View and Path Planning for Scaling 3D Acquisition to Many Objects

Xinyi Fan

A Dissertation
Presented to the Faculty
of Princeton University
in Candidacy for the Degree
of Doctor of Philosophy

Recommended for Acceptance
by the Department of
Electrical Engineering
Adviser: Professor Szymon Rusinkiewicz

June 2018

Abstract

Demand for high-volume 3D scanning of real objects is rapidly growing in a wide range of applications, including quality-control for manufacturing, online retailing, entertainment with virtual reality, as well as archaeological documentation and reconstruction. Fully realizing the potential of 3D acquisition requires scanning large numbers of objects with high quality and at reasonable cost. Although mature technologies exist for high-fidelity 3D model acquisition, deploying them at scale continues to require non-trivial manual labor.

This dissertation focuses on studying practical 3D acquisition for large numbers of objects. The problem is challenging, because it is hard to automatically find a proper set of scanner views that can not only completely cover the surface of multiple objects with different shapes, but also capture high fidelity surface model of the objects. Furthermore, it is non-trivial to position a 3D scanner at each of the desired views accurately and efficiently.

We propose a prototype system for multi-object 3D acquisition, which allows non-expert users to scan large numbers of physical objects within a reasonable amount of time, and with greater ease. Our system uses novel planning algorithms to control a structured-light scanner mounted on a calibrated motorized positioning system. We demonstrate the ability of our prototype to safely, robustly, and automatically acquire 3D models for large collections of small objects.

We propose an objective function for automated view and path planning, taking into account both accuracy and efficiency of the scanning system. We analyze different approaches to optimize for the objective, and discuss their performance and practicality.

In addition, we address the problem of surface inaccessibility to further refine our multi-object 3D acquisition system. We explore solutions for improvement from both the hardware and software ends.

Acknowledgements

First, and foremost, I would like to thank my advisor, Szymon Rusinkiewicz, for his guidance, support, and encouragement throughout my entire journey at Princeton. This work would not have been possible without his thoughtful insight and endless patience.

I would also like to thank my committee, consisting of Tom Funkhouser, Adam Finkelstein, Peter Ramadge, and Yuxin Chen, for their time and interest in this work, and their constructive feedback.

Great thanks to Benedict Brown and Linguang Zhang, for their aid and support in the project that led to this thesis. Thanks to Tim Weyrich, Camillo J. Taylor, James Bruce, Joanna Smith, and Lara Laken, for their suggestions and feedback about this work. Thanks to David Radcliff from EE Lab, Larry McIntyre and Barry Runner from the machine shop, for their technical support. I was fortunate to work on various projects with a number of great collaborators: Benedict Brown, Sema Berkiten, Linguang Zhang, Vladimir Kim, Jovan Popović, Amit Bermano, James Bruce, and Fanglu Liu. Even though not all of these projects directly go into this thesis, I value the experience of working with them, which has a great impact on the shaping of my research and career. Special thanks to Vladimir Kim and James Bruce, for their mentorship during my internships at Adobe Creative Technology Lab and Google Research.

During my PhD years, I have had the honor to be a part of the Princeton Graphics and Vision Group, which has been an invaluable resource of ideas, support, and friendship. I am grateful to all the members from the group, including Sema Berkiten, Cynthia Lu, Thiago Pereira, Ohad Fried, Nora Willett, Maciej Halber, Linguang Zhang, Fisher Yu, Yiming Liu, Tianqiang Liu, Pingmei Xu, Elena Sizikova, Kyle Genova, Riley Simmons-Edler, Zeyu Jin, Huiwen Chang, Shuran Song, Yinda Zhang, Andy Zeng, Amit Bermano, Angel Xuan Chang, Manolis Savva, Adam Finkelstein,

Tom Funkhouser, Olga Russakovsky, Jianxiong Xiao, Aleksey Boyko, Linjie Luo, Vladimir Kim, KatieAnna Wolf, Reid Oda, and many others. Thanks for forming all the coffee and tea "trains". Special thanks go to those who have been my officemates, and have continuously supported my project by bearing with the noise made by the stepper motors in earlier versions of my scanning system. I am also grateful to all the reviewers and participants in the TigGraph retreats over the years, for their helpful suggestions and feedback on my research projects.

I thank the following funding sources for sponsoring this work: National Science Foundation under grant CCF-1027962, IIS-1012147, and IIS-1421435.

I would also like to thank Jeffrey Dwoskin for the LaTeX template and document class that has been used in the creation of this work.

Last but not least, thanks to all my dear friends, for making my PhD life a happy one. My deepest gratitude goes to my family, for their endless love and support. I dedicate my thesis to them.

To my family.

Contents

Abstract iii

Acknowledgements iv

List of Tables ix

List of Figures x

1 Introduction **1**

1.1 3D Acquisition at Scale 1

1.2 Statement of Problem 2

1.3 System Design 4

1.4 Dissertation Outline 7

2 Prototype Multi-object 3D Acquisition System **9**

2.1 System Design for Scanning Multiple Objects 10

2.2 System Pipeline 11

2.3 Scene Exploration 14

2.4 Positioning in the "World" 19

3 Scanning **23**

3.1 3D Scanning Techniques 23

3.2 Implementing A Structured-Light Scanner 25

3.2.1 Illumination Pattern Design 25

3.2.2 Scanner Calibration 28

4 Planning **30**

4.1 Related Work 30

4.2 Objective Setup 32

4.2.1 Search Space Discretization 33

4.2.2 View Quality Metric 35

4.2.3 Overlap-Aware Heuristics 37

4.3 Sequential View and Path Optimization 41

4.3.1 Optimizing the View Planning Objective Function 41

4.3.2 Discussion on Performance 44

4.3.3 Path Finding 47

4.4 Joint Optimization 47

4.4.1 Joint View Selection for Multiple Objects 48

4.4.2 Joint View and Path Planning 55

5 System Evaluation **64**

5.1 View Planning Evaluation 64

5.2 Path Planning Evaluation 67

5.3 System-Level Evaluation 68

5.4 Object Variety Evaluation 70

6 Addressing Surface Inaccessibility **75**

6.1 Augmented Hardware Design 76

6.2 Iterative View Planning 78

6.2.1 New Pose Suggestion 79

6.2.2 View Quality Re-evaluation 80

7 Conclusion, Discussion, and Future Work **84**

Bibliography **88**

List of Tables

2.1 Default setting for the user-adjustable candidate view sampling parameters. 18

5.1 Comparison of total time for our view planning vs. naive strategies employing a fixed number of views per object. 65

5.2 Comparison of total distances and times for our TSP-based path planning vs. a naive path planning strategy. 67

List of Figures

1.1 Scan an object from one point of view. 3

1.2 3D acquisition pipeline. 4

1.3 Typical existing 3D acquisition systems, with references being (a) [9], (b) [35], (c) [2], (d) [45], (e) [61], (f) [62], (g) [3], and (h) [1]. 6

2.1 The physical layout of our prototype scanning system. 11

2.2 Illustration of major system components. 12

2.3 Hardware devices of the compact structured-light scanner with dimensions. 14

2.4 Four binary masks of the four toy soldiers scene obtained from the webcams for silhouette carving. 15

2.5 A scene with four toy soldiers (left) and the approximate models obtained via silhouette carving (right). 16

2.6 A scene with three flat objects (left) and the approximate models obtained via extruding the 2D contours, where mesh triangle edges are shown to better present the model shape (right). 16

2.7 An top view visualization of candidate views computed around four objects. 17

2.8 An top view visualization of candidate views adaptively computed around a long and thin objects. 18

2.9 Actuators in our positioning system. 19

2.10 The positioning system, with a calibration target on the scanning platform. 20
2.11 The AprilTag pattern overlaid with calibrated centers of the tags (left) and the centers reprojected into the world (right). Tag ids are color-coded from blue to red. 21
2.12 Zoomed in view of the calibrated tag centers reprojected in the world, showing the variation across the scanning platform. 22
2.13 The initial scan poses provided by the scanner calibration (left), together with the final result of registration (right). 22

3.1 Stacked gray code column stripe patterns to split the illuminated space into small regions. 26
3.2 Stacked phase shifting column patterns that simulates a sine wave and can interpolate between adjacent light planes defined by the binary stripe patterns. 27
3.3 Diamond pixel for vertical, horizontal, and diagonal lines arranged in the projector's light engine. `http://www.ti.com/lit/ug/dlpu011f/dlpu011f.pdf` 27
3.4 Both the gray code patterns (above) and the phase shift patterns (below) are rotated for 45° to align with the mirror arranging direction in the projector. 28
3.5 The checkerboard pattern used for camera and projector calibration, where the three dots define the center and orientation of the pattern. 29

4.1 The objective function in Equation 4.4 examines surface point samples (left) and views selected from a given candidate set (right) to measure the goodness of a configuration. 34

4.2 View assignment quality visualization with different objective functions. Each scanner view is represented by a camera-projector pair of frustums connected with a dotted line. The object model is represented by surface samples with the view quality value encoded according to the jet color map, as shown in (d). Red means good view quality and green means poor. 39
4.3 Given a scene of "dragon fighting armadillo" with increasing distance between the two objects (a), we visualize the surface sample view quality based on the corresponding selected views (b). The close-up views show that, as the armadillo moves from "close" (left) to "near" (middle), and then "far" (right) from the dragon, the view quality of the head of the dragon improves. 40
4.4 The optimal objective value achieved by different approaches with varying γ, with higher value indicating better overall view quality. . . 46
4.5 The approximate model of the 8 objects used for view planning. . . . 49
4.6 Comparison between the view selection objective value $\mathcal{E}_v$ in log space by optimizing for each object independently and by optimizing for all the objects together, with varying desired view quality parameter ω. . 50
4.7 Comparison between the average view quality achieved by optimizing for each object independently and by optimizing for all the objects together, with varying desired view quality parameter ω. 51
4.8 Comparison between the per-object number of views selected by optimizing for each object independently and by optimizing for all the objects together, with varying desired view quality parameter ω. . . . 53
4.9 Comparison on the CPU time between optimizing for each object independently and by optimizing for all the objects together, with varying desired view quality parameter ω. 54

4.10 Comparison between path length $\mathcal{L}$ achieved by jointly and sequentially optimizing $\mathcal{E}_v$ and $\mathcal{E}_p$, with varying desired view quality parameter ω. 56
4.11 Comparison between jointly and sequentially optimizing $\mathcal{E}_v$ and $\mathcal{E}_p$ from the overall planning objective $\mathcal{E} = \mathcal{E}_v + \mathcal{E}_p$ in log space, with varying desired view quality parameter ω. 57
4.12 Comparison between the average per-object view quality $\omega \cdot \mathcal{Q}$ achieved by jointly and sequentially optimizing $\mathcal{E}_v$ and $\mathcal{E}_p$, with varying desired view quality parameter ω. 58
4.13 Comparison between the number of views selected by jointly and sequentially optimizing $\mathcal{E}_v$ and $\mathcal{E}_p$, with varying desired view quality parameter ω. 59
4.14 Comparison on the breakdown in the objective change introduced by the joint optimization, with varying desired view quality parameter ω. 61
4.15 Visualization of a path (296 seconds long, 57 views) obtained by the joint approach (left) and a path (336 seconds long, 54 views) obtained by the sequential approach (right). 62
4.16 Comparison on the CPU run time between the joint and the sequential approaches (left) and comparison between the saved runtime for the acquisition system and the extra CPU time introduced by using joint view and path planning (right). 63

5.1 Front (left) and back (right) side of the reconstructed models of four objects scanned simultaneously with adaptive view planning. Models with the same color correspond to each other. 65

5.2 We compare the scans obtained using our view planning (right) to those acquired with a naive method employing five (left) or nine (middle) views, equally spaced around the centroid of the object. Our view planning result also selects nine views in this case. 66
5.3 Comparison of our TSP-based path planning (a) and naive path planning (b). The former reduces motion time by approximately 15%. . . 67
5.4 The two plots show the total amount of time a human interacted with the scanning system (left) and the total amount of idle time when the human did not need to attend the system between two adjacent interactions (right) on scanning batches of fresco fragments using both our system and the turntable system. Our system required less interaction time overall and afforded far larger gaps between interactions during which the operator was free to do other work undisturbed. 68
5.5 Reconstructed models of increasingly-large sets of fresco fragments, as used in our scalability experiment. The batch size is, from left to right, 9, 16, and 25. 69
5.6 We scan a scene with ten toy soldiers based on views (a) planned from approximate, silhouette-carved models (b), and reconstruct high resolution 3D models (c). Each toy soldier is about 5 cm tall, and our system reconstructs sub-millimeter geometric detail (d). 70
5.7 A 20 cm tall angel figurine (left) is scanned and reconstructed (right) using our system, with the object platform adjusted to three heights. 72
5.8 Two differently sized soldiers (left) are scanned together and reconstructed (right) with our system. 72
5.9 A flower with a long, thin stem (a) is scanned and reconstructed (d) with our system. Elongated objects such as this are a worst case for a turntable-based system with equally-spaced views (c). 73

5.10 An $8 \times 6 \times 3$ cm reproduction cuneiform tablet (left) is scanned and reconstructed (right) with our system. 74

6.1 Detail in certain region cannot be perfectly reconstructed due to missing data caused by surface inaccessibility. 76

6.2 An imagined augmentation to the current acquisition system design, with a transparent scanning platform and a coupled scanner from below. 77

6.3 We scan the front side of an object using our system as usual but the back side through a sheet of transparent thin acrylic (left) and obtained the final models nicely reconstructed for both the front (right) and back (middle) side, which are then merged together. 78

6.4 An iterative 3D acquisition pipeline. 78

6.5 The reconstructed 3D model (left) and its convex hull (right. 80

6.6 The reconstructed 3D model (left) re-oriented at different stable poses (middle and right). 80

6.7 The view quality visualization on point samples representing the object before (left) and after (right), where red stands for good view quality and green for bad. 81

6.8 The view quality visualization on point sampled from the reconstructed model which is oriented into two new stable poses suggested by Algorithm 2, where red stands for good view quality and green for bad. . . 82

6.9 Side by side comparison among the reconstruction after 1 iteration (left), 2 iteration (middle), and the merged data acquired from iteration 2 (right). 83

Chapter 1

Introduction

"It's supposed to be automatic, but actually you have to push this button."

