Development and Evaluation of 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 that support the accessibility of these algorithms for application developers, especially to developers who are not experts in the field.

We present a novel interface, specifically for 3D reconstruction techniques, designed to allow users to reconstruct the shape of an object without knowledge of algorithmic details. The interface hides the details of algorithms by using a description of visual and geometric properties of the object. 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 robustness of interpreter using a synthetic and real-world dataset 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

Thank those people who helped you.

Don’t forget your parents or loved ones.

You may wish to acknowledge your funding sources.

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, 4, 13, 36] and passive image-based methods [19, 20, 23, 35]. Among the existing approaches, active techniques such as laser scanners [36], Structured Light (SL) systems [13], and Photometric Stereo (PS) [64], as well as passive methods such as Multi-View Stereo (MVS) [53], 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 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.

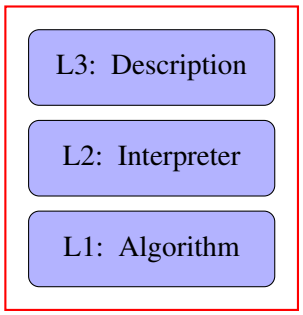


Figure 1.1: The three layers of the 3D reconstruction interface.

1.1 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.

1.1.1 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.

1.1.2 A Taxonomy of 3D Reconstruction

The proposed taxonomy categorizes algorithms not based on *how* they work, but on the problem space that they can reliably work under. First, the problem space is defined, with each axis representing a key property of the object's material or

geometry. Next, a selected classes of algorithms are registered to the problem conditions based on relevant literature reports.

1.1.3 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.

1.1.4 A Mapping of 3D Reconstruction

To derive 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.

1.1.5 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: 1) 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; 2) we use synthetic and real-world objects to test the interpretability of the interpreter.

1.2 Contributions

The main contribution of this thesis is the development of an interface for 3D reconstruction problem, which hides algorithmic details and allows users to describe conditions surrounding the problem. This description 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 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.

1.3 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 taxonomy of 3D reconstruction algorithms based on the conditions surrounding a problem. In Chapter 4, we provide a formal description of the 3D reconstruction problem. In Chapter 5, we evaluate the performance of a selection of algorithms under varied problem conditions, from which a mapping from problem space to algorithms can be derived. 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

Section 2.1 discusses the existing toolboxes for 3D reconstruction. Section 2.2 presents a comprehensive review of the field of image-based 3D reconstruction 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 [16], VLFeat [60], VXL [5] and multiple Matlab libraries [34, 40]. 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.

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 [46]. 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. 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 theis.

2.2.1 Stereo Correspondence

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 the MVS algorithms based on the taxonomy proposed in [53], which divides the field into four classes based on reconstruction method, and categorize the SL algorithms by projection patterns.

MVS: Volumemetric stereo

The first class of MVS algorithms computes the cost function in a 3D volume, then extracts 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 [52]. Another thread of work formulates the problem in the Markov Random Field (MRF) framework and extracts the optimal surface by Graph-Cut algorithms [48, 61, 62].

MVS: Surface Evolution

The second class of MVS algorithms works 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 [41] by iteratively removing inconsistent voxels from the scene [35]. Level-set techniques cast the problem as a variational one, and use a set of PDE's as cost functions, which are deformed from an initial set of surfaces towards the detected objects [19]. Other approaches use a deformable model and represent the scene as surface meshes that moves as a function of internal and external forces [18]. 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.

MVS: Region Growing

The third class of MVS algorithms starts with a sparse set of scene points, propagates 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. The essence of this algorithm is as fol-

lows: 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. A two-view quasi-dense approach first sorts the list of point correspondences into a list of seed points by correlation score. At each step of the propagation, a 'best' seed point is chosen. Then 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 [37, 38]. 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 [20]. A stereoscopic approach called PatchMatch Stereo, which is inspired by an approximate nearest neighbour matching algorithm called PatchMatch [9]. This method 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 extend PatchMatch Stereo to multi-view scenarios [21, 59, 66] and to a proposal of new propagation schemes to increase the computational efficiency [21].

MVS: Depthmap Merging

The fourth class of MVS algorithms computes 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 function are derived to account for occlusion or non-Lambertian effects which may add noise to the photo-consistency score [23, 62]. 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 depth set 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.

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

Structured Light

Structured light is considered one of the most accurate reconstruction techniques. 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 [49]. Temporal techniques generate the codeword by projecting a sequence of patterns. Spatial codification represents each codeword in a unique pattern. Direct codification techniques define a codeword for every pixel, which is equal to its grey level or colour.

Temporal encoding For temporally encoded SL, a sequence of patterns is successively projected onto the surface, the codeword for a given pixel is formed by the 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. We further classify these techniques as follows: 1) binary codeword; 2) n-ary codeword; 3) gray code combined with phase shifting; 4) hybrid techniques.

Spatial encoding This technique concentrates 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.

Direct encoding There are methods to directly represent the codeword in each pixel. 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 clour of the surface. Therefore, reference images must be taken. This kind of coding can be classified as: 1). codification based on grey levels; 2). codification based on colour.

2.2.2 Shading

Shading variations can reveal the surface normal orientation, which can be further integrated into a 2.5D height map. Shading variation depends on the shape (surface normal orientation), reflectance (material), and lighting (illumination). Thus an ill-posed problem arises because different shapes illuminated under different light conditions may produce the same image. This 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 [63]. 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 the intensity variation is first proposed by Horn [28]. 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

$$I(x,y) = R(p(x,y),q(x,y))$$
$$I(x,y) = \rho(\vec{n},\vec{l})\vec{n}^\top\vec{l} \quad \text{(Lambertian model)}$$

where $(p,q) = (z_x, z_y)$ are surface gradients. Unfortunately, measurements of the brightness at a single pixel only provide one constraint, whereas surface orientation requires two. Thus, additional constraints such as smoothness or integrability are required to estimate (p,q).

Photometric Stereo

Category	Camera	Light source	Reflectance
Original 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.

Original Photometric Stereo This method, first proposed by Woodham [64], utilized multiple light sources from different directions to overcome the ambiguity of Shape from Shading. Assuming there are P pixels per image, and Q illumination directions, the intensity of the ith pixel under jth illumination would be

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

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, and the normal can be estimated by

$$N = IL^+$$
$$\rho_i = \|N_{i,:}\|$$
$$n_i = \frac{N_{i,:}^\top}{\|N_{i,:}\|}$$

The key problem is how to generalize the assumptions of 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.

Generalization of Lighting It is possible to estimate the surface orientation without knowing light directions, a case also known as *uncalibrated Photometric Stereo*, see Table 2.2. Most uncalibrated techniques assume Lambertian techniques and are based on factorization technique proposed in [25]. Recall the Irradiance Equation:

$$I = N^\top L$$

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

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 class of techniques relax the assumption of Lambertian reflectance.

Outlier rejection The fact that the reflectance of non-Lambertian surfaces can be approximated by the sum of a diffuse and a specular lobe has been exploited extensively. The specular pixels are considered as outliers in [17] and [10]. The assumption that the color of the specular lobe differs from that of the diffuse lobe allows the separation of the specular and diffuse components [39, 50, 51].

Reference object A separate approach uses a reference object that has the same material as the target object. This is proposed in [55] and later revisited in [26]. It can deal with arbitrary BRDFs as long as the reference and target object has the same material. Multiple reference objects are needed for spatially-varying BRDFs as the BRDF at each point on the target object is a linear combination of the basis BRDFs defined by the set of reference objects.

Parametric reflectance model More sophisticated BRDF models can replace the reference objects. 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 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, etc [7, 67].

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

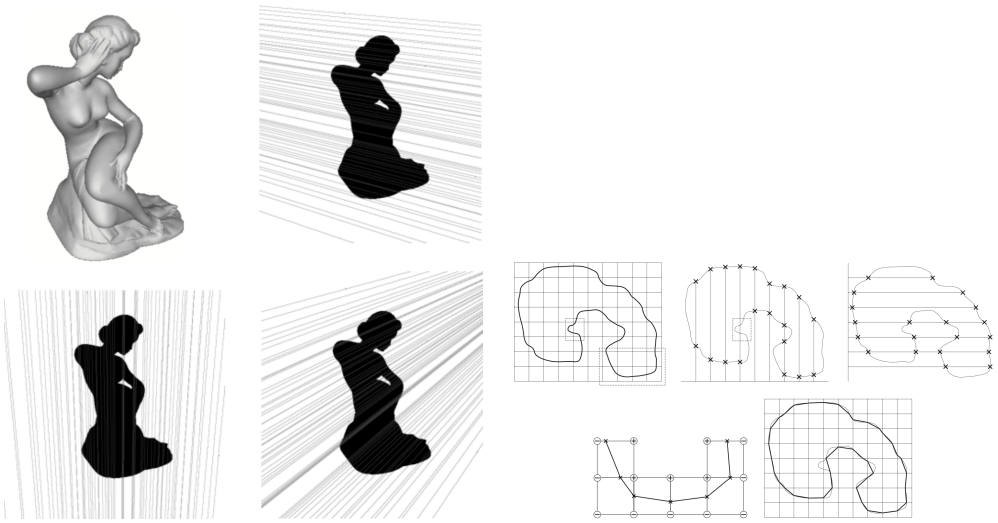
First the object space is split up into a 3D grid of voxels; each voxel is intersected with each silhouette volume; only voxels that lie inside all silhouette volumes remain part of the final shape.

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 [47] 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.

All of the cues above are most widely used ones, and achieved decent results. These following two cues haven't resulted in as much success. Therefore, we only discuss the general idea rather than the technical details.

2.2.4 Texture

The basic principle behind shape from texture is the *distortion* of the individual texel. In general, the image formation process introduces three distortion effects: the *distance effect*, which makes objects in view appear larger when they are closer to the image plane; the *position effect* which makes objects appear differently when the angle between the line of sight and the image plane different; and the *forshortening effect*, which distort the objects depending on the angle between the surface normal and the line of sight. Besides, different effects take place under different projection models: the orthographic projection captures only the foreshortending effect whereas the perspective projection captures all three. Therefore, shape from texture methods which use orthographic projection are valid only in a limited domain, where the other two effects can be ignored, and the perspective model captures all three effects, but the resulting algorithms are complicated and involves the solution of nonlinear equations.

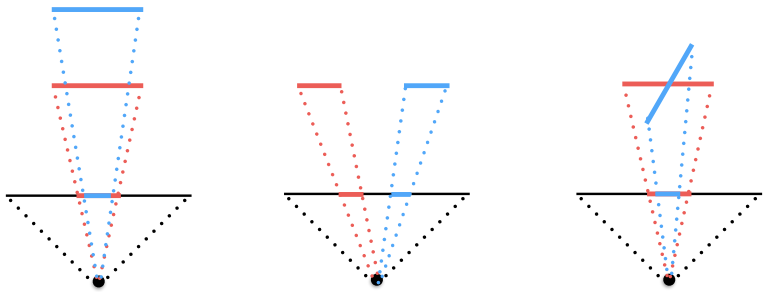


Figure 2.2: Three distortion effect: distance distortion, position distortion, and foreshortening distortion.

To calculate the surface curvature at any point is far from trivial. Therefore, the surface shape is reconstructed by calculating the surface orientation (surface normal). A map of surface normals specifies the surface's orientation only at the points where the normals are computed. But, assuming that the normals are dense enough and the surface is smooth, the map can be used to reconstruct the surface shape.

2.2.5 Defocus

Shape from focus A strong cue for object depth is the amount of blur, which increases as the object moves away from the camera's focusing distance. As shown in Figure 2.3, moving the object surface away from the focus plane increases the circle of confusion.

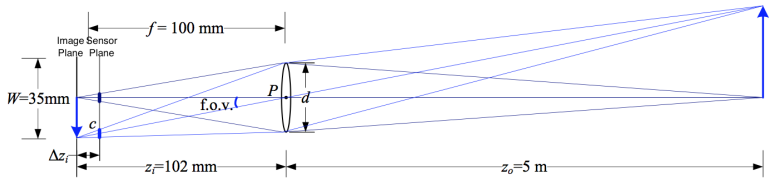


Figure 2.3: A thin lens of focal length f focuses the light from a plane a distance z_0 in front of the lens at a distance z_i behind the lens, where $\frac{1}{z_o}+\frac{1}{z_i}=\frac{1}{f}$. If the sensor plane moved forward Δz_i, the image are no longer in focus and the *circle of confusion* c depends on the distance of the sensor plane motion Δz_i relative to the lens aperture diameter d.

Figure 2.3 shows the basic geometric image formation. The relationship between the object distance z_o, focal distance of the lens f, and the image distance z_i, is given by the Gaussian lens law:

$$\frac{1}{z_o}+\frac{1}{z_i}=\frac{1}{f}$$

All light rays that are radiated from the object and intercepted by the lens to converge at a single point on the image plane, thus a *focused* image $I_f(x,y)$ is formed on the image plane. If, however, the sensor plane does not coincide with the image plane and is displaced from the image plane by a distance Δz_i, the energy received from the object is uniformly distributed over a circular patch on the sensor plane.

The relationship between the radius c of the circle of confusion and the sensor displacement Δz_i is as follows:

$$c = \frac{\Delta z_i r}{z_i}$$

The defocused images can be obtained in three ways: by displacing the sensor with respect to the image plane, by moving the lens, or by moving the object with respect to the object plane. The first two ways cab cause the following problems:

- The magnification of the system varies, thereby causing the image coordinates of the object points to change.
- The area on the sensor plane over which light energy is distributed varies, thereby causing a variation in image brightness.

To address this issue, the degree of focus is changed by moving the object with respect to a fixed configuration of the optical system and sensor. This approach ensures that the focused areas of the image are always subjected to the same magnification.

The idea is as follows: the stage is moved in increaments of Δd, and an image is captured at each stage position ($d = n\Delta d$). By studying the behaviour of the focus measure, an interpolation method is used to compute the accurate depth estimates from a small number of focus measures. An important feature of this method is the local nature, the depth estimate at an image point is computed only from focus measures recorded at that point.

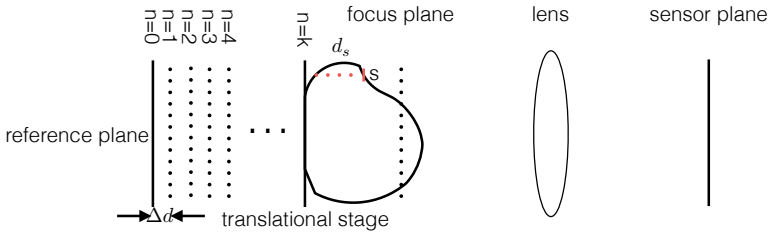


Figure 2.4: shape from focus

Chapter 3

A Taxonomy of 3D Reconstruction

Existing taxonomies of 3D reconstruction techniques generally classify algorithms based on algorithmic details: the survey on Multi-view Stereo algorithms uses taxonomies that differentiate the key properties of each algorithm [53]. Reviews on Structured Light techniques typically classify techniques based on the type of projection pattern used [22, 49]. Photometric Stereo algorithms are classified by the assumptions or generalizations made, such as, unknown/known reflectance, unknown/known light conditions (uncalibrated/calibrated), and so on [54]. However these taxonomies have the following limitations: 1) they are algorithm centric, classifying algorithms based on *how* an algorithm solves the problem, giving little to none insight to the conditions that allow a specific algorithm to work well, thus requires vision knowledge to fully take advantage of these algorithms; 2). they provide means to categorize within-category algorithms, but are unsuitable to compare the performance of between-category algorithms. It is well known that such algorithms target limited categories of objects, and are highly likely to fail when targeting a diverse set of object categories. It is crucial to understand the conditions a specific algorithm performs well when designing an application for reconstruction. Under the previous framework of taxonomy, this knowledge is largely empirical, with each algorithm mapped roughly to a sub-volume in the problem space that is poorly defined. To overcome these limitations, we take a more *object-centered*

approach instead of an *algorithm-centered approach*.

The taxonomy proposed in this chapter defines 3D reconstruction techniques from an object-centered viewpoint, i.e., categorizes algorithms based on the problem conditions that it can reliably work under. This taxonomy 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 working conditions of each algorithm.

3.1 An Object-centered Taxonomy

The proposed object-centered taxonomy categorizes algorithms based on the type of objects/problem conditions that they can reliably work under. We first give an overview of object classes, which serves as the bases to the taxonomy, then proceeds to investigate the working conditions of each class of algorithms based on relevant literature reports.

3.1.1 Object class

In Figure 3.1, we show a taxonomy of object classes based on material and shape properties. However, by no means are these presented object classes complete. There are many other properties not included that are commonely seen in the real world. For instance, effects such as occlusion, discontinuity, and emission, among others, are not considered. Further, we assume **local interaction model**, i.e., global light transport such as transmission, refraction, cast shadow, inter-reflection are not considered. The rationale behind our choice is that most techniques that have been developed over the past few decades mainly tackle object with an opaque, diffuse or mixed surface. For specular, refractive, and translucent or transparent objects, only very specialized algorithms are applicable for reconstruction [29]. We propose the following labels to differentiate object classes. The order of properties for the class label is as follows: translucency, texture, lightness, reflectance model, surface roughness, and concavity. To approach this problem in a feasible manner, six classes of objects are being investigated in depth. They are selected based on the availability of reliable techniques and the diversity of corresponding real-world objects. See the six classes of objects in Table 3.1.

