becomes purely a function of the local surface orientation. The information of reflectance, illumination, and viewing geometry can be combined into a single function called reflectance map $R(p, q)$, that relates surface orientation directly to image intensities:

$$I(x, y) = R(p(x, y), q(x, y))$$

where $(p, q) = (z_x, z_y)$ are surface gradients. Unfortunately, there are more unknown (per-pixel gradient) than there are measurements (per-pixel intensity). More specifically, surface orientation has two unknowns $(p, q)$ whereas measurements of the brightness at a single pixel only provide one constraint. Thus, additional information regarding the surface reflectance and illumination, as well as constraints on surface geometry, such as smoothness or integrability are required to estimate $(p, q)$. One common used constraint is smoothness:

$$\int p_x^2 + p_y^2 + q_x^2 + q_y^2 \, dx \, dy = \int \|\nabla p\|^2 + \|\nabla q\|^2 \, dx \, dy$$

Another is the integrability constraint:

$$\int (p_y - q_x)^2 \, dx \, dy$$

since for a valid depth $z(x, y)$ with $(p, q) = (z_x, z_y)$, we have $p_y = z_{xy} = z_{yx} = q_x$.

Most shape from shading algorithms assume that the surface under consideration is of a uniform albedo and reflectance, and that the light source directions are either known or can be calibrated by the use of a reference object. Thus, they are applicable to textureless surfaces with uniform and known albedo. Besides, a tedious calibration step needs to be carried out to estimate light direction and intensity. However, even by assuming the simplest reflectance model, Lambertian reflectance, the survey by Zhang [71] demonstrated that SfS algorithms generally perform poorly, and none performs well in all cases.
Photometric Stereo

The Classical Photometric Stereo, first proposed by Woodham [69], utilized multiple light sources from different directions to overcome the ambiguity of Shape from Shading. Assuming Lambertian reflectance, $P$ pixels per image, and $Q$ illumination directions, the intensity of the $i$th pixel under $j$th illumination is

$$I_{i,j} = \rho_i \tilde{n}_i^T \tilde{l}_j$$

$$\Rightarrow I = N^T L$$

where $I \in \mathbb{R}^{P \times Q}$ stores the pixel intensity from all images. Each column contains pixels from each image while each rows contains intensity of each pixel under all illumination conditions. $N \in \mathbb{R}^{P \times 3}$ encodes the albedo-scaled surface normal for each pixel, i.e., $N_{i,:} = \rho_i \tilde{n}_i^T$. $L \in \mathbb{R}^{3 \times Q}$ encodes the light source directions, i.e., $L_{:,j} = \tilde{l}_j$. This surface reflectance, i.e., spatially varying albedo $\rho_i$, and the normal $n_i$ can be estimated by

$$N = IL^+$$

$$\Rightarrow \rho_i = \|N_{i,:}\|$$

$$\Rightarrow n_i = \frac{N_{i,:}^T}{\|N_{i,:}\|}$$

Thus, the problem of estimating shape of a Lambertian surface under known lighting conditions has a simple solution. However, this algorithm fails to work once these constraints are violated. Thus, past research efforts have been focused on generalizing various assumptions made by classical photometric stereo. For the camera assumption, orthographic projection can be achieved by using a lens with long focus and placing the objects far from the camera. The nonlinear response can be solved by performing radiometric calibration. The shadow and other global light transportation are a few of the sources of errors, where some approaches consider them as outliers and remove them before normal estimation. The reflectance and lighting assumptions, however, are the most complicated since the reflectance properties depends on material property and microscopic structure. Further, lighting can have either an arbitrary or fixed position, orientation, and intensity. Therefore,
research on Photometric Stereo are generally on two directions: 1) generalization of reflectance; 2) generalization of lighting conditions. A summary of assumptions made by various classes of PS algorithms are presented in Table 2.1.

<table>
<thead>
<tr>
<th>Category</th>
<th>Camera</th>
<th>Light source</th>
<th>Reflectance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classical PS</td>
<td>Orthographic</td>
<td>Directional, known intensity and direction</td>
<td>Lambertian</td>
</tr>
<tr>
<td>Generalized lighting PS</td>
<td>Orthographic</td>
<td>Unknown intensity and direction, ambient</td>
<td>Lambertian</td>
</tr>
<tr>
<td>Generalized reflectance PS</td>
<td>Orthographic</td>
<td>Distant, known intensity and direction</td>
<td>Non-Lambertian</td>
</tr>
</tbody>
</table>

Table 2.1: Assumptions made by different classes of photometric stereo.

**Generalization of Lighting** It is possible to estimate the surface orientation without knowing light directions, a case also known as *uncalibrated Photometric Stereo*, see Table 2.1. Most uncalibrated techniques assume Lambertian techniques and are based on factorization technique proposed in [27]. Recall the irradiance equation:

\[ I = N^\top L \]

However, an infinite number of candidates \( \hat{N} \) and \( \hat{L} \) make the above equality met. In fact, any invertible \( 3 \times 3 \) matrix \( G \) defines a candidate pair \( \hat{N} = N \cdot G, \hat{L} = G^{-1} L \). Thus the normal \( N \) and light source direction \( L \) can only be recovered up to a linear transformation. It has been shown that only a 3-parameter subset of these transformations, known as the Generalized Bas-Relief (GBR) ambiguity, preserve surface integrability [9].

Other generalized lighting conditions are any situations other than the ideal case of using a single distant point light source in a dark room, such as natural ambient light, multiple point light sources with or without ambient lighting, etc. To make the problem more tractable, the reflectance model should no longer be a general one, as this involves too many degrees of freedom that results in many different shapes with incorrectly estimated general reflectance and incorrectly estimated general lighting.
Generalization of Reflectance This direction of research has been to relax the assumption of Lambertian reflectance. This can be broadly divided into four classes of algorithms.

Outlier rejection approach assumes that Non-Lambertian reflectance can be well approximated by the sum of diffuse and specular lobe. The specular pixels are considered as outliers in [15] and [8]. Others assume that the colour of the specular lobe differs from that of the diffuse lobe, which allows the separation of the specular and diffuse components [44, 56, 57].

Reference object approach uses a reference object that has similar material as the target object. This is proposed in [62] and later revisited in [29]. The idea is that surface points with same orientation give similar intensity values under similar reflectance and lighting. It can deal with arbitrary BRDFs as long as the reference and target object has the same material. It can handle spatially-varying BRDFs as long as there are multiple reference objects. Each reference object serves as a “basis” BRDF, and the BRDF at any point on the target object can be approximated as a linear combination of the basis BRDFs.

Parametric reflectance model approach builds upon the idea that an arbitrary BRDF can be approximated by “basis” BRDFs, and replaces the reference objects with sophisticated BRDF models. An isotropic Ward model is used as basis BRDF, and the surface orientation and parameters of the reflectance models are estimated iteratively [26].

Invariants of BRDF approach exploits various physical properties of BRDFs. While parametric reflectance models are very good at reducing the complexity of BRDFs, they are usually only valid for a limited class of materials. An alternative is to exploit the invariants of BRDFs, typically including energy conservation, non-negativity, Helmholtz reciprocity, isotropy, and so on [5, 73].

Photometric Stereo can work extremely well under certain constrained conditions. However, it generally performs poorly once the aforementioned assumptions are violated: the classical PS and generalized reflectance PS fail to work under uncalibrated light conditions. The generalized lighting PS only handle Lambertian surfaces under uncalibrated lighting conditions, but only achieves estimation up to a linear transformation; the classical PS and generalized lighting PS fail to work under generalized reflectance conditions; and lastly, most PS algorithms fail to
work on conditions of generalized lighting and reflectance, one approach that has been proved to work is to place multiple reference objects in the scene with the target object as proposed by [29].

2.2.3 Silhouette

In some cases, it’s an easy task to perform a foreground segmentation of the object of interest, which leads to a class of techniques that reconstructs a 3D volumetric model from the intersection of the binary silhouettes projected into 3D. The resulting model is called a visual hull.

The basic idea of shape from silhouette algorithms is that the object lies inside the intersection of all visual cones back-projected from silhouettes. Suppose there are multiple views $V$ of the target object. From each viewpoint $v \in V$, the silhouette $s_v$ can be extracted, which is the region including the object’s interior pixels and delimited by the line(s) separating the object from the background. The silhouette $s_v$ are generally non-convex and can represent holes due to the geometry of the object. A cone-like volume $cone_v$ called (truncated) extended silhouette is generated by all the rays starting at the centre of projection and passing through all the points of the silhouette. The target object is definitely internal to $cone_v$ and this is true for every view $v' \in V$; it follows that the object is contained inside the volume $c_V = \bigcap_{v \in V} c_v$. As the size of the $V$ goes to infinity, and all possible views are included, $c_V$ converges to a shape known as the visual hull $vh$ of the target object.

Voxel based methods First the object space is split up into a 3D grid of voxels; each voxel is intersected with each silhouette volume; only voxels that lie inside all silhouette volumes remain part of the final shape.

Marching intersections based methods The marching intersection (MI) structure consists of 3 orthogonal sets of rays, parallel to the $X$, $Y$, and $Z$ axis, which are arranged in 2D regular arrays, called the $X$ – rayset, $Y$ – rayset, $Z$ – rayset respectively. Each ray in each rayset is projected to the image plane to find the intersections with the silhouette. These intersections are un-projected to compute the 3D intersection between the ray and the extended silhouette on this ray. This process is repeated for each silhouette, and the un-projected intersections on the same ray are merged by the boolean AND operation.
Once the MI data structure representing the intersection of all extended silhouettes, a triangular mesh is extracted from it. This is done by the MI technique proposed in [53] which traverses the “virtual cells” implicitly defined by the MI, builds a proper marching cube (MC) entry for them that in turn is used to index a MC’s lookup table.

**Exact polyhedral methods** The silhouette is converted into a set of convex or non-convex 2D polygons with holes allowed. The resulting visual hull with respect to those polygonal silhouettes is a polyhedron. The faces of this polyhedron lie on the faces of the original cones. The faces of the original cones are defined by the centre of projections and the edges in the input silhouettes. The idea of this method is: for each input silhouette $s_i$ we compute the face of the cone. Then we intersect this face with cones of all other input silhouettes, i.e., a polygon-polyhedron intersection. The result of these intersections is a set of polygons that define the surface of the visual hull.

Visual Hull algorithms don’t rely on material properties as long as the foreground of the image can be reliably segmented, thus is applicable for objects with arbitrary reflectance properties. However, it fails to carve the concavities on the object surface, thus is unsuitable for concave objects.
Chapter 3

A Problem Space of 3D Reconstruction

We discussed the current landscape of 3D reconstruction in Chapter 2. Previous research has solely focused on developing novel algorithms and software to tackle this problem. Thus, most research efforts have been devoted to improving algorithm performance in terms of accuracy, completeness, computational efficiency, or relax restrictive assumptions so that they can be applied to more general situations. However, this approach, which we call an algorithm-centred approach, faces two challenges: 1) it provides little insight into the conditions that allow a specific algorithm to work successfully; 2) it requires domain-specific knowledge to fine tune algorithm specific parameters to optimize the performance. This knowledge is either unknown or largely empirical, with each algorithm mapped roughly to a sub-volume in the problem space that is poorly defined, thus requires vision knowledge to fully take advantage of these algorithms. In this thesis, problem space is defined as an $N$–dimensional space which encompasses the material and shape properties of objects, and thus the axes of which consist of characteristic material and geometric attributes (called properties). The sub-volume of the problem space is called problem condition(s). We argue below that a well-defined problem space is a critical part of designing an interface of 3D reconstruction, and should receive more research efforts.

We have established in Chapter 2 that most 3D vision algorithms target a lim-
ited set of problem conditions. They may work under one condition, but is highly likely to fail under others. Thus, they are unsuitable to reconstruct objects with a wide range of properties. It is crucial to have multiple algorithms, each registered to a distinct sub-volume of the problem space in order to design an interface for 3D reconstruction problem. To achieve this goal, we need to first propose key attributes of objects as axes of this $N$–dimensional space, which lay the foundation of describing problem conditions in a consistent and rigorous manner. Next, we need to discover the relation between problem conditions and algorithms so that we can know which algorithm performs well given a specific problem condition, which will be discussed in Chapter 5. Recall that problem space is an $N$–dimensional space, of which the axes are material and shape properties of objects. The reason we choose material and shape properties is that they can be visually and perceptually estimated, and are also widely used by typical 3D vision algorithms as reconstruction cues. With a well-defined problem space, we are able to describe the characteristic properties of an object that are crucial for reconstruction. For instance, instead of describing a cup simply as a ‘cup’, we can describe it as ‘a white, glossy porcelain cup with shallow strips on the surface’. The visual and geometric properties, represented by words such as ‘white’, ‘glossy’, and ‘shallow’, are crucial in terms of determining which algorithm is able to perform well. We call it a problem-centred approach. This approach transforms the 3D reconstruction problem from one requiring knowledge and expertise of specific algorithms in terms of how to use them, to one requiring knowledge of problem conditions, which can be perceptually estimated or measured. The advantage of the problem-centred approach are as follows: 1) the properties can be universally used by most objects, without the need of algorithm-specific parameters; 2) the properties of the problem space can be visually or perceptually estimated. Thus, there is no need to understand the meanings of algorithm parameters, i.e., no vision knowledge required.

In this chapter, we first propose a well-defined problem space consisting of visual and geometric properties of objects in Section 3.1. Since the problem space is generally too vast to tackle, we state addition assumptions and underlying rationale to limit the scope of problem space in Section 3.2. Finally, we propose four main problem conditions that we are interested in investigating in this thesis.
3.1 Problem Space

We first give an overview of problem space, which consists of visual and geometric properties of real-world objects, as shown in Figure 3.1. These properties can be conceptualized as axes of the 3D reconstruction problem space. This approach allows us to think of algorithms as “pointers” to volumes within an $N$-dimensional problem space. Existing algorithms can be incorporated into the interface by evaluating the algorithm performance within the problem space, as shown in Figure 3.2. However, by no means is the presented problem space complete. There are many other properties not included that are commonly seen in the real world. For instance, properties such as metalness, emission, occlusion, discontinuity, among others, are not considered. However, the listed set of properties are broad enough to encompass a wide range of real-world objects. To help easy identification of a specific problem condition, we propose the following labels to differentiate object classes, as shown in Figure 3.1.

![Figure 3.1](image)

**Figure 3.1:** A list of properties with labels of the proposed problem space.
3.2 Assumptions

Though 3D reconstruction of opaque surfaces with Lambertian reflectance has been a well-studied problem, specular, translucent, transparent, and refractive scenes still pose challenging problems for acquisition systems [32]. Ihrke et al. conducted a comprehensive review on acquisition approaches for transparent and specular surfaces, and concluded that though different classes of techniques have been developed to deal with these problems and good reconstruction results can be achieved with current state-of-the-art techniques [32]. However, the proposed approaches are still highly specialized and targeted at a very specific object class [32]. Thus, the limits of existing techniques demand that only a subset of problem space can be solved robustly. Further, it is more feasible as a proof of concept demonstration to target a sub-space. Therefore, we make the following assumptions to keep the problem space more tractable:

Simplified Light-Matter Interaction Model

We assume local interaction model, i.e., global light transport such as transmission, refraction, cast shadow, inter-reflection are not considered. The rationale behind our choice is that most techniques that have been developed over the past few decades mainly tackle objects with an opaque, diffuse or mixed surface [32]. Since the majority of reconstruction techniques rely on observing light reflected off a surface, surfaces exhibit significant effect of global light transport present a huge challenge to the reconstruction problem. For specular, refractive, and translucent or transparent objects, only very specialized algorithms are applicable for reconstruction [32]. This is a widely used and accepted model in varied areas of computer vision, including shape from stereo, shading, and so on. As more algorithms become available to tackle these types of objects, they can be embedded to the interface using the same approach that will be discussed in Chapter 5, as shown in Figure 3.2.

