Development and Evaluation of A 3D Reconstruction Framework for General Objects

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 APPLIED SCIENCE

(Department of Electrical and Computer Engineering)

The University of British Columbia

(Vancouver)

April 2017

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.

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.

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 the IEEE Winter Conference on Application of Computer Vision (2018) and is under anonymous review at the moment of thesis submission.

Table of Contents

List of Tables

List of Figures

List of Acronyms

- **3D**: 3-dimensional
- **BRDF**: Bi-directional Reflectance Distribution Function
- **CAD**: Computer Aided Design
- **DoF**: Degree of Freedom
- **EPD**: Effective Problem Domain
- **EPS**: Example-based Photometric Stereo
- **GSL**: Gray code 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

Acknowledgments

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 indebt to Dr. Gregor Miller for his mentorship and constant words of encouragement to keep me sane.

I want to thank my labmates 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

献给我的爷爷吴国利先生

Chapter 1

Introduction

Modeling 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 modeling is to reconstruct a 3D geometric 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 past decades, we have witnessed a variety of tools and approaches to 3D modeling applied successfully to an assortment of sub-domains, such as Computer Aided Design (CAD) tools [1], arm-mounted probes, active methods [2, 3, 10, 37] and passive image-based methods [16, 21, 23, 36]. Among the existing approaches, active techniques such as laser scanners [37], Structured Light (SL) systems [10], and Photometric Stereo (PS) [67], as well as passive methods such as Multi-View Stereo (MVS) [57], have been the most successful. Laser scanners and structured light techniques are seen to generate the most accurate results, but are generally complicated to set up and

calibrate, time consuming to scan, and demanding to store and process in terms of memory. 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 the 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 successes 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 works 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.

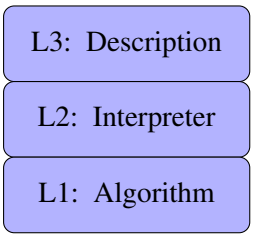


Figure 1.1: The three layers of the 3D reconstruction interface.

1.1 Motivating scenario

We propose a scenario to further emphasize and justify the motivation: Ben is a contributor to the open source 3D creation software, Blender. He wants to create an addon so that 3D artists who don't normally have any experience in coding or 3D vision can easily scan their object, and use them in their scene. He started to look into general computer vision libraries, and came across openCV and one of the modules named *calib3d*. There are many 3D vision algorithms implemented, but there are no out-of-box 3D reconstruction algorithms available. Since he had no previous experience in this field, he decided to keep looking for well established open source 3D recosntruction algorithms. Two of these softwares that he found are VisualSfM and PMVS. He wrote a python addon based on these softwares, which took input images from artists and return a reconstructed scene/object. This addon provided artists with parameter settings to fine tune these algorithms. He released this addon to the Blender community and got some good initial response. However, some artists complained that this addon didn't always achieve successful reconstructions, especially on textureless surfaces. Upon receiving these feedbacks, Ben started to look into ways to work around these issues. He found some open source softwares that claimed to work on textureless surfaces. He integrated these softwares into the addon and created an interface that allows CG artists to choose among different algorithms and parameter settings. However, after the release of the new addon, some artists complained that they didn't have a clue of which algorithm should they choose for their specific object/scene, and the meaning and interpretation of the parameters are confusing and unclear. Since most CG artists don't have a background in vision, it generally takes multiple 'try-and-

error' before getting the desired results. Besides, the meaning of the parameters are way to abstract and confusing, and somewhat counter-intuitive. They asked Ben if there is a way to determine an algorithm for their specific object without multiple 'try-and-error', and provide a more perceptually intuitive and interpretable parameters so that they don't have to deal with algorithm-level parameters. Ben started to do some more research and came across a project called openVL, which provided higher-level abstractions to hide algorithmic details. He found out that this project provided API that allowed users to describe the appearance of the object (problem condition). A synthetic toolbox was provided so that users can set initial parameters by observation, and then fine tune these parameters by comparing the synthetic result to the real object. This description of appearance is interpreted and an algorithm is selected. Ben started to embed openVL into the addon, and provide visually perceptible parameters to the artists. This improved addon is well received and successful among CG artists.

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 taxonomy is proposed that transforms the 3D reconstruction problem from one requiring knowledge of algorithmic details to one that is based on the correlation between the problem space 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. A rigorous 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 softwares and algorithms focus on providing algorithmic solutions to problems, which we call a *algorithm-center approach*. This approach provides little insight to the problem conditions that a specific algorithm is applicable to. We proposed a *problem-centered approach* that gives a well-defined problem space, which allows further investigation of the relation between problem conditions and algorithms. This relation can be used to choose a best possible algorithm based on described problem condition. The problem condition consists of a variety of visual and geometric properties of objects. The collective problem conditions is called *problem condition space*, or in short *problem space*.

1.2.3 A Description of 3D Reconstruction

In previous cases, the mapping from a problem space to an algorithm has been ambigueous due to the problem space that is poorly defined. Here, we set out to provide a rigorous definition of the problem space itself. First, a formal and practical definition of the 3D reconstruction problem based on set theory is proposed. Second, a model consisting of key object properties is developed. Third, the representations of the problem 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 algorithm space, we need to evaluate the performance of the selected algorithms under varied properties and their combinations. We use synthetic datasets to achieve this goal. Part of the challenge in establishing a comprehensive set of experiments for such an evaluation is the large variations of shapes and material properties. To overcome this issue, we first establish the *effective problem domain* (EPS) by finding the effective properties. Then we evaluate the performance of each algorithm within the EPD, which serves as the basis of the mapping.

1.2.5 An Interpretation of 3D Reconstruction

We conduct the evaluation of the interface around two key evaluation questions: 1) can the derived mapping be extended to an object with a different shape; 2) can the proof-of-concept interpreter return a reliable result given the correct description to the problem condition. To answer these questions, we carry out two separate experiments: In Section **??**, we use synthetic objects with the same configurations as the ones used to derive the mapping, and check if the mapping is consistent for different objects across varied problem conditions; in Section 6.4, we use synthetic and real-world datasets to evaluate the interpreter.

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 algorithmic 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 conditions, which consists of varied visual and geometric properties, can be interpreted so that an appropriate algorithm is chosen to reconstruct a successful result. This endeavor 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 expand the problem space by incorporating multiple algorithms. Though it can cover a wider range of problem space than a single algorithm, it is still confined within the space covered by currently existing techniques. 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 prob-

lems 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 3D reconstruction description and the robustness of the proof-of-concept interpreter.

Chapter 2

Related Work

In this chapter, we review the existing softwares and algorithms in the field of 3D computer vision. Section 2.1 discusses the existing softwares 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/geometric cues, which include *stereo correspondence*, *shading*, *silhouette*, *texture distortion*, and *(de)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 [13], VLFeat [63], VXL [4] and multiple Matlab libraries [35, 43]. 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/descriptors are provided by OpenCV but with no indication of under what conditions each works most effectively.

We have witnessed many successful softwares in the field of image-based reconstruction, which is a sub-field of 3D reconstruction. One of the most widely used open source softwares is PMVS developed by Furukawa [21], which is used not only by computer vision/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 [61], and Poisson Surface Reconstruction developed by Michael Misha Kazhdan, which is a surface mesh software that estimate the triangulated surface from oriented point cloud [34]. Some other notable open source softwares include VisualSfM [68], CMP-MVS [26], MVE [20], and openMVG [46]. 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 [50]. 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, (de)focus, etc. This review of algorithms are structured based on these reconstruction cues. Please refer to Table 2.1 for an overview, where the algorithms are organized based on the cue used for reconstruction.

Cue	Algorithm
Stereo correspondence	Stereoscopy Trinocular Stereo Multi-view Stereo (MVS) Laser scanning Structured light (SL)
Shading	Shape from Shading (SfS) Photometric Stereo (PS)
Contour	Shape from Silhouette (SfS)
Texture	Shape from Texture
(De)focus	Shape from (De)focus

Table 2.1: Classes of algorithms that utilize each visual/geometric cue. Note that the abbreviations will be used extensively in the thesis.

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 [57], 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

Volumemetric 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 discretized 3D space in depth-order to identify voxels that have a unique colouring, constant across all possible interpretations of the scene [56]. An-

other thread of work formulates the problem in the Markov Random Field (MRF) framework and extracts the optimal surface by Graph-Cut algorithms [52, 64, 65].

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 [44] by iteratively removing inconsistent voxels from the scene [36]. 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 [16]. Other approaches use a deformable model and represent the scene as surface meshes that moves as a function of internal and external forces [15]. 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 [49]. 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 neighborhood of this seed point, new potential matches are checked and the best points are added to the current list of seed points [38, 39]. 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 [21]. A stereoscopic approach called PatchMatch Stereo, which is inspired by an approximate nearest neighbour matching algorithm called PatchMatch [6], starts by randomly assigning an oriented plane to each pixel in two views. Next, each pixel is taken through three iterations of propagations 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 [22, 62, 70], and develop new propagation schemes to increase the computational efficiency [22].

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 discretized 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 [23, 65]. 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.

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

Though MVS algorithms have relatively minimum hardware requirements, and works reliably even in an unconstrained envirionment. 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 [57]. It turns out that MVS algorithms are able to exploit very weak and intricate image textures, most of which come from shading and/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-diffsue 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 [32, 33].

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 [53]. Temporal techniques generate the codeword by projecting a sequence of patterns. Spatial codification represents each codeword in a unique spatial pattern. Direct codification 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 illuminaiton 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 [53].

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 [53].

Structurd 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 [66]. 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 [29]. 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:

$$\int p_x^2 + p_y^2 + q_x^2 + q_y^2 \mathrm{dxdy} = \int \|\nabla p\|^2 + \|\nabla q\|^2 \mathrm{dxdy}$$

Another is the integrability constraint:

$$\int (p_y - q_x)^2 \mathrm{dxdy}$$

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 [69] demonstrated that SfS algorithms generally perform poorly, and none performs well in all cases.

Photometric Stereo

The **Classical Photometric Stereo**, first proposed by Woodham [67], utilized multiple light sources from different directions to overcome the ambiguity of Shape from Shading. Assuming Lambertian reflectance, P pixels per image, and Q illu-

mination directions, the intensity of the ith pixel under jth illumination is

$$I_{i,j} = \rho_i \vec{n}_i^\top \vec{l}_j$$
$$\Rightarrow \mathbf{I} = \mathbf{N}^\top \mathbf{L}$$

where $\mathbf{I} \in \mathbb{R}^{P \times Q}$ 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. $\mathbf{N} \in \mathbb{R}^{P \times 3}$ encodes the albedo-scaled surface normal for each pixel, i.e., $N_{i,:} = \rho_i \vec{n}_i^\top$. $\mathbf{L} \in \mathbb{R}^{3 \times Q}$ encodes the light source directions, i.e., $L_{:,j} = \vec{l}_j$. This surface reflectance, i.e., spatially varying albedo ρ_i, and the normal n_i can be estimated by

$$\mathbf{N} = \mathbf{I}\mathbf{L}^+$$
$$\Rightarrow \rho_i = \|\mathbf{N}_{i,:}\|$$
$$\Rightarrow n_i = \frac{\mathbf{N}_{i,:}^\top}{\|\mathbf{N}_{i,:}\|}$$

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 2.2.

Generalization of Lighting It is possible to estimate the surface orientation

Category	Camera	Light source	Reflectance
Classical PS	Orthographic	Directional, known intensity and direction	Lambertian
Generalized lighting PS	Orthographic	Unknown intensity and direction, ambient	Lambertian
Generalized reflectance PS	Orthographic	Distant, known intensity and direction	Non-Lambertian

Table 2.2: Assumptions made by different classes of photometric stereo.

without knowing light directions, a case also known as *uncalibrated Photometric Stereo*, see Table 2.2. Most uncalibrated techniques assume Lambertian techniques and are based on factorization technique proposed in [25]. Recall the irradiance equation:

$$\mathbf{I} = \mathbf{N}^\top \mathbf{L}$$

However, an infinite number of candidates $\hat{N}$ and $\hat{L}$ make the above equality met. In fact, any invertible 3×3 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 [8].