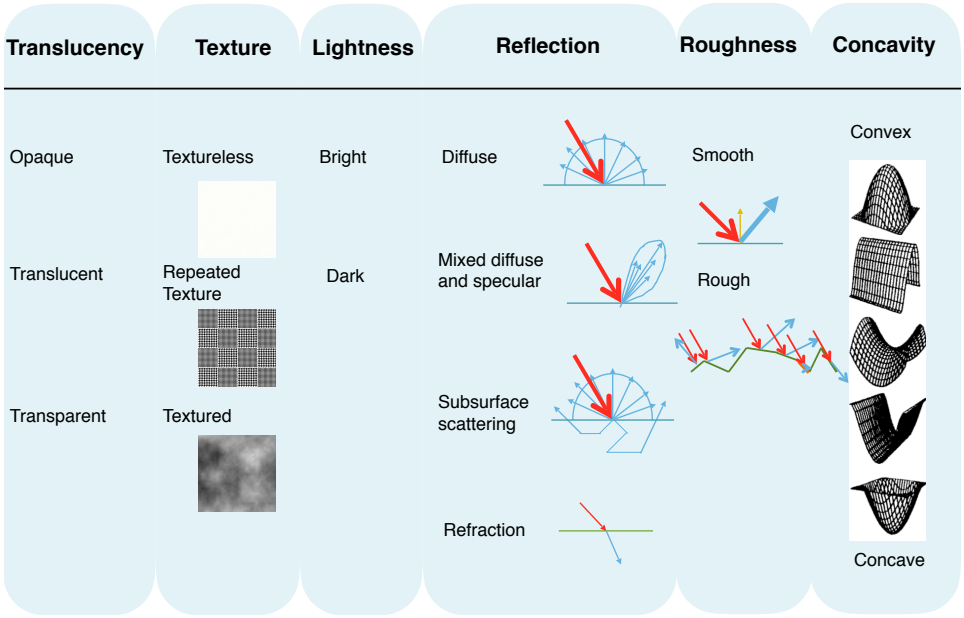


Figure 3.1: A list of properties for object classes.

Class labels

- **Translucency**: **O**: opaque, **Tl**: translucent, **Tp**: transparent.
- **Texture**: **T**: textured, **Tr**: repeated textured, **Tl**: textureless.
- **Lightness**: **B**: bright, **D**: dark.
- **Reflection**: **D**: diffuse model, **S**: specular model, **M**: mixture of diffuse and specular, **Ss**: subsurface scattering, **Rf**: refraction
- **Roughness**: **S**: smooth, **R**: rough
- **Concavity**: **Cx**: convex, **Cv**: concave

A traditional taxonomy

Traditionally, algorithms are categorized into different classes based on the visual cues used for reconstruction, as discussed in Chapter 2. Table 3.2 gives an example of what a typical taxonomy would look like.

Class #	Description	Label
1	Textureless, bright, diffuse reflectance	O-Tl-B-D-R-Cx
2	Textureless, bright, mixed reflectance	O-Tl-B-M-S-Cx
3	Textured, bright, diffuse reflectance	O-T-B-D-R-Cx
4	Textured, bright, mixed reflectance	O-T-B-M-S-Cx
5	Textured, dark, diffuse reflectance	O-T-D-D-R-Cx
6	Textured, dark, mixed reflectance	O-T-D-M-S-Cx

Table 3.1: Labels of six classes of objects.

Algo. class	**Technique**
SfS	Horn [28]
MVS	Furukawa [20], Goesele [23], Vogiatzis [62], Hernández [18], Faugeras [19]
Lamberian PS	Woodham [64], Hayakawa [25], Belhumeur [11], Alldrin [8]
Non Lambertian PS	Coleman [17], Barsky [10], Schluns [51], Sato [50], Mallick [39], Alldrain [6], Goldman [24], Silver [55], Hertzmann [26], Zickler [67]
SL	Inokuchi [31]
VH	Szeliski [56], Matusik [42], Tarini [58]

Table 3.2: A traditional taxonomy that classifies algorithms based on algorithmic details.

3.2 Problem conditions of algorithms

This section investigates the conditions under which each category of algorithms is capable of working based on the reported literature. In this thesis, only visual texture is considered, and it is considered as resulting from non-uniform surface albedo. Thus, uniform surface albedo represents uniform texuture while non-uniform albedo represents textured surfaces.

3.2.1 Multi-view Stereo

The problem conditions of Multi-view Stereo algorithms are summarized in Table 3.3. For a typical MVS algorithm to perform well, the object should have a

textured and diffuse/mixed surface.

Texture: textured

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. However, some MVS algorithms tested on a textureless object "Dino" in the Middlebury MVS benchmark [53] give successful reconstruction results. This demonstrates 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 need to have very high quality.

Reflectance: diffuse/mixed model

Most MVS algorithms require that the object surface with similar or same appearances from different perspectives, and hence, most of the algorithms assume Lambertian reflectance. While pure Lambertian surfaces are rare in reality, it is empirically verified that MVS algorithms perform reasonably well on non-Lambertian surfaces. As long as the cameras can capture the diffuse reflectance component, and 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].

Technique	Texture	Lightness	Reflectance	Roughness	Concavity
MVS	Textured	-	Diffuse or mixed	-	-

Table 3.3: Problem conditions of Multi-view Stereo algorithms.

3.2.2 Shape from Shading

Shape from Shading, first proposed by Horn [28], specifically targets isotropic, known Lambertian surfaces. By assuming orthographic projection, and known light source intensity and direction, surface orientation can be estimated from shading variations. The problem condition is shown in Table 3.4.

Texture: textureless

Typical SfS algorithms asssume surfaces with uniform and known albedo, i.e., textureless surfaces.

Lightness: bright

Active methods such as SfS utilize reflected light to estimate surface depth or orientation information. In this case, the intensity variation is used to estimate surface normal, thus the surface should have sufficiently high albedo, otherwise, the intensity variation would be hard to detect.

Reflectance: Lambertian model

Though other reflectance models are feasible, typical SfS algorithms assume Lambertian reflectance model. The reason is that surface lightness is directly related to surface orietation and reflectance model once the light source and viewing direction are fixed. This is generally an ill-posed problem even with Lambertian model since there is only one intensity value per pixel to solve for surface orientation, which has two DoF. However, even in the simplest case, the survey by Zhang [65] demonstrate that SfS algorithms generally perform poorly, and none performs well in all cases.

Concavity: convex

Typical SfS algorithms can not deal with effects caused by global light transport, such as cast shadow, inter-reflection, and so on. The surface lightness would be corrupted by light transported from other surface facets. Thus, objects exhibit any form of concavity will pose great challenge to SfS algorithms.

Technique	Texture	Lightness	Reflectance	Roughness	Concavity
SfS	Textureless	Bright	Lambertian	-	Convex

Table 3.4: Working condition of Shape from Shading algorithms.

3.2.3 Photometric Stereo

Photometric Stereo can be considered as an extension of SfS algorithms, which adds additional light sources to remove the ambiguity faced by SfS algorithms. The working conditions of Photometric Stereo algorithms are summarized in Table 3.7.

Texture: N/A

Though SfS algorithms require uniform and known albedo, typical PS algorithms can be used easily on surfaces with spatially varying albedo. For instance, the albedo-scaled normal can be estimated, then the albedo is retrieved as the magnitude of the scaled normal [64].

Lightness: bright

Active methods such as PS utilize reflected light to estimate surface depth or orientation information. In this case, PS algorithms work more reliably on surfaces with sufficiently strong albedo. This is because the algorithm exploits the intensity variation as a visual cue, which is more challenging to detect on surfaces with low intensity values.

Reflectance: Lambertian model

The Lambertian PS algorithms can be divided into two groups: calibrated, and uncalibrated method. The original PS proposed by Woodham [64] can be considered as calibrated Lambertian PS. Later, more uncalibrated Lambertian PS algorithms have been proposed to avoid this tedious process.

Silver [55] proposed a look-up scheme that relies on a reflectance object with the same reflectance as the target, which in this case is a uniform Lambertian surface. This approach is later adapted to surfaces with non-Lambertian reflectance with varying albedo or material in [26].

Another successful uncalibrated approach used six or more pixels with the same albedo, and was able to solve for normals up to a rotation ambiguity[25]. It can be further proved that a 3-parameter subset of these transformations, known as the Generalized Bas-Relief (GBR) ambiguity, preserve surface integrability [11]. Thus, given three or more imges of a Lambertian object acquired under light sources of unknown direction and strength, the surface can be reconstructed up to GBR transformation by enforcing surface integrability, see Figure 3.2 for the effect of GBR-ambiguity.

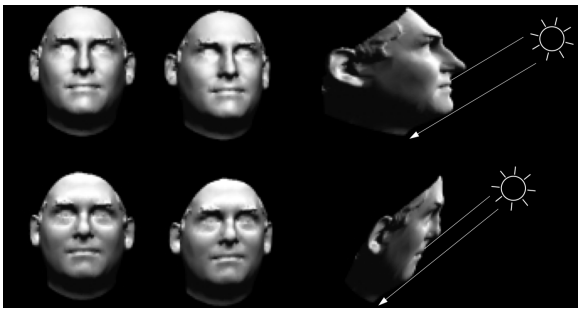


Figure 3.2: The effect of GBR ambiguity. Two sets of shape and light source configurations can produce exactly the same images.

Reflectance: Non-Lambertian model

Much of the emphasis in PS has been to relax the assumption of Lambertian reflectance. There are four classes of Non-Lambertian PS algorithms: the first class treats specular component as outliers [10, 17, 30, 44]; the second class uses reference object that has the same material as the target object [26]; the third class assumes that the reflectance can be modeled as a linear combination of a set of basis reflectance models [6, 24]; and the fourth class exploits the physical properties common to large classes of BRDFs, such as energy-conservation, non-negativity, Helmholtz reciprocity, isotropy, and so on [7, 57, 67].

Convexity: convex

Active methods that assumes a **local interaction model**, such as most PS algorithms, can work more reliably on surfaces without casting shadow and inter-

reflection. Thus surfaces with concavities pose a great challenge for this type of techniques since the indensity can be affected by other surface patches.

Technique	Texture	Lightness	Reflectance	Roughness	Concavity
Lambertian PS, uniform albedo	Textureless	Bright	Lambertian	-	Convex
Lambertian PS, non-uniform albedo	Textured	Bright	Lambertian	-	Convex
Non-Lambertian PS	-	Bright	Mixed	-	Convex

Table 3.5: Working conditions of typical Photometric Stereo algorithms.

3.2.4 Structured Light

For stereo correspondence based methods, actively projected patterns have to be used for the lack of surface texture. Since the surface is diffuse, there is no specular reflection to cause severe noise. Refer to Table 3.6 for the working condition of SL.

Lightness: bright

Active methods such as SL 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.

Reflectance: Diffuse model

Traditional Structured Light techniques cannot deal with highly specular surfaces since the specular area will exhibit strong reflection regardless of the magnitude of the light source, which will cause errors in the decoding process.

Concexity: convex

Active methods that assumes a **local interaction model** such as most SL algorithms can work more reliably on surfaces without casting shadow and interreflection. Thus surfaces with concavities pose a great challenge for this type of techniques since the indensity can be affected by other surface patches.

Technique	Texture	Lightness	Reflectance	Roughness	Concavity
Structured Light	-	Bright	Diffuse	-	Convex

Table 3.6: Working condition of typical Structured Light algorithms.

3.2.5 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 visual properties. However, it fails to carve the concavities on the object surface, thus is unsuitable to concave objects.

Technique	Texture	Lightness	Reflectance	Roughness	Concavity
VH	-	-	-	-	Convex

Table 3.7: Working condition of typical Visual Hull algorithms.

3.3 Summary

Our taxonomy categorizes algorithms based on the problem conditions that they can reliably work under. The problem conditions consist of various visual and geometric properties, as shown in Figure 3.1. These properties can be conceptualized as dimensions of the 3D reconstruction problem space. This taxonomy provides an abstraction which allows us to think of algorithms as volumes within an $n-$dimensional problem space. Existing algorithms can be introduced into this framework based on the performance within the problem space. The aforementioned analysis provide an initial mapping of the space that is summarized below in Figure 3.3.

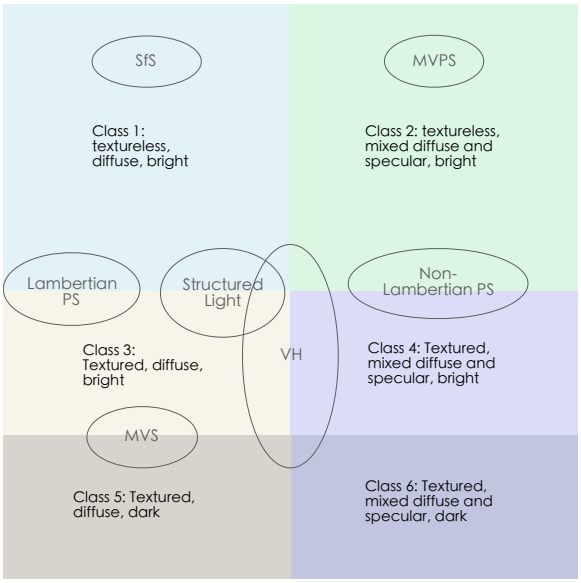


Figure 3.3: Six classes of objects of interest, and the algorithms that could work reliably for these classes.

Chapter 4

A Description of 3D Reconstruction

In Chapter 3, we introduce a taxonomy of 3D reconstruction which maps algorithms to the problem space according to visual/geometric characteristics of the object. However, without a formal 'language', i.e., a model and representations, this mapping would be largely empirical. Expressing the conditions within which an algorithm works well without a formal definition of the problem space prevents formulating a well defined problem.

In this chapter, we attempt to extend this taxonomy by providing 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 Section 4.1. Next, section 4.2 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 4.3 outlines concrete representations of the proposed model. Section 4.4 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 Definition

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.

4.1.1 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.

4.1.2 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 4.1.

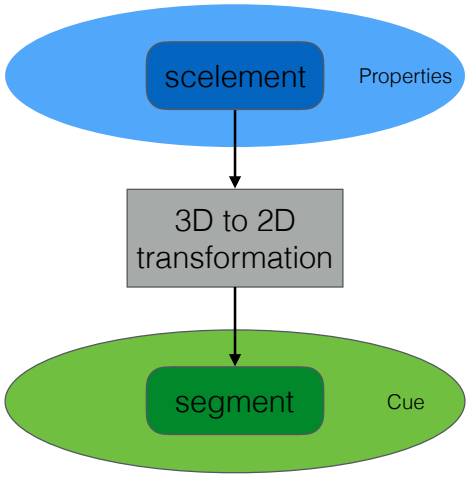


Figure 4.1: Relation between a scell and a segment

4.1.3 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.

4.1.4 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 $\{I_n\}_{n=0}^{N-1}$

4.1.5 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*.

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

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.

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

4.2 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 [15]. 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 3 as the main components of our model. The model consisting of these key properties is shown in Table 4.1.

Property	Texture	Lightness	Reflectance	Roughness	Concavity

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

4.3 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.

4.3.1 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/granularity, 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 ex-

hibits less noticeable elements and displays rather random patterns. Most of the real world textures are mixtures of these two categories. In this thesis, we adopt this simplification and consider *texture randomness*, which is the amount of distortion in the texture. Thus, a uniform texture has low *texture randomness* whereas a highly textured surface has *texture randomness*.

4.3.2 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 most common phenomena experienced and is the easiest to analyze. Generally, we assume that all effects are local, thus global effects such as interreflection 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*.

Definition of Radiometric Terms

Below is a list of radiometry terms, see Figure 4.2 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 $(watt \cdot m^{-2} \cdot steradian^{-1})$.

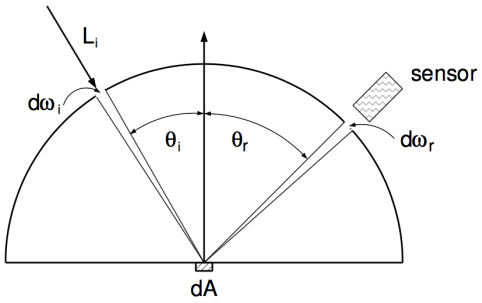


Figure 4.2: Illustration of light-matter interaction.

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 surface 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 4.3: The light-matter interaction.

Diffuse albedo or surface albedo is the proportion of incident light that is reflected by the surface.

Light lens interaction

The assumption made in vision 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. It can be further shown that the image irradiance received by the sensor is proportional to the scene radiance, thus the relation between *scene radiance* and *image irradiance* is linear.

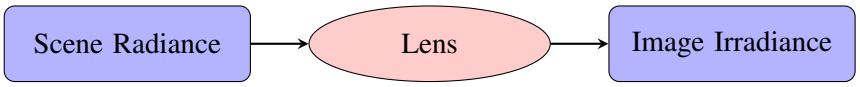


Figure 4.4: The light-lens interaction.

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.