John Brunner

1.1 3D Acquisition at Scale

3D acquisition, namely measuring the 3D shape and reconstructing the 3D digital model of real-world objects, has made realistic capture possible. Given the dimensionality of the world we observe, 3D models naturally become a common and even expected mode of representing tangible objects for a variety of purposes. The completeness and accuracy of the data obtained from 3D acquisition can potentially benefit applications across a wide range of fields:

- **Industrial inspection:** Non-contact 3D scanning has become an important tool for quality control in manufacturing. While manufacturers are faced with the challenge of mass-produce innovative products with high quality at a rapid pace, fast and precise 3D measurement at scale would provide solutions to automatic industrial inspection.

- **Online and automated retail:** 3D digital models with high fidelity could be created for different types of commodities. This would enable retailers to advertise their products in a more realistic way to customers shopping online. Such 3D product catalogs could further spur the development of automated retailing technology.

- **Cultural heritage digitization:** Digitizing the shape of artifacts en masse would enable efficient documentation in a precise and permanent form. Such digital collections would create easy viewing access for a larger audience regardless of location. In addition, new research opportunities could be explored, since the digital model could be studied without fear of damaging the physical artifacts.

- **Virtual, augmented, and mixed reality:** Conveniently acquiring the 3D digital model for real-world objects would naturally yield easy 3D content creation. Combining this with emerging technologies like virtual, augmented, and mixed reality would inspire more exciting applications.

Fully realizing the potential of 3D acquisition, however, will require scanning large numbers of objects with high quality and at reasonable cost. While scan quality and speed are continuously being improved by state-of-the-art scanning systems [35, 45, 9, 63], their dependence on manual labor remains a major bottleneck to scalability. This dissertation focuses on studying the problem of practical 3D acquisition at scale.

1.2 Statement of Problem

Before we start delving into solving the problem of acquiring the 3D models for thousands of objects, let's first look at how a typical 3D acquisition process happens for a single object. As illustrated in Figure 1.1, given an arbitrary object, a 3D scanner

can be placed in the "view space" around it, to capture the depth information about the object surface. In order to obtain a complete 3D model of the object, the scanner

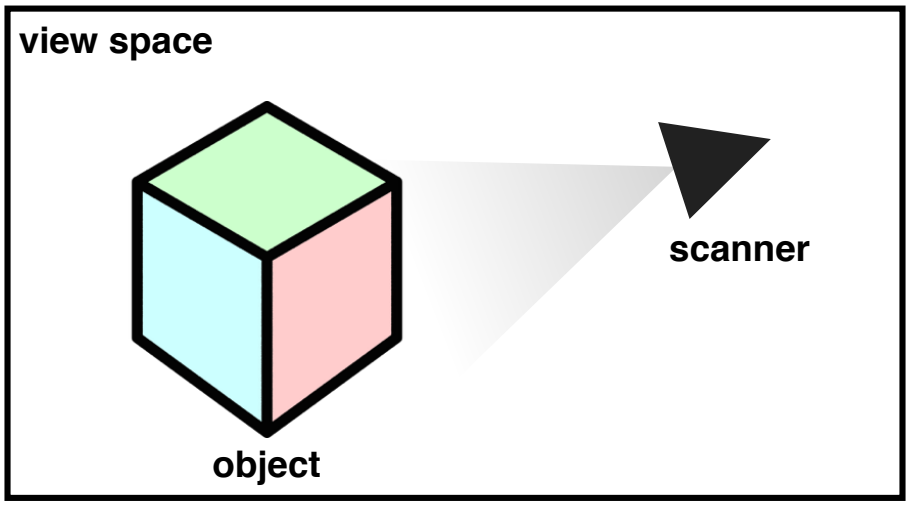


Figure 1.1: Scan an object from one point of view.

needs to scan the object surface from more than one point of *view*. In our context, a *view* specifies the intrinsic parameters of the scanner, as well as the relative pose between the scanner and the object, assuming the object to be la rigid body. Data obtained from multiple views are to be registered in a common coordinate system, and integrated into a single complete model, representing the 3D shape of the object. The use of a single 3D scanner, which is mostly the case, the relative pose between the object and the scanner needs to be altered to achieve multiple views.

To obtain a complete model of an object with perfectly high fidelity, one could imagine scanning the object from an infinite number of views sampled in the view space around the object. In practice, however, it takes non-trivial time to scan per view, and therefore, it is necessary to *select* only a subset of all possible views for the scanner to scan from. The process of determining a suitable set of views for a scanner to capture the complete surface of an object is defined as *view planning*. Given the set of views selected through view planning, the process of altering the relative pose between the scanner and the object is usually called *positioning*. View planning, and

positioning, together with the following scanning and reconstruction, constitute a general pipeline of a 3D acquisition system, as shown in Figure 1.2.

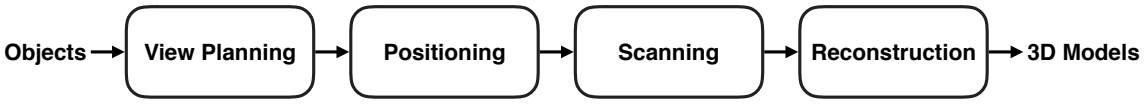


Figure 1.2: 3D acquisition pipeline.

Our goal is to make 3D acquisition practical in a scenario with large numbers of objects. It is non-trivial to generalize the pipeline in Figure 1.2 from a single object to many. The key obstacle preventing this from scaling up in fact, is the manual effort involved, in particular for selecting and altering the views. In this dissertation, we study different approaches to solving the view planning problem, which applies to arbitrary multi-object scanning scenarios. In addition, we propose an end-to-end 3D acquisition system, composed of a view planning module, a positioning system, a scanner, and the subsequent reconstruction, to validate the practicality of our design.

1.3 System Design

We argue that the key to making 3D acquisition at scale practical is to reduce the manual effort required per object. This is in contrast to the design goals of many existing 3D acquisition systems. Different strategies have been adopted to select views and alter the pose between the scanner and the object being scanned, but none of them can be efficiently applied to a multi-object situation to obtain high-fidelity data:

- **Fixed scanner setup:** Many existing scanning systems are single object oriented. Among those, one typical design is to fix the scanner to face a motorized turntable, which can carry the object being scanned and rotate to achieve different views. For example, Figure 1.3(a) shows the laser scanner used by Brown et al. in their work [9]. In such a setup, the viewing angle between the scanner

and the center of the turntable is fixed, which limits the degree of freedom in the relative poses that can possibly be achieved between the object and the scanner. This is likely to yield inaccessible parts for the scanner to reach, especially when scanning objects with more complicated surface shape, such as concavities. To start a scanning sequence in such a setup, a user needs to interacts with the system every few seconds or minutes by positioning a new object and occasionally rotating the object to uncover parts that could not be seen. In particular, to scan the bottom of an object, the user needs to flip the object on the turntable.

- **Gantry-support scanner:** A gantry-support 3D scanner has been designed in the Digital Michelangelo Project [35] to capture the delicate surface geometry of large statues. During the scanning, the object (statue) is kept static, and a gantry is used to carry the scanner to reach different views surrounding the object (Figure 1.3(b)). The movement of the scanner is carefully controlled by expert human users.

- **Hand-held scanners:** Another category of systems [45, 2] allow the user to hold either the scanner or the objects, as shown in Figure 1.3(c) and (d). The hand-held setup makes it easy to control where to capture data from, and thus solves the problem of covering the complete object surface. At the same time, however, the required human interaction prevents the scanning from scaling up. Recent work by Wu et al. [61] employs robotic arms to hold the scanner and the object (Figure 1.3(e)), which eliminates the manual effort involved in the task of scanning a single object, but switching from one object to another has not been addressed. In this case, either a human operator or a robot capable of moving the object is necessary. Further more, physical contact may rise as another potential issue when the objects being scanned are fragile and/or precious.

- **Mobile robotic scanning systems:** It has been increasingly common for 3D scanning devices to be integrated into mobile robotic systems to collect visual information about certain environments [62, 3, 28]. In these setups, the scanner(s) are mounted on motorized agents, e.g. humanoid robots (Figure 1.3(f)), cars (Figure 1.3(g)), or drones (Figure 1.3(h)). The agents' moving trajectories are either controlled by human operators or some auto-navigation programs [43]. These techniques can be especially useful for applications regarding scene-level acquisition, such as civil mapping and indoor reconstruction. Object level acquisition, on the other hand, requires more careful planning and finer resolution.

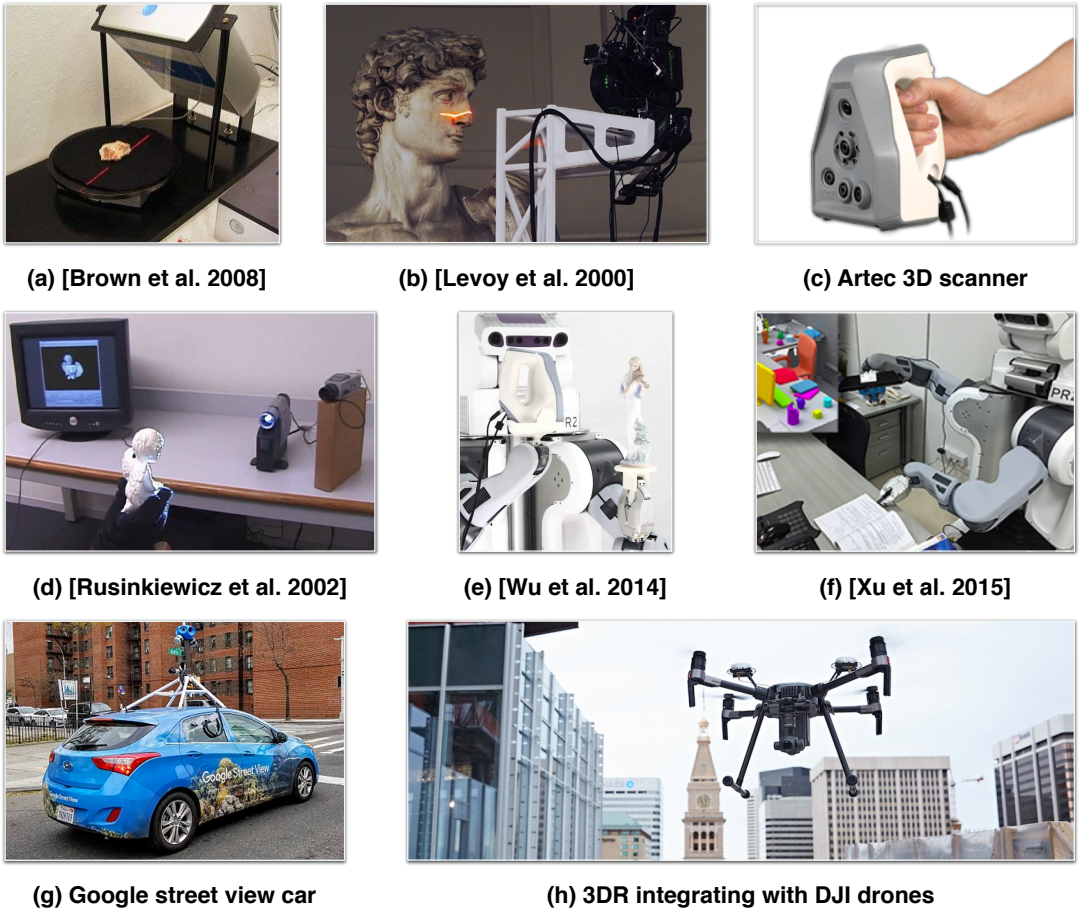


Figure 1.3: Typical existing 3D acquisition systems, with references being (a) [9], (b) [35], (c) [2], (d) [45], (e) [61], (f) [62], (g) [3], and (h) [1].

In summary, it is not trivial to adapt the scene level acquisition systems to perform high quality object scanning, as the view planning and positioning requirement can be very different. Streamlining any existing single object oriented scanner, on the other hand, requires reducing the *number* of user interactions, not just their length.