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/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 [14] and [7]. Others assume that the color of the specular lobe differs from that of the diffuse lobe, which allows the separation of the specular and diffuse components [42, 54, 55].

Reference object approach uses a reference object that has similar material as the target object. This is proposed in [60] and later revisited in [27]. 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 [24].

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, 71].

Photometric Stereo can work extremly 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 [27].

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*.

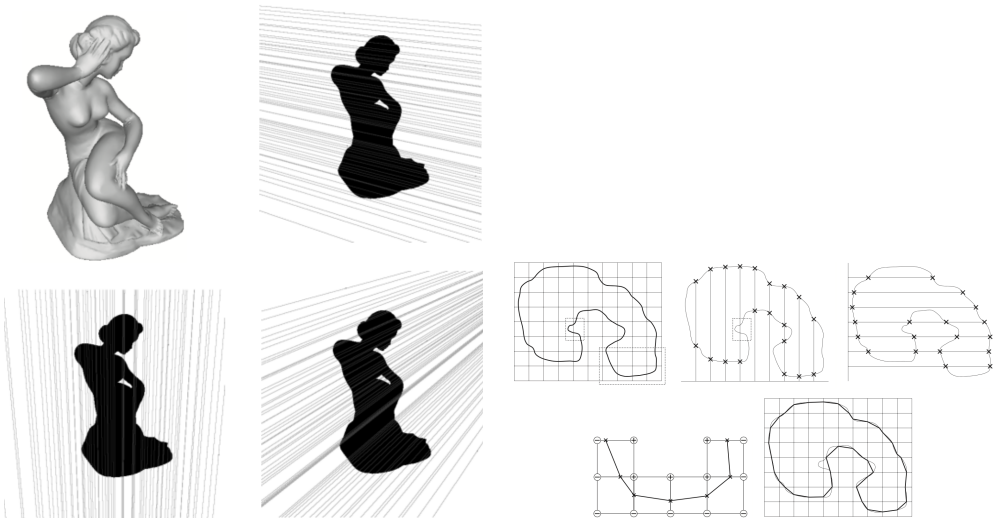
The basic idea of shape from silhouette algorithms is that the object lies inside the intersection of all visual cones back-projected from silhouettes. Suppose there are multiple views V of the target object. From each viewpoint $v \in V$, the silhouette s_v can be extracted, which is the region including 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 center of projection and passing through all the points of the silhouette. The target object is definitely internal to $cone_v$ and this is true fro every view $v' \in V$; it follows that the object is contained inside the volume $c_V = \cap_{v \in V} c_v$. 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.

[computational complexity] intersection of many volumes can be slow. Simple polyhedron-polyhedron intersection algorithms are inefficient. To improve performance, most methods 1) quantize volumes, 2) perform intersection computation in 2D instead of 3D.

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 extrated from it. This is done by the MI technique proposed in [51] 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.



(a) (b)

Figure 2.1: Illustrations of MI-based VH. (a) shows one object (top left) and its silhouette with 2D lines traced over it to find intersections along rays in the X, Y and Z ray-set of the MI, respectively. (b) shows the MI data structure and conversion algorithm in a 2D example. Image courtesy of M. Tarini.

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 silouettes 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 center 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 softwares to tackle this problem. Thus, most research efforts have been devoted to improving algorithmic 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-center* approach, faces two challenges: 1) it provides little to none insight into the conditions that allow a specific algorithm to work successfully; 2) it needs 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 a $N-$dimensional space which encompasses the material and shape of objects, and the axes of which consist of characteristic material and geometric attributes (called properties). The sub-volumn 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. Thus, it is crucial to have multiple algorithms, each registered to a distinct sub-volumn 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 the problem 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 conceptually estimated, and are also wildly 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-centered* 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 measurable. The advantage of the *problem-centered* 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 conceptually 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 rationales to limit the scope of problem space in Section 3.1. 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 dimensions/axes of the 3D reconstruction problem space. This approach allows us to think of algorithms pointing to volumes within an $n-$dimensional problem space. Existing algorithms can be incorporated into the interface by evaluating the algorithmic performance within the problem space, as shown in Figure 3.2. However, by no means are the presented problem space complete. There are many other properties not included that are commonely 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 Table 3.1.

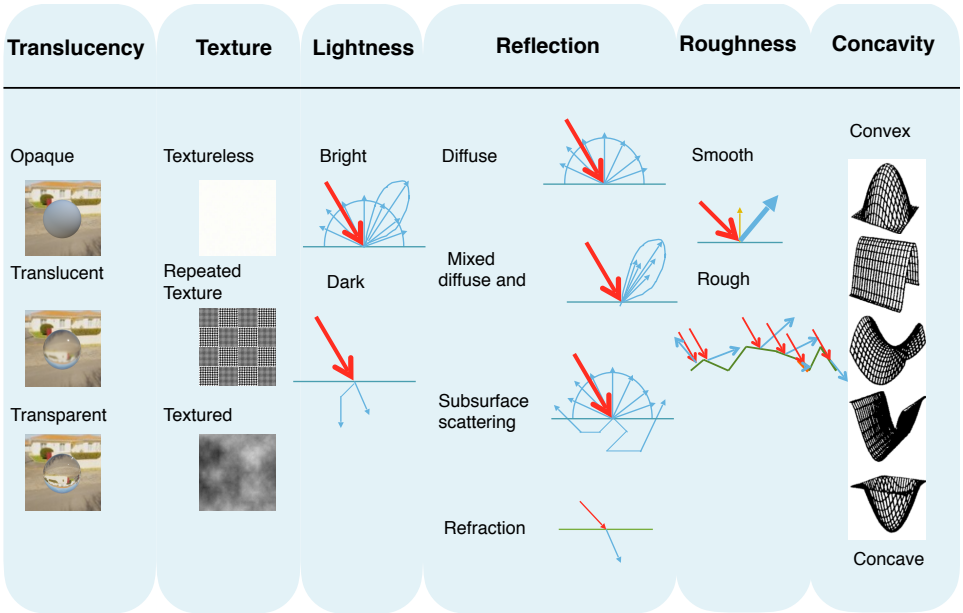


Figure 3.1: A list of properties for problem space.

Property name	Property value	Property label
Translucency	Opaque	**O**
	Translucent	**Tl**
	Transparent	**Tp**
Texture	Textured	**T**
	Repeated textured	**Tr**
	Textureless	**Tl**
Lightness	Bright	**B**
	Dark	**D**
Reflection	Specular model	**S**
	Diffuse model	**D**
	Mixed model	**M**
	Sub-surface scattering	**Ss**
	Refraction	**Rf**
Roughness	Smooth	**S**
	Rough	**R**
Concavity	Convex	**Cx**
	Concave	**Cv**

Table 3.1: Labels of properties

3.2 Assumptions

To limit the scope of the problem space, we make the following assumptions:

Simplified light interaction model

We assume **local interaction model**, i.e., global light transport such as transmission, refraction, cast shadow, inter-reflection, metalic are not considered. The rationale behind our choice is that most techniques that have been developed over the past few decades mainly tackle object with an opaque, diffuse or mixed surface. For specular, refractive, and translucent or transparent objects, only very specialized algorithms are applicable for reconstruction [30]. 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 will be discussed in Chapter 5, as shown in Figure 3.2.

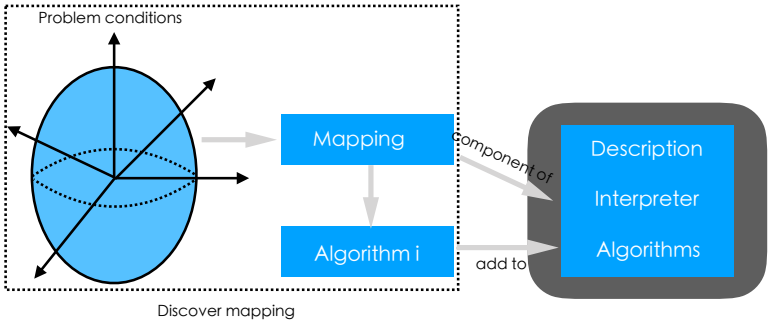


Figure 3.2: Embed algorithms into the interface.

Simplified reflectance model

Since the majority of reconstruciton techniques rely on observing light reflected off a surface, surfaces exhibit significant effect of global light tranport present a huge challenge to the reconstruction problem. Surface exhibits global light transport, including *specular*, *transmission*, *sub-surface scattering*, *inter-reflection*, *self-shadow*, and etc would break the assumptions made by most generic 3D reconstruction algorithms. Thus the global light transport are ignored, and the reflection properties of consideration are *albedo*, i.e., the ratio of reflected light w.r.t the received light, and *specularity*, i.e., the amount of specular reflection. A more comprehensive model should be constructed based on our work to incorporate more complex phenomena to be more comprehensive.

Simplified geometric model

It's 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 prohibitive 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 ignored.

High surface albedo

Existing 3D vision techniques requires 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 candidate algorithms ineffective, including most active techniques. In this thesis, dark surfaces could be challenging for all the underlying algorithms used in this thesis. This is fine in term of designing an interface since there is no need for all underlying algorithms to work well. However, from a demonstrative standpoint, this would fail to demonstrate the performance of interpreter since it would most likely select the baseline method. To avoid such situations, we decided to use objects with bright surfaces so that reconstruction cues to easier to detect.

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 descritpion-based interface that hides algorithm details and return solutions without knowing the underlying algorithms using objects that satisfy these conditions.

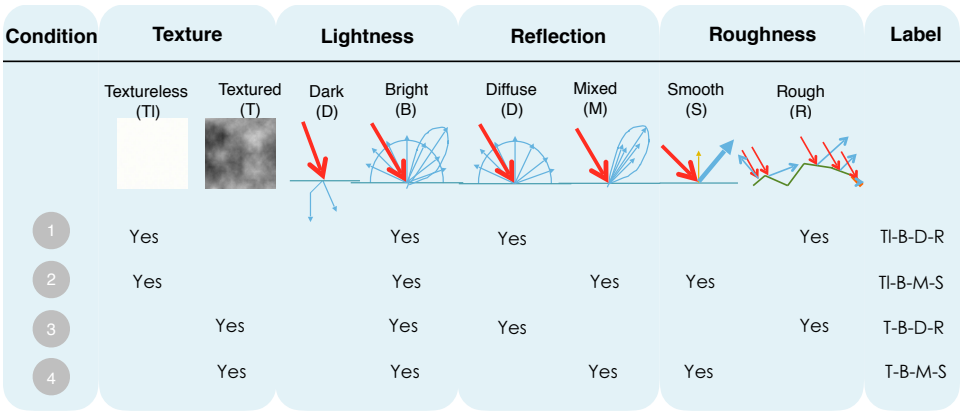


Figure 3.3: Four problem conditions selected based on the definition of problem space and additional assumptions. The description-based interface will be evaluated using objects satisfying these conditions.

Chapter 4

A Description of 3D Reconstruction

In Chapter 3, we introduced a problem space for 3D reconstruction. However, a problem space alone is not sufficient to achieve the goal of creating an interface to 3D reconstruction. Given a problem condition, only a general and vague description is possible for now. For instance, a 'textured, glossy cup'. However, this description is hard to interpret: how is a textured or textureless, matte or glossy surface represented, and how textured or glossy the surface is. Thus, this previous description lacks two key compoents to make it interpretable. First, since it is practically impossible to achieve an exhaustive description, the description should only encompass a fixed set of properties, each with a pre-defined rule for interpretation. The scope of description is determined by the scope of the problem space. For instance, sub-surface scattering property is not included in the description since it is omitted in the problem space. Secondly, 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 (texture element), or 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 quantity of the corresponding property.