Simplified Reflectance Model

Most real-world surfaces are not optically flat, or smooth at a microscopic scale. In most cases, there are irregularities present which are much larger than the light
Figure 3.2: Embed algorithms into the interface. The process is three-fold: 1) reduce problem space by discovering *effective properties* denoted by red; 2) discover mapping from the lower-dimensional problem space to algorithm, denoted by green; 3) mapping from problem space to algorithms denoted by blue.

Wavelength, but too small to be seen or resolved. The *microfacet theory* assumes that the surface is composed of a large collection of *microfacets*, each reflecting light from a given incoming direction into a single outgoing direction which depends on microfacet normal. We assume microfacet reflectance model, which consists of a diffuse and a specular term. The diffuse term can be approximated by a Lambertian model, and the specular term is determined by material, viewing angle, shadowing, surface geometry, and so on.

**Simplified Geometric Model**

It is a challenging task to model geometry using mathematical descriptions. For geometric primitives such as cube, sphere, or cone, etc, it’s possible to describe the shape using concise descriptions. However, the task becomes more challenging when it comes to shapes with varied characteristics. Furthermore, it becomes more ambiguous when natural language is employed. Thus we only consider the microscopic roughness of the surface, which has a direct relation with the reflection. Other prominent geometric properties such as *concavity*, which affects self-shadow, inter-reflection, *depth-discontinuity*, which affects the depth estimation, are not considered.
Surface Reflectance

Existing 3D vision techniques require distinct cues for reconstruction, be it texture, intensity variation, focus change, and so on. This information will become much noisier and less effective on darker surfaces. Surfaces with low albedo will make many algorithms ineffective, including most active techniques. In this thesis, dark surfaces could be challenging for all the underlying algorithms implemented within the interface. This works fine when it comes to design and use such an interface since there is no need for all underlying algorithms to work reliably. However, from a demonstrative standpoint, this would render the interface failing to call any of the selected algorithms except the baseline methods, which fails to demonstrate the performance of interpreter. Thus, we decided to use objects with bright surfaces so that the interpreter can invoke one of the underlying algorithms given accurate description.

Setup of Capturing System

The configuration of capturing system determines the selection of algorithm in a variety of ways. The capturing device determines what type of data is captured, which could be RGB images, or range data depending on whether a camera or laser scanner is used. The lighting condition determines if a passive or active method is selected since active methods need controlled lighting. The number of vantage points determines if data from different viewpoints or under different illuminations are needed. In short, variations in the capturing systems can greatly impact which algorithm is effective, or the performance of algorithms. However, this significantly increases the complication of the problem. Thus, we decide to use a fixed capturing system throughout the thesis. However, note that the results obtained under this capturing system might not be applied to data captured using a different capturing system. However, the approach would not be affected by the hardware setup.

Built upon the previously defined problem space, and additional assumptions, we define four classes of problem conditions that will be investigated in depth, as shown in Figure 3.3. We will demonstrate that it is achievable to design a description-based interface that hides algorithm details and return solutions with-
out knowing the underlying algorithms using objects that satisfy these conditions.

<table>
<thead>
<tr>
<th>Condition</th>
<th>Surface Description</th>
<th>Image formation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Flat, matte</td>
<td>Textureless, near diffuse reflectance, rough</td>
</tr>
<tr>
<td>2</td>
<td>Flat, glossy</td>
<td>Textureless, mixed diffuse and specular reflectance, smooth</td>
</tr>
<tr>
<td>3</td>
<td>Textured, matte</td>
<td>Textured, near diffuse reflectance, rough</td>
</tr>
<tr>
<td>4</td>
<td>Textured, glossy</td>
<td>Textureless, mixed diffuse and specular reflectance, smooth</td>
</tr>
</tbody>
</table>

**Figure 3.3:** Four problem conditions selected based on the definition of problem space and additional assumptions. The description-based interface will be evaluated using objects satisfying these conditions.

### 3.2.1 Discussion and Conclusions

Most vision research has been devoted to developing algorithm novelties and improving performance. However, the conditions under which an algorithm works reliably is often a neglected subject. However, this is a crucial part of designing a 3D reconstruction interface. Since no single algorithm can work reliably under a wide range of conditions, it is required to have multiple algorithms, each covering a distinct or partially overlapping sub-volume. The problem then becomes under which condition can an algorithm work successfully. To answer this question, we first need to define the problem space and problem conditions. Otherwise, it is a poorly-defined task to describe the problem condition of an object.

We first proposed a well-defined problem space, which is an $N$-dimensional space, the axes of which are reflective and geometric properties of objects. The problem condition(s) is then defined as a sub-volume within the problem space. Since existing 3D vision algorithms can only deal with certain classes of objects, we further proposed assumptions to limit the problem space to a more feasible range. Further, to implement a proof of concept interpreter for demonstrative purpose, we choose four problem conditions to demonstrate the proposed interpreter.

Since this thesis studies a sub-space of the entire problem space, a natural ques-
tion is whether this approach extends to a broader space when incorporating properties that have been left out, such as refraction, sub-surface scattering, and so on. This is a challenging task, and depends on two aspects: 1) robust techniques that work successfully under these conditions have been developed; 2) there is a way to describe the corresponding properties. The goal of this thesis is to demonstrate whether it is possible for the proof of concept interpreter to find a reliable algorithm given a description of problem condition. Thus, this is left as a future research direction.
Chapter 4

A Description of 3D Reconstruction

In Chapter 3, we introduced a problem space for 3D reconstruction, which provides a new perspective to understanding algorithms without delving into algorithm details. However, it is not sufficient to merely have a problem space when it comes to creating an interface for 3D reconstruction. It provides little in terms of describing problem conditions that we are interested in solving. There are two key components that are needed in order to provide a consistent description: a model which addresses what to describe, i.e., which property is included or excluded in the model, and corresponding representations addressing how to describe the components of a model, i.e., what characteristics of a property should be of interest. The first issue has already been somewhat addressed in Chapter 3 with the well-defined problem space. Specifically, we can use a subset of properties of the problem space as the components of the model. However, the second issue has yet to be addressed. We have not discussed which aspect of a property is crucial to problem solving, and how it is measured or estimated. Besides, it is pointless to discuss the estimation of property parameter without determining the corresponding representation beforehand. Thus, it is practically impossible to provide a consistent description without a model and corresponding representations. Here is a concrete example to further stress the importance of model and representations: how to describe a porcelain vase? First and foremost, we need to decide which properties
to describe. We use surface ‘texture’ as an example. Next we need to address how this property, ‘texture’ in this example, is represented, i.e., what characteristics of the property is used to measure the strength of ‘texture’. We can choose the scale of texel (texture element), or randomness of texel as representation of texture. Otherwise, it make no sense to quantify a property before determining the corresponding representation. Thus, this chapter addresses these two issues: first, we propose a model consisting of a subset of properties from the problem space. We recognize that it is practically impossible to achieve an exhaustive description, and the scope of description is determined by the scope of the problem space. For instance, sub-surface scattering property should not be included in the description since it is omitted in the problem space as well. Secondly, we need to determine the representation of each property. Each property might have multiple facets that are essential to problem solving. For instance, when we talk about texture, some important features include randomness of texture, scale of texel, magnitude and orientation of texture, and so on. Without determining a proper representation of the property, there is no way we can proceed to perceptually estimate the strength of the property.

This way of thinking is not specific to our problem. Typical computer vision problems require, among other factors, a model of the problem space and appropriate representations [43]. The model characterizes the relevant properties of the elements in the domain and analyze their relations. The representations describe an object’s properties selected by the model to facilitate a solution of the problem. For instance, surface orientation is a part of the surface geometry model, and the corresponding representation can be surface normal or curvature. Another example is colour, which is a component of a material model, and the RGB values could be one of the representations of this property. By using this formal description consisting of a model and corresponding representations, expressing the conditions within which an algorithm works reliably becomes more well-defined. Thus, this chapter is devoted to proposing a model and representations for 3D reconstruction problem, which are the key components of the interpretable description.

In this chapter, we attempt to provide a description of the 3D reconstruction problem which allows for a well defined specification of the conditions surrounding the problem. This description abstracts away from the functional specification
of how to estimate a reconstruction. We first propose a formal definition of the 3D reconstruction problem in Appendix A.1. Next, Section 4.1 proposes a model to 3D reconstruction by selecting various key characteristics of the problem space that are crucial for describing the appearance of the object. Section 4.2 discusses corresponding representations of the proposed model. Section 4.3 provides examples of expressing 3D reconstruction problems using the proposed model and representations. These following four layers represent the description of our accessible 3D reconstruction framework: Definition, Model, Representation, and Expression.

4.1 Model

Model and representations are fundamental for vision problem solving. Model selects characteristic properties of an object, and representations describe the model selected object properties to facilitate a solution of a class of problem. A model facilitates the representation of aspects in reality that are useful in a particular problem domain [13]. There are two criteria in choosing the model: 1) the model components need to be useful for problem solving; 2) the components need to be visually or perceptually identifiable so that it is easy for users to learn to describe problem conditions using the proposed model. As discussed in Chapter 2, visual and geometric information, such as texture, shading variations, and silhouette are critical to 3D vision algorithms. In Chapter 3, we summarized key visual and geometric properties of an object that are of great importance for humans to identify and distinguish objects. It is clear that a subset of information that both vision algorithms and humans utilize overlaps, for instance, both textural and brightness information are utilized. Thus, we select a subset of properties that are used by vision algorithms as reconstruction cues, and also easily identifiable by humans as the main components of our model.

The selection of algorithm depends not only on material and shape of objects, but also on the capturing system. The lighting condition determines if certain categories of algorithms can be used. For instance, active methods need controlled lighting condition, which typically use a light source or projector whereas passive methods work under ambient lighting. The number of vantage points determines whether a single vantage point algorithm or multiple vantage points algorithm is
utilized. Lastly, the static or dynamic nature of the scene determines whether an algorithm that can dynamically update the scene is needed. The key components of the model are shown in Table 4.1.

<table>
<thead>
<tr>
<th>Model</th>
<th>Lighting</th>
<th>Vantage points</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Nature of object or scene</td>
<td>Texture</td>
</tr>
<tr>
<td></td>
<td>Brightness</td>
<td>Reflectance</td>
</tr>
<tr>
<td></td>
<td>Roughness</td>
<td>Concavity</td>
</tr>
</tbody>
</table>

Table 4.1: Model of the 3D reconstruction problem. Visual and geometric properties are selected from the problem space in Chapter 3.

4.2 Representation

Based on the proposed model of the 3D reconstruction problem, we need to further define the representations so that the 3D reconstruction problem can be expressed using the proposed model and representations.

4.2.1 Setup of Capturing System

The model of the capturing system includes the lighting condition, the number of vantage points, and the nature of the object or scene. In this thesis, we use a fixed capturing system, thus the configurations of the capturing system is constant throughout the thesis, thus is omitted in the following discussions for the sake of brevity. The configuration of the capturing system is shown in Table 4.2.

4.2.2 Texture

Texture is one of the most important cues for many computer vision algorithms. It is generally divided into two categories, namely tactile and visual textures. Tactile textures refer to the immediate tangible feel of a surface, whereas visual textures refer to the visual impression that textures produce to the human observer, which
<table>
<thead>
<tr>
<th>Model</th>
<th>Representations</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nature of scene</td>
<td>Static</td>
<td>Ambient light</td>
</tr>
<tr>
<td></td>
<td>Dynamic</td>
<td></td>
</tr>
<tr>
<td>Lighting</td>
<td>Passive</td>
<td>Light source</td>
</tr>
<tr>
<td></td>
<td>Active</td>
<td>Projector</td>
</tr>
<tr>
<td></td>
<td>Mixed</td>
<td>Ambient &amp; light source &amp; projector</td>
</tr>
<tr>
<td>Vantage point</td>
<td>Single vantage point</td>
<td>I</td>
</tr>
<tr>
<td></td>
<td>Multiple vantage points</td>
<td>Small (&lt; 10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Medium (10 – 50)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Large (&gt; 50)</td>
</tr>
</tbody>
</table>

Table 4.2: The representations of the capturing system. The configuration of the current capturing system is label in red.

are related to local spatial variations of simple stimuli like colour, orientation and intensity in an image. Here we focus only on visual textures, as they are more widely used in the stereo vision research. The term ‘texture’ hereafter refers exclusively to ‘visual texture’ unless mentioned otherwise.

Although texture is an important component in computer vision, there is no precise definition for the notion of texture itself. The main reason for this is that natural textures often exhibit separate yet contradicting properties, such as regularity versus randomness, or uniformity versus distortion, which can hardly be described in a unified manner.

There are various properties that make texture distinguishable: scale/size/granularity, orientation, homogeneity, randomness, etc. However, due to the diverse and complex nature of textures, it is a challenging task to generate a synthetic texture solely from these semantic properties, or the other way around, derive parameters from a given texture. The stereo vision community often takes a simplified approach, classifying textures into two categories, regular and stochastic, by degree of randomness. A regular texture is formed by regular tiling of easily identifiable elements (texels) organized into strong periodic patterns. A stochastic texture exhibits less noticeable elements and displays rather random patterns. Most of the real world textures are mixtures of these two categories. In this thesis, we adopt
this simplification and consider texture randomness, which is the amount of distortion in the texture. Thus, a uniform texture has low texture randomness whereas a highly textured surface has high texture randomness.

4.2.3 Brightness

When light strikes a surface, it may be reflected, transmitted, absorbed, or scattered; usually, a combination of these effects occurs. The intensity or colour information received by a sensor is thus determined, among other factors, by the amount of light available after these interactions. We assume that all effects are local, thus global effects such as inter-reflection and transmission, among others, are omitted. This is called a local interaction model. In order to understand the contributing factors of pixel intensity or colour, we need an in-depth understanding of reflection, i.e., how light is reflected off of a surface patch, and the relation between material and intensity values. The radiometric formation of an image consists of three separate processes: light-matter interaction, light-lens interaction, and light-sensor interaction, as discussed in Appendix A.2. The conclusion is that image intensity is proportional to diffuse reflectance (albedo).

Diffuse reflectance represents light that is refracted into the surface, scattered, partially absorbed, and re-emitted. The Lambertian diffuse model assumes that the refracted light has scattered enough that it has lost all directionality and thus the diffuse reflectance is constant. However, very few materials exhibit a Lambertian response. Many materials show a drop in grazing retro-reflection, and many others show a peak. This is strongly correlated to surface roughness — smooth surfaces tend to have a shadowed edge, and rough surfaces tend to have a peak instead of a shadow. This grazing shadow for smooth surfaces can be explained by the Fresnel equations: at grazing angles, more energy is reflected from the surfaces and less is refracted into the surface to be diffusely re-emitted. However, diffuse models don’t generally consider the effect of surface roughness on Fresnel refraction and either assume a smooth surface or ignore the Fresnel effect. In this thesis, we adopt this diffuse model that neglects surface roughness, thus the albedo is considered constant, and not affected by grazing angle and surface roughness.