Our system: We present a system for automatically scanning multiple 3D objects at a time. In our system, the user places several to several-dozen objects in the working volume, and the system automatically acquires their rough shapes and positions. The system then plans an optimal set of views to scan the objects at high quality, as well as the exact path along which the scan head should move. Using a 3-degree-of-freedom positioning system, our scanner automatically performs the 3D scans, restricting the necessary user interaction to placing the objects initially, then flipping them over halfway through scanning if necessary. The latter interaction could be avoided by placing the objects on a sheet of glass and using a second scan head to scan from below; we run a preliminary experiment to explore the feasibility of this further refinement in this paper. We also explore scanning of larger objects by adding a manually-adjusted fourth degree of freedom.

The main benefit of our system, therefore, is that it allows the entire scanning process to happen with no human interaction. In contrast, scanning the same number of objects one-by-one with an existing system might require a similar total scanning time, but with human interaction required every few seconds or minutes.

1.4 Dissertation Outline

The remainder of this dissertation is organized as follows. Chapter 2 describes the workflow of our automated multi-object 3D acquisition system prototype, and justifies major design decisions. It also introduces the design and implementation for the positioning component of the acquisition system, and elucidates the process of

registering all the sensors and the motors of the acquisition system to a common coordinate system. Chapter 3 describes the 3D scanning techniques used in our system, with detail about how the structured-light based scanner is implemented. Chapter 4 analyzes the view and path planning problem for multi-object scanning. It starts by formulating the objective function in consideration of how well the object surface is viewed, as well as the time budget. Different approaches to optimize for the objective function are studied and compared, including (1) sequentially solving the view planning and the path planning problem, and (2) jointly solving the view and path planning problem. Chapter 5 evaluates each component of our acquisition system, as well as system level performance. Both visual and quantitative results are presented. Chapter 6 further studies the refinement for the multi-object 3D acquisition system, by addressing the problem of surface inaccessibility. Possible solutions are explored, including both improving the hardware design for the system, and refining the view planning by making it iterative. Finally, Chapter 7 presents conclusions about this work, discusses its limitations, and suggests ideas for future work.

Chapter 2

Prototype Multi-object 3D Acquisition System

"Moreover, since the sun remains stationary, whatever appears as a motion of the sun is really due rather to the motion of the earth."

Nicolaus Copernicus

Capturing from multiple view points is necessary for a scanner to acquire a complete and high-fidelity 3D model of an object. The choice of views directly influences the overall scan quality since it determines whether there is full coverage of the object and whether good data can be captured for every part of every object (scan quality is generally affected by the object's distance from the scanner and angle of incidence). Most existing motorized scanning systems uniformly sample view space, often by rotating a turntable. Objects with deep concavities or other irregularities often need very dense sampling in this scenario, even if large parts of the object are convex and can be covered by few scans.

Our system, in contrast, optimizes the number and position of views using a low-resolution overview model acquired with a set of webcams. Because we move the (small) scan head rather than the (large) table of (potentially fragile or unsteady) objects, we can support a wider range of motion and obtain more optimal views. Our scanner supports automatic motion with three degrees of freedom — two directions of horizontal translation as well as rotation around the vertical axis — as a reasonable compromise between engineering complexity and flexibility. It works well for scanning collections of small objects. For larger objects that need scans from different heights, we can manually raise or lower the scanning platform between sets of scans. In any case, the view planner handles arbitrary degrees of freedom if the scanning stage provides them.

2.1 System Design for Scanning Multiple Objects

Figure 2.1(left) illustrates the physical layout of our scanning system [18]. The objects are placed on a flat, stationary platform, with the scanner mounted overhead at a fixed, 45° tilt, as shown in Figure 2.1(right). The scanner has three degrees of freedom of motion, which allows it to translate in the x and y directions (parallel to the platform) and rotate about the z axis. The tilt and height are both fixed, although they can be adjusted manually to accommodate objects of different sizes.

Automatic scanning systems typically move either the object or the scan head in order to obtain multiple views. Moving the object is more common, because a motorized turntable works well for single objects, is relatively easy to build, and does not take much space. Alternatively, robot- or vehicle-mounted scanners work well for navigation applications and for scanning buildings and large outdoor scenes.

Scanning many closely spaced, small objects at once falls into neither of these categories. Full coverage requires a denser set of views than a turntable can provide.

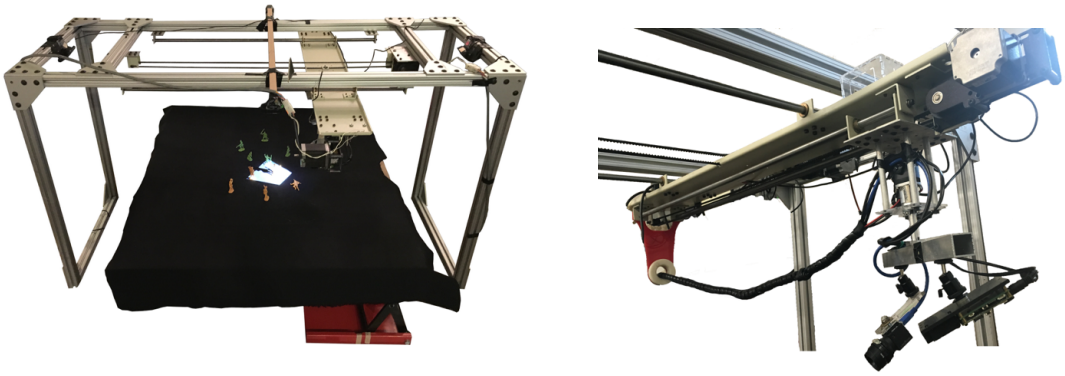


Figure 2.1: The physical layout of our prototype scanning system.

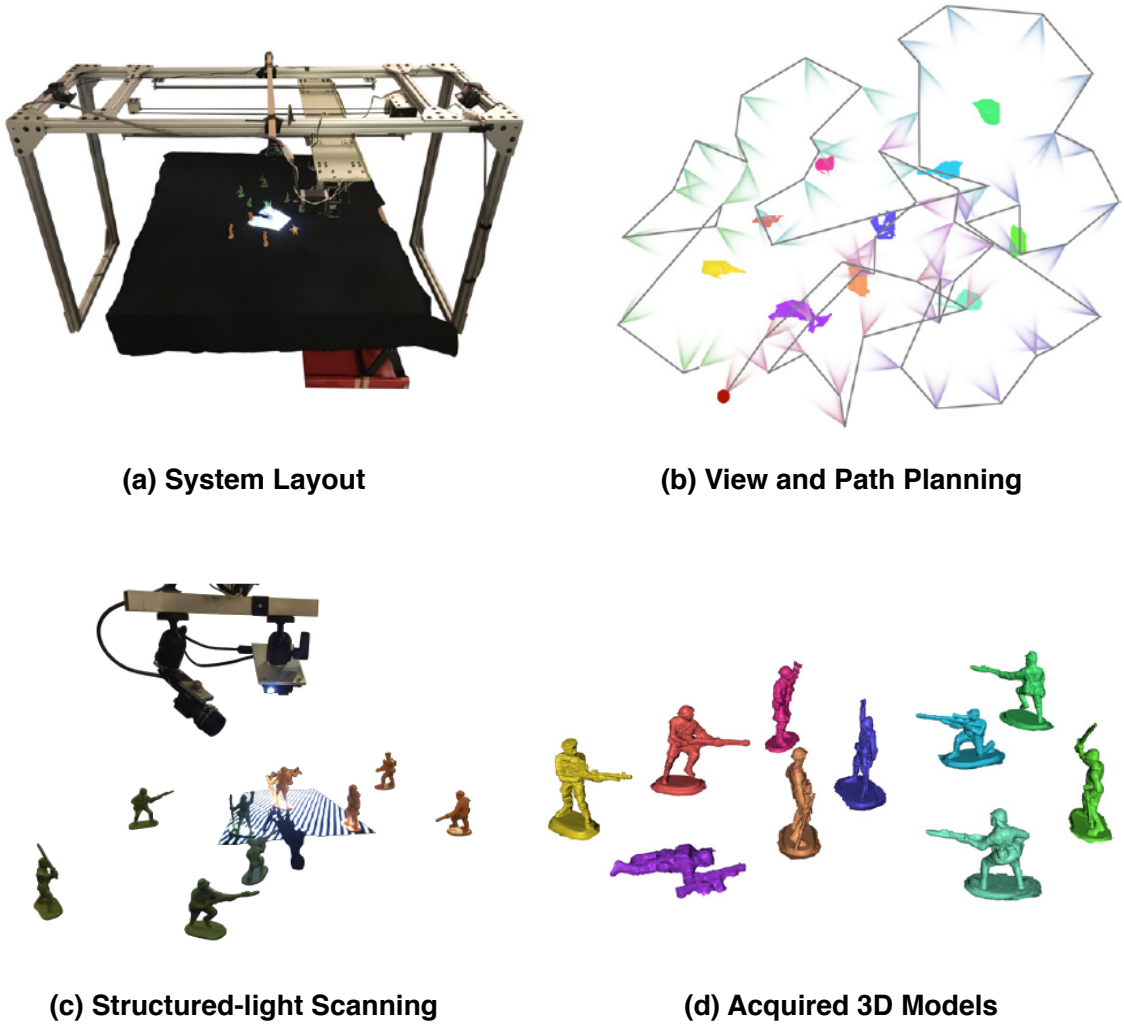
The scanning stage would need, at a minimum, to move forward, back, and side-to-side, as well as rotating around its axis. To prevent objects from tipping over, breaking, or crumbling, vibrations would need to be damped. Because the scan head is small and can tolerate vibration, we believe that moving the scan head is a simpler and cheaper option to engineer.

A free-moving robot would run into a different problem in our scenario: it is an alternative way to move the scan head rather than the objects, but it would still need to navigate *between* the objects. Unless the objects are spaced far apart, this is a physical impossibility.

2.2 System Pipeline

The design of our automatic acquisition system follows the work-flow shown in Figure 1.2 and Figure 2.2. The system starts with a scene exploration process, which examines the shapes and poses of the objects to be scanned. This information is passed to the view planning algorithm, which outputs an optimized set of scanner poses. Following an optimized path, the scanner is then brought to these desired poses by a calibrated positioning system and stops at each to perform a scan. Standard

registration and integration algorithms are applied to the captured data to generate high-fidelity 3D models for the objects.



(a) System Layout **(b) View and Path Planning**

(c) Structured-light Scanning **(d) Acquired 3D Models**

Figure 2.2: Illustration of major system components.

Scene exploration. Our system starts by finding the rough geometry of all the objects in the scene, then generating a set of candidate scanner views that will be a superset of the final selected views. The availability of cheap sensors makes it possible to quickly acquire sufficient information about the scene to enable view planning. Specifically, with the objects placed on the scanning platform, we use a set of fixed calibrated webcams to capture the layout of the objects from the top, then perform

silhouette carving [33] to obtain approximate object models for the view plan. We present the design and implementation of the scene exploration in the following section (2.3).

View planning. Based on the rough models produced by the scene exploration stage, we plan an optimal set of scanner poses (also referred as *views*). These adaptively cover the accessible part of the objects, while also ensuring a fair amount of overlap between adjacent views. Our view planning selects the best views by optimizing a view-quality-based objective function. Details of the view planning are presented in Chapter 4 and 6, which discusses several alternative approaches for optimizing the same objective.

Path planning and positioning. With the set of best views computed by view planning, we use a calibrated motion system to position the scanner at the desired poses. Due to stability concerns we assume that the scanner moves at a moderate speed, and hence in a scaled-up scenario with a large number of objects, the total travel time will be non-trivial. This motivates us to equip the positioning system with a path-planning component, which computes an approximate shortest path to traverse all the desired scanner poses. We discuss our positioning system in Section 2.4 and the path planning in Chapter 4.

Scanner setup. Once we have positioned the scan head, we acquire 3D data using a standard structured-light technique adapted from the work of Taylor [54]. More detail will be discussed in Chapter 3. As illustrated in Figure 2.2c and Figure 2.3, the scanner consists of a compact camera (a 3.2-megapixel PointGrey Flea3) and projector (a 0.4-megapixel TI DLP LightCrafter), mounted at an angle of approximately 20° to each other. Both devices are compact enough to be attached to the positioning system, and the center of the rig is attached to the rotational axis of the position-

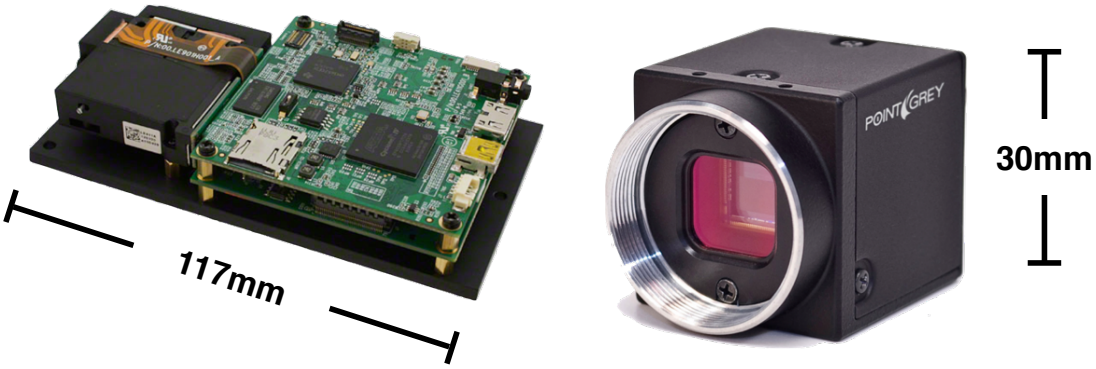


Figure 2.3: Hardware devices of the compact structured-light scanner with dimensions.

ing system. Details of the scanner implementation and calibration are discussed in Chapter 3.