Thus, this chapter is concerned with defining an interpretable description to

select an algorithm, which consists of a model and corresponding representations. Computer vision problems require, among other factors, a model of the problem space and appropriate representations [41]. 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. The description essentially tries to answer two questions: *what to describe*, i.e., what properties should be included or excluded in the model, and *how to describe a property*, i.e., what characteristic feature of property should be of interest. By using this formal description, expressing the conditions within which an algorithm works reliably becomes more specific.

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 *aspects* of the problem space that are crucial for describing the appearance of the object. Section 4.2 outlines concrete representations of the proposed model. Section 4.3 provides examples of expressing common 3D reconstruction problems using the proposed model and representations. These following four layers represent the description of our accessible 3D reconstruction framework: Definition, Representation, Model, and Expression.

4.1 Model

Models and representations are fundamental for vision problem solving. Models select characteristic properties of an object, and representation describes 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 [12]. For instance, surface orientation is one component 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 where RGB space is the corresponding representation of the colour.

We select the subset of properties used for object taxonomy in Chapter **??** as

the main components of our model. The model consisting of these key properties is shown in Table 4.1.

Model	Texture
	Lightness
	Reflectance
	Roughness
	Concavity

Table 4.1: Model of the 3D reconstruction problem. Properties are selected from the taxonomy in Chapter 3.

4.2 Representation

Based on the proposed definition and model of the 3D reconstruction problem, we need to further define our representations so that the 3D reconstruction problem can be expressed using our proposed model. We need to turn our attention to how to represent the properties used in the proposed model.

4.2.1 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 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/gran-

ularity, orientation, homogeneity, randomness, etc. However, due to the diverse and complexe 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 no/low *texture randomness* whereas a highly textured surface has high *texture randomness*.

4.2.2 Lightness

When light strikes a surface, it may be reflected, transmitted, absorbed, or scattered; usually, a combination of these effects occurs. The intensity/colour information received by a sensor is thus determined, among other factors, by the amount of light available after these interactions. Here, we consider intensity as caused solely by reflection, since this is one of the most common phenomena experienced and is the easiest to analyze. Generally, 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/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*.

Light-matter interaction

The relation between the incoming illumination and reflected light is modelled using the *bidirectional reflectance distribution function* (BRDF). The BRDF is defined as:

Definition (BRDF) the ratio of the scene radiance $\mathbf{L_r}(\theta_\mathbf{r}, \phi_\mathbf{r})$ to the irradiance $\mathbf{E_i}(\theta_\mathbf{i}, \phi_\mathbf{i})$, i.e., $f(\theta_i, \phi_i, \theta_r, \phi_r) = \frac{E^{surface}(\theta_i, \phi_i)}{L^{surface}(\theta_r, \phi_r)}$.

Figure 4.1: The light-matter interaction. Scene radiance is linear related to incident radiance.

For Lambertian model, BRDF can be simplified as *Diffuse albedo* or surface albedo, 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, it follows that image irradiance is proportional to the scene radiance. Thus, the relation between *scene radiance* and *image irradiance* is linear.

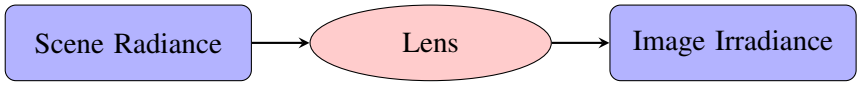


Figure 4.2: The light-lens interaction. Image irradiance is linearly related to scene radiance.

Light sensor interaction

The camera response function relating image irradiance at the image plane to measured pixel intensity values is a non-linear mapping. A linear relation can be retrieved by radiometric calibration.

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 related to surface reflectance, which is characterized by BRDF. There are 4 DoF in spatially-invariant BRDF, and for the simplified case

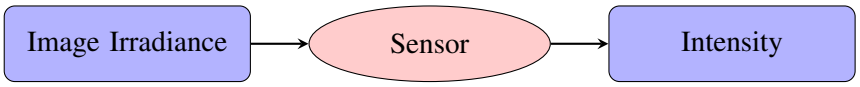


Figure 4.3: The light-sensor interaction. Pixel intensity is linearly related to image irradiance assuming linear radiometric mapping.

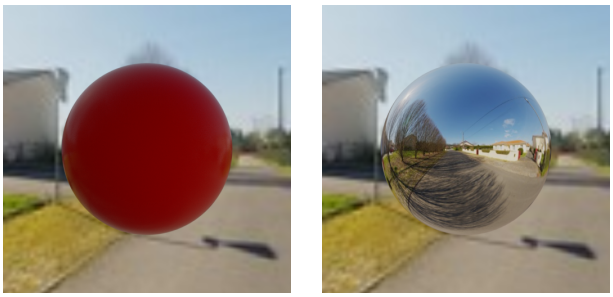
- Lambertian reflectance, the BRDF is degenerated to *diffuse albedo*, which is the representation we adopt for lightness.

4.2.3 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 [47]. Unlike diffuse reflections, where we experience the lightness 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 4.4 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. A commonly used model treats mixed reflectance as a weighted combination of diffuse and specular component. Thus the ratio of incident light that is specularly reflected is considered as the representation of sepcularity, with 0 being completely diffuse, and 1 being completely specular (mirror like).

4.2.4 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



(a) (b)

Figure 4.4: (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 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.

4.2.5 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, reflectance and transmittance 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. 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

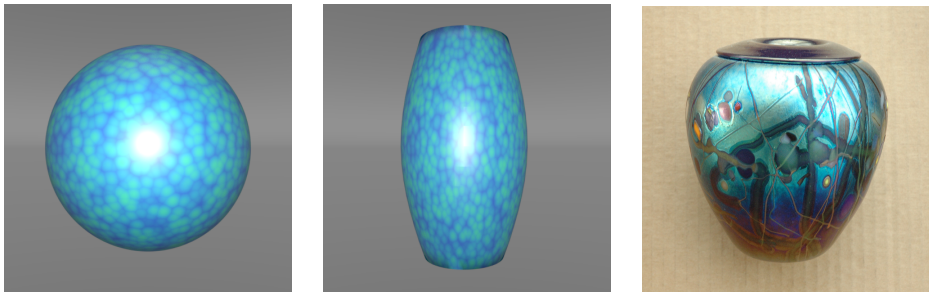
of material perception, though the answer is currently far from clear. But our intuition and many empirical studies support the view that humans are exceptional at material perception.

Everyday experience suggests that, human's visual system are extremly 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. 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 judgements of material properties to determine whether fruit is ripe, whether soup has bee left to go cold or whether bread is going stale [17]. There are numerous experimental evidence 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. Fleming et al. showed subjects photographs of materials from different categories and asked them to rate various subjective qualities, such as hardnesss, glossiness and prettiness. 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 judgements of their properties.

Empirical studies have focused on the 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 [48], 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 [18]. 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 [17]. 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 enviriontment and object. Then a "try-and-see" approach was taken to obtain the parameters. 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 [9] where Berkiten and Rusinkiewicz used a synthetic dataset to find the contributing factors of various Photometric Stereo algorithms.

The lightness of the object is controlled by the albedo value. Albedo is defiend as the ratio of reflected light with respect to incident light. The albedo 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. We demonstrate the process using the following example data 'vase', as shown in Figure 4.5.



(a). synthetic sphere (b). synthetic vase (c). real-world vase

Figure 4.5: The UI for determining the property settings, including albedo, specular, and roughness of the surface. In this case shown above, the problem condition is: texture (0.8), albedo (0.8), specular (0.2), roughness(0.2). (a) demonstrates the effect of the property settings on a sphere, (b) on a teapot, and (c) shows the real-world object.

4.3 Expression

In this section, we introduce the expression of 3D reconstruction problem using the proposed model and representations, as shown in Table 4.2. As discussed in Section 4.2.5, it is intuitive for humans to perceptually estimate material properties. However, as previously discussed, the conditions under which a specific algorithm works in largely unclear, it is practically impossible to fully take advantage of an asortment of vision algorithms without sophisticated vision knowledge.

Model	**Representation**
Texture	*Texture randomness*
Lightness	*Diffuse albedo*
Specularity	*Specular reflectance*
Roughness	*SD of facet slopes*

Table 4.2: A Model and corresponding representations of the 3D reconstruction problem.

Now that we have a proposed model and representations of 3D reconstruction problem, we can express the four proposed problem conditions using this description. Given that all perceived estimates would likely be low resolution from users, we use three discrete scales to parameterize these properties: *low* (0.2), *medium* (0.5), and *high* (0.8). The expression of the reconstruction problem is shown in table 4.3.

Object	Texture	Albedo	Specular	Rough	**Label**
Class 1	low/med	high	low/med	high	Tl-B-D-R
Class 2	low/med	high	high	low/med	Tl-B-M-S
Class 3	high	high	low/med	high	T-B-D-R
Class 4	high	high	high	low/med	T-B-M-S

Table 4.3: Expression of the reconstruction problem for the four problem conditions proposed in Section 3.

Chapter 5

A Mapping of 3D Reconstruction

Most vision work focuses on developing algorithmic 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 *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 [57], and the 'DiLiGenT' dataset targets Photometric Stereo algorithms [59]. This makes such benchmarks only suitable for evaluation of within-category algorithms. Besides, there is few dataset 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/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 is too vast of a space to cope with. To make the problem space more tacklable, 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 dimensionality by considering only these effective properties. Effective properties of an algorithm are those that have a main effect or 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 discuss 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 algorithmic performance so that the problem space can become more manageable. Section 5.3 presents the lookup tables, represented by 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 baseline methods. Then the setups of synthetic capturing systems are presented with

example images. Lastly, quantitative measure used to evaluate the performance of algorithms are proposed.

5.1.1 Selected and baseline methods

We have selected one representative algorithm from three major classes of algorithms presented in Chapter 3: the PMVS proposed in [21], the example-based Photometric Stereo proposed in [27], and the Gray-code 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 these two kinds of patterns individually. A depth consistency step is performed to reject erroneous triangulations.

Technique	Summary
PMVS	Patch-based, seed points propagation MVS.
EPS	Example-based Photometric Stereo.
GSL	Gray code Structured Light technique.
VH	Volumetric Visual Hull.
LLS-PS	Linear least squares Photometric Stereo.

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

We use two baseline approaches 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 algorithms that work 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 $35mm$ or $1400pix$. The synthetic setups are shown in Table 5.2, and some example synthetic images generated using the setups are shown in Figure **??**.

Method	Hardwares	Arrangement
MVS	41 camera	5 rings, each has 1, 8, 8, 12, 12 camera
PS	1 camera+25 light sources	4 rings, each has 1, 8, 8, 8, 8 light sources
SL	1 camera&projector	baseline angle: 10°

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.

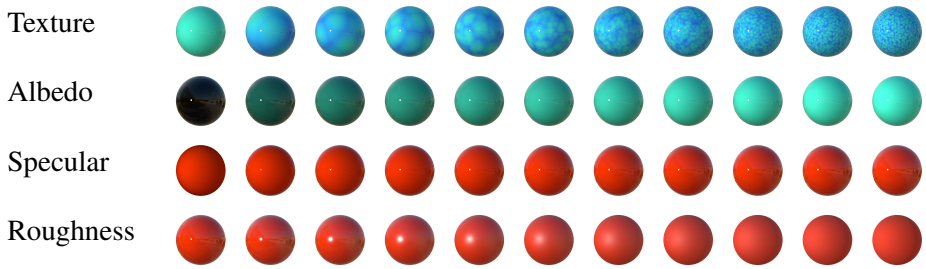


Figure 5.1: Example synthetic images. The value of each property ranges from 0 to 1.

5.1.3 Quantitative measures

We use the metrics proposed in [57] 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 Main effects and interactions of properties

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 3×3 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 choose to interpret the main effects and interactions through graphs. There is a main effect of a property if there is color variation along the corresponding axis. If color changes monotonically along the diagonal, then there is no interaction between the two properties, otherwise, there is an interaction effect.