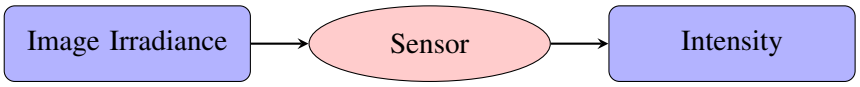


Figure 4.5: The light-sensor interaction.

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 a simplified case - Lambertian reflectance, the BRDF is degenerated to *diffuse albedo*, which is the representation we adopt for the property of intensity.

4.3.3 Specularity

Specular surfaces reflect light in nearly a single direction when the microscopic surface irregularities are small compared to light wavelength, and no subsurface scattering is present [43]. 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. In Figure 4.6, the image no longer reflects the original colour of the surface, which is red. Instead it shows a distorted image of the environment. A purely specular surface is a mirror, which is rare in nature. Most natural materials exhibit a mixture of specular and diffuse reflections. 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 use a numeric *specular* value to denote the proportion of a material's specularity, with 0 being completely diffuse, and 1 being completely specular (mirror like).

Figure 4.6: 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.

4.3.4 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. We need prior knowledge of the microscopic surface irregularities, or a model of the surface itself, to determine the reflection of incident light.

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

4.3.5 Concavity

Concavity can cause self-shadow or interreflection effects, which can severely impede the accuracy of active methods. Since concavity is invisible in the silhouette images, methods that utilize silhouette information may also fail to reconstruct concavities. Concavity can be measured by *surface curvature*, which is defined as the inverse of the radius of a tangentially intersecting circle/sphere. We include concavity in our model for the sake of completeness. However, we decide to omit this property since concavity will introduce global light transport including cast shadow and interreflection, and so on, which violates our assumption of a **local interaction model**.

4.4 Expression

Now that we have a proposed model and representation of 3D reconstruction problem, we can express the six proposed object classes using this description. The expression of the reconstruction problem is shown in table 4.2.

Class #	Texture randomness	Albedo	Specularity	Roughness	Concavity
Class 1	0.2	0.8	0.2	0.8	0.2
Class 2	0.2	0.8	0.5	0.2	0.2
Class 3	0.8	0.8	0.2	0.8	0.2
Class 4	0.8	0.2	0.2	0.8	0.2
Class 5	0.8	0.8	0.5	0.2	0.2
Class 6	0.8	0.2	0.5	0.2	0.2

Table 4.2: Expression of 3D reconstruction problem for the object classss proposed in Chapter 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 application domain or problem conditions. This section builds upon the 3D description proposed in Chapter 4, and attempts to find the problem conditions surrounding each algorithm.

To achieve this goal, we need a dataset to evaluate the performance of each algorithm under varied conditions, which is not available since most 3D benchmarks, to the best of our knowledge, focus on one specific class of algorithms, and do not have objects with changing properties. For example, the Middlebury dataset targets MVS algorithms [53], and the 'DiLiGenT' dataset targets Photometric Stereo algorithms [54]. This makes such benchmarks only suitable for the evaluation of within-category algorithms. There are no datasets that evaluate 3D reconstruction across differ categories, nor are there any datasets that cover a range of properties of materials and geometry and their combinations. Reasons for the lack of such a dataset are: 1) it is already tedious to create a real-world dataset for one specific category of algorithms, so it would be even more challenging to create a dataset for a larger range of algorithms with ground truth; 2) it's practically impossible to change one property, e.g., surface texture, material, etc., while fixing others when creating a real-world dataset.

In response to these challenges, we propose a synthetic dataset created by

phiyically-based rendering software (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.

5.1 Synthetic setup

We use Blender's physically-based rendering engine, Cycles, to generate the synthetic dataset. For each technique, the configuration of the camera remains fixed. The image resolution is 1280×720, with a focal length of $35mm$ or $1400pix$.

For the Multi-View Stereo setup, there are five rings of cameras, of which the elevation angles are 15°, 30°, 45°, 60°, 90°. The between-angles of two neighbouring cameras are 30°, 30°, 45°, 45°, and 360°. Thus, there are in total $12+12+8+8+1=41$ cameras.

For the photometric stereo setup, since increasing the number of images is only important up to a point, the experimental results showed that most algorithms reaches their optimum when 15 images are used [12]. To strike a balance between algorithm performance and rendering time, we use 25 light sources, which are distributed on four different rings with elevation angles of 90°, 85°, 60°, and 45°. The azimuth angle between two neighbouring light sources is 45°.

For the structured light setup, the baseline angle between the camera and the projector is 10°, and only one camera is used, thus only a portion of the object is visible. The resolution of the projector is 1024×768, thus 10 Gray code patterns are needed. To counter the effect of inter-reflection, each pattern and its inverse are projected, which descreases sensitivity to scattered light.

5.2 Structure of Datasets

Due to the number of properties and levels for each property, it would be unrealistic to render all their combinations. For instance, if there are N properties and each is discretized into L levels, the number of different combinations is L^N, and for each combination, there are in total $41+25+42=108$ images to render. Therefore,

we take another approach: 1) we investigate the *effective problem domain* which consists of only the *effective* and *dependent* properties; 2) we generate synthetic images for the *effective* and *dependent* properties and all of their combinations. The structure of the dataset is as follows:

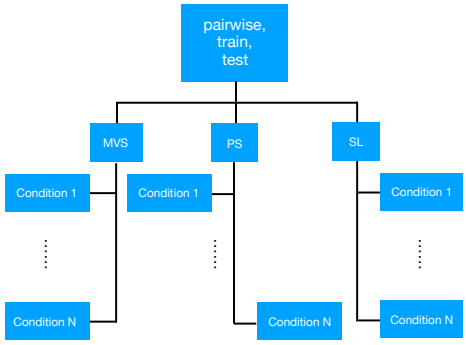


Figure 5.1: Structure of the synthetic dataset.

5.3 Evaluation metrics

We use the metric proposed in [53] 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. Thus 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 es-

timated 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.

5.4 Selected methods

We have selected one representative algorithm from three major classes of algorithms presented in Chapter 3: the PMVS proposed in [20], the example-based photometric stereo proposed in [26], 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 checking step is performed to reject erroneous triangulations.

Technique	Texture	Albedeo	Specular	Roughness
PMVS: patch-based, seed points propagation MVS.				
PMVS	High	-	Low	-
EPS: example-based Photometric Stereo				
EPS	-	High	Low	High
GSL: Gray code Structured Light technique				
GSL	-	High	Low	High

Table 5.1: Summary of the selected algorithms for the framework, and the corresponding working conditions in theory.

5.5 Baseline

A baseline algorithm that works sufficiently well under most conditions should be chosen so that it is possible to determine the performance of a selected algorithm within the framework. We choose the Visual Hull technique as one of our baseline algorithms 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

Technique	Texture	Albedeo	Specular	Roughness
VH: volumetric Visual Hull				
VH	-	-	-	-
LLS-PS: linear least squares Photometric Stereo.				
LLS-PS	-	High	Low	High

Table 5.2: Summary of the baseline algorithms for the framework, and the corresponding working conditions in theory.

true scene is always enclosed by the reconstruction result, so the outcome is always predictable.

A simple linear least squares based Photometric Stereo (LLS-PS) is selected to evaluate Photometric Stereo algorithms. However, there is currently no such algorithm that works reasonably well under various conditions. Thus, we run this baseline algorithm under the optimal condition to ensure that a best possible result is achieved. To assess the performance of the selected PS algorithm against the baseline, we compare the following characteristics of the angular error:

Measures of Central Tendency

Mean and median are both valid measures of central tendency, but as the skewness increases, the mean is dragged in the direction of the skew. As a result, the median is generally considered to be the best representative of the central location of the data. The more skewed the distribution, the greater the difference between the median and mean, and the greater the emphasis should be placed on using the median as opposed to the mean. See Figure 5.2.

Variation

The variation of the angular error is measured by both *interquatile range* ($Q_3 - Q_1$) and *standard deviation* (std).

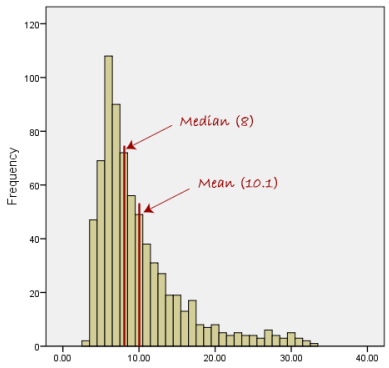


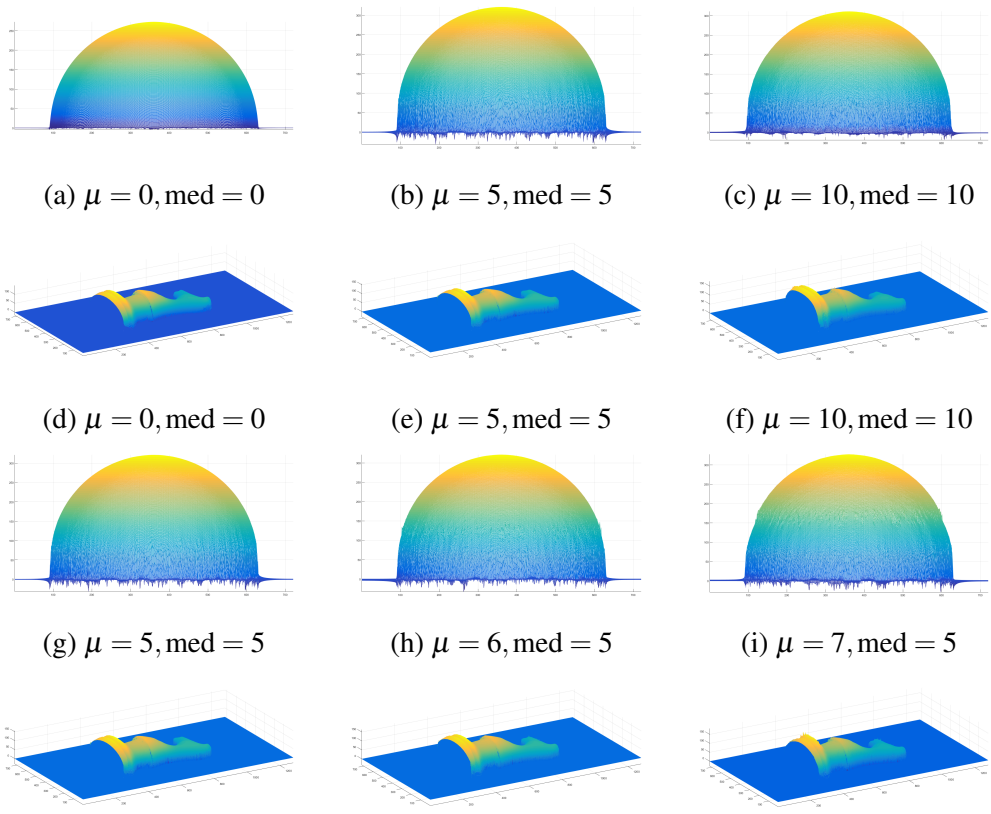
Figure 5.2: A right-skewed distribution, which is a typical graph of the angular error.

Skewness (right/positive-skewness)

The normal estimation is less sensitive to angular error when the mean and median are close. The shape can be recovered relatively well with a mean/median angular error of 10° (see Figure 5.3 (a)-(f)). The normal estimation becomes more susceptible as the difference between the mean and median increases (see Figure 5.3 (g)-(l)). This can be explained as follows: when the normals are reliably estimated, the angular error should follow a Gaussian distribution. Therefore, the mean and median are close to each other. However, if normals are poorly recovered, the mean would become larger since large angular errors exist, while the median would change far less since only a small amount of pixels are affected. Thus, the difference between mean and median would be larger when surface normals are poorly recovered.

5.6 Effective Problem Domain (EPD)

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 **??**, is to reduce the dimensions of problem space by establishing the *effective problem domain* (EPD), which consists of only effective



(a) $\mu = 0, \text{med} = 0$ (b) $\mu = 5, \text{med} = 5$ (c) $\mu = 10, \text{med} = 10$

(d) $\mu = 0, \text{med} = 0$ (e) $\mu = 5, \text{med} = 5$ (f) $\mu = 10, \text{med} = 10$

(g) $\mu = 5, \text{med} = 5$ (h) $\mu = 6, \text{med} = 5$ (i) $\mu = 7, \text{med} = 5$

(j) $\mu = 5, \text{med} = 5$ (k) $\mu = 6, \text{med} = 5$ (l) $\mu = 8, \text{med} = 5$

Figure 5.3: Shape estimation results with varied mean and median values. (a) - (f): the algorithm is less sensitive to large mean or median values when mean and median are close. (g) - (l): as the difference between the mean and median increases, the estimated shape becomes more susceptible to error, as shown by the spikes in the reconstruction.

properties. Then, Section **??** evaluates performance of selected algorithms within EPD.

A naive way of finding the *effective properties* would be evaluating algorithmic performance by changing one property at a time. However, this approach ignores the dependency between properties, i.e., the effect of property *A* might be insignificant when property *B* is absent. Thus, we investigate the pairwise relation between any two properties. The process is demonstrated using PMVS as an example. More details on the analysis and results for the other chosen techniques can be found in the supplementary material.

5.6.1 EPD of PMVS

We evaluate the performance of PMVS in terms of accuracy and completeness under varied combinations of properties. The settings of these properties are listed in Table 5.3.

Group	Texture	Albedo	Specular	Roughness
(a)	[0.2, 0.8]	[0.2, 0.8]	0.0	0.0
(b)	[0.2, 0.8]	0.8	[0.2, 0.8]	0.0
(c)	[0.2, 0.8]	0.8	0.0	[0.2, 0.8]
(d)	0.8	[0.2, 0.8]	[0.2, 0.8]	0.0
(e)	0.8	[0.2, 0.8]	0.0	[0.2, 0.8]
(f)	0.8	0.8	[0.2, 0.8]	[0.2, 0.8]

Table 5.3: Problem conditions for establishing the *effective problem domain* of PMVS.

Effective and Dependent Properties

In this section, we investigate how each property affects the reconstruction in terms of accuracy and completeness.

(a) Texture and Albedo texture has a positive effect on the reconstruction in terms of accuracy and completeness, while the effect of albedo is negligible.

(b) Texture and Specularity specularity has a negative effect on both the accuracy and completeness of the reconstruction. However, the level of impact varies as the texture varies. More specifically, the effect of specularity is more substantial

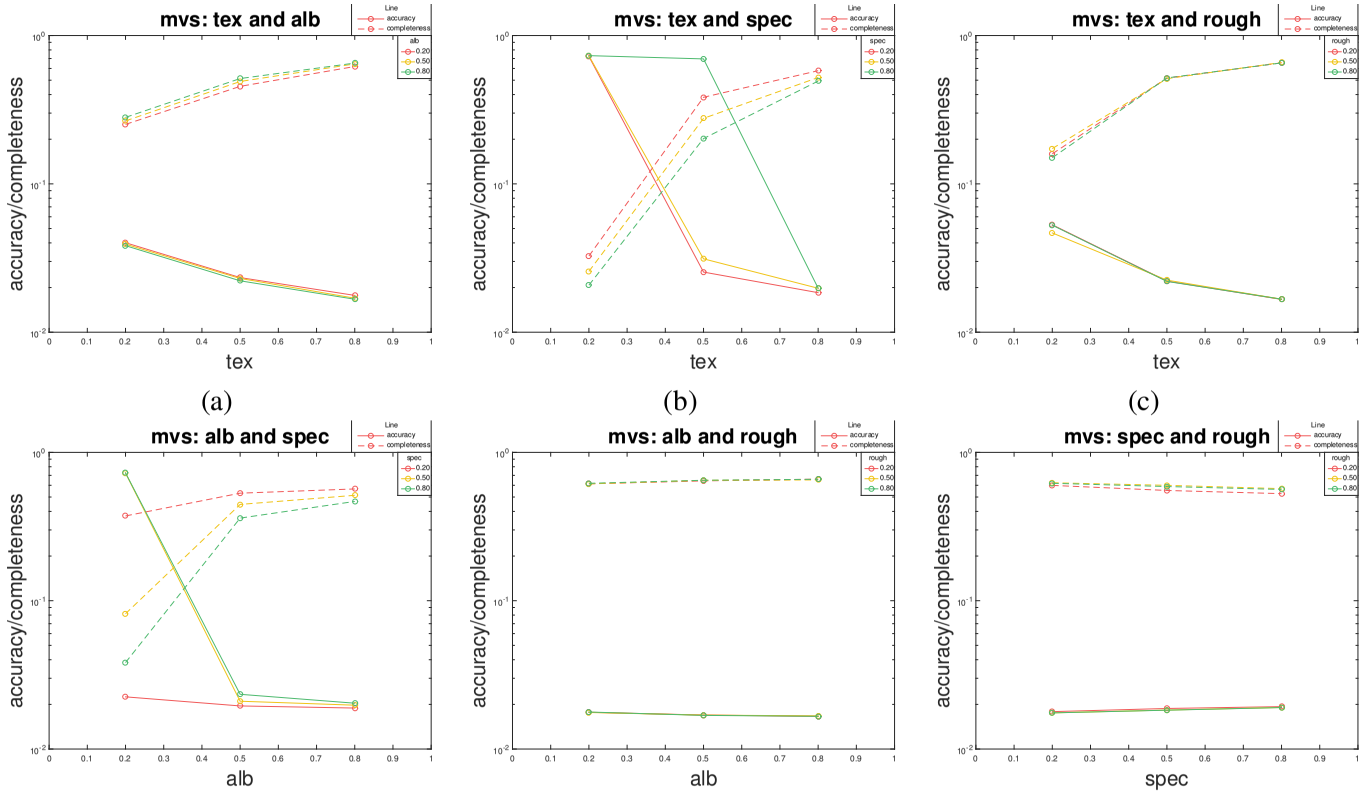
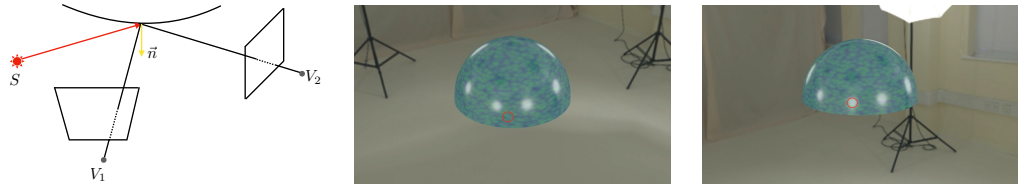