Registration and integration. We adopt standard techniques to register and integrate the scanned data into a complete 3D surface model. We perform ICP [47] to align multiple scans of a single object, with the initial poses provided by the calibrated positioning system. The aligned meshes are merged into a single complete model using VRIP [14] and screened Poisson surface reconstruction [30].

2.3 Scene Exploration

We perform a scene exploration step to obtain an approximate model of the scene with proxy geometry for all objects to be scanned. Our system generates a set of candidate scanner views based on this rough geometry, and passes both the rough geometry and candidate views to the view planner.

Approximate object models. The objects are placed on the scanning platform, which is covered in black cloth for ease of object segmentation. Four static, calibrated webcams positioned around the platform capture images of the scene from above.

The webcam poses are calibrated using the patterns in Figure 2.10, which will be described in Section 2.4. We run background subtraction to segment the objects from the captured images — Figure 2.4 shows the object masks. Silhouette carving

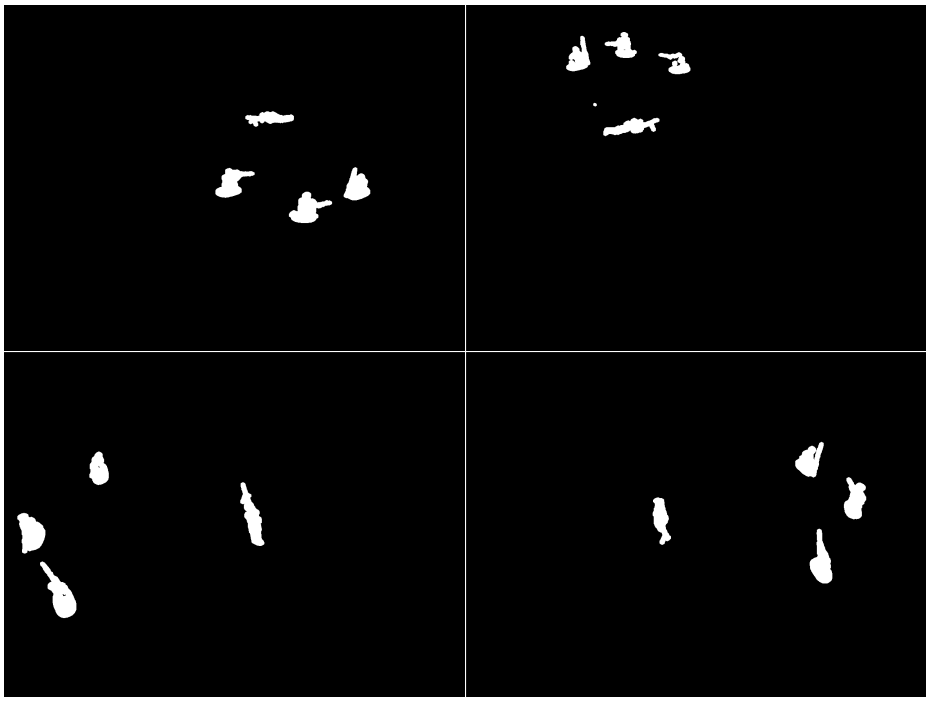


Figure 2.4: Four binary masks of the four toy soldiers scene obtained from the webcams for silhouette carving.

is performed on these masks, and the carved volume surfaces are triangulated into meshes, where the (user-specified) n largest connected components are detected as the approximate models, as illustrated in Figure 2.5. We use a $200 \times 200 \times 200$ voxel grid and have not observed any view planning problems from missing data, but it is possible to expose the grid size as a parameter to handle objects with finer detail such as thin protrusions.

For flat objects we simplify the carving process by extracting 2D contours and extruding them upwards by a user-specified height to approximate the 3D shapes, as shown in Figure 2.6. The 2D contours are extracted from the masks and simplified using a dynamic programming based polygon approximation algorithm [16]. In our

experiments, view planning has never been sensitive to variations in the thickness to which we extrude the contours.

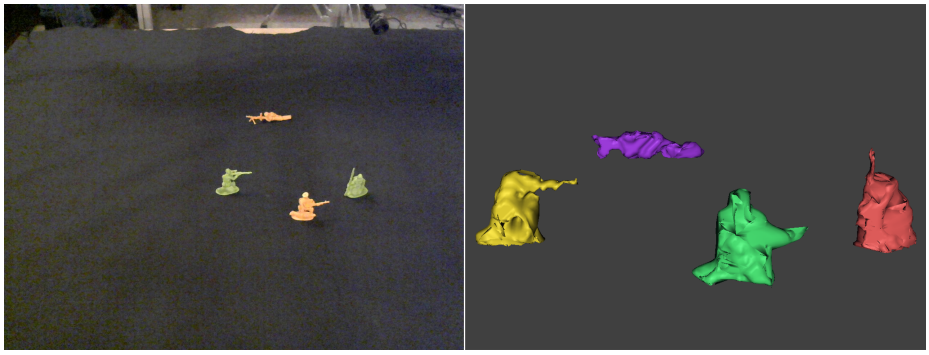


Figure 2.5: A scene with four toy soldiers (left) and the approximate models obtained via silhouette carving (right).

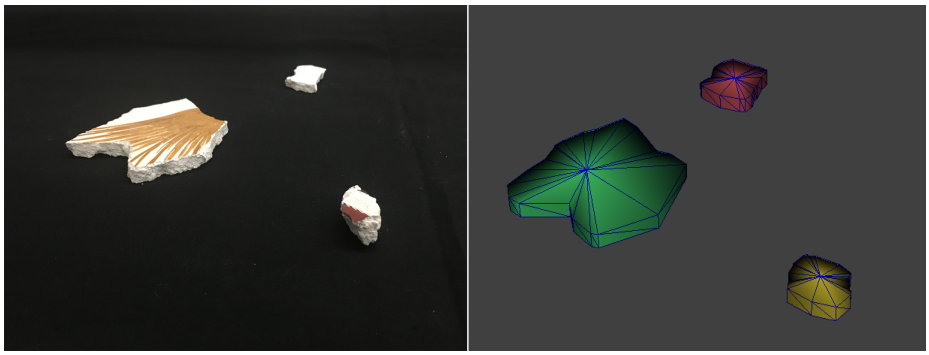


Figure 2.6: A scene with three flat objects (left) and the approximate models obtained via extruding the 2D contours, where mesh triangle edges are shown to better present the model shape (right).

We believe that depth sensors may also provide a solution for obtaining rough 3D models, and in some situations they may work better than silhouette carving. For small objects, however, we observe that the resolution of currently available depth sensors, such as Kinect, is inadequate to improve upon the models produced by silhouette carving.

Candidate scanner views. Our view planning approach (discussed in Chapter 4) is based upon selecting a subset of available views that provide sufficient coverage of the 3D surface of an object. To generate these candidate views, we first fit an

elliptical cylinder to the approximate object model, then dilate the ellipse by several different amounts corresponding approximately to the scanner's "standoff" (i.e., the distance between the camera position and points ranging from the front to the back of the scanner's working volume). The candidate scanner positions are obtained by uniformly sampling angles on the ellipses. At each potential scanner position, we consider a number of scanner orientations centered around the direction facing the middle of the object. Figure 2.7 shows an example of candidate views computed for four objects. The view planner selects a small subset of these candidate views that provides both complete coverage of the object and enough overlap between views to support scan registration.

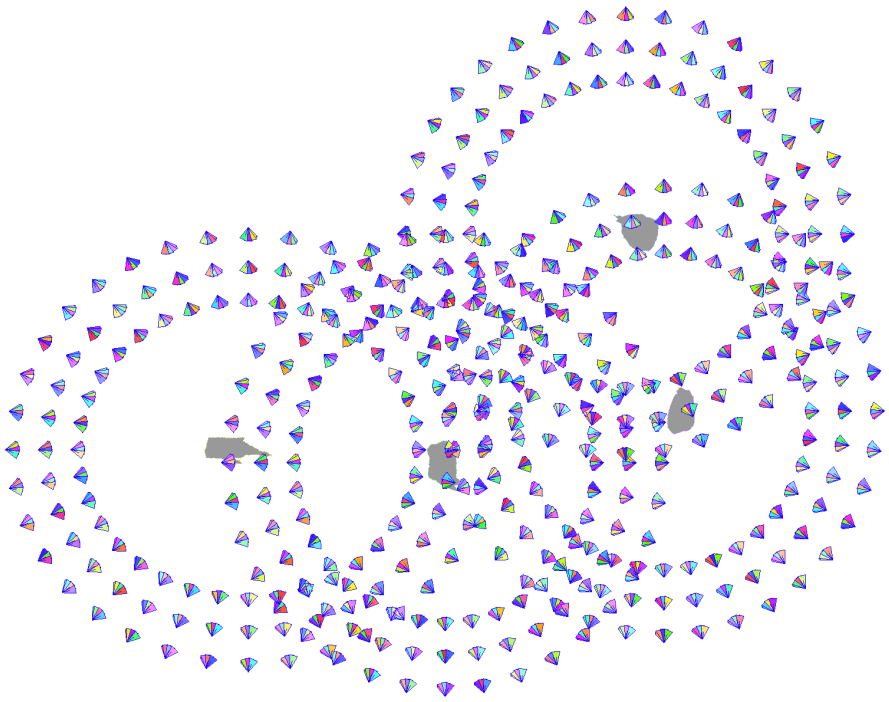


Figure 2.7: An top view visualization of candidate views computed around four objects.

In our current setup, the user-adjustable parameters for candidate view sampling are the radii of the ellipses around which we select views, the different heights of the scanner (constant for small objects), the angular density of views around each object,

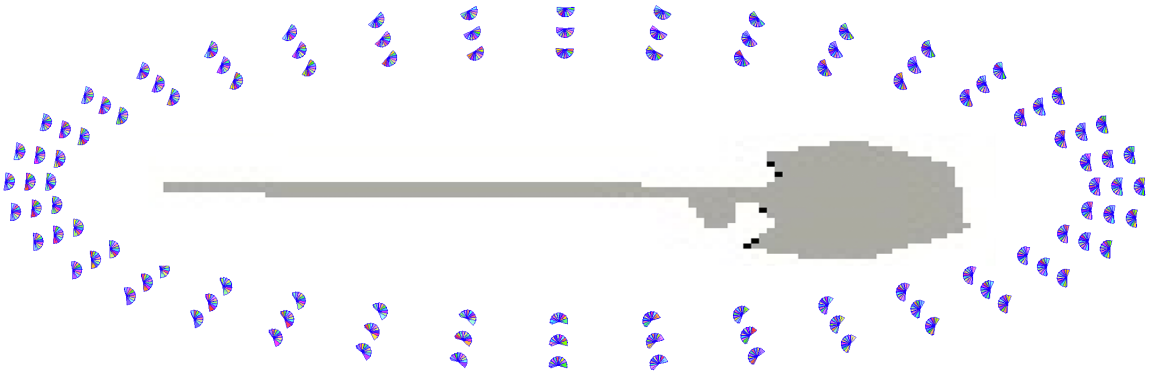


Figure 2.8: An top view visualization of candidate views adaptively computed around a long and thin objects.

and the maximum angular deviation of the scanner from each object center. The angular deviation can be increased when the object shape is extreme, e.g. the long thin geometry as shown in Figure 2.8 and 5.9. Table 2.1 shows the parameters used in our experiments.

Table 2.1: Default setting for the user-adjustable candidate view sampling parameters.

parameter	default setting
ellipse radii	10 to 20 cm plus the object bounding box diagonal radius
scanner heights	1.5 to 3 multiples of the object bounding box height
angular density	10°
angular deviation	$\pm 20^\circ$

2.4 Positioning in the "World"

We propose a novel positioning system that is designed to support efficient 3D acquisition of multiple objects. Motion of the system is calibrated so that the scanner is able to arrive at desired poses based on the view planning results.

Motion ability As shown in Figure 2.9, the positioning system consists of three linear axes (two from a motorized linear translation stage and one from a manual lift table) orthogonal to each other, and a (motorized) rotational axis orthogonal to the plane defined by the two motorized linear axes. The scanner is attached to the rotational axis. The automatic system is driven by three stepper motors: two for translation with a step size of 0.05 mm and one for rotation with a step size of 0.9°. Speed and acceleration of the motors is controlled by an Arduino-based microcontroller.