		Property i		
		0.2	0.5	0.8
Property j	0.2			
	0.5			
	0.8			

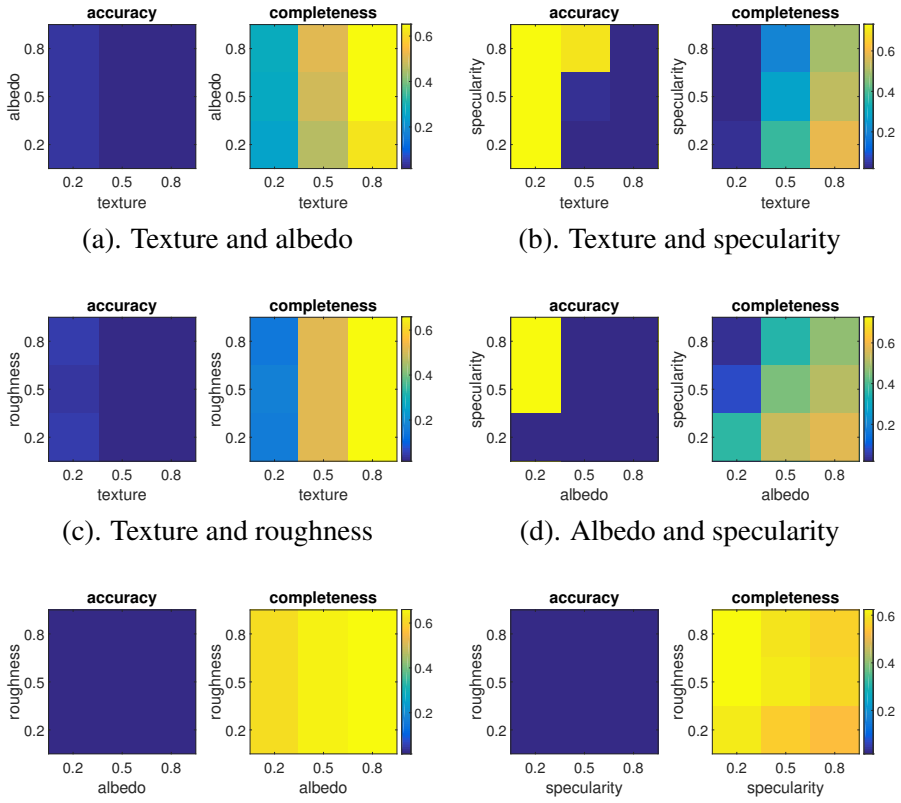
Table 5.3: This is a 3×3 factorial design. Every two properties are selected to test the main effects and interaction, there are in total $\binom{N}{2}$ combinations.

5.2.1 Main effects and interactions: 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 texture 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



(a). Texture and albedo

(b). Texture and specularity

(c). Texture and roughness

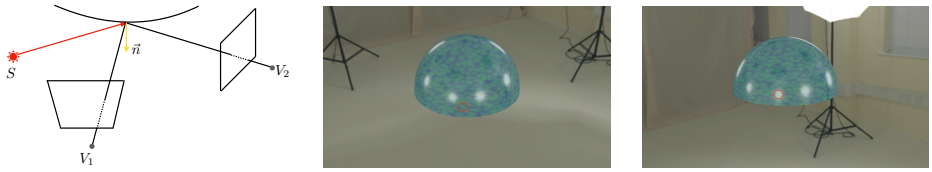
(d). Albedo and specularity

(e). Albedo and roughness

(f). Specularity and roughness

Figure 5.2: Performance of PMVS under four 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 color variation along the corresponding axis. The monotonic color 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.

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.



(a) Image formation (b) V_1 (c) V_2

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 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 spcularity 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.

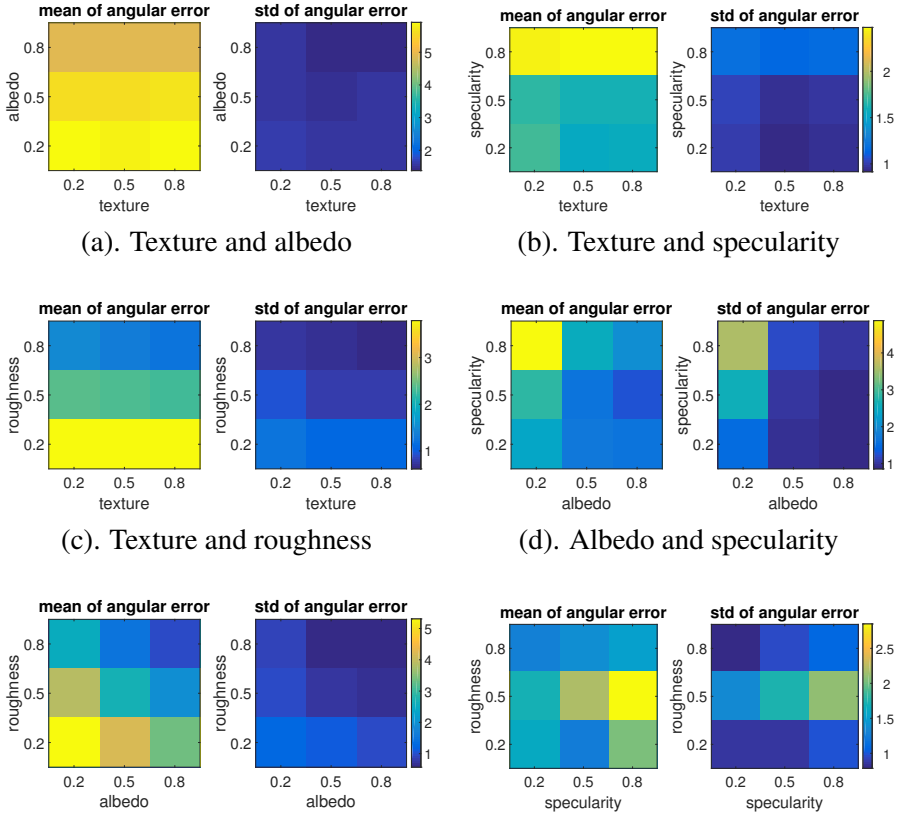
5.2.2 Main effects and interactions: EPS

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 speculairty 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 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



(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 5.4: 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 color variation along the corresponding axis. The monotonic color 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.

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 most significant on low albedo surfaces. There is also an interaction between specularity and roughenss such that surfaces with medium roughness lead to higher angular error compared to surface with low or high roughness.

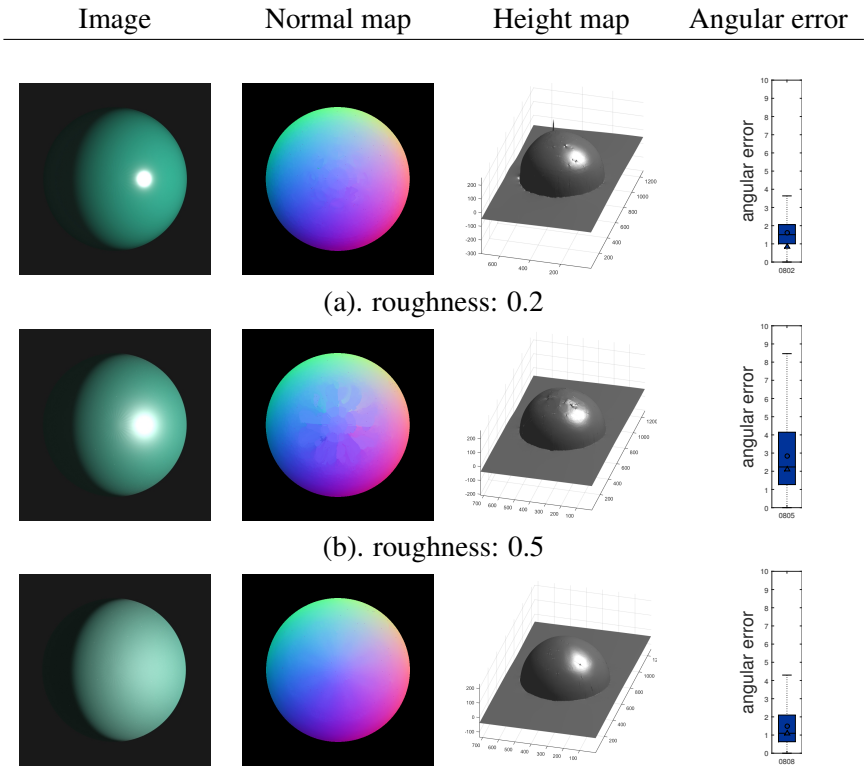
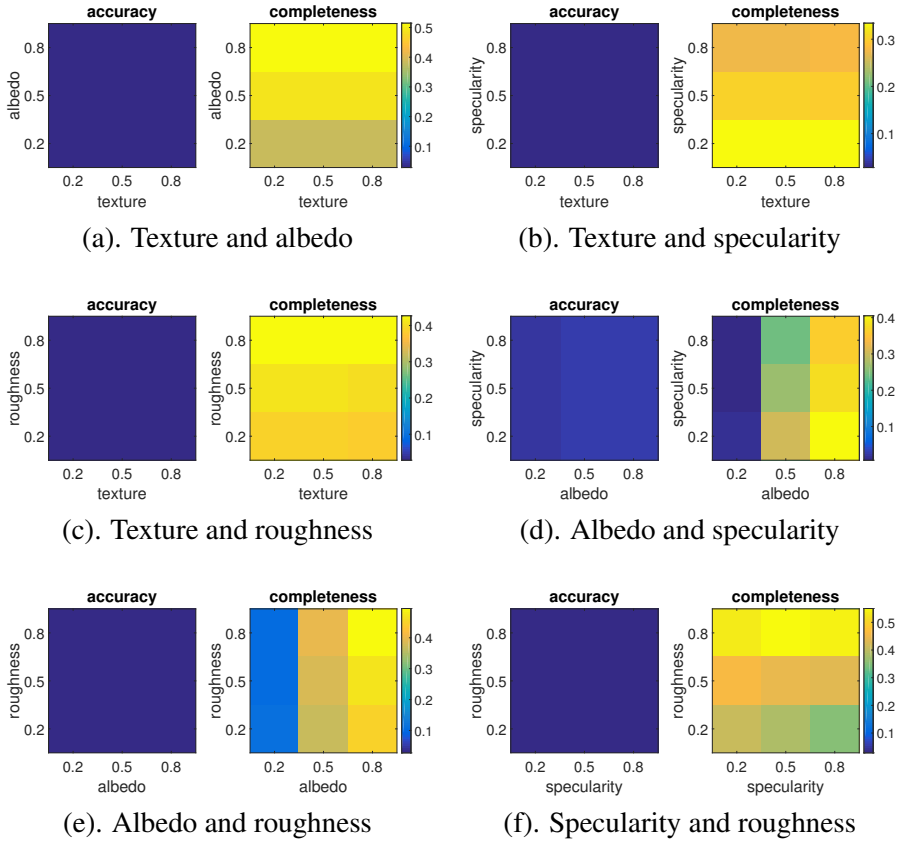


Figure 5.5: An example illustrates the effect of roughness on PS. Albedo is set as 0.8, and specular is set as 0.8. The first column shows the input images, the second column shows the estimated normal map, the third column shows the integrated surface, and last column shows the angular error. We can 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.

5.2.3 Main effects and interactions: 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

(b). Texture and specularity

(c). Texture and roughness

(d). Albedo and specularity

(e). Albedo and roughness

(f). Specularity and roughness

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 color variation along the corresponding axis. The monotonic color 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.

(a) Texture and Albedo The main effect of texture on accuracy and complete-

ness 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 and roughness in terms of accuracy and completeness.