Figure 5.4: Performance of PMVS under six pairwise conditions. For instance, (a) shows the performance under changing *texture* and *albedo* values, while the others are fixed. The property values are set based on settings in Table 5.3.

on a less textured surface than on one of higher textured. This 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 5.5 (a) and (c). Cameras positioned otherwise would observe the true surface, such as camera V_1 shown in Figure 5.5 (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

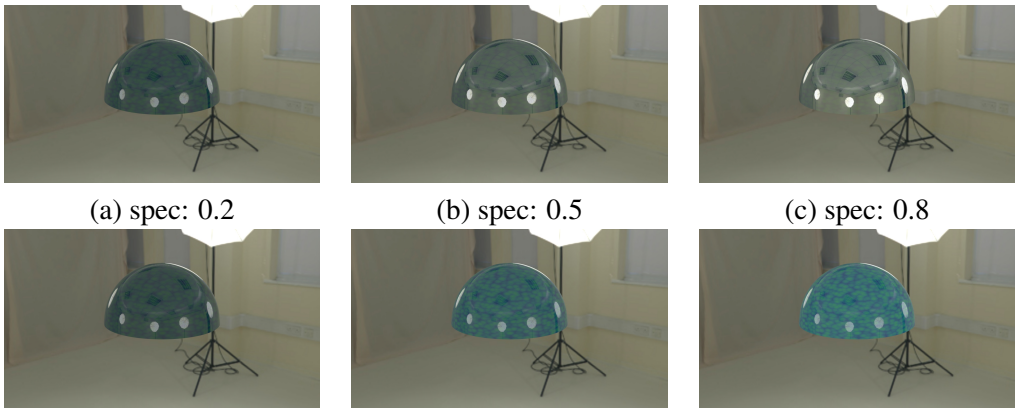
Figure 5.5: (a) shows the reflection of light off a specular surface. V_1 received the diffuse component while V_2 receives the specular component. (b), (c) shows the images observed from these two views. The specular area (red circle) observed in V_2 is visible in V_1.

(c) Texture and Roughness roughness doesn't have a significant effect on the results.

(d) Albedo and Specular albedo has a positive effect whereas specular has a negative effect on the reconstruction. Furthermore, the effect of specular is far more substantial on a lower albedo surface than that on a higher albedo surface. This can be explained as follows: according to the energy conservation law, as the specular component increases, the diffuse component decreases, resulting in a less discernible diffuse area. See Figure 5.6 (a)-(c). Increasing the diffuse albedo can counteract the effect of specularity and make the texture visible again. See Figure 5.6 (d)-(f).

(e) Albedo and Roughness the albedo and roughness have a negligible effect on the results.

(f) Specular and Roughness surface roughness can effectively diminish the specular component and make the surface appear more diffuse. Thus, in theory, roughness should have a positive impact on the reconstruction. However, this is



(a) spec: 0.2 (b) spec: 0.5 (c) spec: 0.8

(d) alb: 0.2 (e) alb: 0.5 (f) alb: 0.8

Figure 5.6: (a)-(c). The albedo is set as 0.2, (d)-(f). The specularity is set as 0.2. According to energy conservation, as the specular component increases, the diffuse component decreases.

valid for surfaces with medium texture since this is the only case where specularity has a significant impact, as shown in Figure 5.4(b). Further, since medium specular, high roughness surfaces visually resemble low specular surfaces (and achieve similar reconstruction results as well), it makes sense to incorporate the effect of roughness to specularity, and ignore roughness for simplicity.

Summary

The effective properties of PMVS are: texture, albedo, and specular, of which texture is the most important. Specularity can deteriorate the reconstruction for lower textured and lower albedo surfaces. The effective problem domain is shown in Table 5.4.

Metric	Texture	Albedo	Specular	Roughness
Accuracy	✓	✓	✓	✗
Completeness	✓	✓	✓	✗

Table 5.4: The *effective problem domain* of PMVS in terms of accuracy and completeness.

5.6.2 EPD of EPS

We evaluate the performance of example-based PS in terms of angular error under varied combinations of properties. The statistical measures that we used include median, mean, standard deviation, and the first and third quartiles of the angular error. We investigate two properties at a time. The settings of the properties are listed in Table 5.5.

Group	Texture	Albedo	Specular	Roughness
(a)	[0.2, 0.8]	[0.2, 0.8]	0.0	0.0
(b)	[0.2, 0.8]	0.8	[0.2, 0.8]	0.2
(c)	[0.2, 0.8]	0.8	0.0	[0.2, 0.8]
(d)	0.0	[0.2, 0.8]	[0.2, 0.8]	0.2
(e)	0.0	[0.2, 0.8]	0.0	[0.2, 0.8]
(f)	0.0	0.8	[0.2, 0.8]	[0.2, 0.8]

Table 5.5: Problem conditions for establishing the *effective problem domain* of EPS.

Effective and Dependent Properties

We investigate how each property affects reconstruction in terms of the statistics of the angular difference.

(a) Texture and Albedo texture has no effect on angular difference while albedo has a positive effect on normal estimation.

(b) Texture and Specular texture has no effect on angular difference while specular has a negative effect on normal estimation since the difference of mean and median gets larger as the specular increases. This can be explained as follows: as the specular increases, only the specular regions exhibit erroneous normal estimation while the rest of the surface is reliably estimated. See Figure 5.8. This is why the median value exhibits far less change, while the mean value increases significantly as shown in Figure 5.7 (b).

(c) Texture and Roughness texture has no effect on angular difference while roughness has a positive effect on normal estimation.

(d) Albedo and Specular the albedo has a positive impact on normal estimation (see Figure 5.9 (a)-(c)), whereas the specularity has a negative impact on

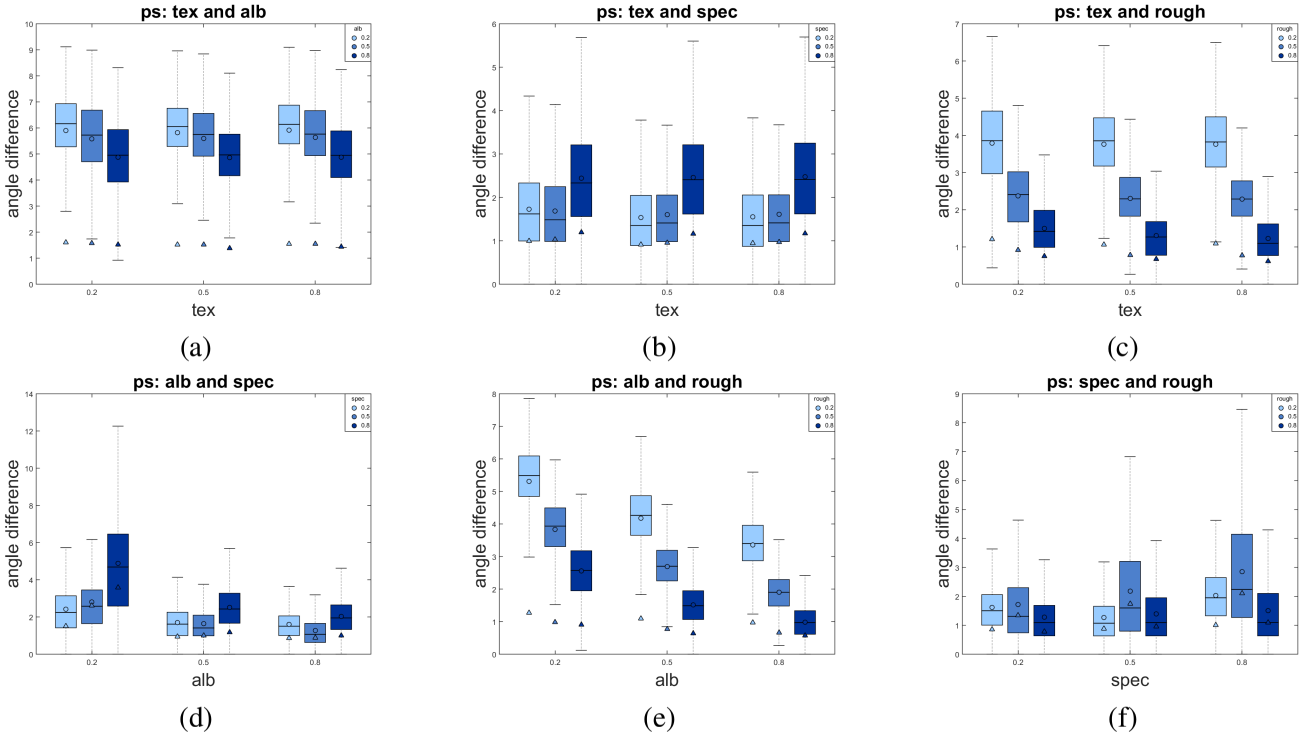


Figure 5.7: Performance of Example-based PS under six pairwise conditions. For instance, (a) shows the performance under changing *texture* and *albedo* values. The property values are assigned based on the settings in Table 5.5 (a).

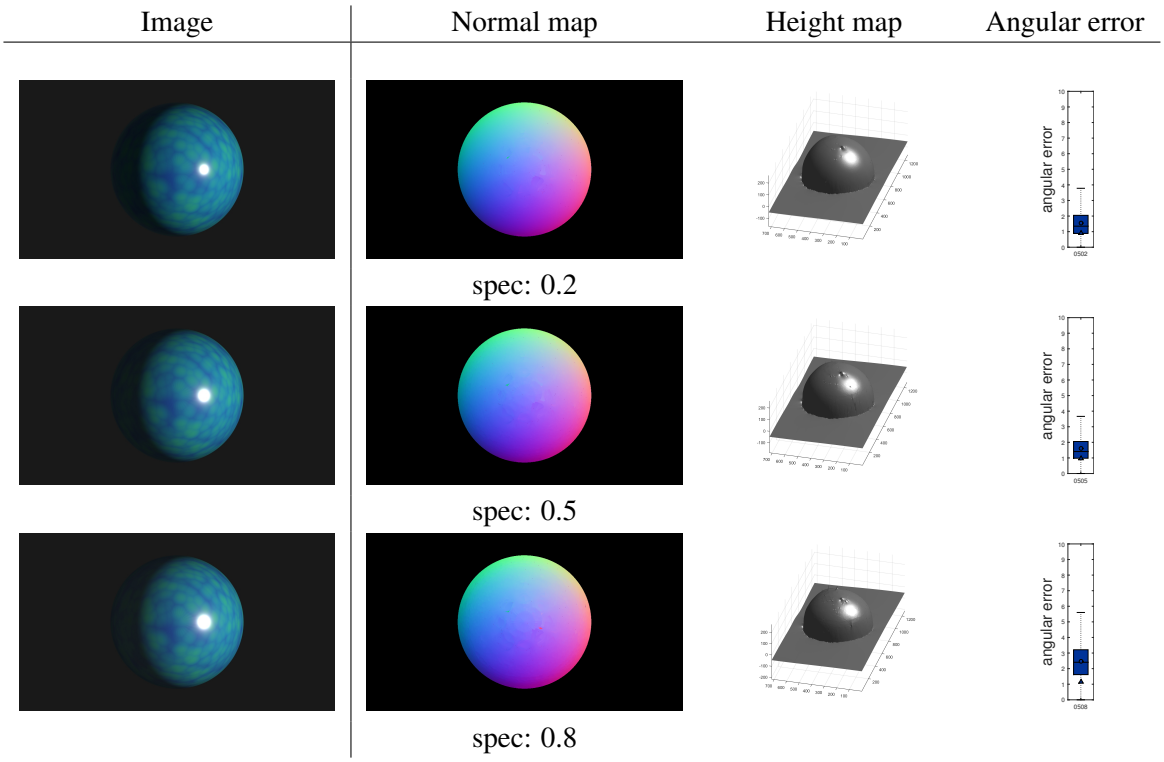


Figure 5.8: (a)-(c). The texture is set as 0.5. The estimated normal map and recovered surface becomes consistently worse as the specular level rises.

normal estimation (see Figure 5.9 (d)-(f)).

(e) Albedo and Roughness both albedo and roughness have a positive effect on normal estimation.

(f) Specular and Roughness the specular has a negative impact on normal estimation, while roughness has a more complicated effect. We observed that the reconstruction becomes worse when roughness is 0.5, which is counter-intuitive at first sight. However, we argue that it's because the roughness is not strong enough to counteract the specular component, the result is a smoothed and blurred specular lobe with a larger area, leading to a poorer reconstruction result. This effect is also demonstrated in the training stage. See Figure 5.10 for visual examples.

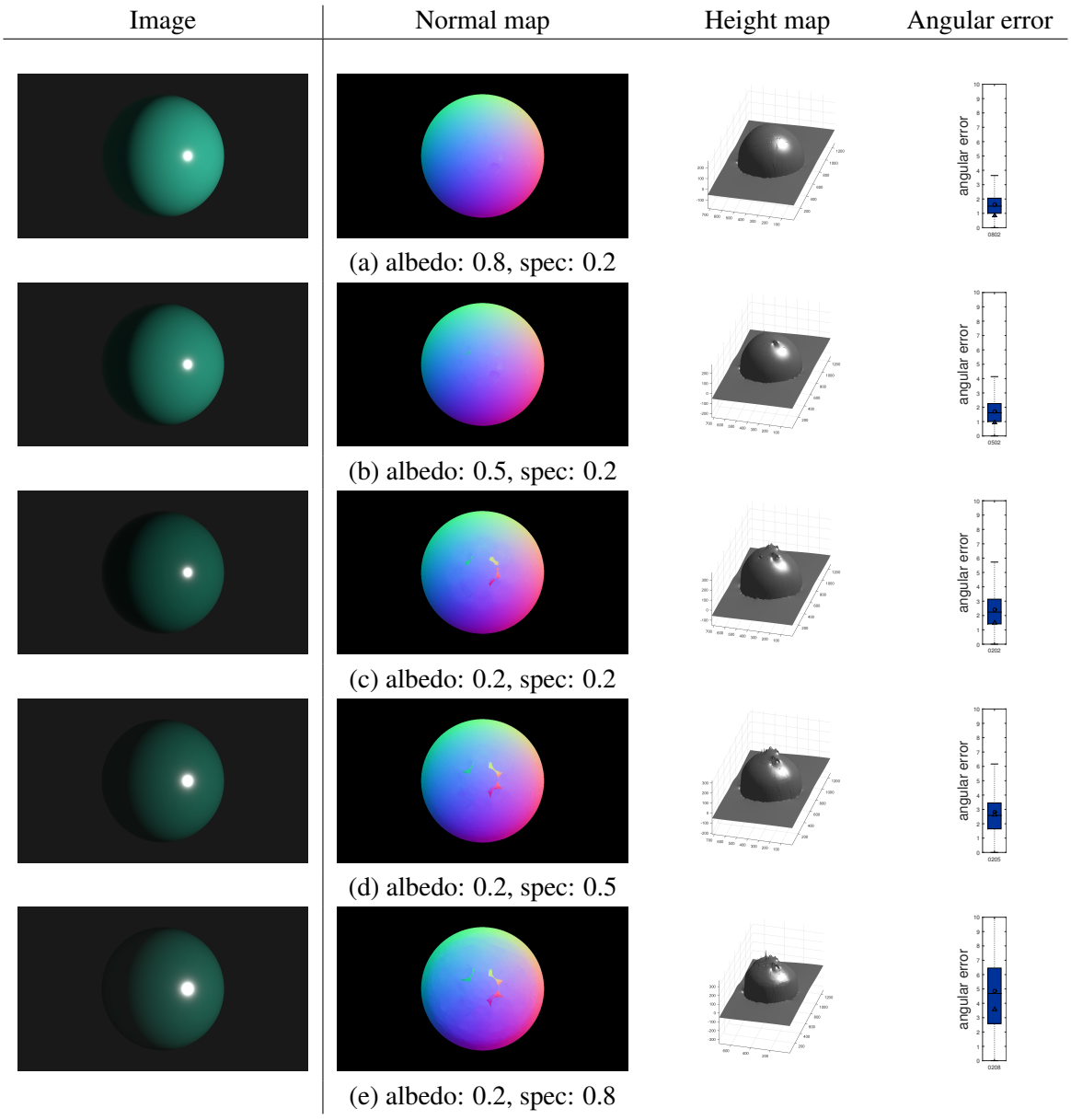


Figure 5.9: According to energy conservation, as the specular component increases, the diffuse component decreases. (a)-(c): the estimated normal map and recovered height map become consistently worse as the albedo decreases; (c)-(e): the estimated normal map and recovered height map become consistently worse as the specular increases.