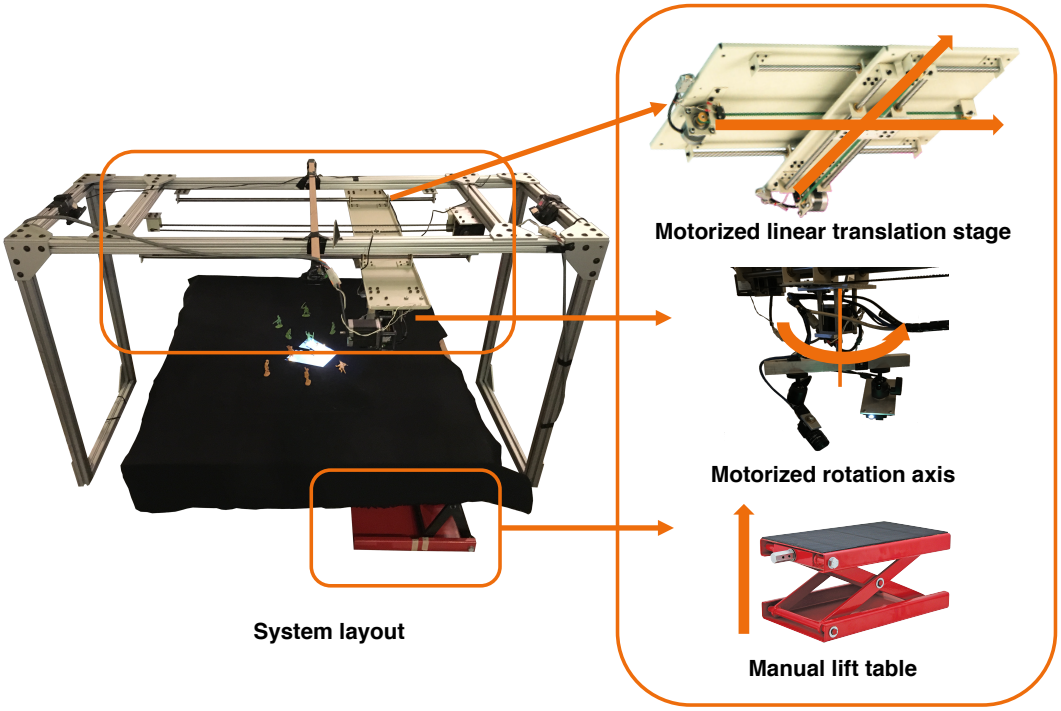


Figure 2.9: Actuators in our positioning system.

Global coordinates The scene exploration, view planning, and scanning need to happen in a unified coordinate system, so that the positioning system is able to accurately position the scanner to reach the poses specified by view planning, and scanned data from different views can be registered and integrated into a complete model. A global coordinate system is defined by employing the AprilTags fiducial system [40], where each tag is a unique 2D bar code. Our calibration pattern uses 256 tags arranged in a 16×16 2D array and glued onto the scanning platform, as shown in Figure 2.10. This pattern is used to calibrate all the sensors employed in our acquisition system, including the four fixed RGB cameras used for scene exploration and the camera in the structured light rig for scanning. Camera extrinsic parameters are estimated by taking a picture of the calibration pattern and detecting the unique tags with their poses known in the global coordinates.

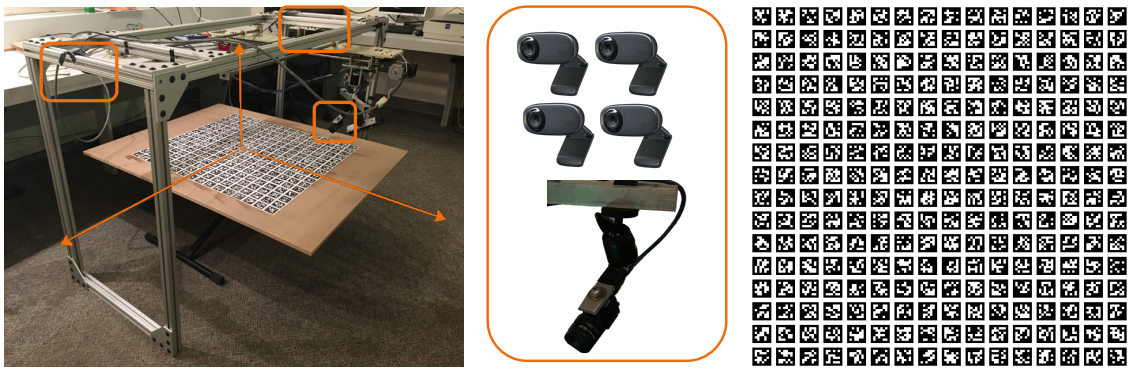


Figure 2.10: The positioning system, with a calibration target on the scanning platform.

Motion calibration The motor controller receives commands in the form of (x^s, y^s, θ_z^s) triplets, but these need not correspond to the global coordinate system ("the world") defined by the AprilTags. To calibrate the motor coordinates, we employ an interpolation-based strategy. During the calibration phase, the positioning system moves the scanner to a set of sparse samples in (x^s, y^s, θ_z^s) space, and

at each stop the scanner captures an image of the calibration pattern to compute its corresponding pose in the global coordinates. Our calibration process is able to

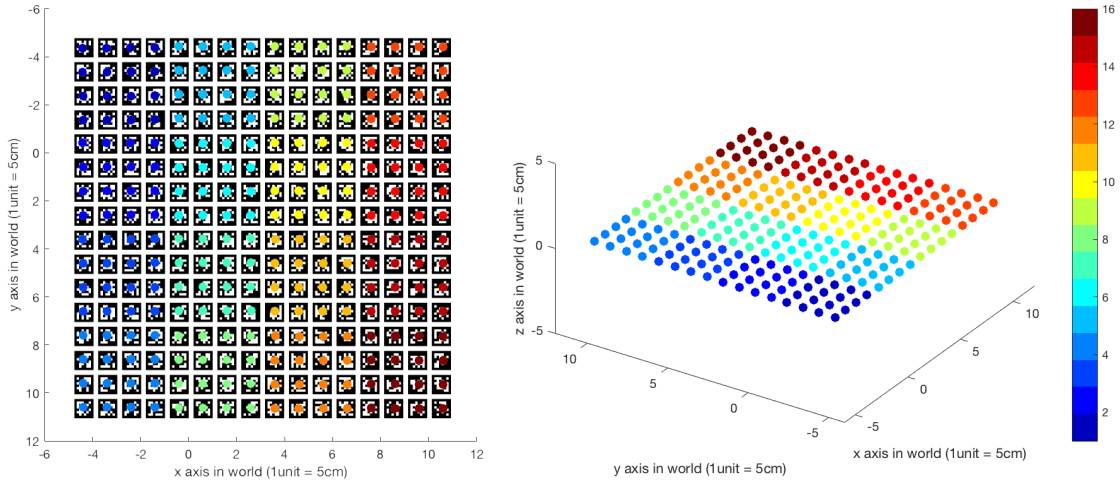


Figure 2.11: The AprilTag pattern overlaid with calibrated centers of the tags (left) and the centers reprojected into the world (right). Tag ids are color-coded from blue to red.

capture the subtle variations across the scanning platform. Ideally, the top of the scanning platform should be a plane and the height of the scanner to the plane is fixed. In practice, however, this is not the case. This is due to the deformation of the wooden plate serving as the platform, the flex of the linear rails along which the axes move, as well as the fact that the rotational axis of the scanner is not perfectly perpendicular to the platform. Figure 2.12 shows a closeup view of the calibrated AprilTag centers in the global coordinates.

To compensate for such variations, a quadratic, rather than linear, model is chosen to fit to the sampled data, to interpolate the transform from any desired pose in global coordinates to a motor command triplet. After calibration, the positioning system achieves 0.5 cm accuracy over approximately a 1 m $\times$ 1 m area. Figure 2.13 shows the accuracy provided by our initial calibration, and the good final alignment achieved with automatic registration beginning with those poses.

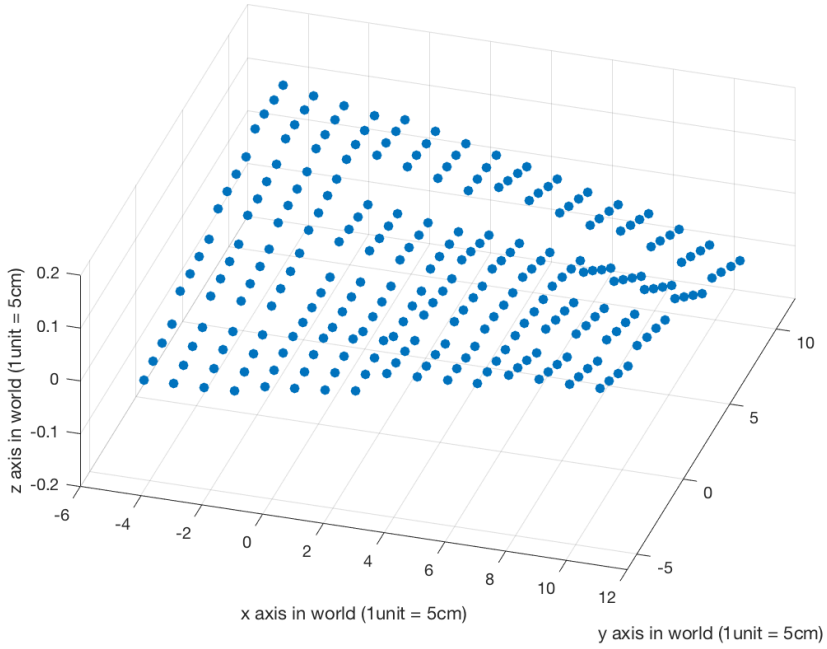


Figure 2.12: Zoomed in view of the calibrated tag centers reprojected in the world, showing the variation across the scanning platform.

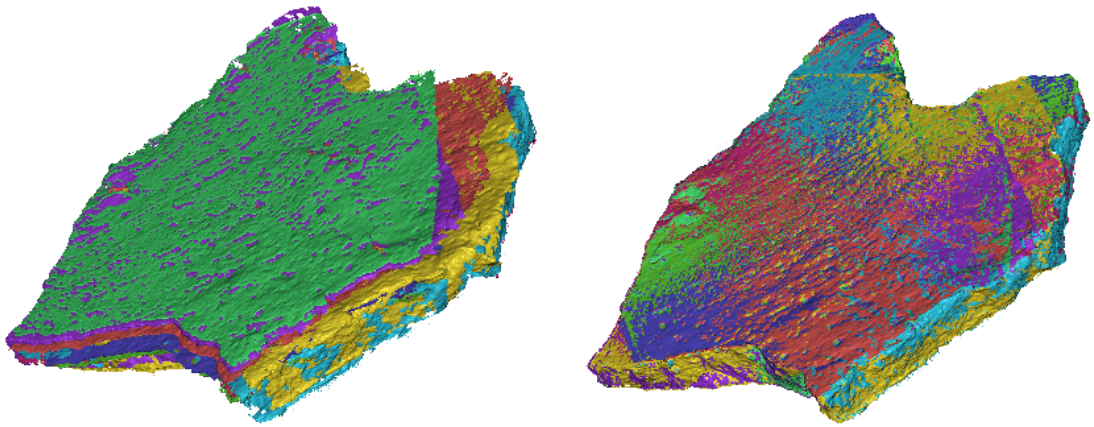


Figure 2.13: The initial scan poses provided by the scanner calibration (left), together with the final result of registration (right).

Chapter 3

Scanning

An experiment is a question which science poses to Nature, and a measurement is the recording of Nature's answer.

Max Planck

In this section, we first briefly review the taxonomy of 3D scanning techniques, with a focus on the triangulation based methods. Considering both accuracy and efficiency, we implement a structured-light scanner in our prototype 3D acquisition system as the main capture device.

3.1 3D Scanning Techniques

3D scanning is the process of measuring the 3D shape of real-world objects. Techniques developed for 3D scanning can be broadly categorized into two major types - contact and non-contact methods.

Contact scanning devices, for instance, a coordinate measuring machine (CMM) [26], probe the object's surface through physical touch. This can be very

efficient in some cases, and is widely used in applications like industrial inspection. However, the physical touch inherently limits measuring efficiency and the variety of objects that can be probed.

Non-contact methods, on the other hand, offers more flexibility. The non-contact family of 3D scanning techniques can be further divided into active and passive methods, depending on whether or not there is emission of light or radiation to "probe" the object surface by detecting reflection. Passive methods examines the 3D shape from only by detecting reflected ambient radiation (e.g., color images), and therefore can usually be fast and cheap. Much research has been conducted to estimate the object's shape from silhouette [37, 38] or stereo [19, 51]. In addition to utilizing a regular camera, the potential of light field imaging systems has also been studied for depth recovery [4, 39, 52] through stereo and refocus.

Active scanning methods typically perform better compared to passive methods, especially in the absence of textures. By controlling the illumination, they are likely to yield more accurate and densely-sampled data from the shape. These includes techniques based on depth from defocus, time of flight [25], photometric stereo [44], and projected-light triangulation [42, 41, 15]. Various work has digitized objects at fine resolution using active 3D scanning techniques [6]. In our context, we especially focus on triangulation based methods.

Laser stripe triangulation has been widely used in previous work to acquire the 3D shape of archaeological artifacts of different size. Levoy et al. [35] built a system which employs laser triangulation rangefinders to digitize large statues by Michelangelo. Brown et al. [9] proposed a 3D model acquisition system for large numbers of fresco fragments with a laser scanner. In both works, laser scanners provide sub-millimeter resolution, but the data quality is proportional to scanning

time, since slower sweeping of the laser stripe across the surface leads to higher-density acquisition.

Structured-light triangulation accelerates the scanning process by projecting a set of temporally-coded patterns onto the object and returning a full range image at a time, as opposed to a single stripe of 3D data from a laser scanner. Various structured-light based acquisition systems have been designed to obtain high-quality 3D geometry data. Bernardini et al. [7] used a lower-resolution structured light system coupled with photometric stereo to digitize Michelangelo's Florentine Pietà. Structured light scanning systems can be fast enough to achieve real-time 3D model acquisition [45, 60]. Data resolution can be enhanced by optimizing pattern design [48], and by combining fine details obtained from normal maps [5].