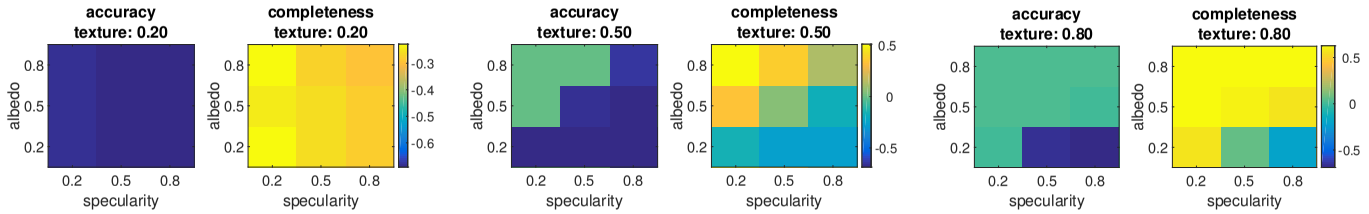
(f) Specular and Roughness The main effects of specularity and roughness on accuracy are not significant whereas the main effect on completeness are significant such that the results increases as the roughness increases or specularity decreases. There is no significant interaction effect between specularity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

Summary: GSL

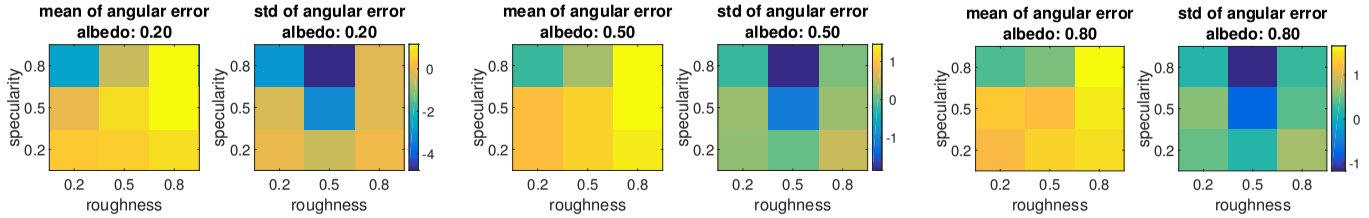
There is no main effects or interaction effects observed on the algorithm in terms of accuracy. Albedo, specularity, and roughness all have main effects on the algorithm in terms of completeness. There are no interaction effects observed.

5.3 Construction of Mapping

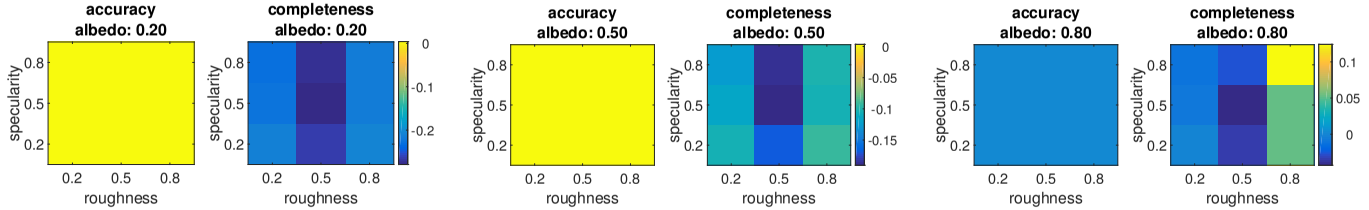
In the previous section, we have examined the performance of algorithms with two changing properties at a time. This is equivalent to examine the performance of algorithms on a $2-$dimensional plane embedded in a $N-$dimensional space. It gives us insights into which properties have significant impacts 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, i.e., positive value represents result better than that of baseline, we inverse the results of accuracy and angular error. Thus, higher value indicates better reconstruction, which indicates that the corresponding algorithm is a successful candidate. 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 ε, which is the minimum distance between the quantitative measures of selected algorithms and those of baseline methods. Any algorithm which is at least ε above the baseline methods are considered algorithms that can work reliably under the specified problem condition. By default, $\varepsilon = 0$.



(a). Look-up table of PMVS.



(b). Look-up table of EPS.



(c). Look-up table of GSL.

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.

Metric	Texture	Albedo	Specular	Roughness
Accuracy	0.5	0.5	0.2	-
&Completeness	0.5	0.8	0.2	-
	0.5	0.8	0.5	-
	0.8	0.2	0.2	-
	0.8	0.5	0.2	-
	0.8	0.8	0.2	-
	0.8	0.5	0.5	-
	0.8	0.8	0.5	-
	0.8	0.5	0.8	-
	0.8	0.8	0.8	-

Table 5.4: The problem conditions under which PMVS works successfully in terms of the two metrics *accuracy* and *completeness*.

Metric	Texture	Albedo	Specular	Roughness
Angle difference	-	0.2	0.2	0.8
	-	0.2	0.5	0.8
	-	0.2	0.8	0.8
	-	0.5	0.2	0.8
	-	0.5	0.5	0.8
	-	0.5	0.8	0.8
	-	0.8	0.2	0.2
	-	0.8	0.2	0.8
	-	0.8	0.5	0.2
	-	0.8	0.5	0.8
	-	0.8	0.8	0.2
	-	0.8	0.8	0.8

Table 5.5: The problem conditions under which example-based PS works successfully in terms of the metric *angular error*.

Metric	Texture	Albedo	Specular	Roughness
Accuracy	-	-	-	-
Completeness	-	0.8	0.2	0.2
	-	0.8	0.5	0.2
	-	0.8	0.8	0.2
	-	0.8	0.2	0.8
	-	0.8	0.5	0.8
	-	0.8	0.8	0.8

Table 5.6: The problem conditions under which Gray-code SL works successfully in terms of the two metrics *accuracy* and *completeness*.

5.4 Summary

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.

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-centric 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 an 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 suffice to demonstrate the usage of interface. The interpreter is responsible for receiving a description from the user, and select 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 interpreter with less accurate, or inaccurate descriptions as inputs. It gives us insights into the sensitivity of the interpreter, and

demonstrate 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 testbench is made publicly available to encourage the testing of additional reconstruction algorithms. Further, the testbench can be extended to include new visual/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 centered 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 algorithmic 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 centered 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 an inaccurate description give a poorer reconstruction result than an accurate description?

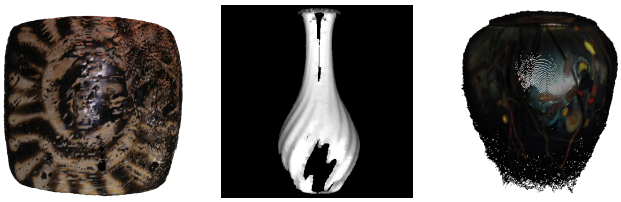
3. Will a inaccurate description give a poor reconstruction result?

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 sensitiveity, 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 comparision 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 the 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.



Rough surface Surface hole Surface spike

Figure 6.1: Visual phenomena that indicate the quality of reconstruction results [better images to be changed].

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 return 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 algorithmic performance across difference ranges of problem space to create a suitable interpreter.

However, if the interpreter relies solely on the mapping, it is possible that a user-specified description has more than one corresponding algorithm available. 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 necessary achieve high completeness. In light of this, the user can choose an algorithm based on the priority level of accuracy and completeness. If multiple algorithm with comparable accuracy or completeness exists, 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 6.2. 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.

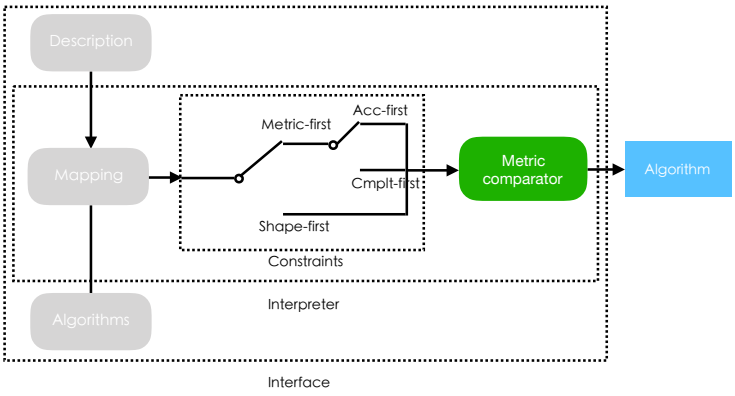


Figure 6.2: Two components of the Interpreter layer. The metric comparator compares the quantitative measures based on the constraints. If 'acc-first' is selected, the comparator favours algorithms with lower accuracy value, whereas if 'cmplt-first' is selected, the comparator favours algorithms with higher completeness. If 'shape-first' is selected, the comparator favours algorithms with lower angular error.

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 [57], DiLiGenT [59], DTU Robot Image Data Sets [31], and so on.

The synthetic dataset contains four objects, as shown in Figure 6.3, and the real-world dataset contains nine objects, four of which are shown in Figure 6.3. Please refer to Figure A.1 for the complete dataset. It covers materials that are rep-

resentative of 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 6.3.

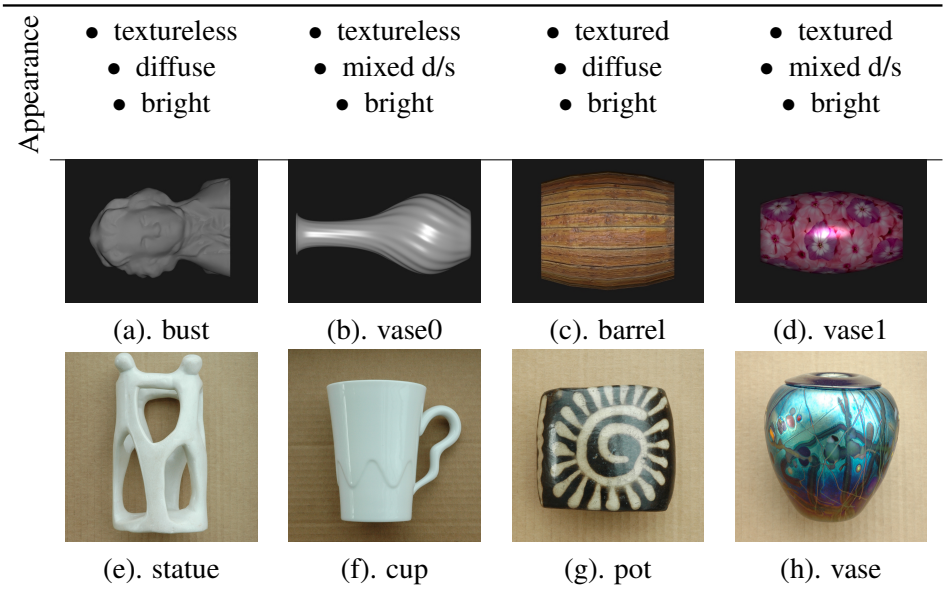


Figure 6.3: Representative objects of the four problem condition discussed in Chapter 3. (a)-(d): Synthetic objects, and (e)-(h) real-world objects.

6.3.1 Calibration

For MVS, a calibration pattern proposed in [40] is imaged under the object, which is used for calibrating the camera position and orientation by Structure from Motion softwares, such as VisualSfM [68]. The focal length is known a priori and remains fixed duing 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 recontruction.

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. Thus, we did not perform the radiometric calibration step. No geometric calibration of the camera is needed.

For SL, a opensource camera-projector calibration software developed by Moreno and Taubin is used for calibration. This technique works by projecting temporal patterns onto the calibration pattern, and uses local homography to individually translate each checkerboard 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 synthetic dataset is largely 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.

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 in at the center 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 12 image per height.

For PS, a $70-200mm$ lens, a handheld 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 everying 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.

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.

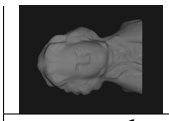
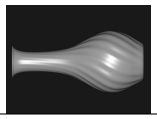
	Class #	1	2	3	4
Objects					
Semantic Description		• textureless • diffuse • bright	• textureless • mixed d/s • bright	• textured • diffuse • bright	• textured • mixed d/s • bright
	Tex	0.2	0.2	0.8	0.8
	Alb	0.8	0.8	0.8	0.8
	Spec	0.2	0.8	0.2	0.8
	Rough	0.8	0.2	0.2	0.2
Alg		• EPS • GSL	• EPS	• PMVS • EPS • GSL	• PMVS • EPS

Figure 6.4: The repsentatives 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 that matches 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 in colour red.