In conclusion, as long as the light-sensor interaction is considered as a linear
mapping (as most vision algorithms do) or calibrated in a pre-processing step, the pixel intensity value is linearly proportional to diffuse reflectance, which is the representation we adopt for brightness.

4.2.4 Specularity

Specular surfaces reflect light in nearly a single direction when microscopic surface irregularities are small compared to light wavelength, and no subsurface scattering is present [49]. Unlike diffuse reflections, where we experience the brightness and colour of an object, specular reflections carry information about the structure, intensity, and spectral content of the illumination field. In other words, specular reflection is simply an image of the environment, or the illumination field, distorted by the geometry of the reflecting surface. For instance, the specular sphere in Figure 4.1 shows a distorted image of the environment instead of the underlying surface colour. A purely specular surface is a mirror, which is rare in nature. Most natural materials exhibit a mixture of specular and diffuse reflections. We use the microfacet model as the reflectance model in this thesis, which is discussed in Appendix A.2.2. Thus the amount of specular component is determined by the Fresnel reflectance, also named specular reflectance.

![Figure 4.1](image)

(a) (b)

**Figure 4.1:** (a). A red diffuse sphere; (b). a red specular sphere. The surface reflects light in a mirror-like way, showing a distorted environment. Since no diffuse reflection exists, the colour of the surface is no longer visible.

The Fresnel reflectance is the fraction of incoming light that is reflected (as opposed to refracted) from an optically flat surface of a given substance. It varies based on the light direction and the surface (in this case microfacet) normal. Fresnel reflectance tells us how much of the light hitting the relevant microfacets (the
ones facing in the half-angle direction) is reflected. Fresnel reflectance depends on refraction index (in other words, what the object is made of) and the incoming light angle (which is plotted here on the x-axis), as shown in Figure 4.2. As the angle increases, the Fresnel reflectance barely changes for the first 45 degrees; afterwards it starts changing, first slowly up to about 75 degrees, and then for very glancing angles, it rapidly goes to 100% at all wavelengths. Since the Fresnel reflectance stays close to the value for 0° over most of the range, we can think of this value $F(0^\circ)$ as the characteristic specular reflectance of the material. This value has all the properties of what is typically thought of as a “colour” - it is composed of RGB values between 0 and 1, and it is a measure of selective reflectance of light. For this reason, we will also refer to this value as the specular colour of the surface, denoted as $c_{spec}$. We choose specular reflectance as the representation of specularity.

**Figure 4.2:** Fresnel reflectance for external reflection from a variety of substances. Since copper and aluminum have significant variation in their reflectance over the visible spectrum, their reflectance is shown as three separate curves for R, G, and B. Copper’s R curve is highest, followed by G, and finally B (thus its reddish colour). Aluminum’s B curve is highest, followed by G, and finally R. (Image from “Real-Time Rendering, 3rd edition”, ©2008 A K Peters, by permission)
4.2.5 Roughness

Roughness, which refers to the microscopic shape characteristics of a surface, contributes to the way in which light is reflected off of a surface. A smooth surface may reflect incident light in a single direction, while a rough surface may scatter the light in various directions. Thus, variations in microscopic surface geometry can cause specular reflections to be scattered, blurring the image of the environment in an amount proportional to surface roughness. We need prior knowledge of the microscopic surface irregularities, or a model of the surface itself, to determine the reflection of incident light.

Possible surface models are divided into 2 categories: surfaces with exact known profiles and surfaces with random irregularities. An exact profile may be determined by measuring the height at each point on the surface by means of a sensor such as the stylus profilometer. This method tends to be cumbersome and impractical, hence, it is more reasonable to model the surface as a random process, where it is described by a statistical distribution of either its height above a certain mean level, or its slope with respect to its mean (macroscopic) slope. We use the second statistical approach as the representation of roughness.

In conclusion, the proposed model and representation of 3D reconstruction can be summarized in Table 4.3.

<table>
<thead>
<tr>
<th>Model</th>
<th>Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nature of scene</td>
<td>Static</td>
</tr>
<tr>
<td>Lighting</td>
<td>Mixed: ambient, projector, light sources</td>
</tr>
<tr>
<td>Vantage point</td>
<td>Medium: 10 - 50</td>
</tr>
<tr>
<td>Texture</td>
<td>Texture randomness</td>
</tr>
<tr>
<td>Lightness</td>
<td>Diffuse reflectance</td>
</tr>
<tr>
<td>Specularity</td>
<td>Specular (Fresnel) reflectance</td>
</tr>
<tr>
<td>Roughness</td>
<td>Distribution of microfacet normal</td>
</tr>
</tbody>
</table>

Table 4.3: A Model and corresponding representations of the 3D reconstruction problem.
4.3 Expression

In this section, we discuss the expression of 3D reconstruction problems using the proposed model and representations, as shown in Table 4.3. There are two related issues that need to be addressed: 1) how intuitive is the proposed model for humans to describe problem conditions; 2) how can a user estimate the parameters of properties. We first address the issue of property perception in Section 4.3.1. We can show that it is intuitive for humans to perceptually identify materials and estimate material properties. Next, we will use the proposed description to express the four problem conditions proposed in Chapter 3.

4.3.1 Perception of Properties

Different materials can be visually distinguished because they structure light in a particular, characteristic way. The way light is structured depends heavily on the shape of object, reflective and transmittive properties of material, and illumination field. This process is called the ‘forward optics’ (image formation) process. The material perception problem is, in some sense, the ‘inverse optics’ problem: determining what combination of surface geometry, surface material, and illumination field generated a given image [6]. Thus, in order to recover the reflectance properties of materials, the visual system must somehow disentangle the contributions of the illumination field and geometry. This is arguably the central problem of material perception, though the answer is currently far from clear [6]. But our intuition and many empirical studies support the view that humans are exceptional at material perception.

Everyday experience suggests that human’s visual system are extremely good at estimating material properties. We can effortlessly distinguish numerous different categories of material: textiles, stones, liquids, foodstuffs, and so on, and can recognize many specific materials within each class such as silk, wool and cotton [18]. Besides, being able to visually distinguish between materials and infer their properties by sight, is invaluable for many tasks. For instance, when determining edibility, we can make subtle visual judgments of material properties to determine whether fruit is ripe, whether soup has been left to go cold or whether bread is going stale [18]. There are numerous experimental evidences to support
this intuition. For example, Sharan et al. have shown that subjects can identify a wide range of materials from photographs even with brief presentations [60]. Fleming et al. showed subjects photographs of materials from different categories and asked them to rate various subjective qualities, such as hardness, glossiness and prettiness [20]. Even though subjects were not explicitly informed that the samples belonged to different classes, the subjective ratings of the individual samples were systematically clustered into categories, suggesting that subjects could theoretically classify materials through visual judgments of their properties.

There have been some empirical studies focused on visual estimation of specific properties of materials, such as glossiness, translucency or surface roughness. For instance, on the topic of glossiness, Nishida and Shinya showed that subjects can judge the specular reflectance of computer simulated glossy surfaces [50], and Fleming et al. showed that this ability generalizes across differences in lighting, as long as the illumination has statistical structure that is typical of the natural environment [19]. Fleming argued that the visual system does not actually estimate physical parameters of materials and objects, for instance, parameters of a reflectance model. Instead the brain is remarkably adept at building ‘statistical generative models’ that capture the natural degrees of variation in appearance between samples [18]. Though there is currently no universally accepted theory on visual perception of material, both our intuition and empirical results do suggest that human are exceptionally good at ‘estimating’ material properties.

Despite its subjective ease, material perception still poses the visual system with some unique and significant challenges, because a given material can take on many different appearances depending on the lighting, viewpoint and shape. Thus, we provide a generative approach to visual perception: a simulation software is used to generate images of a synthetic object with varied property settings to aid the estimation of properties. The user can choose an illumination field and object shape that is closest to the real-world environment and object. Then this “trial-and-error” approach is used to estimate parameters of properties. More specifically, the user would change the value of each property and see if the rendered result resembles the real object. A similar approach can be found in [10] where Berkiten and Rusinkiewicz used a synthetic dataset to find the contributing factors of various Photometric Stereo algorithms. We demonstrate this approach using the following
example shown in Figure 4.3: the brightness of the object is controlled by the albedo value, which is determined by the value channel of HSV colour space. To determine the specularity and roughness of the object, we experiment with varying parameters to get the most realistic image.

![Image](image)

**Figure 4.3:** The UI for determining the property settings, including albedo, specular, and roughness of the surface. As shown in (a), the problem condition in this case is: texture (0.8), albedo (0.8), specular (0.2), roughness(0.2). (b) demonstrates the effect of the property settings on a sphere, (c) on a teapot, and (d) shows the real-world object.

Now that we have proposed the model and representations of 3D reconstruction problem, we can express the four proposed problem conditions using this description. Given that all user perceived estimates would likely to be of low resolution, we use three discrete scales to set these properties: low (0.2), medium (0.5), and high (0.8). The expression of the reconstruction problem is shown in table 4.4.

<table>
<thead>
<tr>
<th>Object</th>
<th>Texture</th>
<th>Albedo</th>
<th>Specular</th>
<th>Rough</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1</td>
<td>low/med high</td>
<td>low/med high</td>
<td>Tl-B-D-R</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class 2</td>
<td>low/med high</td>
<td>high</td>
<td>low/med</td>
<td>Tl-B-M-S</td>
<td></td>
</tr>
<tr>
<td>Class 3</td>
<td>high</td>
<td>low/med high</td>
<td>T-B-D-R</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class 4</td>
<td>high</td>
<td>high</td>
<td>low/med</td>
<td>T-B-M-S</td>
<td></td>
</tr>
</tbody>
</table>

**Table 4.4:** Expression of the reconstruction problem for the four problem conditions proposed in Section 3.
4.4 Discussion and Conclusions

This chapter builds upon the problem space proposed in Chapter 3, and aims to provide a description to allow users to define problem condition of an object definitively. The description consists of two parts: a model and corresponding representations. The model answers the question of what characteristic features of an object to describe, and the representations address the issue of how to describe the selected set of features of an object.

We selected a subset of properties from the problem space as components of the model, and proposed corresponding representations for this model, as shown in Table 4.3. We further analyze how easy it is to perceptually estimate properties, and express the four problem conditions using the proposed description. The proposed description is the first layer of our three-layer description-based interface. It works as the input of interpreter, and from which, an algorithm that works reliably will be chosen to reconstruct a successful digital shape.

Since we only explore a subset of problem space, the proposed model and representations may be insufficient to describe some classes of object, especially objects with translucent, transparent, or refractive surfaces. However, the application of the description to a broader domain is limited by the availability of existing techniques. As long as reliably algorithms have been developed, we can extend the current description to incorporate complex reflective and geometric properties.
Chapter 5

A Mapping of 3D Reconstruction

Most vision work focuses on developing algorithm novelties, and as we have mentioned, very few investigate the rigorous conditions under which the algorithms themselves work. Thus, this knowledge is only known empirically, without a rigorous definition of the problem conditions. This relation between problem space and algorithms (termed as \textit{mapping}) is one of the key components of the interpreter, and is responsible for selecting one of the best possible algorithms based on described problem condition. The mapping is essentially a look-up table that returns a list of successful algorithms given a problem condition. This section builds upon the description proposed in Chapter 4, and attempts to find the problem conditions surrounding each algorithm empirically. To achieve this goal, we need to evaluate the performance of algorithms under varied problem conditions.

Two challenges need to be addressed, the first of which is to evaluate the performance of algorithms under a variety of problem conditions. This requires a dataset containing objects captured for between-category algorithms under a variety of problem conditions. To the best of our knowledge, there is no such benchmark available since most 3D benchmarks focus on one specific class of algorithms. For example, the Middlebury dataset targets MVS algorithms [59], and the ‘DiLiGenT’ dataset targets Photometric Stereo algorithms [61]. This makes such benchmarks only suitable for evaluation of within-category algorithms. Besides, there are few datasets with objects that cover a range of problem conditions. The reason for the lack of such a dataset is that it is practically impossible to create multiple versions
of the same object with one property, e.g., surface texture, material, and so on, varied while everything else is kept constant. In response to this challenge, we created a synthetic dataset using the physical-based rendering engine of Blender to evaluate the 3D reconstruction algorithms. Our dataset includes a collection of images of a scene under different materials and lighting conditions. The camera and projector’s intrinsic and extrinsic parameters are computed directly from the configurations of the synthetic setup, and the ground truth, including the 3D model point cloud and normal map, are generated directly from Blender.

The second challenge is: the problem space is an $N$-dimensional space, consisting of $N$ properties, each with $L$ levels, thus the total number of combinations is $L^N$, which increases exponentially as $N$ increases. This is too vast of a problem space to cope with. To make the problem space more feasible to handle, we adopt the three-point scale, where $L = 3$ (Low = 0.2, Medium = 0.5, and High = 0.8). Further, We conduct $\binom{N}{2} L \times L$ factorial studies to determine the properties that have a significant effect on performance of the algorithms. We can then reduce the space dimension by considering only these effective properties. Effective properties of an algorithm are those that have a main effect, interaction effect, or both on the performance of this specific algorithm. We illustrate the performance of algorithms under varied conditions as heatmaps so that the main effects and interactions between properties can be easily detected.

The chapter is organized as follows: section 5.1 discusses the selected algorithms, and the process of creating a synthetic dataset for the evaluation of selected algorithms. Section 5.2 discusses the procedure of identifying properties with a significant effect on algorithm performance so that the problem space can become more manageable. Section 5.3 presents the lookup tables, represented as heatmaps, from problem conditions to performance of algorithms, which is served as mapping from problem condition to algorithms.

5.1 Construction of Dataset

This section discusses the construction of a synthetic dataset that is used to evaluate between-class algorithms under varied problem conditions. First we choose three representative algorithms from three distinct algorithm classes, as well as two base-
line methods. Then the synthetic capturing systems are presented with synthetic images generated as examples. Lastly, quantitative measures used to evaluate the performance of algorithms are proposed.

5.1.1 Selected and Baseline Methods

We have selected three representative algorithms, each from one of three major classes of algorithms presented in Chapter 3: the PMVS proposed in [23], the example-based Photometric Stereo proposed in [29], and the Gray-coded Structured Light technique. See Table 5.1 for a summary of the selected algorithms. The current implementation of SL projects both column and row patterns, and depth values are computed using both patterns individually. A depth consistency step is performed to reject erroneous triangulation results.

<table>
<thead>
<tr>
<th>Technique</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMVS</td>
<td>Patch-based, seed points propagation MVS.</td>
</tr>
<tr>
<td>EPS</td>
<td>Example-based Photometric Stereo.</td>
</tr>
<tr>
<td>GSL</td>
<td>Gray coded Structured Light technique.</td>
</tr>
<tr>
<td>VH</td>
<td>Volumetric Visual Hull.</td>
</tr>
<tr>
<td>LLS-PS</td>
<td>Linear least squares Photometric Stereo.</td>
</tr>
</tbody>
</table>

Table 5.1: Summary of the selected and baseline algorithms implemented for the interface.