3.2 Implementing A Structured-Light Scanner

In our prototype system, a structured-light scanner is adopted to capture the depth data, as aforementioned in the system pipeline description in Section 2.2. The structure-light based scanning technique, in its simplest form, uses a projector to illuminate an object or scene with a sequence of time varying patterns, and a camera to capture a corresponding sequence images under these varying illuminations. Correspondences are established between points in both the projector frame and the camera frame by decoding the resulting sequence of images.

3.2.1 Illumination Pattern Design

We adopt a sequence of patterns as a combination of Gray code and phase shift encoding. More specifically, our pattern sequence consists of 10 Gray code column patterns, 10 Gray code row patterns, 8 phase shifting column patterns and 8 phase

shifting row patterns, plus an all black and an all white pattern for subtracting background illumination.

It is common to use a sequence of binary code based stripe patterns to identify the integer coordinates of each illuminated pixel in the scene. Such patterns only have two levels of illumination, which are coded as 0 (dark) and 1 (bright). To reduce decoding error, we use Gray code to encode the stripe patterns, since the Hamming distance between any adjacent pair of Gray code words stays 1. The stripes are implemented both vertically and horizontally. Figure 3.1 shows the stacked 10 Gray code column patterns used in our system.

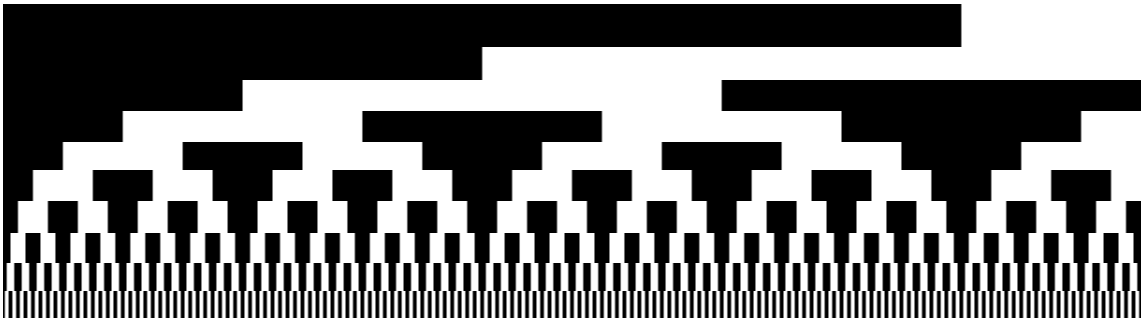


Figure 3.1: Stacked gray code column stripe patterns to split the illuminated space into small regions.

The discrete nature of binary code based patterns limits the spatial accuracy, bounded by the projector's resolution, which is usually lower than that of a digital camera. In our system, the projector has a 0.4-megapixel resolution, whereas the camera has a 3.2-megapixel resolution. To achieve accuracy at sub-pixel level, continuous patterns, such as phase shifting sinusoidal patterns, are commonly adopted together with the binary patterns, since the binary patterns eliminates the ambiguity caused by the periodic sinusoidal patterns, while the phase shifting patterns improve the spatial resolution. As the binary stripes divide the space in regions, the phase shifting patterns are essentially projected light with a sine wave coded intensity that

can interpolate between adjacent light planes. Figure 3.2 shows the stacked 8 phase shifting column patterns used in our system.

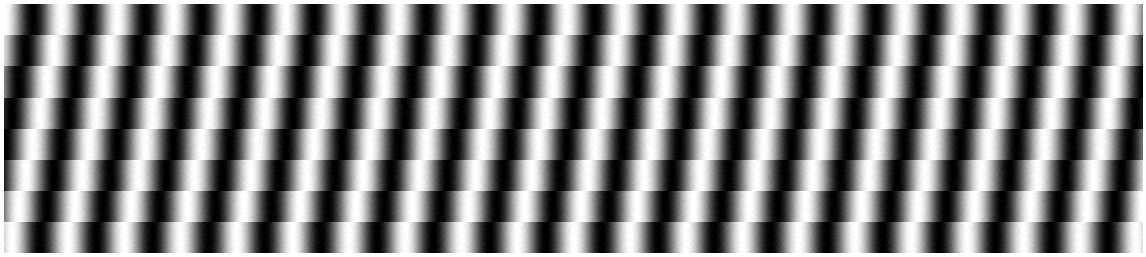


Figure 3.2: Stacked phase shifting column patterns that simulates a sine wave and can interpolate between adjacent light planes defined by the binary stripe patterns.

As shown in Figure 2.3(left), we choose the TI DLP LightCrafter projector because it's compact and fast. The "pixel frame" of the projector is in fact a Digital Micromirror Device (DMD) [27] with 1039680 mirrors, arranged in 912 columns by 1140 rows with the diamond pixel array geometry and configuration, as illustrated in Figure 3.3.

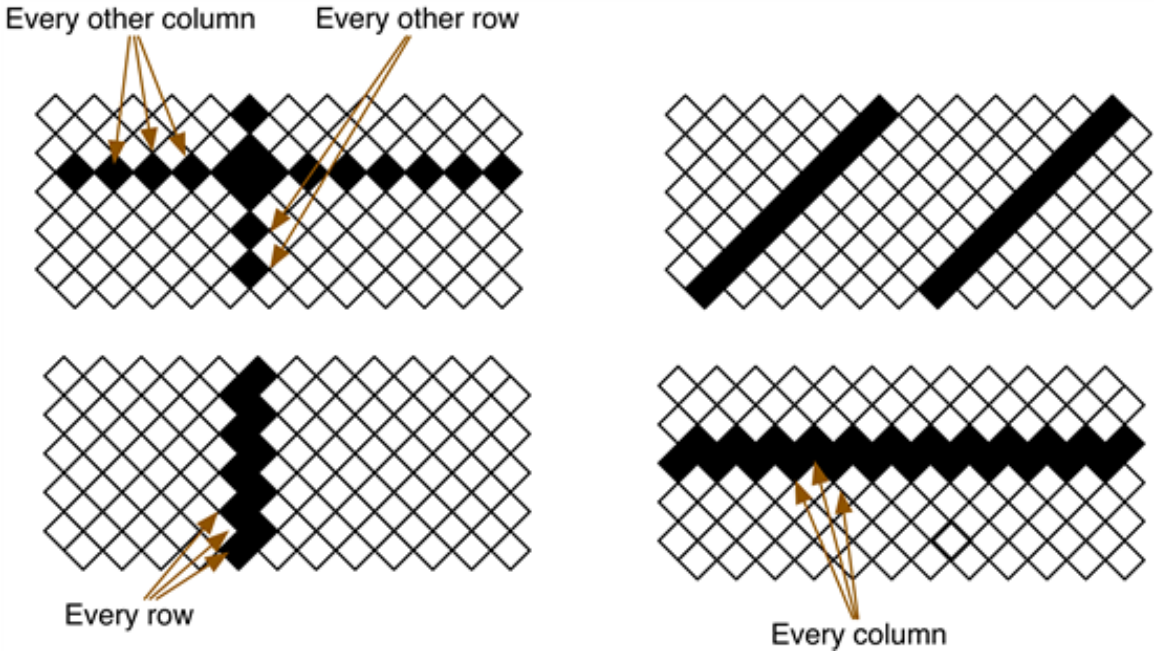


Figure 3.3: Diamond pixel for vertical, horizontal, and diagonal lines arranged in the projector's light engine. `http://www.ti.com/lit/ug/dlpu011f/dlpu011f.pdf`

To fully leverage resolution of the projector, we adapt our illumination patterns to align with diagonal pixel arrays by rotating the stripes for 45°. Figure 3.4 gives an example of how the vertical stripe patterns look like after the rotation. The horizontal stripes are also similarly rotated.

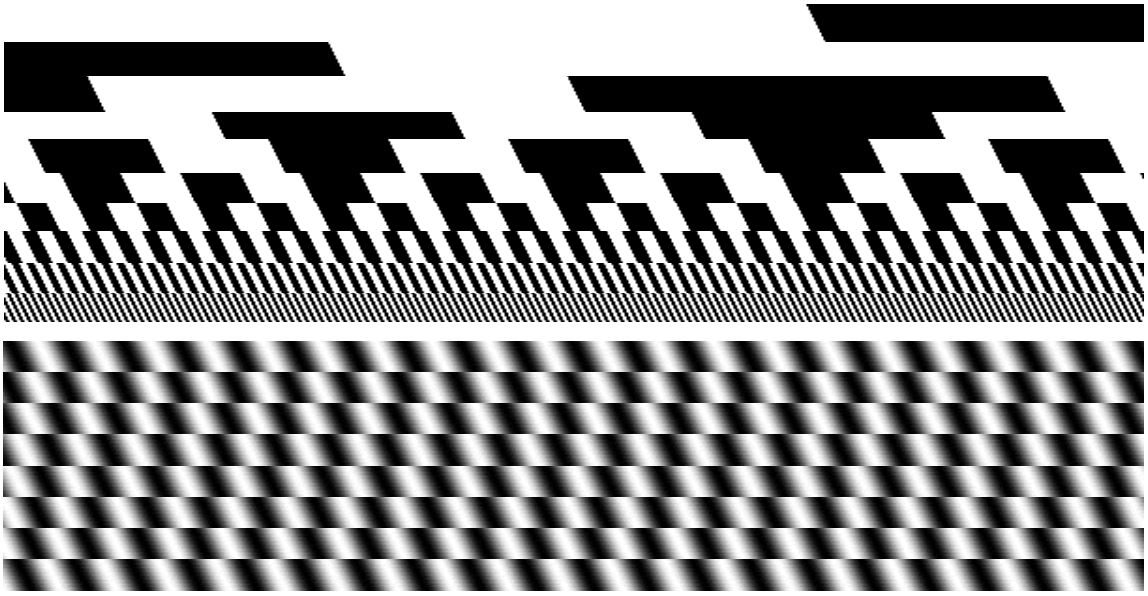


Figure 3.4: Both the gray code patterns (above) and the phase shift patterns (below) are rotated for 45° to align with the mirror arranging direction in the projector.

3.2.2 Scanner Calibration

A checkerboard calibration pattern, as shown in Figure 3.5, is utilized to calibrate the intrinsic parameters of the camera and the projector, as well as the relative pose between them. During the calibration process, the checkerboard pattern plane is posed in different angles and distance relative to the scanner view, and at each pose, the entire 38 frames of illumination patterns are sequentially projected onto the plane and captured by the camera. The set of fully illuminated (using the all white pattern) images of the checkerboard pattern at each pose are processed first to detect corners in each frame. Standard camera calibration techniques [56, 64] are applied to estimate the camera intrinsics. By decoding the structured-light patterns, correspondences are

constructed between the camera and the projector. A camera model is then iteratively fitted to the projector using a RANSAC [21] based optimization.

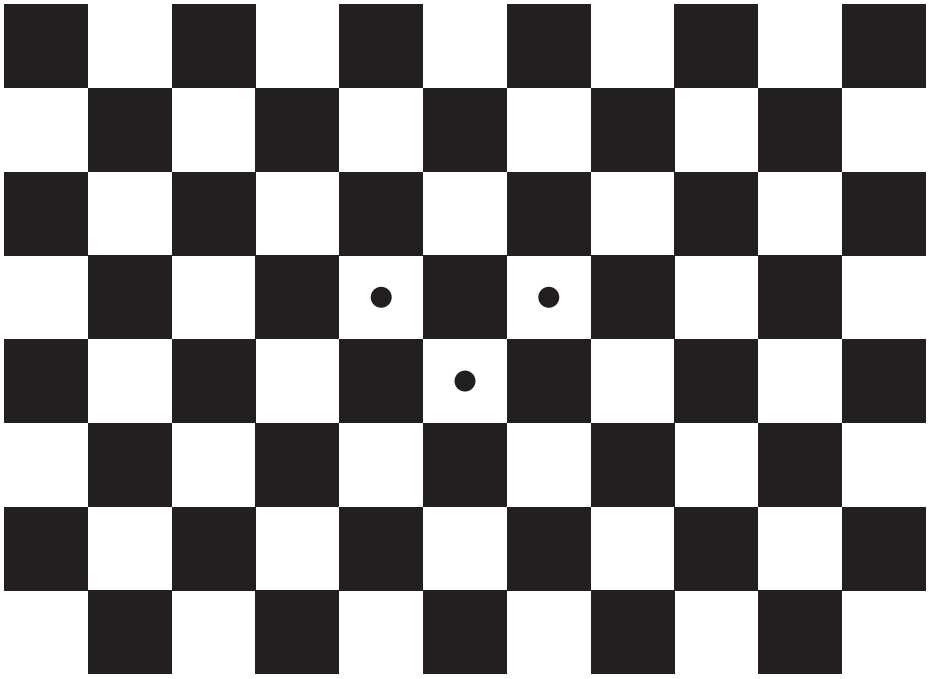


Figure 3.5: The checkerboard pattern used for camera and projector calibration, where the three dots define the center and orientation of the pattern.

Chapter 4

Planning

"It is a narrow mind which cannot look at a subject from various points of view."