6.3.4 Real-world Dataset

We select four real-world objects, each represents the one of the four problem condtiions 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.

	Class #	1	2	3	4
Objects					
Verbal		• textureless • diffuse • bright	• textureless • mixed d/s • bright	• textured • diffuse • dark/bright	• textured • mixed d/s • dark/bright
Description	Tex	0.2	0.2	0.8	0.8
	Alb	0.8	0.8	0.8 (0.2)	0.8 (0.2)
	Spec	0.2	0.8	0.2	0.8
	Rough	0.8	0.2	0.2	0.2
Alg		• EPS • GSL	• EPS •	• PMVS • GSL	• PMVS

Figure 6.5: The repsentatives 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 that matches 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 in colour red.

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 accuracy; 3) given an inaccurate descrip-

tion, 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 6.3.

6.4.1 Evaluation 1: accurate description, successful result

In this section, we evaluate whether the interpreter can return a successful reconstruction result given a valid description of problem condition, see results in Figure 6.6. The figure is divided into two sections, one for synthetic dataset and another for real-world dataset. The 'Algo' row shows the algorithm selected by the interpreter, which also appear as the algorithm in red text in Figure 6.4, 6.5. The 'Results' row shows the reconstruction results of the corresponding algorithm, and the 'Baseline' row demonstrate the results of the baseline method. 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, obtrutions, 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 selection of objects do no favor any algorithm. In fact, as we will show below, it is highly possible to get a poor result if another algorithm is chosen instead of the interpreted one. The significance of this study is that we demonstrated 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 param-

eters.

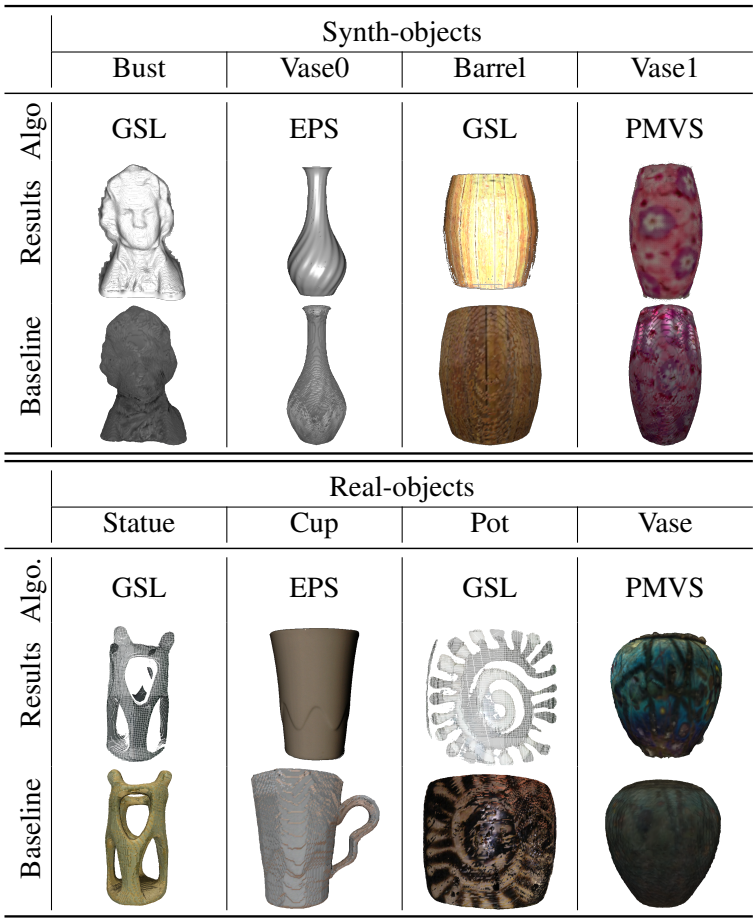


Figure 6.6: Evaluation 1: correct description leads to successful reconstruction result. The baseline results are provided so that we can determine the quality of result returned by the algorithm chosen by the interpreter.

6.4.2 Evaluation 2: less accurate description, less successful result

We have demonstrated in last section that the interpreter can achieve successful results given accurate description. However, how would the interpreter perform given an inaccurate description? There are two guesses: 1) the interpreter would return poor result given an inaccurate 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 inaccurate descriptions, and the cases where it would fail or succeed. We provide four inaccurate desriptions by iterating through four properties, and each time one and only one property is set correctly. For instance, prop_i is correctly estimated for desc_i while the rest of the properties are set incorrectly, $i \in \{1,2,3,4\}$. Since a description could be mapped to multiple algorithms, a description that does not match the object could potentially have a successful reconstruction result as well.

The layout of the graph is as follows: columnwise, $\text{Desc}_i, i \in \{1,2,3,4\}$ presents results where an inaccurate description is provided while the last column presents the results where the correct description is provided. Rowwise, for each object, the result is divided into three segments: The description of problem condition is presented in the first segment, which is coloured-coded to indicate the correctly set property. The order of properties in the description is: texture, albedo, specularity, and roughness. Since there is one and only one correctly estimated property in each inaccurate description, it is coloured in red while the incorrectly 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.

Let's denote the algorithm(s) returned by the mapping given description Desc_i as $\{\text{Algo}_i\}$. For instance, the algorithms for object *bust* given description Desc_2 is $\text{Algo}_2 = \{PMVS, EPS, GSL\}$. Let's also denote the algorithm(s) returned by the correct description as $\{\text{Algo}\}$. Thus in the case of object *bust*, $\{\text{Algo}\} = \{EPS, GSL\}$. We observe that an inaccurate description can possibly lead to a successful reconstruction if and only if

$$\{\text{Algo}_i\} \bigcap \{\text{Algo}\} \neq \emptyset$$

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 an inaccurate description, a successful reconstruction result is still achievable. Let's use *bust* as an example, Desc_1 and Desc_3 lead to poor results whereas

Object	Descriptions and Results				
	$Desc_1$	$Desc_2$	$Desc_3$	$Desc_4$	Correct Desc
Bust	02020802 • BL	08080802 • PMVS • EPS • GSL	08020202 • PMVS	08020808 • EPS	02080208 • EPS • GSL
					
vase0	02080208 • EPS • GSL	08080208 • PMVS • EPS • GSL	08020802 • BL	08020202 • PMVS	02080802 • EPS
					
barrel	08020802 • PMVS	02080802 • EPS • GSL	02020202 • BL	02020808 • EPS	08080208 • PMVS • EPS • GSL
					
vase1	08020208 • PMVS • EPS	02080208 • EPS • GSL	02020808 • BL	02020202 • BL	08080802 • PMVS • EPS
					

Figure 6.7: Evaluation 2: less accurate description may lead to poor result. Desc$_i$ represents inaccurate descriptions. For each object, the first row represent the description, with the correctly estimated property coloured in red while the incorrect 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.

Object	Descriptions and Results				
	$Desc_1$	$Desc_2$	$Desc_3$	$Desc_4$	Correct Desc
statue	02020802 • BL	08080802 • PMVS • EPS • GSL	08020202 • PMVS	08020808 • EPS	02080208 • EPS • GSL
					
cup	02080208 • EPS • GSL	08080208 • PMVS • EPS • GSL	08020802 • BL	08020202 • PMVS	02080802 • EPS
					
pot	08020802 • PMVS	02080802 • EPS • GSL	02020202 • BL	02020808 • EPS	08080208 • PMVS • EPS • GSL
					
vase	08020208 • PMVS • EPS	02080208 • EPS • GSL	02020808 • BL	02020202 • BL	08080802 • PMVS • EPS
					

Figure 6.8: Evaluation 2: less accurate description may lead to poor reconstruction results. $Desc_i$ represents inaccurate descriptions. For each object, the first row represent the description, with the correctly estimated property coloured in red while the incorrect 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.

Desc_2 and Desc_4 result in successful reconstructions. We can see that neither Desc_1 nor Desc_3 is mapped to any algorithm that is in common with those returned given accurate description. Further, both Desc_2 and Desc_4 have at least one mapped algorithm that coincides with the algorithms returned given accurate description. The take-away message is that for an incorrect 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.

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 algorithms mapped from inaccurate description coincide with algorithms returned from accurate description, it is still possible to achieve successful reconstruction. In this section, we are interested to see if this conclusion can also be applied to complete 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.

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 chosen by the interpreter coloured in red. The reconstruction result by the interpreted algorithm is shown as well.

From the study of less accurate desription, we have already found out that it is still possible to achieve a successful result given an inaccurate description. However, it becomes more challenging, and more likely to select a less successful algorithm or the baseline method. There is a certain limit to the tolerance of inaccuracy of property estimation. Thus, the take-away message is that the interpreter, in some cases, can still performs 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.

Object	Bust	Vase0	Barrel	Vase1
Incorrect Desc	08020802 • BL	08020208 • PMVS • EPS	02020802 • BL	02020208 • EPS
				
Correct Desc	02080208 • EPS • GSL	02080802 • EPS	08080208 • PMVS • EPS • GSL	08080802 • PMVS • EPS
				

Object	Statue	Cup	Pot	Vase
Incorrect Desc	08020802 • BL	08020208 • PMVS • EPS	02020802 • BL	02020208 • EPS
				
Correct Desc	02080208 • EPS • GSL	02080802 • EPS	08080208 • PMVS • EPS • GSL	08080802 • PMVS • EPS
				

Figure 6.9: Evaluation 3: inaccurate description may lead to poor result. For each description, the first row represent the settins 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 then those of accurate descriptions.

6.5 Summary

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 6.10.

The figure layout is as follows: each description Desc_i only matches the problem condition of object i, which is the object in the ith 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 complex descriptions of object geometry and material, incorporating new algorithms, and improving mapping are all ongoing processes to improve the interface presented.

Desc #	Bust	Vase0	Barrel	Vase1	Interp Algo.
1					GSL
2					EPS
3					GSL
4					PMVS

Desc #	Statue	Cup	Pot	Vase	Interp Algo.
1					GSL
2					EPS
3					GSL
4					PMVS

Figure 6.10: 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*, which is labeled 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 labeled objects indicates success/failure of the interpreter.

Chapter 7

Conclusions

The main contribution of this thesis is the development and application of an interface to 3D reconstruction problem. The significance of this thesis is that not only does it provide a means to reconstruct more general objects, but also does so without requiring knowledge of vision expertise. With the increased sophistication of the vision algorithms, comes the increased barriers to applying these algorithms in real-world applications. Just as personal computer did not become mainstream until the graphical user interface, computer vision algorithm will not become game changing until creative minds from all discplines 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.

In the thesis we have presented a taxonomy for 3D reconstruction algorithms based on the conditions surrounding the problem. This object centric taxonomy allows application developers access to advanced vision techniques without requiring sophisticated vision knowledge of the underlying algorithms.

We then discussed a description to 3D reconstruction building upon the proposed taxonomy, which provides an abstraction layer above the underlying vision algorithms. The proposed description provides a formal and definitive way to represent the problem conditions of a 3D reconstruction problem, and based on which, the performance of the algorithm can be evaluated, allowing for a better understanding of the working conditions surround the specific algorithm.

Once we obtained the means to represent problem conditions, we set out to investigate the optimal working conditions surrounding algorithms. This information provides a deeper understanding of performance of algorithms and potentially providing insights into how they may be improved.

Lastly, we demonstrated the usage of the interface by a proof-of-concept interpreter, which returns a successful reconstruction result given a valid description of object's visual and geometric properties.

Extending beyong 3D reconstruction, our proposed general framework of vision is designed to allow all vision tasks more accessible to application developers by providing a description of vision which allows for the description of the problem conditions. In order to provide such accessibility, a representation of the problem space and a means to find the optimal problem space must be well defined. This is a non-trivial task and requires an understanding of the field and an ability to abstract away algorithmic complexity.

7.1 Future directions

3D reconstruction has been one of the most important sub-fields 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 this opens up some potential future directions that can improve the work completed in this thesis.