We use two baseline methods to compare our results: Visual Hull and a simple linear least squares based Photometric Stereo (LLS-PS). We use Visual Hull since it works relatively well as long as the silhouette of the object can be reliably extracted, thus being insensitive to material properties. In addition, the true scene is always enclosed by the reconstruction result, so the outcome is always predictable. We use LLS-PS to evaluate Photometric Stereo algorithms. However, there is currently no such PS algorithm that works reasonably well under a variety of conditions. Thus, we run this baseline algorithm under the optimal condition to ensure a best possible result.
5.1.2 Synthetic Setups

We use the physical-based rendering engine of Blender, Cycles, to generate the synthetic datasets. For each technique, the configuration of the camera remains fixed. The image resolution is 1280×720, with a focal length of 35 mm or 1400 pixels. The synthetic setups are shown in Table 5.2, and some example synthetic images generated using the setups are shown in Figure 5.1.

<table>
<thead>
<tr>
<th>Method</th>
<th>Hardware</th>
<th>Arrangement</th>
</tr>
</thead>
<tbody>
<tr>
<td>MVS</td>
<td>41 camera</td>
<td>5 rings, each has 1, 8, 8, 12, 12 camera</td>
</tr>
<tr>
<td>PS</td>
<td>1 camera+25 light sources</td>
<td>4 rings, each has 1, 8, 8, 8, 8 light sources</td>
</tr>
<tr>
<td>SL</td>
<td>1 camera&amp;projector</td>
<td>baseline angle: 10°</td>
</tr>
</tbody>
</table>

Table 5.2: Summary of synthetic capturing systems for three classes of algorithms.

The effects of properties simulated by the rendering engine are shown in Figure 5.1.

![Texture](image1)
![Albedo](image2)
![Specular](image3)
![Roughness](image4)

Figure 5.1: Example synthetic images. The value of each property ranges from 0 to 1.

5.1.3 Quantitative Measures

We use the metrics proposed in [59] to evaluate MVS and SL algorithms. More specifically, we compute the accuracy and completeness of the reconstruction. For accuracy, the distance between the points in the reconstruction $R$ and the nearest points on ground truth $G$ is computed, and the distance $d$ such that $X\%$ of the points on $R$ are within distance $d$ of $G$ is considered as accuracy. A reasonable $d$ value
is between \([3,5]\) mm, and \(X\) is set as 95. The lower the accuracy value, the better the reconstruction result. For completeness, we compute the distance from \(G\) to \(R\). Intuitively, points on \(G\) are not “covered” if no suitable nearest points on \(R\) are found. A more practical approach computes the fraction of points of \(G\) that are within an allowable distance \(d\) of \(R\). Note that as the accuracy improves, the “accuracy value” goes down, whereas as the completeness improves, the “completeness value” goes up.

For photometric stereo, depth information is lost since only one viewpoint is used. Thus, the previous metrics are not applicable. Here we employ another evaluation criteria that is widely adopted, which is based on the statistics of angular error. For each pixel, the angular error is calculated as the angle between the estimated and ground truth normal, i.e., \(\arccos(n_g^T n)\), where \(n_g\) and \(n\) are the ground truth and estimated normals respectively. In addition to the mean angular error, we also calculate the standard deviation, minimum, maximum, median, first quartile, and third quartile of angular errors for each estimated normal map.

### 5.2 Reduction of Problem Space Dimension

The greatest challenge in constructing a mapping from problem space to algorithms is the large variations in shapes and material properties, which results in a problem space that is too large to cope with. Suppose there are \(N\) properties, each with \(L\) discrete levels, then there are in total \(L^N\) different problem conditions. Thus, the first step, discussed in Section 5.2.1, 5.2.2, 5.2.3, is to reduce the dimensions of problem space by discovering the properties that have effects on performance of algorithms.

This study is a \(3 \times 3\) factorial design with two properties (factors), each with three levels, i.e., low (0.2), medium (0.5), and high (0.8), see Table 5.3. We are interested in one-way interaction (main effect) and two-way interaction of the properties (factors). A main effect is the effect of one of the independent variables on the dependent variable, ignoring the effects of all other independent variables. An interaction occurs when the effect of one independent variable on the dependent variable changes depending on the level of another independent variable. In our current design, this is equivalent to asking whether the effect of property \(i\) changes
depending on property \( j \), where \( i \neq j, i, j \in \{1, 2, 3, 4\} \). The easiest way to communicate an interaction is to discuss it in terms of the simple main effects, which is defined as the main effect of one independent variable (e.g., property \( i \)) at each level of another independent variable (e.g., property \( j \)). We observe an interaction between two factors whenever the simple effects of one change as the levels of the other factor are changed. Assume that the error variance is small, so that differences in performance that are apparent on the graph are also statistically significant. Thus, we are able to interpret the main effects and interactions through graphs. There is a main effect of a property if there is colour variation along the corresponding axis. If colour changes monotonically along the diagonal, then there is no interaction between the two properties, otherwise, there is an interaction effect.

\[
\begin{array}{c|ccc}
\text{Property } i & 0.2 & 0.5 & 0.8 \\
\hline
\text{Property } j & 0.2 & 0.5 & 0.8 \\
\end{array}
\]

**Table 5.3:** This is a \( 3 \times 3 \) factorial design. Every two properties are selected to test the main effects and interaction, there are in total \( \binom{N}{2} \) combinations.

### 5.2.1 Reduction of Problem Space Dimension: PMVS

We study the main effects and interaction effects of properties on the performance of PMVS in terms of accuracy and completeness. The performance of the algorithm is visualized in Figure 5.2.

(a) **Texture and Albedo** The main effects of texture and albedo on accuracy, and the main effect of albedo on completeness are not significant whereas the main effect of texture on completeness is significant such that surfaces with higher texture leads to results of higher completeness than less textured surfaces. There is not significant interaction effect between texture and albedo on either accuracy or completeness.

(b) **Texture and Specularity** The main effects of texture and specularity on both accuracy and completeness are significant such that surfaces with lower tex-
(a). Texture and albedo

(b). Texture and specularity

(c). Texture and roughness

(d). Albedo and specularity

(e). Albedo and roughness

(f). Specularity and roughness

Figure 5.2: Performance of PMVS under six conditions. For instance, (a) shows the performance under the condition of changing texture and albedo levels, while the others are fixed. The main effect of a property is illustrated by the colour variation along the corresponding axis. The monotonic colour variation diagonally indicates no interaction between the two properties, otherwise, there is an interaction effect. Thus in (a), we observe a main effect of texture on completeness, no other main effects and interaction effects are present.
ture or surfaces with higher specularity leads to higher accuracy value. However, we argue that this main effect are caused by the interaction effect between texture and specularity. As we can see, the effect of specularity on both accuracy and completeness are less noticeable for lowly and highly textured surfaces whereas the effect of specularity is most substantial for surfaces with medium texture. This effect can be explained as follows: the specular lobe can only be observed by cameras positioned and oriented towards the specular lobe, such as camera $V_2$ shown in Figure 5.3 (a) and (c). Cameras positioned otherwise would observe the true surface, such as camera $V_1$ shown in Figure 5.3 (a) and (b). The algorithm would then exploit the texture information provided by views like $V_1$, and thus would be able to reconstruct a specular surface.

![Figure 5.3](image)

**Figure 5.3:** (a) shows the reflection of light off a specular surface. $V_1$ received the diffuse component while $V_2$ receives the specular component. (b), (c) shows the images observed from these two views. The specular area (red circle) observed in $V_2$ is visible in $V_1$.

(c) **Texture and Roughness** The main effects of texture and roughness, and that of roughness on completeness are not significant whereas the main effect of texture on completeness is significant such that surfaces with higher textures leads to results with higher completeness. There is no significant interaction effect between texture and roughness on either accuracy or completeness.

(d) **Albedo and Specularity** The main effects of albedo and specularity on accuracy is not significant whereas the main effects of albedo and specularity on completeness are significant such that surfaces with higher albedo or lower specularity leads to higher completeness. There is no significant interaction effect between albedo and specularity in terms of accuracy and completeness as the value varies monotonically along the diagonal.

(e) **Albedo and Roughness** The main effects of albedo and roughness on ac-
accuracy and completeness are not significant. There is no interaction effect between albedo and roughness on either accuracy or completeness.

(f) Specularity and Roughness The main effects of specularity and roughness on accuracy and completeness are not significant. There is no interaction effect between specularity and roughness on either accuracy or completeness.

Summary: PMVS
For accuracy, specularity has a main effect such that higher specularity leads to higher accuracy value. There are no main effects for other properties. There is a significant effect between texture and specularity such that the effect of specularity is most substantial on surfaces with medium texture. There is also a significant interaction effect between albedo and specularity such that specularity is most substantial on surfaces with low albedo value.

For completeness, there is a significant main effect from texture, and no other main effects observed. There is a significant interaction effects between texture and specularity on completeness such that the negative effect of specularity is most significant on surfaces with medium level texture. There are no other significant interaction effects observed.

5.2.2 Reduction of Problem Space Dimension: EPS
We study the main effects and interaction effects of properties on the performance of EPS in terms of mean and standard deviation (SD) of angular error. The performance of the algorithm is visualized in Figure 5.4.

(a) Texture and Albedo The main effects of texture on mean and SD of angular error, and the main effect of albedo on SD of angular error are not significant. The main effect of albedo on mean value of angular error is significant such that the angular error decreases as the albedo increases. There is no significant interaction effect between texture and albedo on mean and SD of angular error.

(b) Texture and Specularity The main effects of texture on mean and SD of angular error are not significant whereas the main effects of specularity are significant such that both values increases as specularity increases. There is no significant interaction effect between texture and specularity in terms of mean and SD of an-
Figure 5.4: Performance of Example-based PS under six problem conditions. For instance, (a) shows the performance under the condition of changing texture and albedo levels, while the others are fixed. The main effect of a property is illustrated by the colour variation along the corresponding axis. The monotonic colour variation diagonally indicates no interaction between the two properties, otherwise, there is an interaction effect. Thus in (a), we observe a main effect of albedo on mean angular error, no other main effects and interaction effects are present.
angular error.

(c) **Texture and Roughness** The main effects of texture on mean and SD of angular error, and the main effect of roughness on SD of angular error are not significant whereas the main effect of roughness on mean of angular error is significant such that the mean angular error decreases as roughness increases. There is no interaction between texture and roughness in terms of mean and SD of angular error.

(d) **Albedo and Specularity** The main effects of albedo and specularity on mean and SD of angular error are significant such that the mean and angular error decrease as albedo increases or specularity decreases. There is also a significant interaction effect between albedo and specularity such that the effect of specularity on mean and SD of angular error is most significant when the surface albedo is low.

(e) **Albedo and Roughness** The main effects of albedo and roughness on mean of angular error are significant such that the mean angular error decreases as the albedo or roughness increases whereas the main effects on SD of angular error are not significant. There is no significant interaction effect between albedo and roughness.

(f) **Specularity and Roughness** The main effect of specularity on mean and SD of angular error is significant such that the values vary as the specularity changes. There is an interaction effect between specularity and roughness. More specifically, the mean and SD of angular error do not decrease monotonically as the roughness increases. More specifically, the angular error becomes worse for surfaces with medium roughness, which is counter-intuitive at first sight. However, we argue that this is because the roughness is not strong enough to counteract the specular highlights, causing a smoothed and blurred specular region with larger area, thus leading to a poorer normal estimation. See Figure 5.5 for visual illustrations.

**Summary: EPS**

Albedo, specularity, and roughness all have a main effect on angular error. More specifically, higher albedo and roughness lead to lower mean angular error, higher specularity leads to higher mean and SD angular error. There is a significant interaction effect between albedo and specularity such that the effect of specularity is

55
Figure 5.5: An example illustrates the effect of roughness on PS. Albedo is set as 0.8, and specular is set as 0.8. The first column shows the input images, the second column shows the estimated normal map, the third column shows the integrated surface, and last column shows the angular error. We can see from the qualitative results (normal map and height map), and quantitative result (angular error) that a medium level roughness would lead to a worse normal estimation since it blurs the specular lobe.
most significant on low albedo surfaces. There is also an interaction between specularity and roughness such that surfaces with medium roughness lead to higher angular error compared to surface with low or high roughness.

5.2.3 Reduction of Problem Space Dimension: GSL

We interpret the main effects and interactions of properties on the performance of GSL in terms of accuracy and completeness. The performance of the algorithm is visualized in Figure 5.6.

(a) Texture and Albedo The main effect of texture on accuracy and completeness and the main effect of albedo on accuracy are not significant. The main effect of albedo on completeness is significant such that the results increases as the albedo increases. There is no significant interaction effect between texture and albedo in terms of accuracy and completeness.

(b) Texture and Specularity The main effect of texture on accuracy and completeness and the main effect of specularity on accuracy are not significant. The main effect of specularity on completeness is significant such that the results decreases as the specularity increases. There is no significant interaction effect between texture and specularity in terms of accuracy and completeness.

(c) Texture and Roughness The main effect of texture on accuracy and completeness and the main effect of roughness on accuracy is not significant. The main effect of roughness on completeness is significant such that the results increases as the roughness increases. There is no significant interaction effect between texture and roughness in terms of accuracy and completeness.

(d) Albedo and Specularity The main effects of albedo and specularity on accuracy are not significant whereas the main effects on completeness are significant such that the results increase as albedo increase or specularity decreases. There is no significant interaction effect between albedo and specularity in terms of accuracy and completeness.

(e) Albedo and Roughness The main effects of albedo and roughness on accuracy, and the main effect of roughness on completeness are not significant. The main effect of albedo on completeness is significant such that the results increases as the albedo increases. There is no significant interaction effect between albedo
Figure 5.6: Performance of Gray-coded SL under six problem conditions. For instance, (a) shows the performance under the condition of changing texture and albedo levels, while the others are fixed. The main effect of a property is illustrated by the colour variation along the corresponding axis. The monotonic colour variation diagonally indicates no interaction between the two properties, otherwise, there is an interaction effect. Thus in (a), we observe a main effect of albedo on completeness, no other main effects and interaction effects are present.
and roughness in terms of accuracy and completeness.

(f) Specular and Roughness

The main effects of specularity and roughness on accuracy are not significant whereas the main effect on completeness are significant such that the results increases as the roughness increases or specularity decreases. There is no significant interaction effect between specularity and roughness in terms of accuracy and completeness.

Summary: GSL

There is no main effects or interaction effects observed on the algorithm in terms of accuracy. Albedo, specularity, and roughness all have main effects on the algorithm in terms of completeness. There are no interaction effects observed.