George Eliot

The goal of planning in our context is to automatically find a suitable sequence of views, from which the scanner can efficiently acquire the complete surface of many objects with high quality. An abundance of literature can be found that studies the selection of views for one application or another, but few of them can be trivially adapted to achieve the planning goal in our scenario. We propose a general purpose quality measurement as an objective function which works in arbitrary setting of view and path planning tasks for multi-object 3D acquisition, taking into account both *efficiency* of the system operating and *accuracy* of the acquired 3D model. We study different approaches to optimize for this objective function.

4.1 Related Work

A variety of research has addressed the problem of view planning for 3D reconstruction and inspection [49]. Existing approaches can be categorized as model-based or non-

model-based. Model-based methods can be divided into subcategories according to different techniques for model representation, including visibility matrices [53, 50], aspect graphs [53, 8], and "art gallery" floor plans [57]. Solutions can be found in the research field of set theory, graph theory, and computational geometry. Similar work on optimal camera placement from the field of distributed sensor networks [23, 65] can also be adapted to solve view planning for 3D acquisition. However, none of these methods incorporate explicit quality goals for the reconstructed object model, nor do they consider view-overlap constraints for registration. Recent work by Xu et al. [62] proposed an automatic object-in-scene scanning system, but it requires physically moving the objects using the robot. This is incompatible with our goal of safe, contact-less scanning of valuable objects. Incorporating general robotic systems into 3D acquisition [10, 32] is an interesting topic, but it would be difficult to guarantee safety for the objects being scanned.

Non-model-based view planning is also known as the Next-Best-View (NBV) problem. It seeks to find the viewpoint that provides the greatest expected reduction in uncertainty about the object being scanned [49]. Most of these methods assume no a priori knowledge about the object, and plan each view iteratively based on the acquired data. Wu et al. [61] presented a Poisson-guided autonomous scanning method and demonstrated high-quality reconstruction. However, their method is quality-driven and does not aim to minimize the number of scans needed to cover the object's surface; it will therefore require a large number of views to scan multiple objects, with correspondingly long acquisition times.

Another related technique is based on viewpoint entropy [58]. It is generally concerned with selecting informative views, which is a little different from our need of complete coverage. However, among the choices for views that give full coverage, those that contain maximum entropy in the overlapping areas tend to align and merge most robustly.

Research on view and path planning can also be found in robotics [59, 11, 17], where the trajectory of agents is designed based on simplified, abstract models that capture environment features.

4.2 Objective Setup

We start by formulating the view planning problem in a model-based approach. Given a set of view configurations and certain information of the objects surface, we propose an objective function that measures the goodness of the configuration, taking into consideration both accuracy and efficiency at the same time:

$$\mathcal{E} = \mathcal{Q} - \mathcal{C} \tag{4.1}$$

$\mathcal{Q}$ is the view quality term representing how well the objects surface are viewed, and $\mathcal{C}$ is the view cost term representing the time consumption introduced by the given set of views. The view cost can be further divided into two parts: the total time $\mathcal{S}$ needed for the scanner to scan at each view, and the total time $\mathcal{L}$ needed for the scanner to travel (by the positioning system) among all the views.

$$\mathcal{C} = \mathcal{S} + \mathcal{L} \tag{4.2}$$

In order to make the view quality term and the view cost term comparable, it would be nice to measure $\mathcal{Q}$ in unit of time as well. We multiply $\mathcal{Q}$ by a constant ω, which indicates the "amount of time increase" by scanning a unit-area surface with adequate quality. The physical meaning of the new view quality term $\omega \cdot \mathcal{Q}$ then becomes the amount of time needed to achieve certain scan accuracy on the entire surface area considered.

As a result, we re-write the objective function as:

$$\mathcal{E} = \omega \cdot \mathcal{Q} - \mathcal{S} - \mathcal{L} \tag{4.3}$$

The remainder of this section describes how we set up the view quality term $\mathcal{Q}$ - we first define a per-point view quality score, and then integrate it over multiple views and many surface samples to measure the quality of a complete set of views.

4.2.1 Search Space Discretization

The objective function $\mathcal{E}$ is evaluated based on the objects' surface geometry and a set of views selected from certain view space. We use a set of oriented point samples P to model the object surface geometry, and discretize the view space into a set of candidate views V. The objective $\mathcal{E}$ in Equation 4.1 then becomes a function of P and V^*,

$$\begin{aligned} \mathcal{E}(P, V^*) &= \omega \cdot \mathcal{Q}(P, V^*) - \mathcal{S}(V^*) - \mathcal{L}(V^*) \\ &= \omega \cdot \mathcal{Q}(P, V^*) - \mathcal{s} \cdot |V^*| - \mathcal{L}(V^*) \end{aligned} \tag{4.4}$$

where $V^* \subset V$ is a selected set of views, and $\mathcal{s}$ denotes the average scanning time spent at a single view in V^*

Sampling surface geometry. The model-based view planning assumes some knowledge about the objects to be scanned. Usually, the "model" that the view planning bases on is an approximation of the final output from the 3D acquisition pipeline. For example, it can be obtained as an approximate surface model from the scene exploration process described in Section 2.3. Notice that the final output is essentially a surface interpolated from a dense set of samples obtained by a 3D scanner. Since the approximate model does not capture sufficient details on the

object surface geometry, it is reasonable to downsample the approximate surface. We uniformly sample oriented points from the approximate surface, such that each sampled point represents a surface patch with unit area. In the view planning, these point samples are used as a proxy of the object surface geometry to compute the view quality measurement $\mathcal{Q}$. A surface is represented as a triangular mesh in our context. To uniformly sample the surface by area, each triangular face is selected with a probability proportional to its area, and then, a point within this triangle is uniformly sampled at random, as shown in Figure 4.1(left).

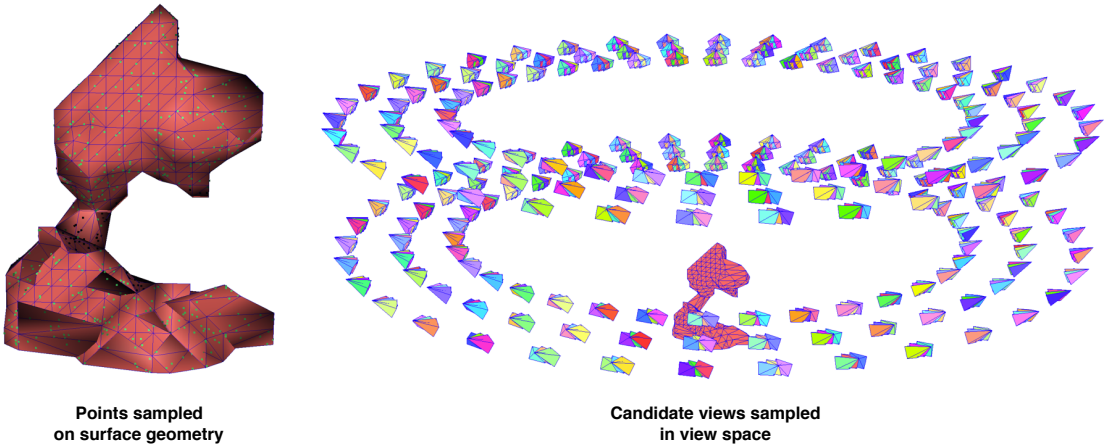


Figure 4.1: The objective function in Equation 4.4 examines surface point samples (left) and views selected from a given candidate set (right) to measure the goodness of a configuration.

Sampling the view space. In practice, the positioning system that moves the scanner to each selected view is limited by the precision of its motors and gears; it is therefore reasonable to examine the space of scanner candidate views as a discrete space. Different approaches can be utilized to discretize the candidate view space, as long as it provide sufficient coverage of the object surface to be scanned. One might imagine to use a 3D voxel grid to represent space of where each view can locate, and sample the viewing direction on a sphere at each location. A more efficient way of sampling is to constrain the candidate views by each object to be scanned, as

described in Section 2.3. Figure 4.1(right) gives an illustration on how the candidate views are constrained by the approximate object surface model.

4.2.2 View Quality Metric

Given the oriented point set P and candidate view set V, we begin by defining a view quality function that measures how well one single point $p \in P$ is "seen" by a single scanner view $v \in V$:

$$q(p, v) = \hbar(p, v) \cdot g(p, v), \tag{4.5}$$

where $\hbar(p, v)$ is a visibility term and $g(p, v)$ is a geometry term. Each point p is defined by 6 parameters $[x, y, z, n_x, n_y, n_z]$, where x, y, and z specify its location, and n_x, n_y, and n_z specify the direction in which the surface patch around p is oriented. Note that a scanner view v is defined by all its constituent optical devices, and those devices can be either sensors (e.g., cameras) or lighting devices (e.g., projectors). The prototype we built in Chapter 2 adopts a one-camera, one-projector structured-light configuration, but the metric developed in this work applies generally to any multi-camera, multi-projector setup. Each optical device at a scanner view is defined by its intrinsic and extrinsic parameters.

Visibility term. A point is *visible* to a device view if the point is within the field of view of that device and the point is not occluded by other parts of the surface model. We define the visibility term as a binary function

$$\hbar(p, v) = \begin{cases} 1 & \text{if } p \text{ is } \textit{visible} \text{ to all device views at } v, \\ 0 & \text{otherwise.} \end{cases} \tag{4.6}$$

The field of view of each optical device can be obtained through calibration process, such as the one described in Section 3.2.2. Occlusion is checked by performing efficient ray-mesh intersection [46].

Geometry term. For points that are visible to a scanner view, we define the geometry term to quantify how well each point is "seen" from the view:

$$g(p, v) = \max\{0, \min\{\vec{c}_v^{(1)} \cdot \vec{n}_p, \vec{c}_v^{(2)} \cdot \vec{n}_p, ..., \vec{c}_v^{(K)} \cdot \vec{n}_p\}\}, \tag{4.7}$$

where $\vec{n}_p$ is the surface normal vector at point p, K is the total number of optical devices in the scanner setup, and $\vec{c}_v^{(k)}$ is the viewing vector from p to the center of projection of device k. The dot products are clamped at 0 because the surface becomes invisible when the angle between the two vectors is less than 90°. The geometry term ensures that all the optical devices "see" the point frontally.

Integration. Given a candidate scanner view set V and a surface sample set P, we integrate the per-point, per-device view quality scores $q(p, v)$ over V and P to form an overall measurement of the quality of any subset of scanner views from V. For some selected set of scanner views $V^* \subset V$, we therefore define the best view for each point as

$$\beta_1(p) = \arg\max_{v \in V^*} q(p, v). \tag{4.8}$$

The overall view quality measurement can then be written as

$$\mathcal{Q}(P, V^*) = \sum_{p \in P} q(p, \beta_1(p)), \tag{4.9}$$

Note that for each point p we do not take the summation of its view qualities over all views, but instead over only the best one. This will ensure that each point has at least one "good" view, as opposed to a larger number of views with mediocre quality.

4.2.3 Overlap-Aware Heuristics

The basic view quality term in Equation 4.9 encourages full "good view" coverage over all the points. It does not, however, necessarily guarantee *overlap* between scans from adjacent views, which is essential to the subsequent registration step. We therefore propose heuristics to improve the objective function so that it addresses view overlap.

Second-best views. In order to acquire more accurate data and encourage view overlap for registration, we would like each point to be "seen" by the scanner from at least *two* views as opposed to only *one* as indicated in Equation 4.9; and hence for some selected set of views V^* we define the second-best views for each point as

$$\beta_2(p) = \underset{v \in V^* \setminus \{\beta_1(p)\}}{\arg\max} \; \boldsymbol{q}(p, v). \tag{4.10}$$

The view quality term is then re-written as

$$\mathcal{Q}_2(P, V^*) = \sum_{p \in P} \boldsymbol{q}_2(p, \boldsymbol{\beta}(p)), \tag{4.11}$$

where

$$\boldsymbol{q}_2(p, \boldsymbol{\beta}(p)) = (1 - \epsilon) \cdot \boldsymbol{q}(p, \beta_1(p)) + \epsilon \cdot \boldsymbol{q}(p, \beta_2(p)). \tag{4.12}$$

In this equation, $\epsilon \in [0, 1]$ is a user specified weight that defines how much we rely on the quality of the second-best views. Now we ensure that each point has at least two "good" views.