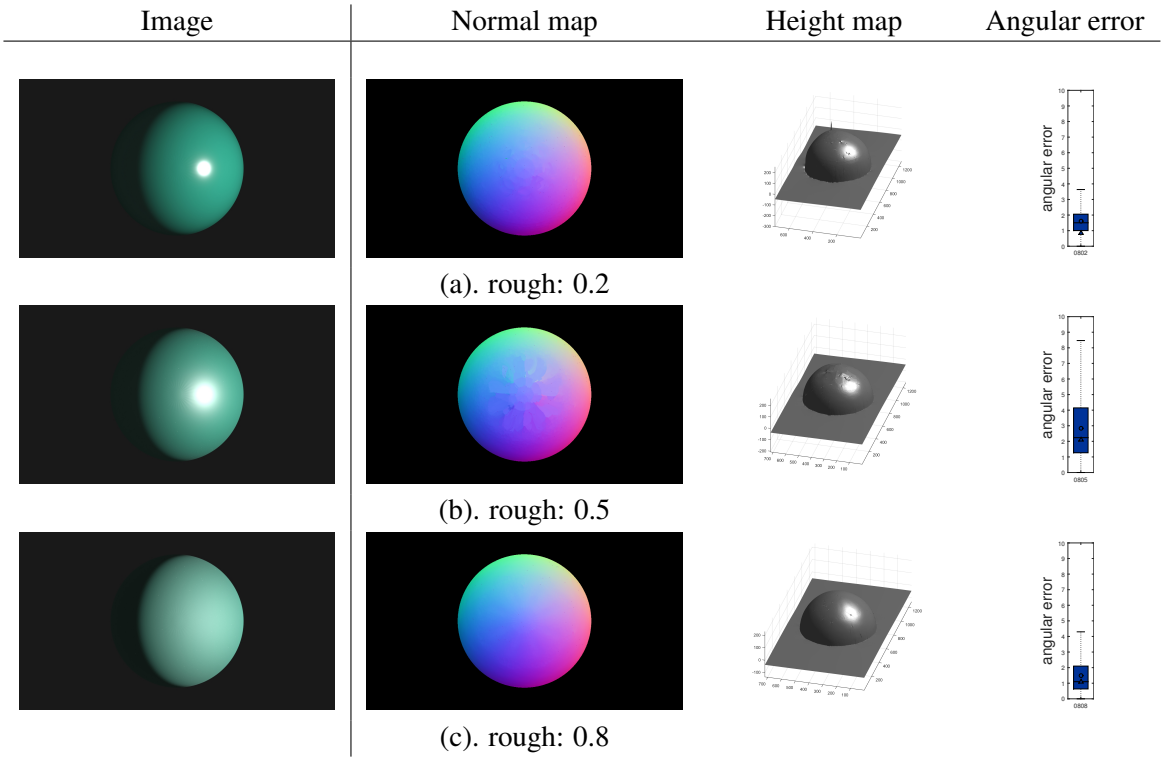


Figure 5.10: The 'peculiar' effect of roughness on PS. Albedo is set as 0.8, and specular is set as 0.8. (b) demonstrates that a medium level roughness would lead to worse normal estimation since it blurs the specular lobe.

Summary

The properties that have an effect on EPS are: albedo, specular, and roughness, as shown in Table 5.6. Therefore, we will only consider these three properties for all forthcoming discussion of EPS.

Metric	Texture	Albedo	Specular	Roughness
Angle difference	✗	✓	✓	✓

Table 5.6: The *effective problem domain* of EPS in terms of the *angular difference*.

5.6.3 EPD of GSL

We evaluate the performance of Gray-code SL in terms of accuracy and completeness under a varied combination of properties. The settings of the properties are listed in Table 5.7.

Property	Texture	Albedo	Specular	Roughness
(a)	[0.2, 0.8]	[0.2, 0.8]	0.0	0.0
(b)	[0.2, 0.8]	0.8	[0.2, 0.8]	0.0
(c)	[0.2, 0.8]	0.8	0.0	[0.2, 0.8]
(d)	0.0	[0.2, 0.8]	[0.2, 0.8]	0.0
(e)	0.0	[0.2, 0.8]	0.0	[0.2, 0.8]
(f)	0.0	0.8	[0.2, 0.8]	[0.2, 0.8]

Table 5.7: Problem conditions for establishing the *effective problem domain* of GSL.

Effective and Dependent Properties

We investigate how each property affects the reconstruction in terms of accuracy and completeness. A depth check step is performed to remove erroneous depth, causing the accuracy to remain nearly constant across all cases.

(a) Texture and Albedo texture has no significant effect, whereas albedo has a positive effect on completeness. Both properties have no significant effect on accuracy.

(b) Texture and Specular texture has no significant effect whereas specular has a negative effect, on completeness. Both properties have no significant effect on accuracy.

(c) Texture and Roughness texture has no significant effect whereas roughness has a slightly positive effect on completeness. Both properties have no significant effect on accuracy.

(d) Albedo and Specularity albedo has a positive effect (see Figure 5.12 (a)-(c)) whereas specularity has a negative effect on completeness (see Figure 5.12 (d)-(f)). The effect of specular becomes less substantial as albedo increases (see Figure 5.11 (d)). We conclude that the effect of specularity is most significant when the albedo is low. Neither property has a significant effect on accuracy.

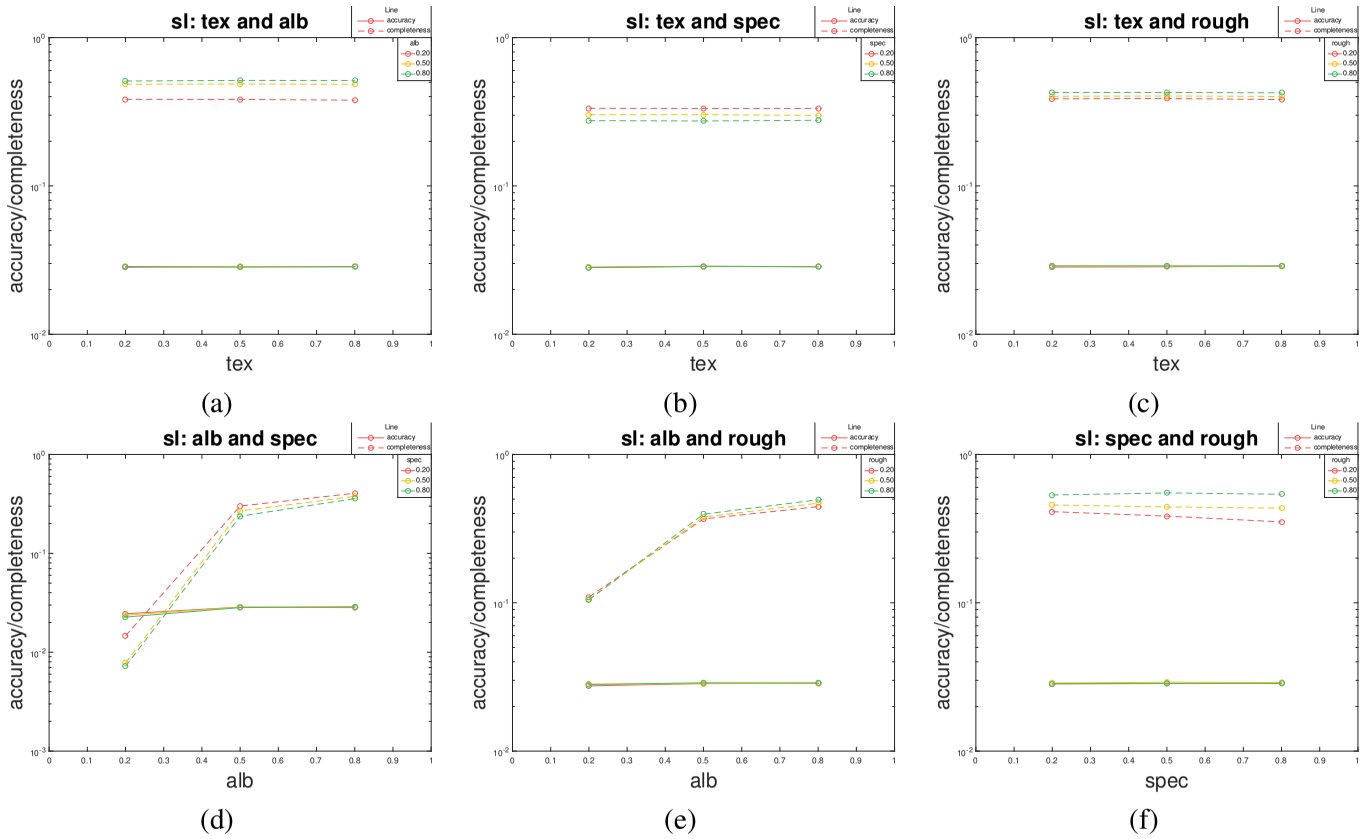
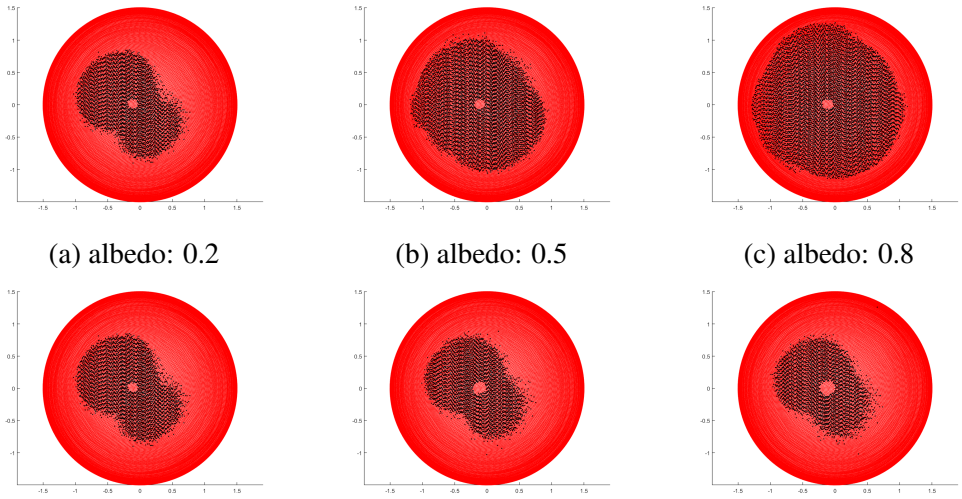


Figure 5.11: Performance of Gray-encoded SL under six pairwise conditions. For instance, (a) shows the performance under changing *texture* and *albedo* values. The property values are assigned based on settings in Table 5.7 (a).



(a) albedo: 0.2 (b) albedo: 0.5 (c) albedo: 0.8

(d) specular: 0.2 (e) specular: 0.5 (f) specular: 0.8

Figure 5.12: (a)-(c): the specular is set as 0.2, albedo has a positive effect on completeness; (d)-(e): the albedo is set as 0.2, specular has a negative effect on completeness.

(e) Albedo and Roughness albedo has a positive effect, whereas roughness has a slightly positive effect on completeness. Both properties have no significant effect on accuracy.

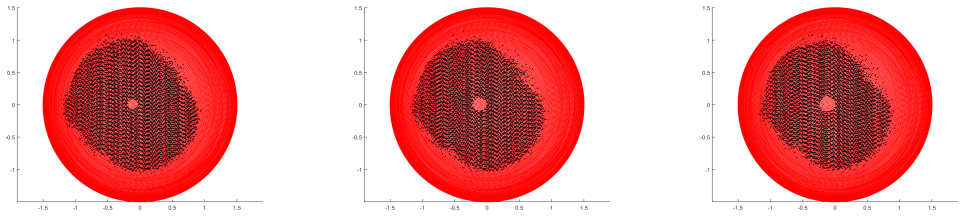
(f) Specular and Roughness specular has a negative effect (see Figure 5.13 (a)-(c)) whereas roughness has a positive effect on completeness (see Figure 5.13 (d)-(f)). Neither property has a significant effect on accuracy.

Summary

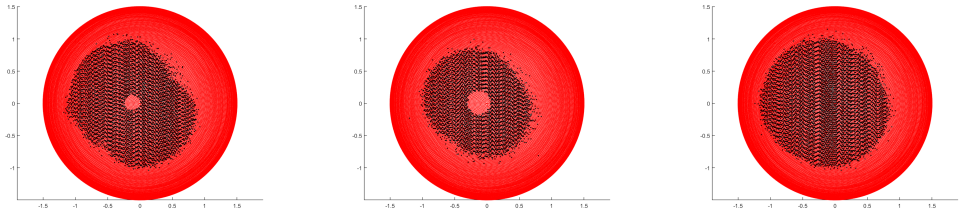
The properties that have an effect on the GSL are: texture, albedo, specular, as shown in Table 5.8. Therefore, we will only consider these three properties for all forthcoming discussion of GSL.

5.7 Mapping Construction

Once we have discovered the EPD, the algorithm is evaluated solely within the EPD. We present the results as follows: the $27(3^3)$ results are divided into three



(a) specular: 0.2 (b) specular: 0.5 (c) specular: 0.8



(d) roughness: 0.2 (e) roughness: 0.5 (f) roughness: 0.8

Figure 5.13: (a)-(c): the roughness is set as 0.2, and specular has a negative effect on completeness; (d)-(e): the specular is set as 0.8, roughness has a positive effect on completeness.

Metric	Texture	Albedo	Specular	Roughness
Accuracy	✗	✗	✗	✗
Completeness	✗	✓	✓	✓

Table 5.8: The *effective problem domain* of GSL in terms of accuracy and completeness.

plots. Each plot illustrates the results of one fixed property P_1, and two changing properties P_2 and P_3. To have a better understanding of the pairwise relation between any two properties, each effective property would be chosen as P_1 once. Therefore, we end up with three groups of graphs, each consisting of three plots, as shown in Figure 5.14, 5.15, and 5.16.

5.7.1 Mapping of PMVS

The performances of PMVS under different combinations of property values are shown in Figure 5.14, along with the performance of the baseline method. The conditions under which PMVS works well are listed in Table 5.9. We make the

following observations from the training results:

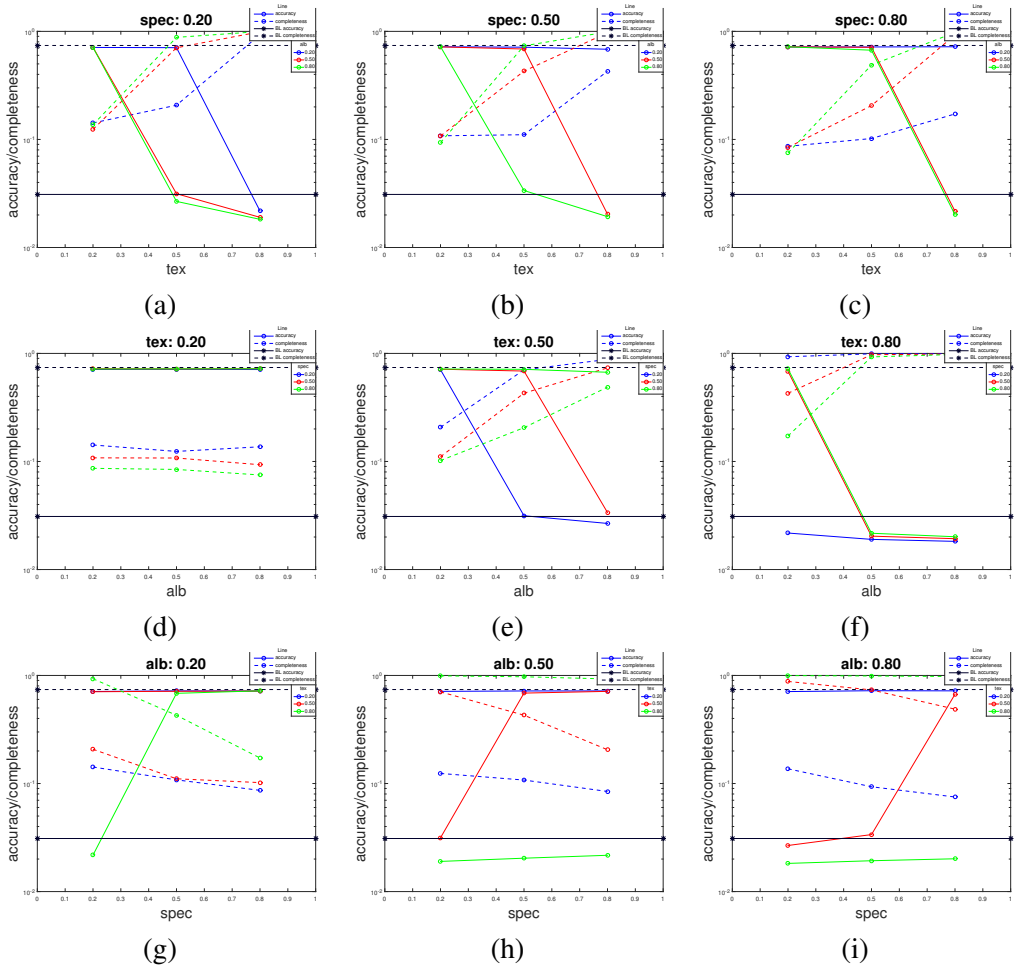


Figure 5.14: Performance of PMVS under varied conditions of changing property values. The baseline method serves as the guidelines to determine the performance of PMVS.