5.3 Construction of Mapping

In the previous section, we have examined the performance of algorithms with two changing properties at a time. This is equivalent to examine the performance of algorithms on a 2-dimensional plane embedded in a \( N \)-dimensional space. It gives us insights into which properties have significant impacts on the performance of algorithms. In this section, we examine the problem conditions consisting of only properties that have significant main or interaction effects on the algorithms. This is a much more feasible problem since only a subset of all \( N \) properties have a significant effect on a specific algorithm. The result is a one-to-many mapping from problem condition to algorithms. To determine if an algorithm achieves a successful reconstruction, we compare the quantitative results to those of the baseline methods. More specifically, an algorithm is considered as a successful candidate if it achieves better reconstruction result than that of baseline methods in terms of quantitative measures, such as accuracy, completeness, and angular error. To illustrate the results, all quantitative measures of baseline methods are subtracted from those of selected algorithms, the results of which are visualized using heatmaps, as shown in Figure 5.7. To keep the results consistent across all quantitative measures, i.e., positive value represents better result than the baseline methods, we inverse the results of accuracy and angular error. The mapping is essentially a look-up table that returns a list of successful algorithms given a problem condition,
and these heatmap graphs can be viewed as mapping from problem condition to algorithms: given a problem condition, the users can specify a threshold value $\epsilon$, which is the minimum tolerance of the quantitative measures between the selected algorithms and those of baseline methods. Any algorithm which is at least $\epsilon$ above the baseline methods are considered algorithms that can work reliably under the specified problem condition. By default, $\epsilon = 0$. Thus, positive value indicates better reconstruction, which indicates that the corresponding algorithm is a successful candidate. The mapping of the three selected algorithms are shown in Table 5.4.

5.4 Discussion and Conclusions

It is a non-trivial task to find a mapping from problem conditions to algorithms based on the description. By no means is the aforementioned approach the only way, or a perfect way, since it potentially has the problem of suffering from property scaling issue. Nonetheless, the factorial studies remains a valuable approach to obtain insights of the effective properties of a specific algorithm. The development of the mapping is an on-going process. For instance, we can include more quantitative metrics such as colour accuracy, ‘ghost’ reconstruction, and so on. In order to make the mapping applicable to objects with more complex shapes, we need to consider more sophisticated geometric properties besides roughness, such as concavity, depth-discontinuity, occlusion, etc. Furthermore, the incorporation of more algorithms is another way to ensure that the problem space is well covered.
Figure 5.7: Performance of PMVS, EPS, and GSL under all problem conditions. These are look-up tables that provide information regarding the performance of selected algorithms compared to baseline methods. Once a threshold value $\epsilon$ is specified, these look-up tables can be used as mapping from problem condition to algorithms, i.e., return successful algorithms given a problem condition.
<table>
<thead>
<tr>
<th>Texture</th>
<th>Albedo</th>
<th>Specular</th>
<th>Roughness</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>0.2</td>
<td>-</td>
</tr>
<tr>
<td>0.5</td>
<td>0.8</td>
<td>0.2</td>
<td>-</td>
</tr>
<tr>
<td>0.5</td>
<td>0.8</td>
<td>0.5</td>
<td>-</td>
</tr>
<tr>
<td>0.8</td>
<td>0.2</td>
<td>0.2</td>
<td>-</td>
</tr>
<tr>
<td>0.8</td>
<td>0.5</td>
<td>0.2</td>
<td>-</td>
</tr>
<tr>
<td>0.8</td>
<td>0.8</td>
<td>0.5</td>
<td>-</td>
</tr>
<tr>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 5.4: The problem conditions under which PMVS, EPS, and GSL work successfully in terms of the two metrics accuracy and completeness.
Chapter 6

An Interpretation of 3D Reconstruction

Now that we have proposed a well-defined problem space of 3D reconstruction problem, as well as a mapping from problem space to algorithms, the interpretation from the problem-centred description to reliable algorithms must be shown. There are many possible ways to interpret the description, our proof of concept interpreter is based on the direct evaluation of performance of each 3D reconstruction algorithm under different problem conditions presented in Chapter 5. This mapping from problem space to algorithms and additional constraints allow a successful algorithm to be definitively selected.

In this chapter, we first propose a proof of concept interpreter, which makes the three-layer description-based interface complete. By no means is the proposed interpreter the best possible interpreter, but it suffices to demonstrate the usage of the interface. The interpreter is responsible for receiving a description from the user, and selects an algorithm from a suite of algorithms. We are interested in evaluating the performance of the interpreter under the four problem conditions, which are summarized in Chapter 3. More specifically, we are interested in finding out whether the interpreter is able to translate a user-specified description into an algorithm so that a successful reconstruction result can be achieved. Further, we are also interested in the performance of the interpreter with less accurate, or inaccurate descriptions as inputs. It gives us insights into the sensitivity of the interpreter, and
demonstrates cases where it is crucial to provide accurate descriptions.

Three algorithms and two baseline methods that are introduced in Chapter 5 have been implemented and integrated into the interpreter, providing basic but well rounded reconstruction capabilities. Although more algorithms can potentially be added, these methods are sufficient to validate the interface’s ability to translate a description into a reconstruction solution. The integration of new algorithms that are more appropriate under particular situations requires only that they be evaluated by the same process discussed in Chapter 5, as shown in Figure 3.2. This allows researchers to contribute novel algorithms to the interpretation of the interface easily.

We created both synthetic and real-world datasets to evaluate the performance of the interpreter. The datasets is made publicly available to encourage the testing of additional reconstruction algorithms. Further, the datasets can be extended to include new visual and geometric properties, thus providing a wider coverage of the problem space.

This chapter is organized as follows: Section 6.1 provides a detailed roadmap of our evaluation that is centred around three key evaluation questions. Section 6.2 proposes a proof of concept interpreter. Section 6.3 gives an overview of the creation of dataset, including hardware calibration, and image capturing, and example images of synthetic and real-world dataset. Section 6.4 addresses the evaluation question by demonstrating the performance of interpreter under the four problem conditions using synthetic and real-world datasets, where a satisfactory reconstruction result is returned given the correct description of the object.

6.1 Evaluation Methodology

This thesis proposed a description-based interface to 3D reconstruction problem that hides algorithm details. This interface consists of three separate layers, a description, an interpreter, and an algorithm layer. The interpreter is responsible for selecting an appropriate algorithm for reconstruction based on user-specified description. The goal of our evaluation is to validate if the proposed proof of concept interpreter can translate a user-specified description into an algorithm that gives a successful reconstruction result. The evaluation is centred around the following
three evaluation questions:

1. Can the proof of concept interpreter return one of the best-suited algorithms that achieves a successful reconstruction given the correct description?

2. Will a less accurate description give a poorer reconstruction result than an accurate description?

3. Will an inaccurate description give a poor reconstruction result?

The first evaluation question address the issue of robustness, i.e., can the interpreter deal with objects with varied material and geometric variations. The other two questions address the issue of sensitivity, i.e., can the interpreter still perform reliably when the description is less accurate. The criteria of determining whether a reconstruction is successful are detailed below, along with the evaluation steps.

6.1.1 Criteria

How to determine if the reconstruction achieved by the algorithm selected by the interpreter is successful? In this thesis, visual inspection is utilized to determine the quality of reconstruction. More specifically, the result returned by the interface is compared to that of the baseline algorithms to determine if the quality is acceptable. As previously mentioned, the baseline method is chosen so that it always can provide a decent reconstruction under most circumstances.

The reason that qualitative comparison is sufficient is that we are less interested in developing novel algorithms or improving algorithm performance. If that is the goal, we do need a quantitative comparison among algorithms. However, the goal is to validate if the proposed interface can translate a description into an acceptable result, thus we are more interested in showing that this user-specified description can indeed be interpreted and invoke a good-enough algorithm. Instead of computing quantitative values, such as accuracy, completeness, and angular error, we examine visual appearances that represent these quantitative measures: the roughness of reconstructed surface indicates accuracy, the lack of surface holes indicates completeness, and surface spike indicates large angular error, see Figure 6.1.
Figure 6.1: Visual phenomena that indicate poor quality of reconstruction results. (a) has rough surface which indicates poor accuracy, (b) has incomplete holes which indicates poor completeness, and (c) has spikes which indicates poor normal estimation.

6.2 Interpreter

Our interface consists of three separate layers. The upper layer is the description of the problem condition. The middle layer is the interpreter which receives a description from the user and returns an acceptable result. The bottom layer is the actual implementation of the algorithms. The interpreter is responsible for choosing an appropriate 3D reconstruction algorithm based on the description of the problem domain and additional requirements. Thus, it requires an understanding of algorithm performance across different ranges of problem space. However, if the interpreter relies solely on the mapping, it is possible that a user-specified description leads to more than one successful algorithm. We propose to use two additional constraints so that only one appropriate algorithm is selected.

The first constraint is metric-first or shape-first. There are generally two types of reconstruction results: euclidean/metric reconstruction, and shape reconstruction. The former reconstructs a scene with metric information, but generally gives noisier results. The latter can achieve quality that is only matched by laser scanners but lack the depth information. The default setting of this constraint is metric-first.

The second constraint is accuracy-first or completeness-first. Methods that achieve high accuracy do not necessarily achieve high completeness. In light of this, the user can choose an algorithm based on the priority level of accuracy and completeness. If multiple algorithms with comparable accuracy or completeness exist, we use a simple ranking system: if one algorithm has the same accuracy but is more complete, or more complete and equally accurate, it is chosen over the
others. The default is the accuracy-first constraint.

Thus, the proposed proof of concept interpreter that consists of two components, mapping and constraints, as shown in Figure 1. Note that there are many ways of using this mapping information to create an interpreter, and by no means is the proposed one the optimal interpreter.

Algorithm 1 Proof-of-concept interpreter

Require: Description desc
Ensure: A successful algorithm bestAlgo returned

\{	ext{algos}\} \leftarrow \text{MAPPING(desc)}
while \{	ext{alogs}\} \neq \emptyset do
algo \leftarrow \text{Top}\{\text{alogs}\}, \text{Pop}\{\text{alogs}\}
if Euclidean-first then
  if Accuracy-first then
    score \leftarrow \text{QUANTSCORE(desc, algo, Accuracy)}
  else if Completeness-first then
    score \leftarrow \text{QUANTSCORE(desc, algo, Completeness)}
  end if
else if Shape-first then
  score \leftarrow \text{QUANTSCORE(desc, algo, AngularError)}
end if
if score < bestScore then
  bestScore \leftarrow score, bestAlgo \leftarrow algo
end if
end while

function \text{MAPPING(description)}
  return A list of successful algorithms \{\text{algorithms}\}.
end function

function \text{QUANTSCORE(description, algorithm, type)}
  if type == Accuracy then
    return Accuracy score.
  else if type == Completeness then
    return Completeness score.
  end if
  return Angular error. \quad \triangleright \text{Current support three quantitative measures.}
end function
6.3 Overview of Datasets

In this section, we introduce a synthetic and a real-world dataset to evaluate the performance of the interpreter under varied problem conditions. To the best of our knowledge, there is few publicly available dataset that capturing images for algorithms across different categories. Most existing datasets target a specific class of algorithms, such as the famous Middlebury dataset [59], DiLiGenT [61], DTU Robot Image Data Sets [33], and so on.

The synthetic dataset contains four objects, as shown in Figure 6.2, and the real-world dataset contains nine objects, four of which are shown in Figure 6.2. Please refer to Figure A.6, A.7 for the complete dataset. It covers materials that are representative of the four problem conditions proposed in Chapter 3: textureless-diffuse-bright (bust, statue), textureless-mixed-bright (vase0, cup), textured-diffuse-bright (barrel, pot), textured-mixed-bright (vase1, vase). In terms of surface shapes, we have focused mainly on objects with simple geometric properties, namely, smoothly curved surfaces with shallow concavities, see descriptions and example images in Figure 6.2.

![Representative objects of the four problem condition discussed in Chapter 3. (a)-(d): Synthetic objects, and (e)-(h) real-world objects.](image)

**Figure 6.2:** Representative objects of the four problem condition discussed in Chapter 3. (a)-(d): Synthetic objects, and (e)-(h) real-world objects.
6.3.1 Calibration

For MVS, a calibration pattern proposed in [42] is imaged under the object, which is used for calibrating the camera position and orientation by Structure from Motion software, such as VisualSfM [70]. The focal length is known a priori and remains fixed during the image capturing process. Thus the extrinsic parameters of the camera can be retrieved up to a similarity transformation, and the reconstruction result is a metric/euclidean reconstruction.

For most PS algorithms, i.e., calibrated PS algorithms, it is necessary to estimate the light direction and intensity. However, the selected PS algorithm can deal with unknown light sources and spatially-varying BRDFs. Thus, light calibration is not a required step. Though it is preferable to correct the non-linear response of camera, Hertzmann and Seitz discovered that it was unnecessary for EPS [29]. Thus, we did not perform the radiometric calibration step. No geometric calibration of the camera is needed.

For SL, an open source camera-projector calibration software developed by Moreno and Taubin is used for calibration [47]. This technique works by projecting temporal patterns onto the calibration pattern, and uses local homography to individually translate each checker board corner from the camera plane to the projector plane. This technique can estimate both the intrinsic parameter and camera and projector, and the relative position and orientation.

6.3.2 Image Capturing

The creation of the synthetic dataset is mostly the same as that discussed in Chapter 5, thus is omitted in this section. The images of real-world dataset are captured using a Nikon D70S camera with a 18 – 70mm lens. The image resolution is 3004 × 2000 pix. The setup of the capturing systems are shown in Figure 6.3.

For MVS, we capture the dataset by positioning the camera in three different heights. The objects are about 30 – 50cm away from the camera, and stays fixed on a turntable. We have followed the following two steps to acquire data: 1) put the camera at a different height, adjust the orientation so that the object is at the centre of the frame; 2) take pictures while rotating the table. The table rotates approximately 30° every time. We rotate it 12 times and in total, we can obtain 36
images per object.

For PS, a 70 – 200mm lens, a hand held lamp, and two reference objects (diffuse and glossy) are used. The objects are positioned about 3m from the camera to approximate orthographic projection. To avoid inter-reflection, all data are captured in a dark room with everything covered by black cloth except the target object. We use a hand-held lamp as the light source and choose close to frontal viewpoints to avoid severe self-shadowing effect. We take 20 images per object and select 15 plus images depending on the severity of the self-shadow effect.

For SL, we use a Sanyo Pro xtraX Multiverse projector with a resolution of 1024 × 768. The baseline angle of the camera projector pair is approximately 10°. To alleviate the effect of ambient light, all images are captured with room lights off. To counteract the effect of inter-reflection, additional images are captured by projecting an all-white and all-black patterns.

(a). MVS, VH  
(b). PS  
(c). SL

Figure 6.3: Image capturing systems for MVS/VH, PS, and SL.

6.3.3 Synthetic Dataset

Each object in the synthetic dataset represents one of the four problem conditions discussed in Section 3. The description of problem condition of each synthetic object is shown in Figure 6.4, as well as the appropriate algorithm selected by the interpreter. Given a user specified description, the proof of concept interpreter will select an algorithm, and any object that matches this description should be well reconstructed by this selected algorithm.
Figure 6.4: The representatives of the four classes of objects used for evaluation. Example images and problem conditions of synthetic objects are in the first two rows. The correct description of the problem condition of corresponding object is presented in third row. The last row shows the algorithms returned by the mapping, from which the interpreter selects one successful algorithm, which is coloured in red.

### 6.3.4 Real-World Dataset

We select four real-world objects, each represents the one of the four problem conditions proposed in Chapter 3, as shown in Figure 6.5. The descriptions of problem conditions of real-world objects are shown in Figure 6.5, as well as the appropriate algorithm selected by the interpreter. Given a user specified description, the proof of concept interpreter will select an algorithm, and any object that matches this description should be well reconstructed by this selected algorithm.