7.1.1 Geometric Model

The current geometric model fails to capture the complexity of real world objects and focuses mainly on visual properties. Thus, one of the future directions is to develop intuitive geometric models to better describe complex objects.

7.1.2 Property Parameters

We have used a "try-and-see" approach to obtain the property settings of an object, which is based on user judgement and is thus not very rigorous and tends to be tedious. We can use machine learning techniques to obtain visual and geometric information.

7.1.3 Metrics

We have utilized three metrics: accuracy, completeness and angular error. However, there are other measures worth investigating, such as colour accuracy, 'ghost' reconstruction error, and so on. Additional metrics such as these can extend the application of our framework, providing more options for developers to choose from.

7.1.4 Mapping Construction

The construction of mapping requires an evaluation of the corresponding algorithm for pairwise properties, which tends not to scale well with respect to the number of properties. Therefore, we need better ways to discover the dependent relation bewteen any two properties.

7.1.5 Interpreter

Currently, 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.

7.2 Closure

Nowadays, computer vision has been playing a increasing important role in a variety of fields. However, it also takes application developers increasing expertise to apply these technologies to a specific domain. We hope that the efforts in the vision community can be directed to not only develop more advanced algorithms, but to easier access to these algorithms as well.

Bibliography

[1] Autodesk. URL http://en.wikipedia.org/wiki/Autodesk. → pages 1

[2] Lidar. URL http://en.wikipedia.org/wiki/Lidar. → pages 1

[3] Kinect. URL http://en.wikipedia.org/wiki/Kinect. → pages 1

[4] Vxl c++ libraries for computer vision research and implementation. http://vxl.sourceforge.net. → pages 8

[5] N. G. Alldrin and D. J. Kriegman. Toward reconstructing surfaces with arbitrary isotropic reflectance: A stratified photometric stereo approach. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007. → pages 19

[6] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 28(3), Aug. 2009. → pages 11

[7] S. Barsky and M. Petrou. The 4-source photometric stereo technique for three-dimensional surfaces in the presence of highlights and shadows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10): 1239–1252, 2003. → pages 19

[8] P. N. Belhumeur, D. J. Kriegman, and A. L. Yuille. The bas-relief ambiguity. *International journal of computer vision*, 35(1):33–44, 1999. → pages 18

[9] S. Berkiten and S. Rusinkiewicz. An RGBN benchmark. Technical report, Technical Report TR-977-16, Princeton University, Feb. 2016. → pages 37

[10] F. Bernardini, H. Rushmeier, I. M. Martin, J. Mittleman, and G. Taubin. Building a digital model of michelangelo's florentine pieta. *IEEE Computer Graphics and Applications*, 22(1):59–67, 2002. → pages 1

[11] F. Blais. Review of 20 years of range sensor development. *Journal of Electronic Imaging*, 13(1), 2004. → pages 10

[12] R. C. Bolles and P. Horaud. 3dpo: A three-dimensional part orientation system. *The International Journal of Robotics Research*, 5(3):3–26, 1986. → pages 30

[13] G. Bradski and A. Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc.", 2008. → pages 8

[14] E. N. Coleman and R. Jain. Obtaining 3-dimensional shape of textured and specular surfaces using four-source photometry. *Computer graphics and image processing*, 18(4):309–328, 1982. → pages 19

[15] C. H. Esteban and F. Schmitt. Silhouette and stereo fusion for 3d object modeling. *Computer Vision and Image Understanding*, 96(3):367–392, 2004. → pages 11

[16] O. Faugeras and R. Keriven. *Variational principles, surface evolution, pde's, level set methods and the stereo problem*. IEEE, 2002. → pages 1, 11

[17] R. W. Fleming. Visual perception of materials and their properties. *Vision research*, 94:62–75, 2014. → pages 36

[18] R. W. Fleming, R. O. Dror, and E. H. Adelson. Real-world illumination and the perception of surface reflectance properties. *Journal of vision*, 3(5):3–3, 2003. → pages 36

[19] R. W. Fleming, C. Wiebel, and K. Gegenfurtner. Perceptual qualities and material classes. *Journal of vision*, 13(8):9–9, 2013. → pages 36

[20] S. Fuhrmann, F. Langguth, and M. Goesele. Mve-a multiview reconstruction environment. In *Proceedings of the Eurographics Workshop on Graphics and Cultural Heritage (GCH)*, volume 6, page 8, 2014. → pages 9

[21] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8): 1362–1376, 2010. → pages 1, 9, 11, 41

[22] S. Galliani, K. Lasinger, and K. Schindler. Massively parallel multiview stereopsis by surface normal diffusion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 873–881, 2015. → pages 12

[23] M. Goesele, B. Curless, and S. M. Seitz. Multi-view stereo revisited. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2402–2409. IEEE, 2006. → pages 1, 12

[24] D. B. Goldman, B. Curless, A. Hertzmann, and S. M. Seitz. Shape and spatially-varying brdfs from photometric stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(6):1060–1071, 2010. → pages 19

[25] H. Hayakawa. Photometric stereo under a light source with arbitrary motion. *JOSA A*, 11(11):3079–3089, 1994. → pages 18

[26] J. Heller, M. Havlena, M. Jancosek, A. Torii, and T. Pajdla. 3d reconstruction from photographs by cmp sfm web service. In *Machine Vision Applications (MVA), 2015 14th IAPR International Conference on*, pages 30–34, May 2015. doi:10.1109/MVA.2015.7153126. → pages 9

[27] A. Hertzmann and S. M. Seitz. Example-based photometric stereo: Shape reconstruction with general, varying brdfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1254–1264, 2005. → pages 19, 41, 63

[28] V. H. Hiep, R. Keriven, P. Labatut, and J.-P. Pons. Towards high-resolution large-scale multi-view stereo. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1430–1437. IEEE, 2009. → pages 11

[29] B. K. Horn. Shape from shading: A method for obtaining the shape of a smooth opaque object from one view. 1970. → pages 15

[30] I. Ihrke, K. N. Kutulakos, H. Lensch, M. Magnor, and W. Heidrich. Transparent and specular object reconstruction. In *Computer Graphics Forum*, volume 29, pages 2400–2426. Wiley Online Library, 2010. → pages 26

[31] R. Jensen, A. Dahl, G. Vogiatzis, E. Tola, and H. Aanæs. Large scale multi-view stereopsis evaluation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 406–413. IEEE, 2014. → pages 61

[32] H. Jin, S. Soatto, and A. J. Yezzi. Multi-view stereo beyond lambert. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2003. → pages 13

[33] H. Jin, S. Soatto, and A. J. Yezzi. Multi-view stereo reconstruction of dense shape and complex appearance. *International Journal of Computer Vision*, 63(3):175–189, 2005. → pages 13

[34] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006. → pages 9

[35] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing. Available from: <http://www.peterkovesi.com/matlabfns/>. → pages 8

[36] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. *International Journal of Computer Vision*, 38(3):199–218, 2000. → pages 1, 11

[37] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, et al. The digital michelangelo project: 3d scanning of large statues. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 131–144. ACM Press/Addison-Wesley Publishing Co., 2000. → pages 1

[38] M. Lhuillier and L. Quan. Match propagation for image-based modeling and rendering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8):1140–1146, 2002. → pages 11

[39] M. Lhuillier and L. Quan. A quasi-dense approach to surface reconstruction from uncalibrated images. *IEEE transactions on pattern analysis and machine intelligence*, 27(3):418–433, 2005. → pages 11

[40] B. Li, L. Heng, K. Koser, and M. Pollefeys. A multiple-camera system calibration toolbox using a feature descriptor-based calibration pattern. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1301–1307. IEEE, 2013. → pages 62

[41] J. J. Little. *Recovering shape and determining attitude from extended gaussian images*. PhD thesis, 1985. → pages 30

[42] S. P. Mallick, T. E. Zickler, D. J. Kriegman, and P. N. Belhumeur. Beyond lambert: Reconstructing specular surfaces using color. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 619–626. Ieee, 2005. → pages 19

[43] G. Mariottini and D. Prattichizzo. Egt: a toolbox for multiple view geometry and visual servoing. *IEEE Robotics and Automation Magazine*, 3(12), December 2005. → pages 8

[44] D. Marr. Vision: A computational investigation into the human representation and processing of visual information. 1982. → pages 11

[45] D. Moreno and G. Taubin. Simple, accurate, and robust projector-camera calibration. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*, pages 464–471. IEEE, 2012. → pages 63

[46] P. Moulon, P. Monasse, R. Marlet, and Others. Openmvg. an open multiple view geometry library. https://github.com/openMVG/openMVG. → pages 9

[47] S. K. Nayar, K. Ikeuchi, and T. Kanade. Surface reflection: physical and geometrical perspectives. Technical report, DTIC Document, 1989. → pages 34

[48] S. Nishida and M. Shinya. Use of image-based information in judgments of surface-reflectance properties. *JOSA A*, 15(12):2951–2965, 1998. → pages 36

[49] G. P. Otto and T. K. Chau. region-growingalgorithm for matching of terrain images. *Image and vision computing*, 7(2):83–94, 1989. → pages 11

[50] T. Poggio, V. Torre, and C. Koch. Computational vision and regularization theory. *Nature*, 317(6035):314–319, 1985. → pages 9

[51] C. Rocchini, P. Cignoni, F. Ganovelli, C. Montani, P. Pingi, and R. Scopigno. Marching intersections: an efficient resampling algorithm for surface management. In *Shape Modeling and Applications, SMI 2001 International Conference on.*, pages 296–305. IEEE, 2001. → pages 21

[52] S. Roy and I. J. Cox. A maximum-flow formulation of the n-camera stereo correspondence problem. In *Computer Vision, 1998. Sixth International Conference on*, pages 492–499. IEEE, 1998. → pages 11

[53] J. Salvi, J. Pages, and J. Batlle. Pattern codification strategies in structured light systems. *Pattern recognition*, 37(4):827–849, 2004. → pages 13, 14

[54] Y. Sato and K. Ikeuchi. Temporal-color space analysis of reflection. *JOSA A*, 11(11):2990–3002, 1994. → pages 19

[55] K. Schluns. Photometric stereo for non-lambertian surfaces using color information. In *International Conference on Computer Analysis of Images and Patterns*, pages 444–451. Springer, 1993. → pages 19

[56] S. M. Seitz and C. R. Dyer. Photorealistic scene reconstruction by voxel coloring. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 1067–1073. IEEE, 1997. → pages 10

[57] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 519–528. IEEE, 2006. → pages 1, 10, 12, 39, 42, 61

[58] L. Sharan, R. Rosenholtz, and E. Adelson. Material perception: What can you see in a brief glance? *Journal of Vision*, 9(8):784–784, 2009. → pages 36

[59] B. Shi, Z. Wu, Z. Mo, D. Duan, S.-K. Yeung, and P. Tan. A benchmark dataset and evaluation for non-lambertian and uncalibrated photometric stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3707–3716, 2016. → pages 39, 61

[60] W. M. Silver. *Determining shape and reflectance using multiple images*. PhD thesis, Massachusetts Institute of Technology, 1980. → pages 19

[61] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM transactions on graphics (TOG)*, volume 25, pages 835–846. ACM, 2006. → pages 9

[62] Y. Uh, Y. Matsushita, and H. Byun. Efficient multiview stereo by random-search and propagation. In *3D Vision (3DV), 2014 2nd International Conference on*, volume 1, pages 393–400. IEEE, 2014. → pages 12

[63] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008. → pages 8

[64] G. Vogiatzis, P. H. Torr, and R. Cipolla. Multi-view stereo via volumetric graph-cuts. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 391–398. IEEE, 2005. → pages 11

[65] G. Vogiatzis, C. H. Esteban, P. H. Torr, and R. Cipolla. Multiview stereo via volumetric graph-cuts and occlusion robust photo-consistency. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12): 2241–2246, 2007. → pages 11, 12

[66] R. J. Woodham. Photometric stereo: A reflectance map technique for determining surface orientation from image intensity. In *22nd Annual Technical Symposium*, pages 136–143. International Society for Optics and Photonics, 1979. → pages 15

[67] R. J. Woodham. Photometric method for determining surface orientation from multiple images. *Optical engineering*, 19(1):191139–191139, 1980. → pages 1, 16

[68] C. Wu et al. Visualsfm: A visual structure from motion system. 2011. → pages 9, 62

[69] R. Zhang, P.-S. Tsai, J. E. Cryer, and M. Shah. Shape-from-shading: a survey. *IEEE transactions on pattern analysis and machine intelligence*, 21 (8):690–706, 1999. → pages 16

[70] E. Zheng, E. Dunn, V. Jojic, and J.-M. Frahm. Patchmatch based joint view selection and depthmap estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1510–1517, 2014. → pages 12

[71] T. E. Zickler, P. N. Belhumeur, and D. J. Kriegman. Helmholtz stereopsis: Exploiting reciprocity for surface reconstruction. *International Journal of Computer Vision*, 49(2-3):215–227, 2002. → pages 19

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.