We make the following observations from the results: accuracy and completeness improves consistently as the *texture* level increases. Accuracy and completeness results deteriorate consistently as *specularity* increases, and this negative impact is most significant when texture level is medium or albedo value is low. The effect of *albedo* on a surface with low texture is negligible. However, albedo has

a noticeably more significant positive impact on completeness as the texture of a specular surface increases.

We can derive from the training results the problem conditions under which PMVS can perform reliably. These conditions are listed in Table 5.9.

Metric	Texture	Albedo	Specular	Roughness
Accuracy	0.5	0.5	0.2	-
	0.5	0.8	0.2	-
	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	-
Completeness	0.5	0.5	0.2	-
	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 5.9: The condition matrix of PMVS in terms of the two metrics *accuracy* and *completeness*.

5.7.2 Mapping of EPS

The performances of example-based PS under different combinations of properties are shown in Figure 5.15, along with the result of te baseline method. The conditions under which example-based PS works well are listed in Table 5.10. We make the following observations from the training results:

(a)-(c) Albedo: albedo has a consistently positive effect on the reconstruction.

(d)-(f) Specular: specular has a consistently negative impact on the normal estima-

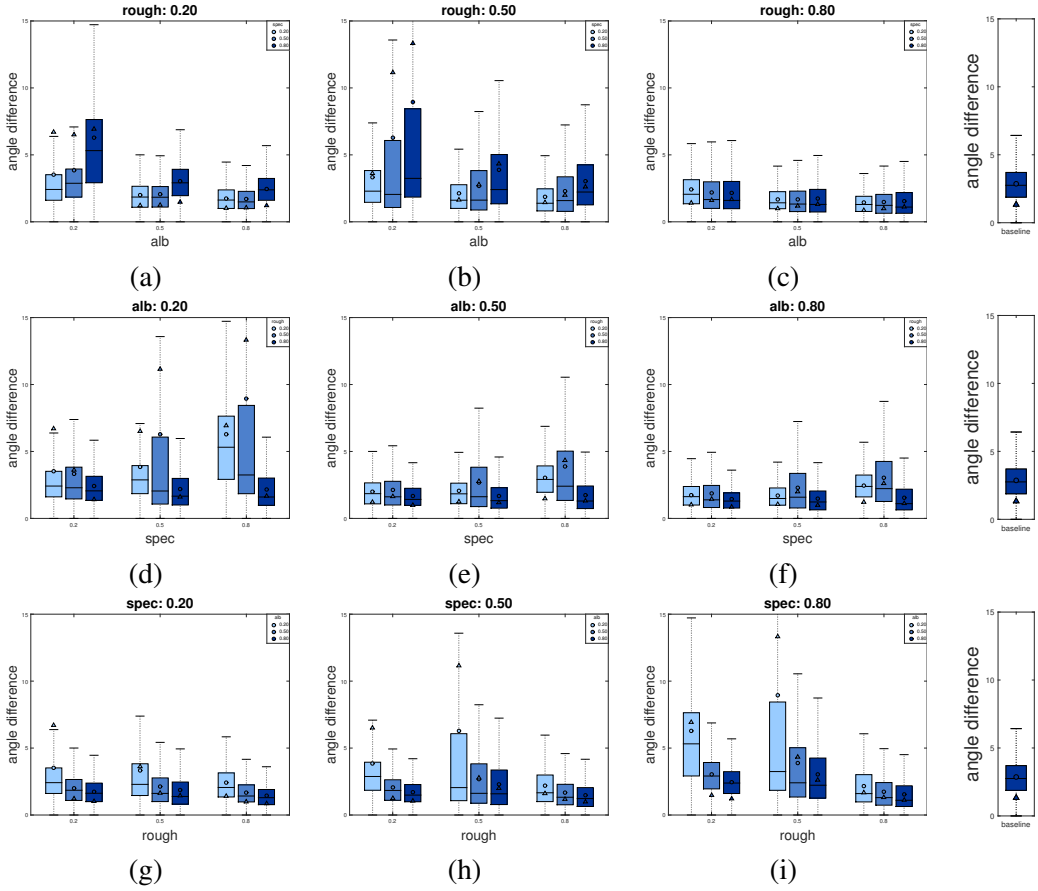


Figure 5.15: Performance of EPS under varied conditions of changing property values. Varied statistical measures of angular error are compared to the baseline method to determine the performance of EPS.

tion, which is manifested by the increasing variation represented by interquartile range and standard deviation.

(g)-(i) Roughness: roughness has a more complicated effect on reconstruction as illustrated in Figure 5.10. For example, medium roughness blurs the specular area, and leads to poor normal estimation and shape recovery.

We could derive from the training results the problem conditions under which EPS could perform reliably. These conditions are in Table 5.10.

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 5.10: The condition matrix of example-based PS in terms of the metric *angular error*.

5.7.3 Mapping of GSL

The performances of Gray-code SL under different combinations of property values are shown in Figure 5.16, along with the results of the baseline method. Only a portion of the scene is visible since there is only one camera, and the percentage of visible surface varies from object to object. This value can be approximated by the completeness value obtained under the optimal reconstruction condition. In this case, we claim that a completeness of 60% of that of the baseline is acceptable. The conditions under which GSL works well are listed in Table 5.11. We make the following observations:

(a)-(i) accuracy remains almost fixed, and is thus irrelevant to all properties.

(a)-(c) Albedo has a consistently positive effect on the completeness of the reconstruction.

(d)-(f) Specular has a negative effect on completeness, as shown in Figure 5.12 (d)-(f). However, we did notice in some cases that the completeness of the reconstruction improves as specular level increases. There are two contributing factors to completeness: 1) specularity decreases the completeness since the pattern can no longer be decoded in the glossy area, thus causing incomplete reconstruction; 2) large roughness can spread the specular lobe into a larger area, leading to a

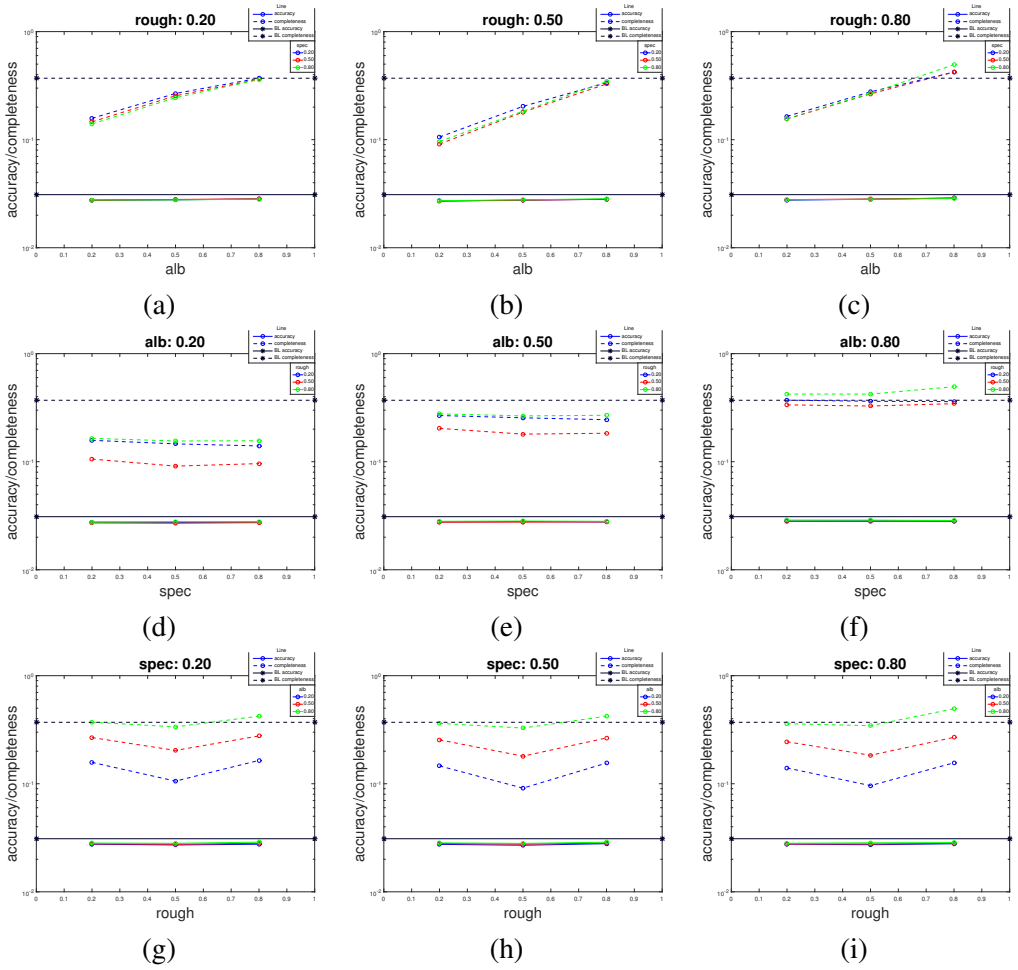


Figure 5.16: Performance of GSL under varied conditions of changing property values. The baseline method serves as the guideline to determine the performance of GSL.

brighter surface, thus increasing the completeness of the reconstruction. These two contradicting factors together determine the completeness of the reconstruction. If the first factor is more substantial, the completeness will decrease whereas if the second factor is more significant, the completeness will increase.

(g)-(i) Roughness has a similar effect to what we found when considering EPS, i.e., medium roughness would blur the specular lobe to a larger area, thus causing larger holes in the reconstruction. See Figure 5.10 (d)-(f). Large roughness can effectively counteract the impact of specular, thus improving the completeness of the reconstruction.

We can derive from the training results the problem conditions under which GSL could perform reliably by considering both the quantitative and qualitative results. These conditions are listed in Table 5.11.

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 5.11: The condition matrix of Gray-code SL in terms of the two metrics *accuracy* and *completeness*.

5.8 Summary

The development of 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

So far we have proposed a well-defined problem space for the 3D reconstruction problem, as well as a precise mapping from the problem space to the algorithm space. This section, we validate that the derived mapping can be reliably applied to an object with a different shape, and propose a proof-of-concept interpreter that translates the description to an appropriate algorithm that can give a reliable reconstruction result. In other words, the interpretability from the problem centric description to a reliable reconstruction result must be shown. Both synthetic and real-world datasets are provided to evaluate the performance of the interpreter.

However, such an evaluation faces several challenges. First, the mapping does not pose very strict constraints on object's geometry, and an exhaustive evaluation would require a vast number of objects to reach to a solid conclusion, which is not a practical approach. Second, we need a selection of algorithms that cover a wide range of problem space so that we can determine more convincingly if an unsuccessful result is due to lack of coverage or the limitness of the framework.

To address the first challenge, we assume **local interaction model**, which is not an uncommon assumption in vision community. Thus, geometric properties, such as concavity, that would violate this assumption will not be considered. Thus, objects with shallow concavities are used for evaluation.

For the second challenge, we need a selection of algorithms that cover a wide

range of problem space. Aside from a Multi-view Stereo algorithm, we implemented a Photometric Stereo and Structured Light techniques as discussed in Chapter 5. We can see that the selected algorithms cover a wide range of problem space, thus is sufficient to validate the framework's ability to translate the descriptive model into a reconstruction. Further, it is relatively straightforward to incorporate new algorithms, as long as they are evaluated with the same problem conditions presented in Chapter 5. This allows researchers to contribute novel algorithms to the framework once they become available.

This chapter is organized as follows: Section 6.1 provides a roadmap of our evaluation that is centered around two key evaluation questions: robustness of the mapping with respect to surface concavity, and robustness of the proof of concept interpeter. Section 6.3 investigates cases of varying surface concavity, and conditions under which the mapping can be reliably applied. Section 6.4 proposes a proof-of-concept interpreter, and Section 6.5 demonstrate the robustness of the interpreter using synthetic and real-world datasets, where a satisfactory reconstruction result can be returned given the correct description of the object.

6.1 Evaluation Methodology

This evaluation intends to 1) validate that the mapping from Chapter 5 can be extended to objects with different shapes, and demonstrate cases where it succeeds and fails; and 2) demonstrate the robustness of the proof-of-concept interpreter using synthetic and real-world datasets. For our first goal, objects with varied degrees of shape (concavity) changes are used, and the corresponding results are compared to the mapping. We attempt to demonstrate if the mapping, to some extent, is robust to the changes of shape, and when it would fail to hold. For the second goal, we use synthetic and real-world objects to demonstrate that the interpreter can return a satisfactory result when provided with a correct description.

6.1.1 Key Evaluation Questions and Steps

The evaluation attempts to *prove that the mapping can be extended to other objects with different geometries*, and *demonstrate that the framework can return a satisfactory reconstruction result given a correct description*.

1. Is the mapping robust to changes of object's shape?

We first need to verify that the mapping derived in Chapter 5 is applicable to objects with different shapes. The variation of geometry is too vast and complicated to model, so it wouldn't be possible to consider all possible conditions. As mentioned before, we focus on one geometric property, surface concavity, that in theory can have an impact on the results. We use three synthetic objects with varied degrees of concavity, and verify if the mapping is applicable under all circumstances, and when it would succeed or fail. We use synthetic data to verify the mapping since it would not be practical to change material properties using real world objects. The details of evaluation include:

Data generation the synthetic data is generated in Blender using the same setups presented in Chapter 5. We consider the four property settings that represent four major classes of real-world objects from Chapter 3.

Validation consisting of two steps: 1) the successful algorithm(s) in each case need be identified based on qualitative and quantitative results; 2) the reliable algorithms in each case should be consistent with the mapping results. If that is not the case, we need to find out how robust each algorithm in within the mapping is with respect to concavity changes, and how concavity change affects the reconstruction results.

Criteria To determine that a method returns a reliable result, the accuracy value should be lower than that of the baseline method while the completeness should be higher, and all the statistical measures of the angular error must also be lower than those of the baseline method, including the mean, median, standard deviation, and interquartile range. The qualitative results are used to further confirm the validity of the quantitative results and give a visual sense of the cause of failed reconstruction.

2. Can the proof-of-concept interpreter return a satisfactory reconstruction given the correct description?

Given a correct description of an object, the algorithm chosen by the proof-of-concept interpreter should give a satisfactory reconstruction result of a real-world object. In this case, visual inspection is utilized to determine the quality of reconstruction results since ground true data is unavailable, thus quantitative results cannot be computed. The framework would use the algorithm determined by map-

ping for reconstruction, the result of which would then compared to the baseline algorithm to determine if the quality is acceptable. As previously mentioned, the baseline method is chosen so that it can always provide a decent reconstruction under most circumstances. The details of evaluation are presented as follows:

Data generation: real-world data are captured using similar setups as the synthetic counterparts: for MVS, a Nikon D70S camera with a $18-70mm$ lens are used; for photometric images, a Nikon D70S camera with a $70-200mm$ lens, a handheld lamp, and two reference objects are used (diffuse and glossy); for structured light techniques, a Nikon 70S camera and a Sanyo Pro xtraX Multiverse projector are used, which is positioned with a between angle of around 10°. We used nine everyday objects with varying textures, reflectance properties, and shapes with low concavity.

Validation: we need to demonstrate that the interpreter would return a reliable reconstructed model given the correct description, and a less successful model given an invalid description. The quality of the reconstruction is determined by comparing the result to the baseline method.

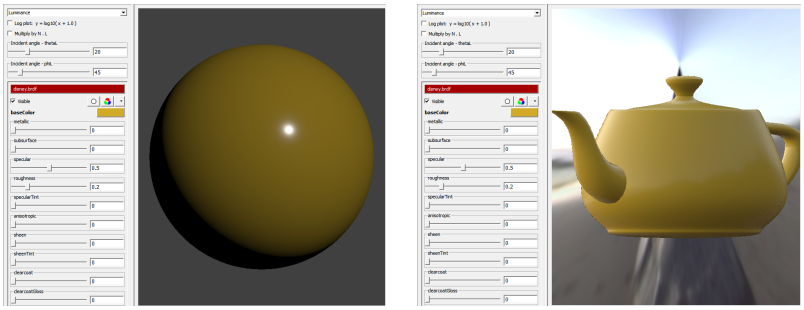
Output: the output of Photometric Stereo algorithms, typically a normal map, is integrated to a surface mesh for visual inspection.