### 6.4 Evaluation of Interpreter

This section evaluates the proof of concept interpreter, which is three-fold: 1) given an accurate description, can the interpreter select an algorithm that achieves a successful reconstruction result; 2) given an less accurate description, would the interpreter perform poorer than given an accurate description; 3) given an inaccurate
<table>
<thead>
<tr>
<th>Class #</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects</td>
<td>![Objects Image 1]</td>
<td>![Objects Image 2]</td>
<td>![Objects Image 3]</td>
<td>![Objects Image 4]</td>
</tr>
<tr>
<td>Condition</td>
<td>• textureless</td>
<td>• textureless</td>
<td>• textured</td>
<td>• textured</td>
</tr>
<tr>
<td></td>
<td>• diffuse</td>
<td>• mixed</td>
<td>• diffuse</td>
<td>• mixed</td>
</tr>
<tr>
<td></td>
<td>• bright</td>
<td>• bright</td>
<td>• dark/bright</td>
<td>• dark/bright</td>
</tr>
<tr>
<td>Tex</td>
<td>0.2</td>
<td>0.2</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>Alb</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8 (0.2)</td>
<td>0.8 (0.2)</td>
</tr>
<tr>
<td>Spec</td>
<td>0.2</td>
<td>0.8</td>
<td>0.2</td>
<td>0.8</td>
</tr>
<tr>
<td>Rough</td>
<td>0.8</td>
<td>0.2</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>Alg</td>
<td>• EPS</td>
<td>• EPS</td>
<td>• PMVS</td>
<td>• PMVS</td>
</tr>
<tr>
<td></td>
<td>• GSL</td>
<td>• GSL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 6.5:** The representatives of the four classes of objects used for evaluation. Example images and problem conditions of real-world objects are in the first two rows. The correct description of the problem condition of corresponding object is presented in third row. The last row shows the algorithms returned by the mapping, from which the interpreter selects one successful algorithm, which is coloured in red.

description, would the interpreter fail to achieve a successful reconstruction. We provide demonstrative results of the interpreter using the synthetic and real-world dataset, as shown in Figure 6.2.

### 6.4.1 Evaluation 1: Accurate Description, Successful Result

In this section, we evaluate the performance of interpreter given a valid description, i.e., whether the interpreter can return a successful reconstruction result given a valid description of problem condition. We provide the detailed process of reconstructing objects using the proposed interface with a example, and show the reconstruction results of the aforementioned datasets.

Let’s revisit the scenario proposed in Chapter 1 using the test objects ‘bust’ and ‘statue’ as an example. First, the user describes the problem condition based on visual and geometric properties. In the example, the description of object ‘bust’
and ‘statue’ is: 0.2 (texture), 0.8 (albedo), 0.2 (specularity), and 0.8 (roughness). The synthetic objects with the same problem condition are shown in Figure 6.6.

![Figure 6.6](image)

**Figure 6.6:** (a) shows the problem condition of object ‘bust’ and ‘statue’ specified in the 3D software. (b) and (c) shows the synthetic object rendered using the problem condition in (a).

Next, the interpreter, consisting of mapping and constraints, selects an appropriate algorithm that could result in a successful reconstruction given the user-specified description. This is achieved by using the mapping discovered in Chapter 5, and additional constraints to select a reliable algorithm for reconstruction, as shown in Figure 6.7. In this example, GSL is selected as the interpreted algorithm for reconstruction.

![Figure 6.7](image)

**Figure 6.7:** The interpreter selects an appropriate algorithm based on description. Based on the discovered mapping from Chapter 5, EPS and GSL can achieve a successful reconstruction, which is labelled in green while PMVS would fail to give successful result, which is labelled in red.

Lastly, the interpreted algorithm accepts images captured using the appropriate capturing system as input, and returns a reconstruction result. We can see that
the reconstructed model of the interpreted algorithm has a much smoother surface than that of the baseline method, which indicates greater accuracy, as shown in Figure 6.8. Thus, we conclude that the interpreted algorithm reconstructs the object successfully.

![Figure 6.8](image)

**Figure 6.8:** (a), (c) show the reconstruction results of the interpreted algorithm, GSL. (b), (d) show the results of the baseline method.

For more results, please refer to Figure 6.9. The figure is organized as follows: ‘Desc’ row shows the description of the objects using the proposed model and representations, the ‘Algo’ row shows the algorithm selected by the interpreter. The ‘Results’ row shows the reconstruction results of the corresponding algorithm, and the ‘Baseline’ row demonstrate the results of the baseline method. As discussed previously, we utilize visual phenomena, such as surface roughness, holes, spikes as indicators of the reconstruction quality. The baseline method achieves decent reconstruction results on all objects. Though due to the resolution of the voxel grids, the surfaces are relatively rough. Further, the surface concavities fail to be carved out for concave objects (Bust). The performance of the interpreter on both synthetic and real-world datasets are consistent in that the algorithms selected by the interpreter consistently outperform the baseline technique. All results have much smoother reconstructed surfaces, especially those of EPS and GSL, which is an indicator that the accuracy of the results are much higher using the selected algorithm. Further, erroneous results, such as surface holes, spikes are absent from the reconstruction results, which indicates that the selected algorithms can achieve completeness, and angular error no worse than the baseline.

Note that the selected objects do no favour any algorithm. The significance of this study is to demonstrate that it is achievable to translate a user-specified description, which has nothing to do with algorithm-level knowledge, to an algorithm
selected from a suite of algorithms and achieve a successful reconstruction. There is no need to understand the underlying algorithms, or set the obscure algorithm specific parameters.

<table>
<thead>
<tr>
<th>AlgoDesc</th>
<th>Results</th>
<th>Synthetic and real-world objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>02080208</td>
<td>GSL</td>
<td>Bust&amp; Statue</td>
</tr>
<tr>
<td>02080802</td>
<td>EPS</td>
<td>Vase0&amp; Cup</td>
</tr>
<tr>
<td>08080202</td>
<td>GSL</td>
<td>Barrel&amp; Pot</td>
</tr>
<tr>
<td>08080802</td>
<td>PMVS</td>
<td>Vase1&amp; Vase</td>
</tr>
</tbody>
</table>

**Figure 6.9:** Evaluation 1: correct description leads to successful reconstruction result. The baseline results are provided so that we can determine the quality of result returned by the algorithm chosen by the interpreter.

### 6.4.2 Evaluation 2: Less Accurate Description, Less Successful Result

We have demonstrated that the interpreter can achieve successful results given accurate description in last section. However, how would the interpreter perform given a less accurate description? There are two guesses: 1) the interpreter would return a poor result given a less accurate description; 2) the interpreter can still
achieve successful results under some specific circumstances. In this section, we are interested to find out how sensitive the interpreter is given a less accurate description, and cases where it would succeed or fail. We provide four less accurate descriptions by iterating through four properties with one and only one property set incorrectly each time. For instance, prop \( i \) is incorrectly estimated for desc \( i \) while the rest of the properties are set correctly, \( i \in \{1, 2, 3, 4\} \). Since a description could be mapped to multiple algorithms, a description that does not match the object could potentially result in a successful reconstruction result as well. We demonstrate the cases where less accurate descriptions lead to poor or successful results, and discuss the underlying rationale.

Let’s revisit the scenario proposed in Chapter 1 using the test object ‘vase0’: the user (Daisy) first sets the parameters as follows: 0.2 (texture), 0.8 (albedo), 0.8 (specularity), and 0.8 (roughness). However, the interpreted algorithm in this case, GSL, fails to reconstruct the glossy area on the surface, resulting in an incomplete result, as shown in 6.10 (a). The user then observes that the surface is smooth enough that the highlighted area is not blurry at all, thus decides to tweak roughness from 0.8 to 0.2. The interpreter selects EPS as the reconstruction algorithm, and then proceeds to achieve a smooth and complete reconstruction result, see Figure 6.10 (b). We can see from this example that less accurate description could lead to less successful reconstruction result.

![Figure 6.10](image.png)

**Figure 6.10:** (a) shows the reconstruction using the incorrect description while (b) shows the reconstruction result using the correct description. The text labelled in red represents incorrectly estimated property.

However, this might not always be the case. Let’s use object ‘bust’ as an example: given four less correct descriptions with exactly one property estimated incorrectly, the algorithm selected by the interpreter can always achieve a success-
ful reconstruction result, see Figure 6.11. We conclude from this example that it is achievable to obtain a successful reconstruction result given a less accurate description.

(a). 08080208 (b). 02020808 (c). 02080808 (d). 02080202 (e). 02080208

Figure 6.11: (a)-(d) show the reconstruction results given a less accurate description while (e) shows the result given an accurate description. The text labelled in red represents the property estimated incorrectly.

The question then becomes, under what conditions, the interpreter can achieve a successful or poor reconstruction result given a less accurate description. Let’s denote the algorithm(s) returned by the mapping given description Desc\(_i\) as \(\{\text{Algo}_i\}\), and the algorithm selected by the interpreter as \(\{\text{Interp}_i\}\). For instance, the algorithms for object \textit{bust} given description Desc\(_1\) is \(\{\text{Algo}_1\} = \{\text{PMVS}, \text{EPS}, \text{GSL}\}\), and \(\{\text{Interp}_1\} = \{\text{GSL}\}\). Let’s also denote the algorithm(s) returned by the correct description as \(\{\text{Algo}\}\). Thus in the case of object \textit{bust}, \(\{\text{Algo}\} = \{\text{EPS}, \text{GSL}\}\). We observe that a less accurate description can possibly lead to a successful reconstruction if and only if

\[
\{\text{Interp}_i\} \cap \{\text{Algo}\} \neq \emptyset
\]

The reason is that there are multiple algorithms that could work successfully for a specific object. Thus if an algorithm other than the interpreted one is chosen given a less accurate description, a successful reconstruction result is still achievable. Let’s use \textit{vase0} and \textit{bust} as an example. The algorithms returned from mapping are \(\{\text{Algo}_{\text{vase0}}\} = \{\text{EPS}, \text{GSL}\}\) given Desc\(_3\). The algorithms selected by the interpreter is \(\{\text{Interp}_3\} = \{\text{GSL}\}\). The algorithm(s) mapped from the correct description are \(\{\text{Algo}_{\text{vase0}}\} = \{\text{EPS}\}\) for \textit{vase0}, and \(\{\text{Algo}_{\text{bust}}\} = \{\text{EPS}, \text{GSL}\}\) for \textit{bust}, respectively. We can see that \(\{\text{Interp}_3\} \cap \{\text{Algo}_{\text{bust}}\} = \{\text{GSL}\}\), thus a less accurate description can lead to a successful result whereas \(\{\text{Interp}_3\} \cap \{\text{Algo}_{\text{vase0}}\} = \emptyset\), thus leading to an incomplete reconstruction result. The take-away message is that for an less accurate description, the reconstruction results are generally worse than
that of the interpreted result. However, for conditions that have multiple working algorithms, there may very well be an acceptable result.

For more results, please refer to Figure 6.12, 6.13. The layout of the graph is as follows: column wise, $\text{Desc}_i, i \in \{1, 2, 3, 4\}$ presents results where a less accurate description is provided while the last column presents the results where the correct description is provided. Row wise, for each object, the result is divided into three segments: The description of problem condition is presented in the first segment, which is coloured-cod to indicate the incorrectly set property. The order of properties in the description is: texture, albedo, specularity, and roughness. Since there is one and only one incorrectly estimated property in each less accurate description, it is coloured in red while the correctly estimated properties are coloured in black. The algorithms returned by the mapping are presented below the description in the second segment, with the algorithm chosen by the interpreter coloured in red. The reconstruction result using the interpreted algorithm is shown in the last segment of each section.

6.4.3 Evaluation 3: Inaccurate Description, Poor Result

We have demonstrated that given a less accurate description, the results may or may not be poor. In cases where the algorithm interpreted from a less accurate description overlaps with the set of algorithms mapped from an accurate description, it is still possible to achieve a successful reconstruction. In this section, we are interested to see if this conclusion can also be applied to completely inaccurate description, i.e., is it still possible to return a successful result. In this section, we evaluate whether the interpreter would return a poor reconstruction result given an inaccurate description of problem condition. We evaluate the performance of the interpreter using both synthetic and real-world objects.

Let’s revisit the scenario proposed in Chapter 1 using test object ‘vase0’: the user (Daisy) is interested to test how sensitive the interpreter is by providing completely incorrect description, i.e., 0.8 (texture), 0.2 (albedo), 0.2 (specularity), and 0.8 (roughness) in this case. The interpreter selects PMVS as the reconstruction algorithm, and then proceeds to achieve a noisy and incomplete reconstruction result, which is worse than that of the interpreted algorithm, see Figure 6.14.
<table>
<thead>
<tr>
<th>Object</th>
<th>Descriptions and Results</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Desc₁</td>
</tr>
<tr>
<td>Bust</td>
<td>08080208</td>
</tr>
<tr>
<td></td>
<td>• PMVS •</td>
</tr>
<tr>
<td></td>
<td>• EPS •</td>
</tr>
<tr>
<td></td>
<td><img src="image1.png" alt="Bust Image" /></td>
</tr>
<tr>
<td>vase₀</td>
<td>08080802</td>
</tr>
<tr>
<td></td>
<td>• PMVS •</td>
</tr>
<tr>
<td></td>
<td>• EPS •</td>
</tr>
<tr>
<td></td>
<td><img src="image6.png" alt="vase₀ Image" /></td>
</tr>
<tr>
<td>barrel</td>
<td>02080208</td>
</tr>
<tr>
<td></td>
<td>• EPS •</td>
</tr>
<tr>
<td></td>
<td>• GSL •</td>
</tr>
<tr>
<td></td>
<td><img src="image11.png" alt="barrel Image" /></td>
</tr>
<tr>
<td>vase₁</td>
<td>02080802</td>
</tr>
<tr>
<td></td>
<td>• EPS •</td>
</tr>
<tr>
<td></td>
<td>• EPS •</td>
</tr>
<tr>
<td></td>
<td><img src="image16.png" alt="vase₁ Image" /></td>
</tr>
</tbody>
</table>

**Figure 6.12:** Evaluation 2: less accurate description may lead to poor result. Descᵢ represents inaccurate descriptions. For each object, the first row represent the description, with the incorrectly estimated property coloured in red while the correct ones in black. The algorithms determined by mapping are below the description with the algorithm selected by interpreter coloured in red (BL: baseline). The last row shows the corresponding reconstruction results.
<table>
<thead>
<tr>
<th>Object</th>
<th>Descriptions and Results</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Desc&lt;sub&gt;1&lt;/sub&gt;</td>
</tr>
<tr>
<td>statue</td>
<td>08080208</td>
</tr>
<tr>
<td></td>
<td>PMVS</td>
</tr>
<tr>
<td></td>
<td>EPS</td>
</tr>
<tr>
<td>cup</td>
<td>08080802</td>
</tr>
<tr>
<td></td>
<td>PMVS</td>
</tr>
<tr>
<td></td>
<td>EPS</td>
</tr>
<tr>
<td>pot</td>
<td>02080208</td>
</tr>
<tr>
<td></td>
<td>EPS</td>
</tr>
<tr>
<td></td>
<td>GSL</td>
</tr>
<tr>
<td>vase</td>
<td>02080802</td>
</tr>
<tr>
<td></td>
<td>EPS</td>
</tr>
<tr>
<td></td>
<td>EPS</td>
</tr>
<tr>
<td></td>
<td>GSL</td>
</tr>
</tbody>
</table>

**Figure 6.13:** Evaluation 2: less accurate description may lead to poor reconstruction results. Desc<sub>i</sub> represents inaccurate descriptions. For each object, the first row represent the description, with the incorrectly estimated property coloured in red while the correct ones in black. The algorithms determined by mapping are below the description with the algorithm selected by interpreter coloured in red (BL: baseline). The last row shows the corresponding reconstruction results.
Figure 6.14: (a) shows the reconstruction result given an incorrect description whereas (b) shows the result given a correct description.