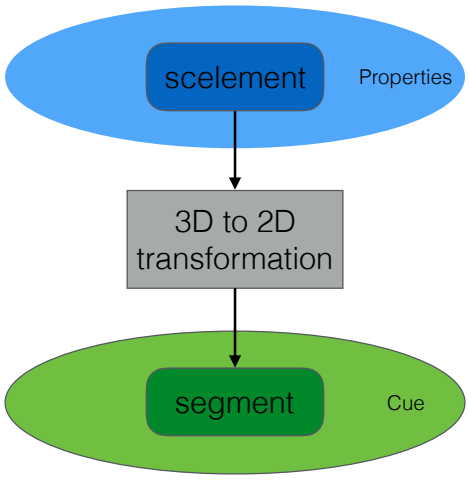


Figure A.1: Relation between a scell and a segment

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*.

$$consist(prop(sc), cue(seg)) = 1 \Rightarrow consistent$$
$$consist(prop(sc), cue(seg)) = 0 \Rightarrow not\ consistent$$

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., $\forall sc_i \in \{sc_m\}_{m=0}^{M-1}$, 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 betweem a scell *sc* and the corresponding segment *seg*.

$$\begin{aligned} &consist(prop(sc), cue(seg)) = x, x \in [0,1] \\ &consist(prop(sc), cue(seg)) = 1 \Rightarrow consistent \\ &consist(prop(sc), cue(seg)) = 0 \Rightarrow not\ consistent \end{aligned}$$

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

$$consist(prop(sc), cue(seg)) > \varepsilon$$

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 $\{seg_{(i,j)}\}_{i=0,j=0}^{M-1,N-1}$ are maximized.

$$\text{maximize} \quad \sum_{j=0}^{N-1} \sum_{i=0}^{M-1} consist(prop(sc_i), cue(seg_{(i,j)}))$$

A.2 Definition of Radiometric Terms

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

- Solid angle ($d\omega$): 3D counterpart of angle, $d\omega = \frac{dA\cos\theta_i}{R^2}(steradian)$.
- Projected solid angle ($d\Omega$): $d\Omega = \cos\theta d\omega$.
- Incident radiance ($\mathbf{L_i}(\theta_\mathbf{i}, \phi_\mathbf{i})$): light flux received from the direction (θ_i, ϕ_i) on a unit surface area, unit $(watt \cdot m^{-2} \cdot steradian^{-1})$.

- Irradiance ($\mathbf{E_i}(\theta_\mathbf{i}, \phi_\mathbf{i})$): light Flux (power) incident per unit surface area from all direction, $\mathbf{E_i}(\theta_\mathbf{i}, \phi_\mathbf{i}) = \int_{\Omega_i} L_i(\theta_i, \phi_i) d\Omega_i (watt/m^2)$.
- Surface radiance ($\mathbf{L_r}(\theta_\mathbf{r}, \phi_\mathbf{r})$): light flux emmited from a unit surface area in the direction (θ_r, ϕ_r), unit $(watt \cdot m^{-2} \cdot steradian^{-1})$.

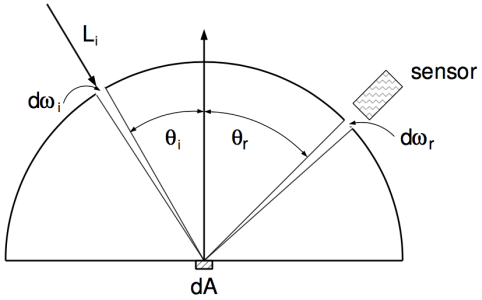


Figure A.2: Illustration of light-matter interaction.

A.3 Material of real-world objects

A.4 Parameters of real-world objects

A.5 Results of real-world objects

(a). box

(b). cat0

(c). cat1

(d). cup

(e). dino

(f). house

(g). pot

(h). statue

(i). vase

Table A.1: Images of the real-world objects.

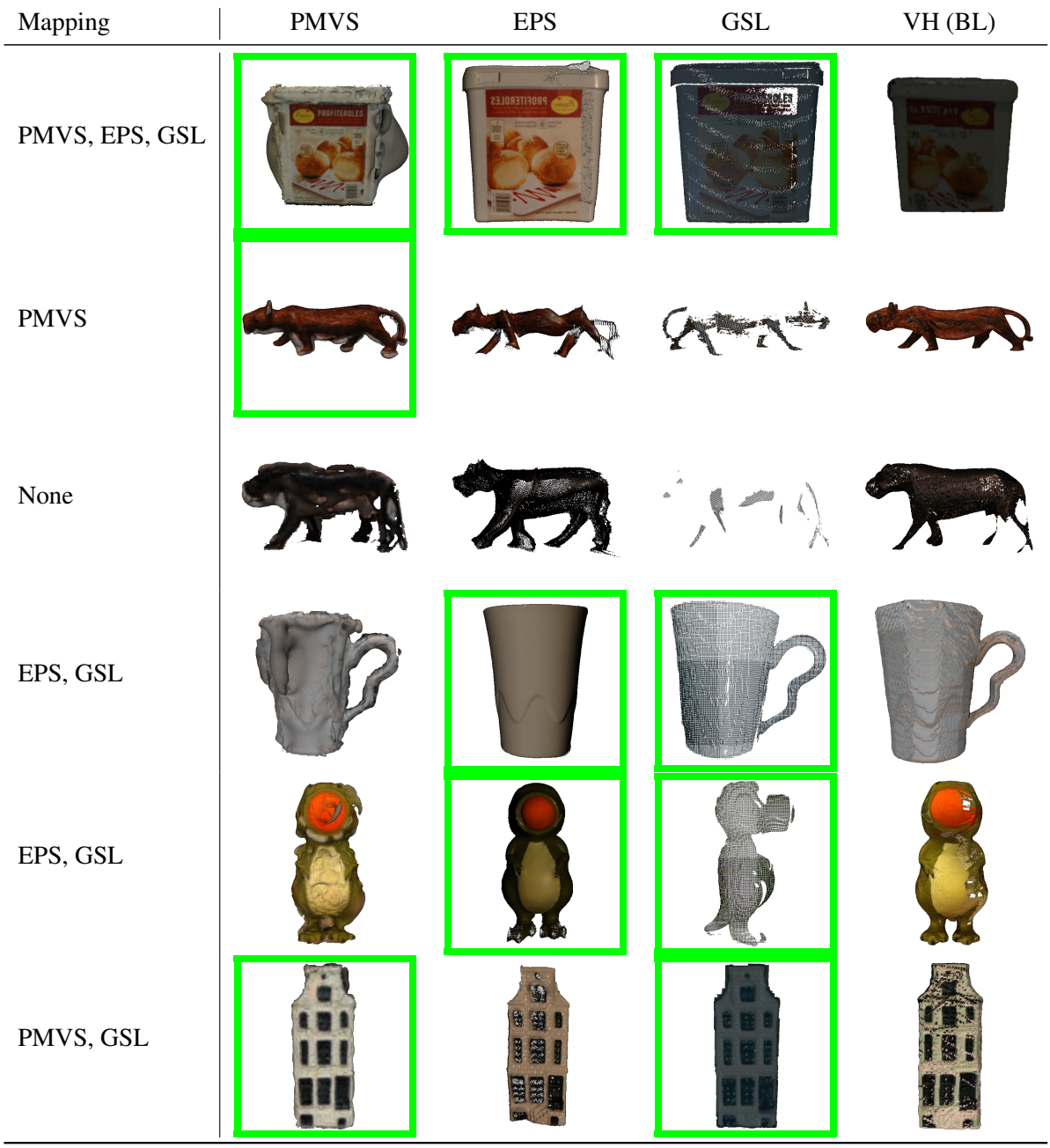


Figure A.3: Reconstruction results of MVS, PS, SL, and the baseline method VH.

Class	Texture	Albedo	Specularity	Roughness	Mapping
box	0.8	0.8	0.2	0.8	PMVS, EPS, GSL
cat0	0.5	0.5	0.2	0.2	PMVS
cat1	0.2	0.2	0.2	0.2	None
cup	0.2	0.8	0.5	0.2	EPS, GSL
dino	0.2	0.5, 0.8	0.2	0.8	EPS, GSL
house	0.8	0.2, 0.8	0.2	0.2	PMVS, GSL
pot	0.8	0.2, 0.5	0.2	0.2	PMVS
status	0.2	0.8	0.2	0.8	EPS, GSL
vase	0.8	0.2, 0.5	0.5	0.2	PMVS

Table A.2: Property list for the real-world objects

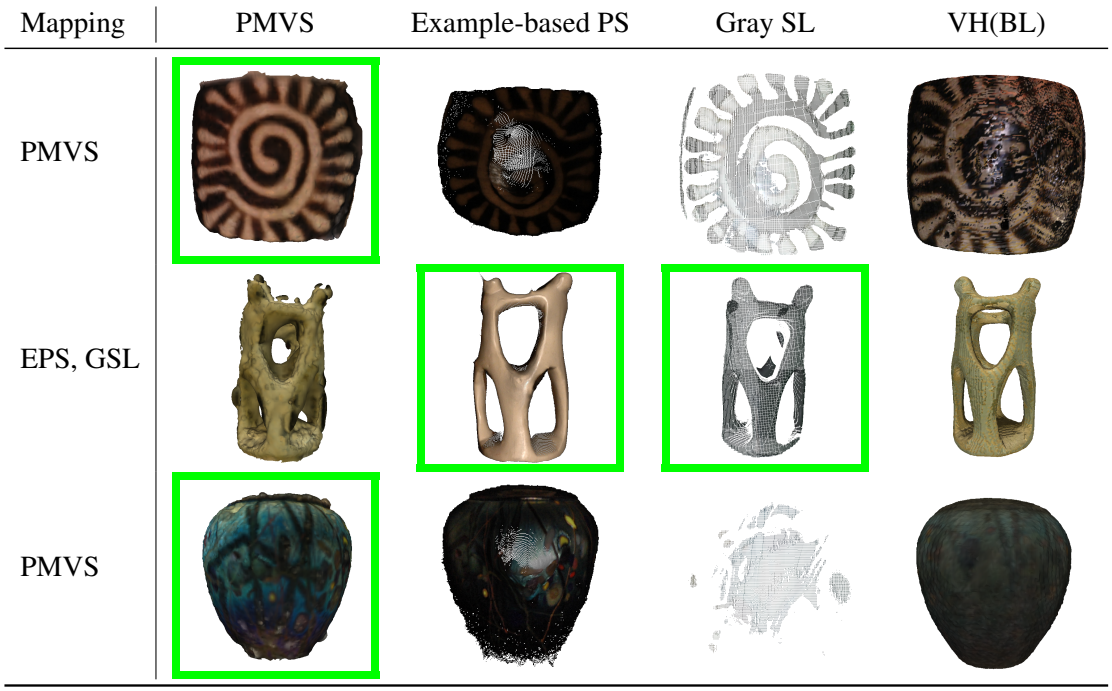


Figure A.4: Reconstruction results of MVS, PS, SL, and the baseline method VH (cont'd).