6.2 Parameter Setting

The first step of using the framework is to estimate the parameters of an object's properties. We use the BRDF explorer developed by Disney Animation [3] to visualize the rendered object with changing properties. 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 looks like the real object. A similar approach can be found in [12] where Berkiten and Rusinkiewicz also use a synthetic dataset to find the contributing factors of various Photometric Stereo algorithms.

6.3 Evaluation of Mapping

We first validate that the derived mapping can be applied to an object with a different shape. Given the description of an arbitrary object, we wish to use all three



(a) Lit sphere (b) Lit teapot

Figure 6.1: The UI for determining the property settings, including albedo, specular, and roughness of the surface. The albedo is set as the value channel of HSV colour space. In this case, the albedo is set as 0.8, and the specular and roughness is set as 0.5, 0.2, respectively. (a) demonstrates the effect of the property settings on a sphere while (b) on a teapot.

techniques for reconstruction, to see if the algorithm that has the best quantitative or qualitative results is consistent to the algorithm chosen by the mapping.

6.3.1 Synthetic Datasets

We use three objects with increasing degree of concavity, which are 'bottle', 'knight', and 'king', as shown in Figure 6.2. We select four property settings representing the four classes of the most commonly seen objects discussed in Chapter 4, which is shown in Table 6.1 to assess the robustness of the mapping. We present the results of the mapping in Table 6.1 as a reference to check if the quantitative and qualitative results from the testing data is consistent with the results of the mapping. The results are shown in Figure 6.3, 6.4, and 6.5.

Data 1: bottle

The first test object is a 'bottle', which has shallow indentations on the surface, i.e., low level concavity. The synthetic object is configured with the four property settings listed in Table 6.1. The first column presents the results of the mapping, and the algorithms that produce acceptable results are labeled in the green box. All quantitative and qualitative results align with those of the mapping, so we claim

Class	Texture	Albedo	Specularity	Roughness	Metrics Accuracy	Completeness	Ang error
(a)	0.2	0.8	0.2	0.8	GSL	GSL	EPS
(b)	0.2	0.8	0.5	0.2	GSL	-	-
(c)	0.8	0.8	0.2	0.8	PMVS, GSL	PMVS, GSL	EPS
(d)	0.8	0.8	0.5	0.2	PMVS, GSL	PMVS	-

Table 6.1: Property settings of the three testing objects: 'bottle', 'knight', 'king', which have increasing degree of concavity.

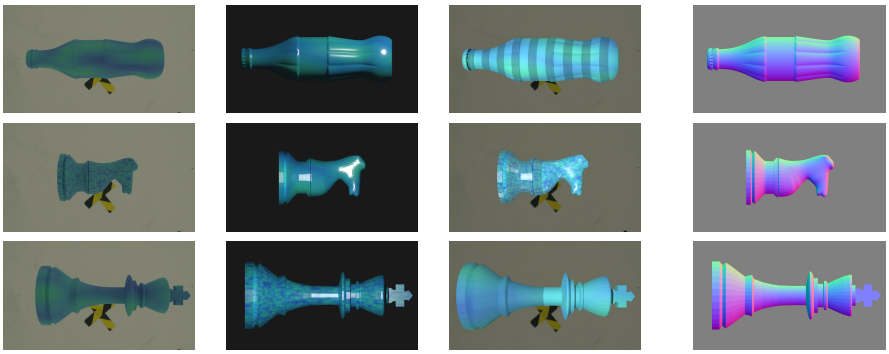


Figure 6.2: The synthetic dataset and groundtruth for the evaluation of the robustness of the mapping to concavity. Three objects with varied degrees of concavity are selected, each is configured with four properties settings listed in Table 6.1.

that the mapping is successfully applied to a surface with low concavity.

Data 2: knight

The second object is a knight chess piece, which has medium concavity. In this case, we can see that the results of PMVS and GSL are still consistent with the mapping. However, the EPS fails to return reliable reconstruction in cases of high specular, such as (b) and (d), which is manifested by an increase in the variation of the angular error, represented by standard variation and the interquartile range. Thus, we claim that the mapping is still valid for PMVS, GSL and for EPS in low specular cases.

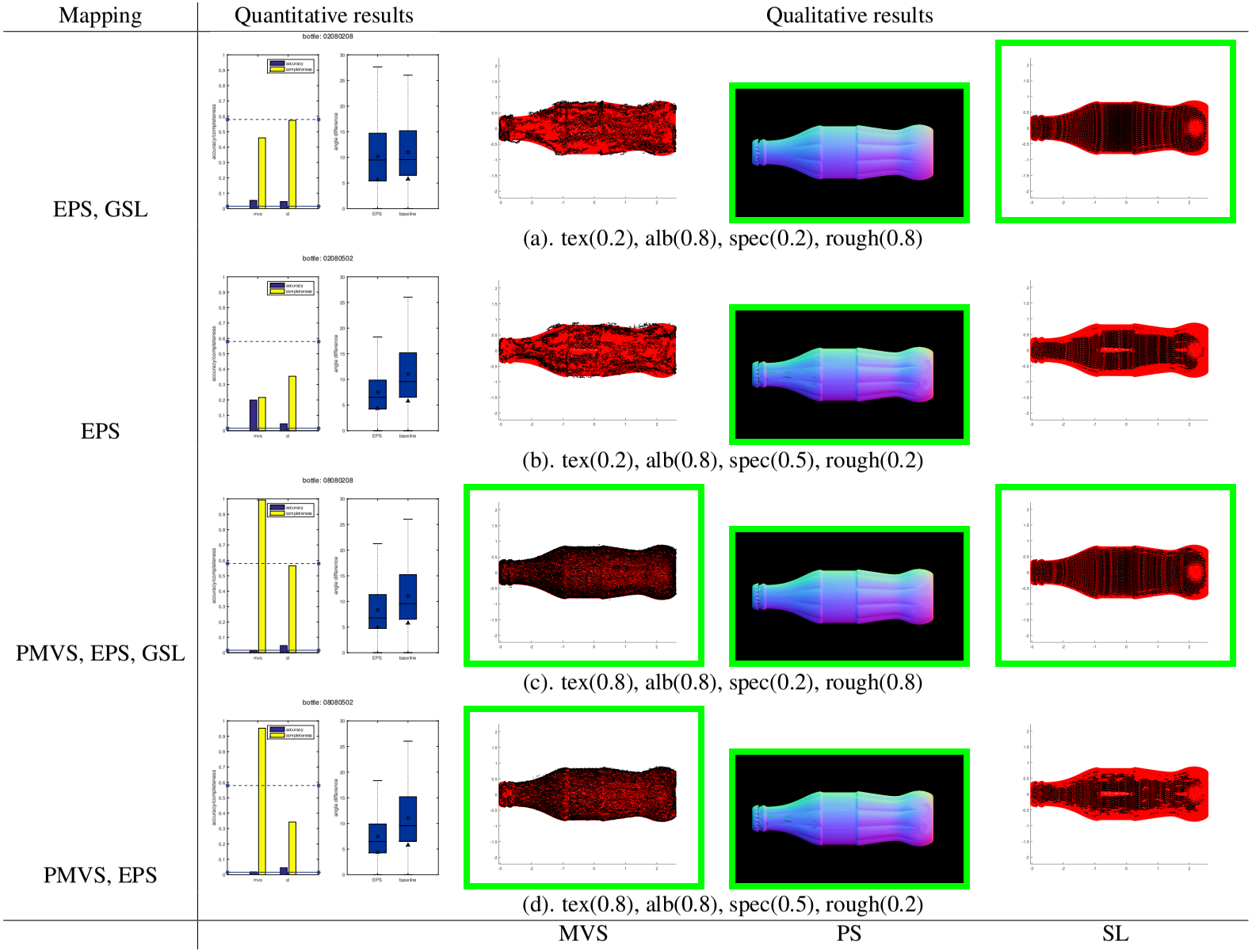


Figure 6.3: The first column shows the best algorithm chosen by the mapping. The quantitative and qualitative performance of each technique on the synthetic dataset is also shown. The red dots represent the ground truth while the black dots represent the reconstruction.

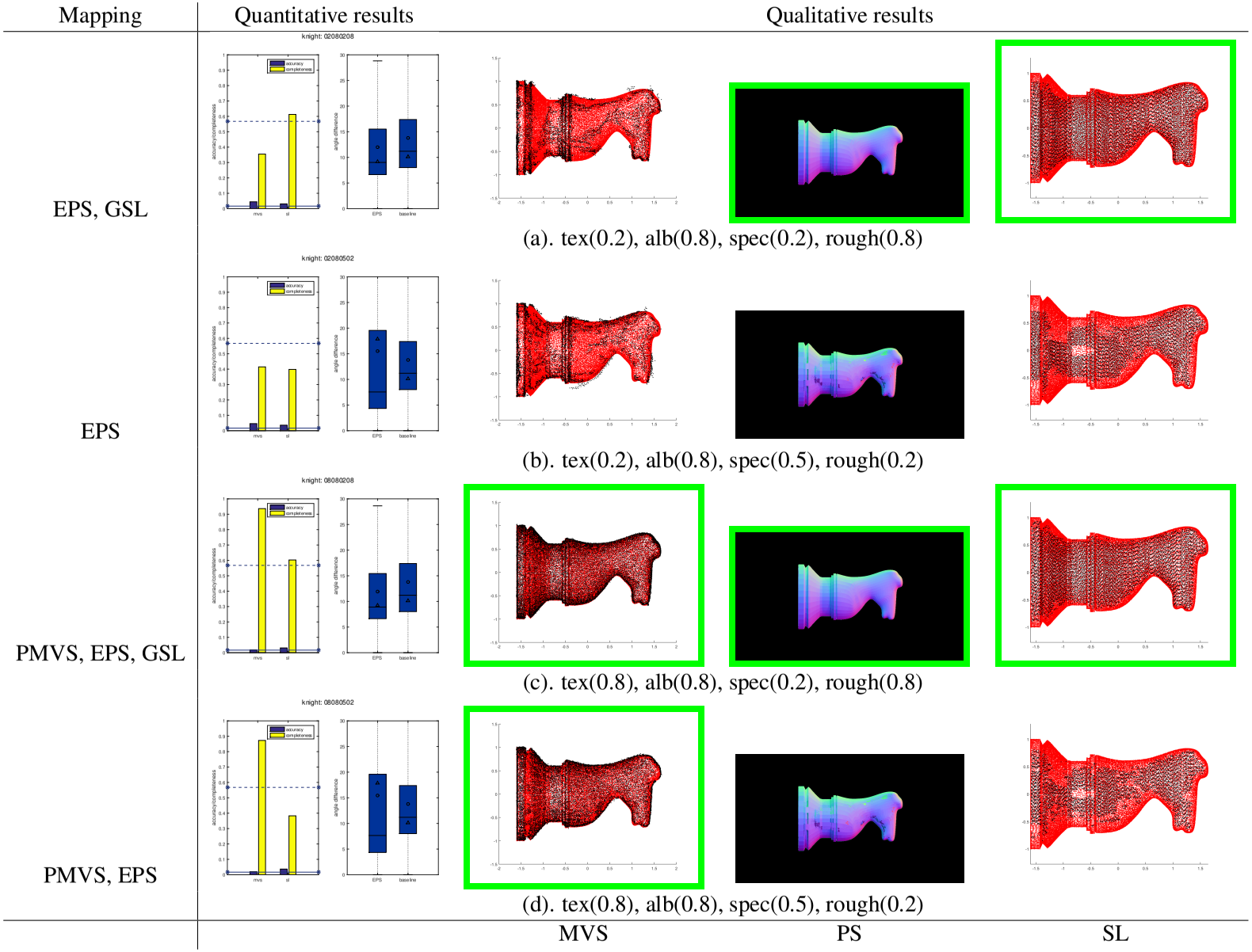


Figure 6.4: The first column shows the best algorithm chosen by the mapping. The quantitative and qualitative performance of each technique on the synthetic dataset is also shown. The red dots represent the ground truth while the black dot represent the reconstruction.

Data 3: king

The last synthetic object is the king chess piece, which has the largest concavity. In the case of high concavity, quantitative results of PMVS and GSL are still consistent with that of the mapping. However, the results of the EPS become inconsistent, which is a result of the cast shadow from the large concavity. Though the result of EPS under conditions (a) and (c) are still better than that of the baseline, the median angular error is above the acceptable threshold, which is 10° in most cases. We can see with more clarity from the normal maps that the cast shadow on the 'cross' leads to completely inaccurate normal estimation, which is labeled by a red rectangle.

Summary

We can conclude that the mapping of PMVS and GSL are robust to concavity, whereas EPS is relatively more sensitive to concavity due to cast shadows. Therefore, we should redirect more research efforts to developing Photometric Stereo algorithms that can reliably deal with cast shadow so that they can be reliably applied to more complex shapes.

6.4 Interpreter

Our framework consists of three separate layers. The upper layer is the description of the problem domain. 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. There are many ways to use the mapping to interpret the problem description. We proposed a proof of concetp interpreter that consists of two components: mapping and constraints.

The mapping constructed using the synthetic dataset from Chapter 5 provides us with a detailed understanding of how different combinations of properties affect the performance of algorithms. This allows us to select an appropriate algorithm

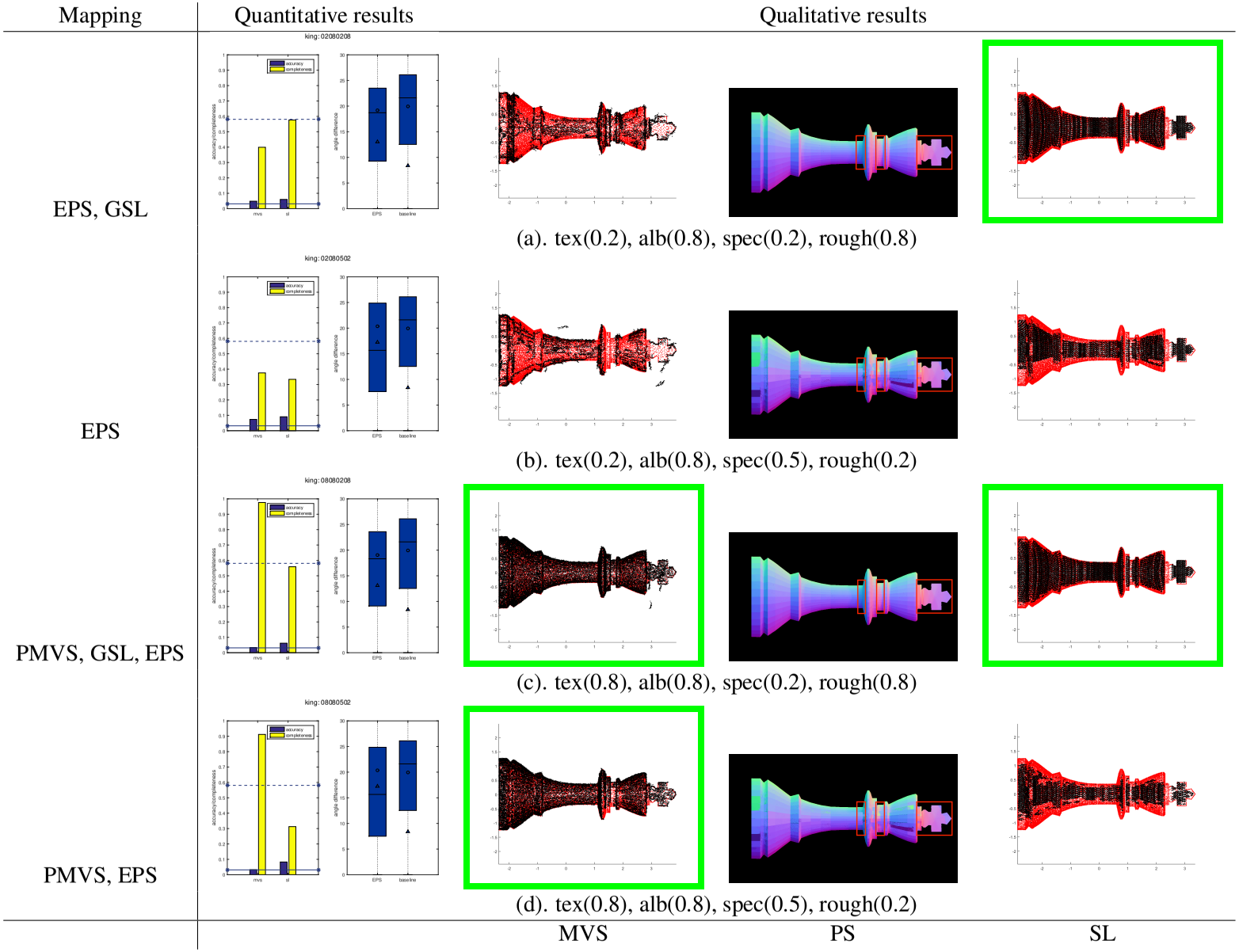


Figure 6.5: The first column shows the best algorithm chosen by the mapping. The quantitative and qualitative performance of each technique on the synthetic dataset is also shown. The green dots represent the ground truth while the black dots represent the reconstruction.

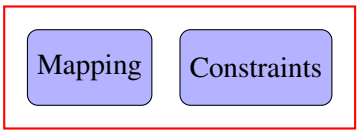


Figure 6.6: Two components of the Interpreter layer.

based on a description of properties.