For more results, please refer to Figure 6.15. The figure is divided into two sections: one for inaccurate description and one for accurate description. The inaccurate description is property-wise opposite to the corresponding accurate description. The mapped algorithms are shown below the description, with the algorithm selected by the interpreter coloured in red. The reconstruction result by the interpreted algorithm is shown as well.

From the study of less accurate descriptions, we have already discovered that it is still possible to achieve a successful result given a less accurate description. However, it becomes more likely to select a less successful algorithm or even the baseline method (BL) given an incorrect description. Conversely, a better description can lead to a better reconstruction result. The interpreter is, to some extent, tolerant to inaccurate descriptions. The take-away message is that the interpreter, in some cases, can still perform reliably given an inaccurate description. However, it becomes more likely to fail to achieve a successful result given an completely inaccurate description of problem condition.

6.5 Discussions and Conclusions

In this chapter, we proposed a proof of concept interpreter, and provide demonstrative results of the performance of the interpreter. We are interested to see if the interpreter is able to handle conditions where accurate or inaccurate descriptions are provided. The findings can be further summarized into one graph, as shown in Figure 6.16.

The figure layout is as follows: each description \( \text{Desc}_i \) only matches the prob-
<table>
<thead>
<tr>
<th>Object</th>
<th>Bust</th>
<th>Vase0</th>
<th>Barrel</th>
<th>Vase1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incorrect</td>
<td>08020802</td>
<td>08020808</td>
<td>02020802</td>
<td>02020208</td>
</tr>
<tr>
<td>Desc</td>
<td>BL</td>
<td>PMVS</td>
<td>BL</td>
<td>EPS</td>
</tr>
<tr>
<td>Correct</td>
<td>02080208</td>
<td>02080802</td>
<td>08080208</td>
<td>08080802</td>
</tr>
<tr>
<td>Desc</td>
<td>EPS • GSL</td>
<td>EPS • EPS</td>
<td>PMVS • EPS</td>
<td>PMVS • EPS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Object</th>
<th>Statue</th>
<th>Cup</th>
<th>Pot</th>
<th>Vase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incorrect</td>
<td>08020802</td>
<td>08020808</td>
<td>02020802</td>
<td>02020208</td>
</tr>
<tr>
<td>Desc</td>
<td>BL</td>
<td>PMVS</td>
<td>BL</td>
<td>EPS</td>
</tr>
<tr>
<td>Correct</td>
<td>02080208</td>
<td>02080802</td>
<td>08080208</td>
<td>08080802</td>
</tr>
<tr>
<td>Desc</td>
<td>EPS • GSL</td>
<td>EPS • EPS</td>
<td>PMVS • EPS</td>
<td>PMVS • EPS</td>
</tr>
</tbody>
</table>

Figure 6.15: Evaluation 3: inaccurate description may lead to poor result. For each description, the first row represent the settings of properties. The algorithms determined by mapping are shown below with the algorithm selected by interpreter coloured in red (BL: baseline). The last row shows the corresponding reconstruction results. We can see that the results of inaccurate descriptions are poorer than those of accurate descriptions.
<table>
<thead>
<tr>
<th>Desc #</th>
<th>Bust</th>
<th>Vase0</th>
<th>Barrel</th>
<th>Vase1</th>
<th>Interp Algo.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>![Bust Image]</td>
<td>![Vase0 Image]</td>
<td>![Barrel Image]</td>
<td>![Vase1 Image]</td>
<td>GSL</td>
</tr>
<tr>
<td>2</td>
<td>![Bust Image]</td>
<td>![Vase0 Image]</td>
<td>![Barrel Image]</td>
<td>![Vase1 Image]</td>
<td>EPS</td>
</tr>
<tr>
<td>3</td>
<td>![Bust Image]</td>
<td>![Vase0 Image]</td>
<td>![Barrel Image]</td>
<td>![Vase1 Image]</td>
<td>GSL</td>
</tr>
<tr>
<td>4</td>
<td>![Bust Image]</td>
<td>![Vase0 Image]</td>
<td>![Barrel Image]</td>
<td>![Vase1 Image]</td>
<td>PMVS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Desc #</th>
<th>Statue</th>
<th>Cup</th>
<th>Pot</th>
<th>Vase</th>
<th>Interp Algo.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>![Statue Image]</td>
<td>![Cup Image]</td>
<td>![Pot Image]</td>
<td>![Vase Image]</td>
<td>GSL</td>
</tr>
<tr>
<td>3</td>
<td>![Statue Image]</td>
<td>![Cup Image]</td>
<td>![Pot Image]</td>
<td>![Vase Image]</td>
<td>GSL</td>
</tr>
<tr>
<td>4</td>
<td>![Statue Image]</td>
<td>![Cup Image]</td>
<td>![Pot Image]</td>
<td>![Vase Image]</td>
<td>PMVS</td>
</tr>
</tbody>
</table>

**Figure 6.16:** The evaluation of interpreter using synthetic objects. The first column presents the description provided to the interpreter. Description \( i \) matches with the problem condition of synthetic object in column \( (i + 1) \), which is labelled in green rectangle. The last column is the algorithm selected by the interpreter. Since the interpreter would return a successful reconstruction given a description that matches the problem condition, the quality of reconstruction of the labelled objects indicates success/failure of the interpreter.
lem condition of object $i$, which is the object in the $(i+1)$th column. Based on previous findings, objects in diagonal should return successful reconstruction whereas the other results may or may not be successful, which is exactly the case.

Building upon our description and mapping, we are able to develop a proof of concept interpreter which interprets the description of the problem, selects the most appropriate algorithm based on the mapping and returns a reliable reconstruction result. The development of more sophisticated descriptions for more complex geometry and material, incorporating new algorithms, and improving the construction of mapping are some potential directions to improve the presented interface.
Chapter 7

Conclusions

With the increased sophistication of vision algorithms comes the increased barriers of applying these algorithms in real world applications. Just as personal computers did not become mainstream until the graphical user interface, computer vision algorithms will be less likely to reach to masses until creative minds from all disciplines can take advantage of these amazing technologies without needing expertise knowledge. To address this challenge, we proposed a three-layer interface, consisting of description, interpreter, and algorithms, to allow users with no vision background to take advantage of these vision algorithms. We review each of the chapters in details below.

A Problem Space of 3D Reconstruction

In this section, we presented a well-defined problem space for 3D reconstruction algorithms based on the conditions surrounding the problem. It is an $N$–dimensional space, the axes of which are visual and geometric properties of objects. This allows users to think of an algorithm as a “pointer” to a sub-volume within the space that it can reliably work under. This object centred problem space allows application developers to have access to advanced vision techniques without requiring sophisticated vision knowledge of the underlying algorithms.
A Description of 3D Reconstruction

After proposing a problem space that allows to associate algorithms to problem conditions, we proceeded to discuss a way of describing the problem condition of a 3D reconstruction problem, which provides an abstraction layer above the underlying vision algorithms. The description consists of a model and corresponding representations. The model selects characteristic visual and geometric properties of objects as components and the representation uses the key aspect of a property to quantify the strength of the specific property. The proposed description provides a formal and definitive way of describing the problem condition. We further discuss the ease of quantifying the properties, and provide concrete examples to demonstrate the process. The performance of the algorithm can be evaluated given a well defined description, allowing a better understanding of the working conditions of the specific algorithm.

A Mapping of 3D Reconstruction

Once we obtained the means of describing the problem conditions, we set out to investigate the working conditions surrounding each algorithm. First, we investigate the impact of pairwise properties on the performance of three algorithms across categories, from which the effective properties are determined. Next, we evaluate the selected algorithms under different problem conditions consisting only of these effective properties. This allows us to derive a mapping from problem conditions to algorithms by comparing the performance to that of the baseline methods. This information provides a deeper understanding of performance of algorithms and potentially insights into how they may be improved.

An Interpretation of 3D Reconstruction

Lastly, we demonstrated the use of the interface by a proof of concept interpreter, which returns a successful reconstruction result given a valid description of an object’s problem condition. The interpreter is the intermediate layer of the interface, which receives a user-specified description and invokes one of the underlying algorithms for reconstruction. We are interested to see how the interface would perform given accurate, less accurate and incorrect descriptions. First, we proposed a proof
of concept interpreter that takes advantage of the discovered mapping and additional constraints. Next, we presented the real-world and synthetic datasets for evaluation. Lastly, we demonstrated the performance of the interpreter by providing accurate and inaccurate descriptions. The performance of the interface echoes with the statement of the thesis that a description-based interface for 3D reconstruction without knowing algorithm detail is achievable.

We have discovered that the interpreter can produce a successful reconstruction result reliably given accurate description. Further, the performance of the interface deteriorates as the description get farther away from the accurate description. Thus, we conclude that it is achievable to design a description-based interface that leads to a successful reconstruction given a correct description of problem condition. Though, the interface, to some extent, is tolerant to the inaccurate description of problem condition. Nonetheless, the correct interpretation of problem condition is key to a successful reconstruction. The main contribution of this thesis is the development and application of an interface to 3D reconstruction problem. The significance of this contribution is that: 1) few algorithms can work for a diverse categories of objects. The interface, to some extent, can cover a wider range of object categories by incorporating multiple algorithms; 2) a description of object problem condition is provided to hide the algorithmic details, thus understanding of the algorithm, or conditions of applying algorithms are not a prerequisite.

Extending beyond 3D reconstruction, our proposed three-layer interface to general vision problems can be applied to allow other vision tasks be more accessible to application developers by providing a description to vision problems which allows the specifications of problem conditions. In order to provide such accessibility, a description of a well defined problem space, and ways to discover the optimal problem conditions must be addressed. This is a non-trivial task to provide such an abstraction as it requires an understanding of the field and an ability to abstract away algorithmic complexity. Further, the construction of mapping requires more than just appropriate datasets, but also means of avoiding property scaling issue, which would make the problem space too vast to handle. In conclusion, this thesis addresses the accessibility of one of vision topics, 3D reconstruction problem, which provides a novel perspective of approaching 3D vision problem without
delving into algorithm details, and establishes a general framework of description-based interface that could be extended to other vision problems.

7.1 Future Directions

3D reconstruction has been one of the most important topics in vision for decades with a range of applications. This thesis focuses on the accessibility of these algorithms instead of developing algorithm novelties. We make several assumptions and simplifications in this thesis, and thus opens up some potential future directions that can improve the work completed in this thesis.

Geometric Model

The current model fails to capture the geometric complexity of real world objects and focuses mainly on visual properties. Target objects with complex geometric properties, such as severe concavity, occlusion, depth (dis-)continuity, and so on are not captured in the dataset. The issue of incorporating these geometric properties is due to the dilemma of mathematical and semantic representation: mathematical representations are easier to estimate but difficult for human interpretation while semantic representations are easier for humans to understand but challenging to be modelled mathematically. For instance, surface concavity can be mathematically modelled by curvature, but this is non-trivial to estimate the parameter of curvature and translate the numeric value to semantically meaningful terms. Concavity can also be represented as a continuum with two extremities: convexity and concavity, which is intuitive to humans. However, it is hard to interpolate to obtain any semantic terms in between, or get it translated to the mathematical counterpart. Thus, one of the research directions is to develop intuitive and semantically meaningful geometric model to better describe objects with complex shapes. This could significantly expand the problem space and allows a much more powerful interface that target a broader categories of objects.

Property Estimation

We have used an approach of visual inspection and “trial-and-see” to simulate and estimate the property settings of an object, which is based on user input and thus
is relatively subjective, tedious, less rigorous and prone to error. A more robust approach is to utilize machine learning techniques to obtain visual and geometric information directly from images.

**Quantitative Measure**

We have utilized three metrics: accuracy, completeness and angular error. However, there are other measures worth investigating. For instance, colour accuracy that measures the accuracy of the colour information of the reconstructed model, and ‘ghost’ error, which measures the amount of erroneously reconstructed object or scene, and so on. We provide the users with more options and controls over the reconstruction result that suits their needs by providing additional quantitative measures.

**Mapping Construction**

The key component of the interpreter is the mapping from problem conditions to algorithms. This mapping information gives the interpreter information regarding what algorithms work reliably under a given problem condition. Every algorithm that is to be integrated into the interpreter needs to be evaluated under a variety of problem conditions. However, this process might suffer from property scaling issues. For instance, if the number of effective properties is \(N\) and each property has \(L\) levels, the number of different combinations is \(L^N\), which increases exponentially as the number of properties increases. Therefore, we need better ways to discover this mapping relating problem space and algorithms. Otherwise, the mapping construction suffers from what we call *property scaling issue*.

**Interpreter**

Interpreter is the intermediate layer of the three-layer interface, which is responsible for receive user-specified description and invokes one of the successful underlying algorithms. Currently, the implementation of the proof of concept interpreter is simplistic and does not fully take advantage of the information we have obtained from the mapping construction process. Therefore, we should develop a more sophisticated interpreter that is more powerful and offers more flexibility and options.
to the users.

7.2 Closing Words

Nowadays, computer vision has been playing an increasingly important role in a variety of fields. However, it also requires application developers increasing expertise to apply these technologies to a specific application domain. We hope that the efforts in the vision community can be directed to not only develop more advanced algorithms, but also easier access to these algorithms as well. The development of such a system across vision problems will require significant efforts in the form of detailed problem space, comprehensive description, sophisticated methods of interpretation, and so on. This is the work of the entire community of vision researchers, and could lead to advancements in the field similar to those seen in Computer Graphics following the creation of OpenGL. We hope that this thesis provides a starting point for such a significant undertaking, highlighting our initial approach to the problem, and the challenges that researchers who choose to follow may face.
Bibliography


94


[70] C. Wu et al. Visualsfm: A visual structure from motion system. 2011. → pages 11, 69


Appendix A

Supporting Materials

A.1 Definition of 3D Reconstruction

We will first provide definitions of some basic concepts, which include general computer vision concepts such as scene, camera, and image. We then define a few other terms that are closely related to the reconstruction problem. We then provide reasonable approximations for a more practical definition of the problem as a whole.

A.1.1 Basic Notations

We will use the following notations: \( \{C_n\}_{n=0}^{N-1} \) represents the camera set, which includes both intrinsic and extrinsic parameters; \( \{I_n\}_{n=0}^{N-1} \) represents the set of all images; \( \{L_n\}_{n=0}^{N-1} \) represents the set of light sources.

**Definition 1 (Scene)** The scene \( S \) is the four-dimensional joint spatio-temporal target of interest.

**Definition 2 (Image)** The image refers to the 2D observation of the 3D scene \( S \) on the image plane of camera \( C_i \) at time \( t_0 \), which is modelled as: \( I_i = T(S, C_i, L_0, t_0) \), or on the image plane of \( C_0 \) under the light source \( L_i \) at time \( t_i \), \( I_i = T(S, C_0, L_i, t_i) \), where \( T \) is the geometric/radiometric transformation.

\( T \) can be a geometric transformation which determines the 2D coordinates of a 3D point, or a radiometric transformation which determines the intensity/irradiance information from the information of illumination, viewing direction and surface
orientation.

### A.1.2 Segment and Scell

**Definition 3 (Segment)** A segment ($seg$) is a distinct region in the image, and is the most basic element in the image, which can be considered as a generalized pixel.

For instance, a segment can be a pixel, a window area, an edge, a contour, or a region of arbitrary size and shape.

**Definition 4 (Cue)** Cues are the visual or geometric characteristics of the segments $seg$ that can be used for reconstruction, denoted as $cue(seg)$.

For instance, the cue can be the texture within a window area, the intensity/colour value of a pixel, or the object contour, etc.

**Definition 5 (Scell)** A scell (scene element, denoted as $sc$) is a volume in the scene which corresponds to at least one segment. A scell can be considered as a generalization of a voxel.

**Definition 6 (Property)** Properties are the visual and geometric characteristics of the scell $sc$, which would influence the cues of a segment, denoted as $prop(sc)$.

The property of the scell can be the 3D position or orientation information, visual texture, reflectance, surface orientation, roughness, convacity, etc.

The relation between the terms defined above is shown in Figure A.1.

![Figure A.1: Relation between a scell and a segment](image-url)
A.1.3 Consistency

Every photograph of a 3D scene taken from a camera $C_i$ partitions the set of all possible scenes into two families, those that reproduce the photograph and those that do not. We characterize this constraint for a given shape and a given radiance assignment by the notion of consistency.

**Definition 7 (Consistency criterion)** The consistency criterion checks whether the properties of a scell $sc$ can produce the cues observed in the corresponding segment $seg$.

$$consist(prop(sc), cue(seg)) = 1 \Rightarrow \text{consistent}$$

$$consist(prop(sc), cue(seg)) = 0 \Rightarrow \text{not consistent}$$

**Definition 8 (Segment consistency)** Let $S$ be the scene. A scell $s \in S$ that is visible from $C_i$ is consistent with the image $I_i$ if and only if the consistency criterion is true.

**Definition 9 (Image consistency)** A scene $S$ is image consistent with image $I_i$ if any scell $\forall s \in S$ visible from the camera $C_i$ is segment consistent with this image.

**Definition 10 (Scene consistency)** A scene $S$ is scene consistent with a set of images $\{I_n\}_{n=0}^{N-1}$ if it’s image consistency with each image $I_i \in \{I_n\}_{n=0}^{N-1}$ in the set.

A.1.4 Formal Definition

**Definition 11 (3D reconstruction problem)** Given a set of images $\{I_n\}_{n=0}^{N-1}$ captured by cameras $\{C_n\}_{n=0}^{N-1}$, or under a set of light sources $\{L_n\}_{n=0}^{N-1}$, find a set of scells $\{sc_m\}_{m=0}^{M-1}$ such that any scell is consistent with the visible images in the set $\{I_n\}_{n=0}^{N-1}$, i.e., $\forall sc \in \{sc_m\}_{m=0}^{M-1}$, we the have following:

$$consist(prop(sc_i), cue(seg(i,j))) = 1.$$

where $seg(i,j)$ is the corresponding segment of $sc_i$ in camera $C_j$. Alternatively, 3D reconstruction tries to find a set of scells $\{sc_m\}_{m=0}^{M-1}$ that are scene consistent with the image set $\{I_n\}_{n=0}^{N-1}$.
A.1.5 Applied Definition

While the definition presented above gives a formal definition of the problem of 3D reconstruction, it is not necessarily applicable in a practical setting. In this section, we extend this formal definition to an approximate, yet applied version.

**Definition 12 (Consistency score)** The consistency score measures the similarity between a scell \( sc \) and the corresponding segment \( seg \).

\[
\text{consist}(\text{prop}(sc), \text{cue}(seg)) = x, x \in [0, 1]
\]

\[
\text{consist}(\text{prop}(sc), \text{cue}(seg)) = 1 \Rightarrow \text{consistent}
\]

\[
\text{consist}(\text{prop}(sc), \text{cue}(seg)) = 0 \Rightarrow \text{not consistent}
\]

**Definition 13 (Applied consistency criterion)** A scell \( sc \) and a segment \( seg \) are considered consistent if the consistency score is above a pre-defined threshold \( \varepsilon \).

\[
\text{consist}(\text{prop}(sc), \text{cue}(seg)) > \varepsilon
\]

**Definition 14 (Applied 3D Reconstruction Problem)** Given a set of images \( \{I_n\}_{n=0}^{N-1} \) captured by cameras \( \{C_n\}_{n=0}^{N-1} \), or under a set of light sources \( \{L_n\}_{n=0}^{N-1} \), find a set of scells \( \{sc_m\}_{m=0}^{M-1} \) such that the consistency score between the set of scells and their corresponding segments \( \{seg(i,j)\}_{i=0}^{N-1} \) are maximized.

\[
\text{maximize} \sum_{j=0}^{N-1} \sum_{i=0}^{M-1} \text{consist}(\text{prop}(sc_i), \text{cue}(seg(i,j)))
\]

A.2 Physics-Based Vision

A.2.1 Radiometric Terms

Below is a list of radiometry terms, see Figure A.2 for an illustration:

- Solid angle \( (d\omega) \): 3D counterpart of angle, \( d\omega = \frac{dA \cos \theta}{R^2} \) \( \text{steradian} \).

- Projected solid angle \( (d\Omega) \): \( d\Omega = \cos \theta d\omega \).
• Incident radiance \( L_i(\theta_i, \phi_i) \): light flux received from the direction \((\theta_i, \phi_i)\) on a unit surface area, unit \((\text{watt} \cdot \text{m}^{-2} \cdot \text{steradian}^{-1})\).

• Irradiance \( E_i(\theta_i, \phi_i) \): light Flux (power) incident per unit surface area from all direction, \( E_i(\theta_i, \phi_i) = \int_{\Omega_i} L_i(\theta_i, \phi_i) d\Omega_i \text{(watt/m}^2\text{)} \).

• Surface radiance \( L_r(\theta_r, \phi_r) \): light flux emitted from a unit surface area in the direction \((\theta_r, \phi_r)\), unit \((\text{watt} \cdot \text{m}^{-2} \cdot \text{steradian}^{-1})\).

\[ L_i(\theta_i, \phi_i) \]

\[ L_r(\theta_r, \phi_r) \]

**Figure A.2:** Illustration of light-matter interaction.

**Definition 15 (BRDF)** the ratio of the scene radiance \( L_r(\theta_r, \phi_r) \) to the irradiance \( E_i(\theta_i, \phi_i) \), i.e., \( f(\theta_i, \phi_i, \theta_r, \phi_r) = \frac{L_{\text{surface}}(\theta_r, \phi_r)}{E_{\text{surface}}(\theta_i, \phi_i)} \).

The BRDF is used in the reflectance equation:

\[
L_{\text{surface}}(\theta_r, \phi_r) = f(\theta_i, \phi_i, \theta_r, \phi_r)E_{\text{surface}}(\theta_i, \phi_i) = \int_{\Omega} f(\theta_i, \phi_i, \theta_r, \phi_r)L_i(\theta_i, \phi_i) \cos \theta_i d\Omega_i
\]

The phenomena described by the BRDF includes (at least for dielectrics) two distinct physical phenomena - surface reflection and subsurface scattering. Since each of these phenomena has different behaviour, BRDFs typically include a separate term for each one. The BRDF term describing surface reflection is usually called the *specular term* and the term describing subsurface scattering is called the *diffuse term*.
A.2.2 Reflectance Model: Microfacet Model

The basis for most physically-based specular BRDF term is microfacet theory. This theory was developed to describe surface reflection from general (non-optically flat) surfaces. The basic assumption underlying microfacet theory is that the surface is composed of many microfacet, too small to be seen individually. Each microfacet is assumed to be optically flat, and splits light into exactly two directions - reflection and refraction.

The microfacet model postulates that if a surface reflection can occur between a given light vector $l$ and view vector $v$, then there must exist some portion of the surface, or microfacet, with a normal aligned halfway between the $l$ and $v$. This “half vector”, sometimes referred to as the microsurface normal, is thus defined as $h = \frac{l + v}{|l + v|}$. Not all microfacets for which $n = h$ will contribute to the reflection: some are blocked by other microfacets from the direction of $l$ (shadowing), from the direction of $v$ (masking), or from both. Microfacet theory assumes that all shadowed light is lost from the specular term; in reality, due to multiple surface reflections some of it will eventually be visible, but this is not accounted for in microfacet theory. This is not typically a major source of error in most cases (rough metal surfaces are a possible exception).

With these assumptions (optically flat microfacets, no inter-reflection), the microfacet specular BRDF term has the following form:

$$f(l, v) = \frac{D(\theta_h)F(\theta_d)G(\theta_l, \theta_v)}{4 \cos \theta_l \cos \theta_v}$$

where $\theta_l$ and $\theta_v$ are the angles of incidence of $l$ and $v$ vectors with respect to the normal $\theta_h$ is the angle between the normal and the half vector, and $\theta_d$ is the “difference” angle between $l$ and the half vector. For the specular term, $D$ is the microfacet distribution function and is responsible for the shape of the specular peak, $F$ is the Fresnel reflection coefficient, and $G$ is the geometric attenuation or shadowing factor.

Although there are several model for subsurface local reflection in the literature, the most widely-used on by far is the Lambertian BRDF term. The Lambertian BRDF is actually a constant value; the well known cosine of $(n \cdot l)$ factor is part of the reflection equation, not the BRDF. The exact value of the Lambertian
BRDF is

\[ f_{\text{Lambert}}(l, v) = \frac{c_{\text{diff}}}{\pi} \]

Here \( c_{\text{diff}} \) is the fraction of light which is diffusely reflected. It is an RGB value with R, G, and B restricted to the 0 - 1 range, and corresponds closely to what most people think of as a “surface colour”. This parameter is typically referred to as the **diffuse colour**.

Most physically plausible models not specifically described in microfacet form can still be interpreted as microfacet models in that they have a distribution function, a Fresnel factor, and some additional factor which could be considered a geometric shadowing factor. The only real difference between microfacet models and other models is whether they include the explicit \( \frac{1}{4 \cos \theta_l \cos \theta_i} \) factor that comes from the microfacet derivation. For models that exclude this factor, an implied shadowing factor can be determined by multiplying the model by \( 4 \cos \theta_l \cos \theta_i \) after factoring out the \( D \) and \( F \) factors.

### A.2.3 Image Formation: Radiometric Perspective

This section discusses the radiometric perspective of image formation. Specifically, we discuss the contributing factors that determine the pixel intensity of an image.

#### Light-Matter Interaction

The relation between the incoming illumination and reflected light is modelled using the **bidirectional reflectance distribution function** (BRDF), refer to the Appendix A.2.1 for definitions of radiometric terms.

![Figure A.3: The light-matter interaction. Scene radiance is linear related to incident radiance.](image)

As we can see from the definition of BRDF, scene radiance is a function of BRDF given a fixed incident radiance. BRDF is a bidirectional function, which
depends on both incoming and outgoing directions. It can be simplified under specific reflectance models. For instance, BRDF can be simplified as *Diffuse albedo* or surface albedo when using Lambertian reflectance, which is the proportion of incident light that is reflected by the surface.

**Light-Lens Interaction**

A common assumption made in vision community is that radiance is constant as it propagates along a ray. Therefore, the scene radiance is the same as the radiance passing through the lens, which is the same as the radiance received by the sensor. Since image irradiance is the radiance accumulated on a unit surface area, it follows that image irradiance is linear related to the scene radiance. Thus, it can be shown that *image irradiance* is proportional to *scene radiance* [21].

![Figure A.4: The light-sensor interaction. Image irradiance is linearly related to scene radiance.](image)

**Light-Sensor Interaction**

The camera response function relating image irradiance at the image plane to measured pixel intensity values is a non-linear mapping. It is common to assume that intensity is linear related to image irradiance in many vision algorithms. A linear relation can be retrieved by radiometric calibration.

![Figure A.5: The camera response function is typically non-linear. Thus pixel intensity if non-linear with respect to irradiance. However, this can be corrected by radiometric calibration. More often, it is assumed that pixel intensity is linearly related to image irradiance in many vision algorithms.](image)
## A.3 Results of Real-World Objects

<table>
<thead>
<tr>
<th>Image</th>
<th>PMVS</th>
<th>EPS</th>
<th>GSL</th>
<th>VH (BL)</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="pmvs1.png" alt="PMVS" /></td>
<td><img src="eps1.png" alt="EPS" /></td>
<td><img src="gsl1.png" alt="GSL" /></td>
<td><img src="vh1.png" alt="VH (BL)" /></td>
</tr>
<tr>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="pmvs2.png" alt="PMVS" /></td>
<td><img src="eps2.png" alt="EPS" /></td>
<td><img src="gsl2.png" alt="GSL" /></td>
<td><img src="vh2.png" alt="VH (BL)" /></td>
</tr>
<tr>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="pmvs3.png" alt="PMVS" /></td>
<td><img src="eps3.png" alt="EPS" /></td>
<td><img src="gsl3.png" alt="GSL" /></td>
<td><img src="vh3.png" alt="VH (BL)" /></td>
</tr>
<tr>
<td><img src="image4.png" alt="Image" /></td>
<td><img src="pmvs4.png" alt="PMVS" /></td>
<td><img src="eps4.png" alt="EPS" /></td>
<td><img src="gsl4.png" alt="GSL" /></td>
<td><img src="vh4.png" alt="VH (BL)" /></td>
</tr>
</tbody>
</table>

**Figure A.6:** Reconstruction results of MVS, PS, SL, and the baseline method VH.
<table>
<thead>
<tr>
<th>Image</th>
<th>PMVS</th>
<th>Example-based PS</th>
<th>Gray SL</th>
<th>VH(BL)</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="PMVS" /></td>
<td><img src="image3.png" alt="Example-based PS" /></td>
<td><img src="image4.png" alt="Gray SL" /></td>
<td><img src="image5.png" alt="VH(BL)" /></td>
</tr>
<tr>
<td><img src="image6.png" alt="Image" /></td>
<td><img src="image7.png" alt="PMVS" /></td>
<td><img src="image8.png" alt="Example-based PS" /></td>
<td><img src="image9.png" alt="Gray SL" /></td>
<td><img src="image10.png" alt="VH(BL)" /></td>
</tr>
<tr>
<td><img src="image11.png" alt="Image" /></td>
<td><img src="image12.png" alt="PMVS" /></td>
<td><img src="image13.png" alt="Example-based PS" /></td>
<td><img src="image14.png" alt="Gray SL" /></td>
<td><img src="image15.png" alt="VH(BL)" /></td>
</tr>
<tr>
<td><img src="image16.png" alt="Image" /></td>
<td><img src="image17.png" alt="PMVS" /></td>
<td><img src="image18.png" alt="Example-based PS" /></td>
<td><img src="image19.png" alt="Gray SL" /></td>
<td><img src="image20.png" alt="VH(BL)" /></td>
</tr>
</tbody>
</table>

**Figure A.7:** Reconstruction results of MVS, PS, SL, and the baseline method VH (continued).