Next we turn to defining the constraints of the framework. Constraints are provided to allow the user to describe the expected result so that a model best resembling the user's request can be returned by the framework when multiple algorithms achieve satisfactory results. The following constraints are provided:

- Depth-first/shape-first: methods that reconstructs surface orientations from a single viewpoint cannot retrieve the depth information, and thus are refered to as 2.5D reconstruction. Examples of this are Shape from Shading, Photometric Stereo, and Shape from (de)focus, etc.. However, these methods generally can reconstruct fine scale details, thus can achieve results with much higher quality. Intuitively, depth-first can return model with true depth information whereas shape-first prioritizes fine details over depth information.

- Accuracy-first/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.

Under our current interpreter, if a developer-defiend description and constraints have more than one corresponding algorithm available, by default, depth-first has higher priority over quality-first, and accuracy has higher priority over completeness. Since GSL typically generates more accurate resutlts, the order of the algorithms is: GSL $>$ PMVS $>$ EPS.

6.5 Evaluation of Interpreter

We demonstrate real-world use cases of the proof of concept interpreter, which should return a satisfactory result given a valid description, or a less successful one

given an incorrect description. We choose four different problem conditions where each desrcibes one of the four major classes of objects and provides demonstrative results. Please refer to the appendix for more results.

6.5.1 Synthetic Datasets

We generate a dataset of four objects, each representing one of the four classes discussed in Section 3. The mapping from problem conditions to algorithms are summarized in Table 6.2. Given a specific algorihthm, the proof-of-concept interpreter will select an algorithm, and any object that matches this description should be well reconstructed by this algorithm. We use four descriptions that match the four problem conditions listed in Table so that each object should at least return one well reconstructed result. Since a problem condition could be mapped to multiple algorithms, an object that doesn't match the description could potentially have a satisfactory result as well. The reconstruction results of test objects and those of the baseline method are shown in Figure 6.7.

C#	**Obj**	Tex	Alb	Spec	Rough	Mapping
1	barrel	0.8	0.8	0.2	0.8	PMVS, EPS, GSL
2	vase0	0.8	0.8	0.8	0.2	PMVS , EPS
3	bust	0.2	0.8	0.2	0.8	EPS, GSL
4	vase1	0.2	0.8	0.8	0.2	EPS

Table 6.2: Problem conditions and mapping of the synthetic objects.

Data 1: Barrel

Description 1 matches with object *barrel*, which has a reliable reconstruction result. The selected algorithm GSL is also in the mapped algorithms of object *bust*, as shown in Table 6.2. Thus even though **description 1** doesn't match with the problem condition of *bust*, a satisfactory result is also returned. However, the selected algorithm is not in the mapped algorithms of object *vase0* and *vase1*, thus less successful results are returned.

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

Figure 6.7: The evaluation of interpreter using synthetic objects. The first column presents the description provided to the interpreter. Description *i* matches with condition *i* in Table 6.2. The last column is the algorithm selected by the interpreter. The object of which condition matches the description is labeled in green rectangle. Since the interpreter would return a successful reconstruction given a description that matches the condition, the quality of reconstruction of the labeled objects indicates success/failure of the interpreter.

6.5.2 Real-world Datasets

We use similar setups to the synthetic counterparts and captured a real world dataset of nine objects 6.8. The property of these objects are listed in Table A.3. Since we do not have the ground truth here, we resort to visual analysis to see if the appropriate algorithm gives an acceptable reconstruction compared to that of the baseline method.

We use the aforementioned methods to retrieve the parameters of each property. The decomposition of material for each object is presented in Figure A.2. The property settings of each object is listed in Table A.3.

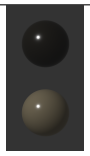
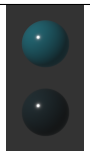
class #	Description	Object	Material
1	textureless diffuse bright		
2	textureless mixed diffuse/specular bright		
3&4	textured diffuse dark/bright		
5&6	textured mixed diffuse/specular dark/bright		

Figure 6.8: The rerepsentatives of the six classes of objects used for evaluation.

Object	Texture	Albedo	Specular	Roughness	Mapping
status	0.2	0.8	0.2	0.8	EPS, GSL
cup	0.2	0.8	0.5	0.2	EPS, GSL
pot	0.8	0.2, 0.5	0.2	0.2	PMVS
vase	0.8	0.2, 0.5	0.5	0.2	PMVS

Table 6.3: Property list for the real-world objects

Data 1: pot

Description 1 matches partially with object *pot* since it has both high and low albedo surface areas. The surface area with high albedo is well reconstructed whereas the surface with low albedo, which doesn't match the description is not reconstructed at all. The selected algorithm is also in the mapped algorithms of object *statue* and *cup*, thus satisfactory results are returned as well. However, this is not the case for object *vase*, thus a very sparse reconstruction is returned.

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

Figure 6.9: The evaluation of interpreter using real-world objects. The first column presents the description provided to the interpreter. Description *i* matches with condition *i* in Table A.3. The last column is the algorithm selected by the interpreter. The object of which the condition matches the description is labeled in green rectangle. Since the interpreter would return a successful reconstruction given a description that matches the condition, the quality of reconstruction of the labeled objects indicate the success/failure of the interpreter.

6.6 Summary

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 framework presented.

Chapter 7

Conclusions

7.1 Future directions

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.

Bibliography

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

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

[3] Brdf explorer. https://www.disneyanimation.com/technology/brdf.html. → pages 71

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

[5] Vxl c++ libraries for computer vision research and implementation. http://vxl.sourceforge.net. → pages 6

[6] N. Alldrin, T. Zickler, and D. Kriegman. Photometric stereo with non-parametric and spatially-varying reflectance. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008. → pages 23, 27

[7] 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 14, 27

[8] N. G. Alldrin, S. P. Mallick, and D. J. Kriegman. Resolving the generalized bas-relief ambiguity by entropy minimization. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–7. IEEE, 2007. → pages 23

[9] 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 9

[10] 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 14, 23, 27

[11] 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 23, 27

[12] S. Berkiten and S. Rusinkiewicz. An RGBN benchmark. Technical report, Technical Report TR-977-16, Princeton University, Feb. 2016. → pages 43, 71

[13] 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

[14] F. Blais. Review of 20 years of range sensor development. *Journal of Electronic Imaging*, 13(1), 2004. → pages 8

[15] 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 35

[16] G. Bradski and A. Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc.", 2008. → pages 6

[17] 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 14, 23, 27

[18] 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 8, 23

[19] O. Faugeras and R. Keriven. *Variational principles, surface evolution, pde's, level set methods and the stereo problem*. IEEE, 2002. → pages 1, 8, 23

[20] 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, 23, 45

[21] 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 9

[22] J. Geng. Structured-light 3d surface imaging: a tutorial. *Advances in Optics and Photonics*, 3(2):128–160, 2011. → pages 20

[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, 9, 23

[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 14, 23, 27

[25] H. Hayakawa. Photometric stereo under a light source with arbitrary motion. *JOSA A*, 11(11):3079–3089, 1994. → pages 13, 23, 27

[26] 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 14, 23, 26, 27, 45

[27] 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 8

[28] B. K. Horn. Shape from shading: A method for obtaining the shape of a smooth opaque object from one view. 1970. → pages 11, 23, 25

[29] 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 21

[30] K. Ikeuchi. Determining surface orientations of specular surfaces by using the photometric stereo method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6):661–669, 1981. → pages 27

[31] S. Inokuchi. Range-imaging system for 3d object recognition. In *Proc. of 7th International Conference on Pattern Recognition, 1984*, 1984. → pages 23

[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 24

[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 24

[34] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing. Available from: <http://www.peterkovesi.com/matlabfns/>. → pages 6

[35] 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, 8

[36] 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

[37] 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 9

[38] 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 9

[39] 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 14, 23

[40] 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 6

[41] D. Marr. Vision: A computational investigation into the human representation and processing of visual information. 1982. → pages 8

[42] W. Matusik, C. Buehler, L. McMillan, and S. J. Gortler. An efficient visual hull computation algorithm. *Tech. Rep., MIT LCS Technical Memo 623, MIT Laboratory for Computer Science*, 2002. → pages 23

[43] S. K. Nayar, K. Ikeuchi, and T. Kanade. Surface reflection: physical and geometrical perspectives. Technical report, DTIC Document, 1989. → pages 39

[44] S. K. Nayar, K. Ikeuchi, and T. Kanade. Determining shape and reflectance of hybrid surfaces by photometric sampling. *IEEE Transactions on Robotics and Automation*, 6(4):418–431, 1990. → pages 27

[45] G. P. Otto and T. K. Chau. region-growingalgorithm for matching of terrain images. *Image and vision computing*, 7(2):83–94, 1989. → pages 8

[46] T. Poggio, V. Torre, and C. Koch. Computational vision and regularization theory. *Nature*, 317(6035):314–319, 1985. → pages 7

[47] 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 16

[48] 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 8

[49] J. Salvi, J. Pages, and J. Batlle. Pattern codification strategies in structured light systems. *Pattern recognition*, 37(4):827–849, 2004. → pages 10, 20

[50] Y. Sato and K. Ikeuchi. Temporal-color space analysis of reflection. *JOSA A*, 11(11):2990–3002, 1994. → pages 14, 23

[51] 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 14, 23

[52] 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 8

[53] 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, 8, 20, 24, 42, 44

[54] 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 20, 42

[55] W. M. Silver. *Determining shape and reflectance using multiple images*. PhD thesis, Massachusetts Institute of Technology, 1980. → pages 14, 23, 26

[56] R. Szeliski. Rapid octree construction from image sequences. *CVGIP: Image understanding*, 58(1):23–32, 1993. → pages 23

[57] P. Tan, S. P. Mallick, L. Quan, D. J. Kriegman, and T. Zickler. Isotropy, reciprocity and the generalized bas-relief ambiguity. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007. → pages 27

[58] M. Tarini, M. Callieri, C. Montani, C. Rocchini, K. Olsson, and T. Persson. Marching intersections: An efficient approach to shape-from-silhouette. In *VMV*, pages 283–290, 2002. → pages 23

[59] 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 9

[60] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008. → pages 6

[61] 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 8

[62] 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 8, 9, 23

[63] 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 11

[64] R. J. Woodham. Photometric method for determining surface orientation from multiple images. *Optical engineering*, 19(1):191139–191139, 1980. → pages 1, 12, 23, 26

[65] 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 25

[66] 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 9

[67] 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 14, 23, 27

Appendix A

Supporting Materials

A.1 Material of real-world objects

(a). box

(b). cat0

(c). cat1

(d). cup

(e). dino

(f). house

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

(g). pot

(h). statue

(i). vase

Table A.2: Images of the real-world objects.

A.2 Parameters of real-world objects

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.3: Property list for the real-world objects

A.3 Results of real-world objects

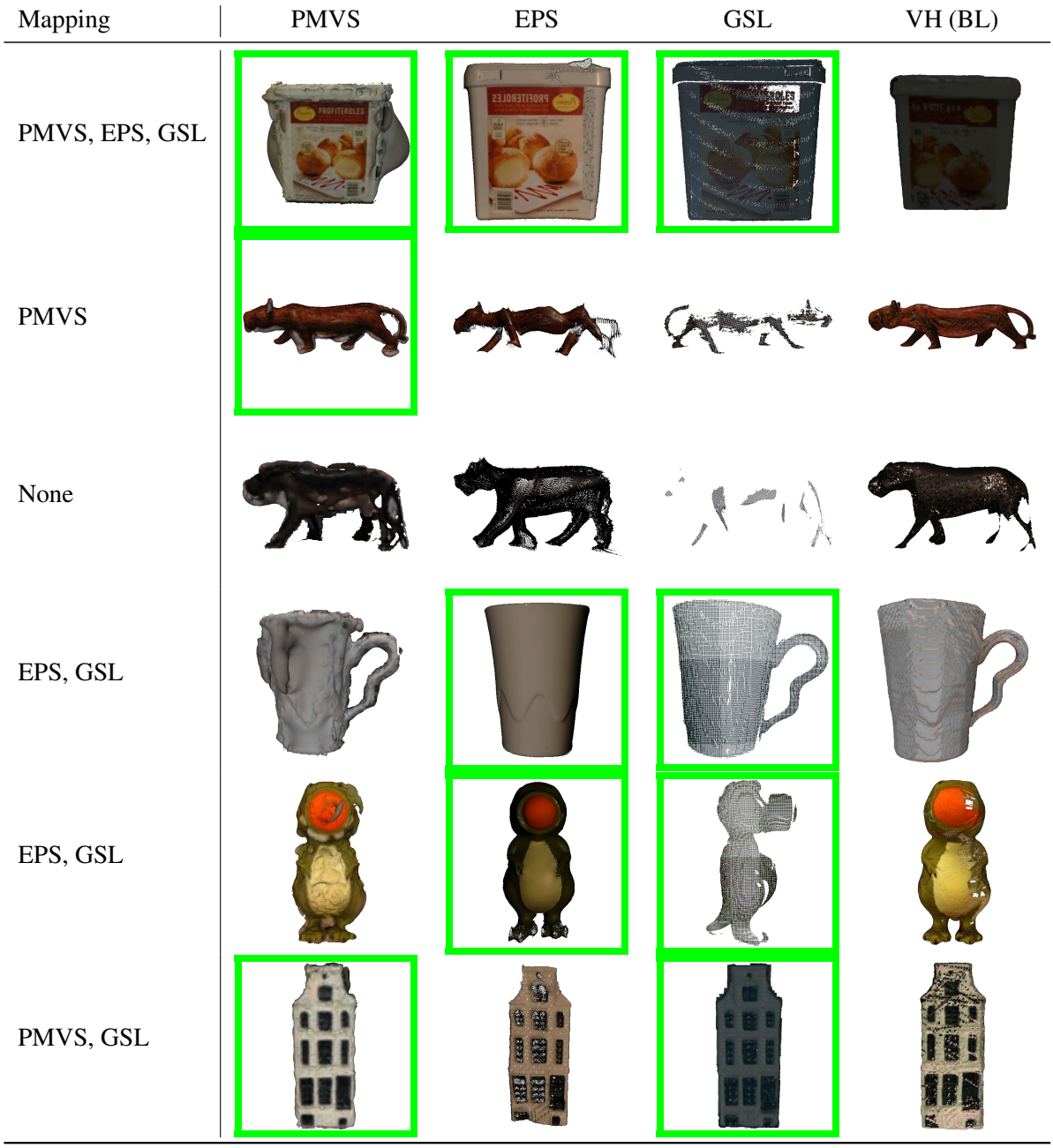


Figure A.1: Reconstruction results of MVS, PS, SL, and the baseline method VH.

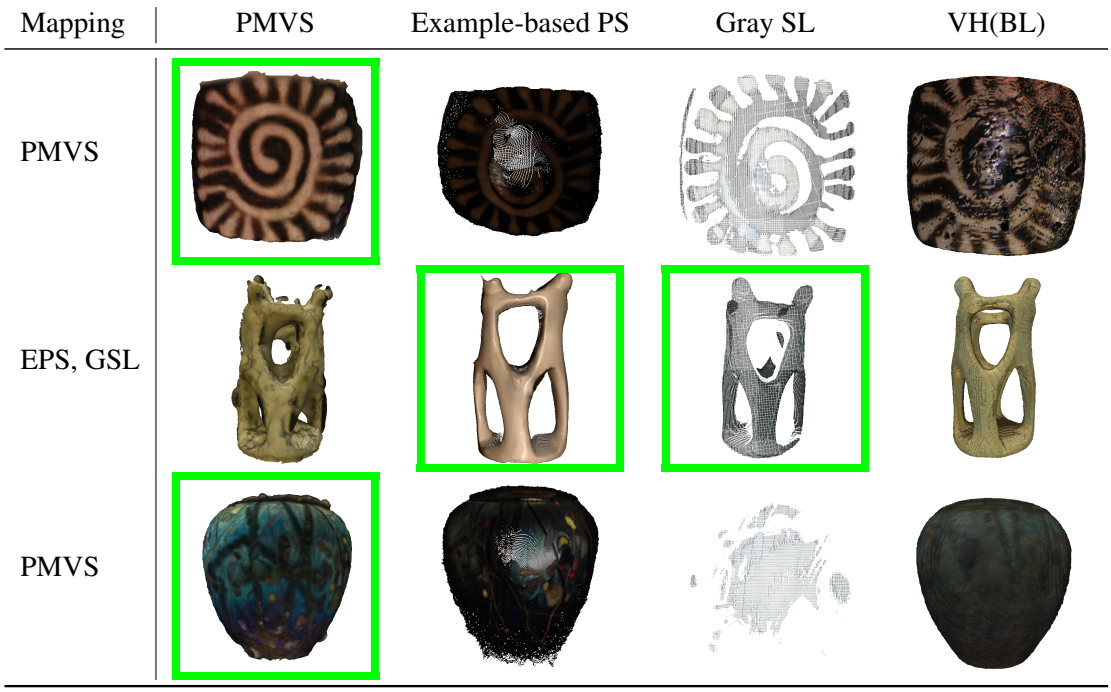


Figure A.2: Reconstruction results of MVS, PS, SL, and the baseline method VH (cont'd).