Neighborhood view quality aggregation. To encourage overlap around sharp corners, we measure the view quality of a point more conservatively by evaluating the view quality of all points in its neighborhood. For any point $p \in P$ with its small neighborhood $\mathcal{N}(p) \subset P$, and a given view v, the neighborhood aggregated

view quality is defined as

$$\boldsymbol{q}_N(p, v) = (1 - \tau) \cdot \min_{p' \in \mathcal{N}(p)} \boldsymbol{q}(p', v) + \tau \cdot \frac{1}{|\mathcal{N}(p)|} \sum_{p' \in \mathcal{N}(p)} \boldsymbol{q}(p', v). \tag{4.13}$$

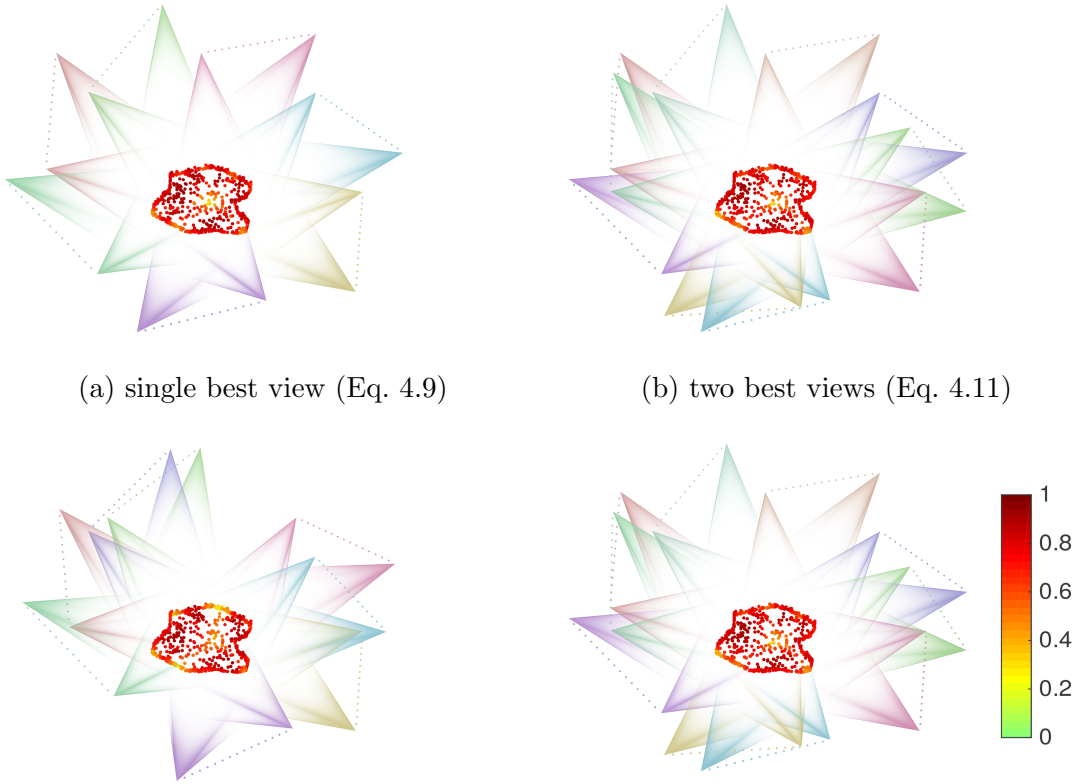
Plugging this into Equations 4.11 and 4.12, we obtain a new view quality measurement

$$\mathcal{Q}_{2,N}(P, V^*) = \sum_{p \in P} \boldsymbol{q}_{2,N}(p, \boldsymbol{\beta}(p)), \tag{4.14}$$

where

$$\boldsymbol{q}_{2,N}(p, \boldsymbol{\beta}(p)) = (1 - \epsilon) \cdot \boldsymbol{q}_N(p, \beta_1(p)) + \epsilon \cdot \boldsymbol{q}_N(p, \beta_2(p)). \tag{4.15}$$

Figure 4.2 shows an example that visualizes view quality without and with different heuristics. We maximize the objective function in Equation 4.4 by plugging in different view quality terms, using the optimization described in Section 4.3. With only a single best view considered and no neighborhood heuristic (Equation 4.9), all the points have very good view quality, but the view assignment is not addressing overlaps. When the second best view is also considered (Equation 4.11), the planning tends to add slightly more views and encourages overlap between some pairs of adjacent views. When neighborhood aggregation is introduced to the objective function, with the same view cost but only a single best view considered (Equation 4.14 with $\epsilon = 0$), it sacrifices per-point view quality in favor of encouraging overlap where it is often difficult to achieve manually, such as around sharp corners. The hybrid objective with both neighborhood aggregation and second-best view (Equation 4.14 with $\epsilon = 0.5$) also gives reasonable results, but usually uses the most views because it is the most conservative. We adopt the single-best-view objective function with neighborhood aggregation (Equation 4.14 with $\epsilon = 0$) in the following experiments.



(a) single best view (Eq. 4.9)

(b) two best views (Eq. 4.11)

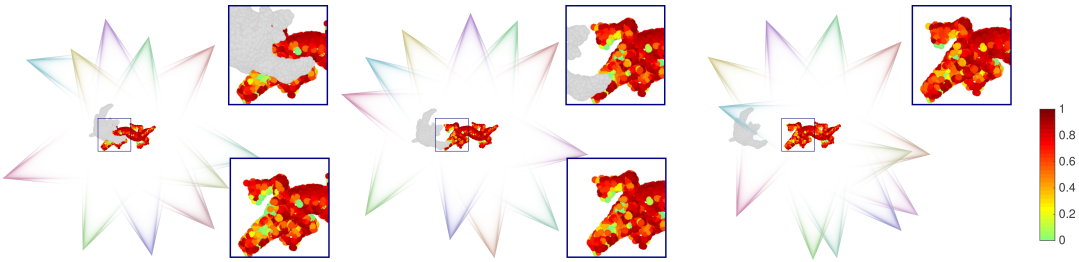
(c) single best view + neighborhood (Eq. 4.14 with $\epsilon = 0$)

(d) two best views + neighborhood (Eq. 4.14 with $\epsilon = 0.5$)

Figure 4.2: View assignment quality visualization with different objective functions. Each scanner view is represented by a camera-projector pair of frustums connected with a dotted line. The object model is represented by surface samples with the view quality value encoded according to the jet color map, as shown in (d). Red means good view quality and green means poor.

Multi-object objective. Our view quality metric easily generalizes to the multi-object case. We sum up the objective function over all objects, modifying the visibility term by checking occlusion from all object surfaces in the scene to avoid inter-object occlusion. Figure 4.3 shows an experiment evaluating occlusion detection performance in three different cases. As shown at right, when there is plenty of space between the two objects, the view rendered in light blue is selected to cover most of the region of the dragon head. However, when the armadillo is moved closer, as shown in the middle and left, the originally desired view can no longer see the dragon's head well due to

(a) Models of "dragon fighting armadillo" as the armadillo moves from "close" (left) to "near" (middle), and then "far" (right) from the dragon. The models are synthesized at similar level of quality with the approximate models acquired from the scene exploration.



(b) Visualization of the surface sample view quality based on the corresponding selected views for the three different scenes: "close" (left), "near" (middle), and "far" (right). Red represents good view quality and green poor. The scanner views are simplified by only visualizing corresponding camera views. Zoomed-in views of the dragon head are shown to illustrate the view quality change. For the "close" and "near" scenes, we show closeups of the dragon head both with and without the armadillo occlusion.

Figure 4.3: Given a scene of "dragon fighting armadillo" with increasing distance between the two objects (a), we visualize the surface sample view quality based on the corresponding selected views (b). The close-up views show that, as the armadillo moves from "close" (left) to "near" (middle), and then "far" (right) from the dragon, the view quality of the head of the dragon improves.

occlusion, and therefore the view planner has to select alternate views from further back. As illustrated in the zoomed-in area, the view-quality visualization provides feedback to the user in these cases: a significant amount of green area suggests that flipping or rearranging the objects will be necessary to acquire a complete model. In most cases, however, inter-object occlusion detection ensures proper view selection to avoid occlusion. To better leverage the information from a scene with multiple objects, we discuss...4.4.

4.3 Sequential View and Path Optimization

Given the objective function in Equation 4.4, different strategies can be employed to maximize it. We start with a simple approach, which decompose the objective into two parts, a view selection objective $\mathscr{E}_v$ and a path finding objective $\mathscr{E}_p$, and tries to optimize for them separately:

$$\mathscr{E} = \mathscr{E}_v + \mathscr{E}_p \tag{4.16}$$

where

$$\mathscr{E}_p = -\mathscr{L}(V^*). \tag{4.17}$$

and

$$\begin{aligned} \mathscr{E}_v &= \omega \cdot \mathscr{Q}(P, V^*) - \mathscr{s} \cdot |V^*| \\ &= \omega \cdot \sum_{i=1}^{n} (\mathscr{Q}(P_i, V_i^*) - \gamma \cdot |V_i^*|) \end{aligned} \tag{4.18}$$

n denotes the total number of objects being scanned. For simplicity we use $\gamma = \frac{\mathscr{s}}{\omega}$ to control the ratio between the amount of accuracy and efficiency to be achieved, and optimize the objective $\mathscr{E}_v$ for each object independently. This of course does not guarantee optimal result, but the simplicity of solving each component independently makes the problem more tractable.

4.3.1 Optimizing the View Planning Objective Function

We explore several different approaches to maximize $\mathscr{E}_v$. Given that each point needs to be seen at least once or twice, depending on the choice of objective function, maximizing the objective reduces to the classic NP-complete set-cover and multicover problems [29]. Therefore, we explore a number of ways of approximating the problem in order to find solutions with practical computation time.

Sequential greedy optimization. An intuitive way of optimizing our view selection objective function is using the classic greedy approach. In fact, there are inapproximability results [20] showing that the sequential greedy approach is the best possible polynomial-time approximation algorithm for set cover. In our scenario, we begin with $V^* = \emptyset$ and iteratively add the view that yields the largest increase in the objective function. The constant γ ensures that, at some point, no new view can be found that leads to an increase in the objective function, terminating the algorithm and controlling the number of views we select.

Simulated annealing. The greedy approach is simple to implement and very efficient, but due to its deterministic nature the objective will not improve once a local optimum is achieved. Simulated annealing [31] is a probabilistic method for approximating the global optimum of an objective function that may possess many local optima, at a cost of relatively long running time.

Algorithm 1 details our implementation of simulated annealing for optimizing the objective in Equation 4.14. The algorithm is initialized with random views, and at each iteration updates a state vector $\vec{X} = [X_1, X_2, ..., X_{|V|}]$ consisting of indicator variables representing whether a candidate view is selected or not, such that $V^* = \{v \,|\, X_v = 1, v \in V\}$. While a basic implementation might simply enable or disable a single view at each iteration, we take advantage of the structure of the candidate view space V to improve efficiency. Specifically, with probability one-half we swap some view v for a neighboring view $v' \in \mathcal{N}(v)$, instead of simply switching a view on or off. The energy function $E(\vec{X})$ guiding whether a state transition is accepted is set equal to the objective function $F(P, V^*)$ with V^* defined by $\vec{X}$, and the annealing temperature T decreases exponentially.

As shown below, we find that simulated annealing, if given a slow-enough annealing schedule and enough iterations, typically outperforms the greedy approach.

Algorithm 1 Simulated Annealing for View Planning

Input: random initialization $\vec{X}_0$
  repeat
    draw Pr from uniform (0, 1) distribution
    if $Pr <$ threshold **then**
      randomly select view $v \in V^*$
      randomly select view $v' \in \mathcal{N}(v)$
      $\vec{X}'_t \leftarrow \vec{X}_t$ with X_v and $X_{v'}$ swapped
    else
      randomly select view $v \in V$
      $\vec{X}'_t \leftarrow \vec{X}_t$ with X_v flipped
    end if
    $T_t = \alpha^t$, $\Delta E = E(\vec{X}'_t) - E(\vec{X}_t)$
    if $\Delta E > 0$ **then**
      $\vec{X}_{t+1} \leftarrow \vec{X}'_t$
    else
      with probability $\exp(\Delta E \cdot T_t)$, $\quad \vec{X}_{t+1} \leftarrow \vec{X}_t$
      with probability $1 - \exp(\Delta E \cdot T_t)$, $\quad \vec{X}_{t+1} \leftarrow \vec{X}'_t$
    end if
    $t \leftarrow t + 1$
  until convergence

Moreover, it automatically decides the exact number of views needed in the optimal solution based on the view cost parameter γ.

Integer programming. Another way to approximate the view planning optimization is to formulate it as a binary integer programming problem. In this case, the objective function needs to be quantized based on a view quality threshold η, and thus given a point p and a view v, measuring the view quality becomes simply checking whether it is "good enough", namely above η. Specifically, we define a set of indicator variables W_{pv}, which are 1 if $f(p, v) > \eta$ and 0 otherwise. The objective function $\mathcal{E}_v$ is then approximated by

$$\sum_{p \in P} \min\{W_{pv} \cdot X_v,\ |\boldsymbol{\beta}(p)|\} - \gamma \sum_{v \in V} X_v. \tag{4.19}$$

where $|\boldsymbol{\beta}(p)|$ is the number of best views considered for each point. A branch-and-bound method [24] is applied to solve this integer program exactly.

4.3.2 Discussion on Performance

We evaluate the performance of the three approaches on the same dataset by comparing the optimal objective values they obtain, as shown in Figure 4.4. In each figure, the blue curve shows the evolution of the objective value against the number of views selected by the sequential greedy algorithm, with the ultimate result of the greedy algorithm being the highest point. The red curve shows the objective value achieved by integer programming, with different values of the view quality quantization threshold η. The scattered orange squares are results from 10 different runs of simulated annealing, using different random seeds.

Varying View Cost. The three plots in Figure 4.4 show results for different values of the view cost multiplier γ. We leave the choice up to the user, to select γ to be the desired increase in the average view quality, as one additional view is added. As γ increases, the required benefit of adding a view increases, and hence the optimal number of views decreases.

Algorithm Comparison. The figures show that simulated annealing achieves better objective values compared to the greedy approach and integer programming. With the threshold η properly chosen, the integer programming can perform as well as the greedy approach, but is less predictable, since the number of views and ultimate quality do not vary monotonically with η. While simulated annealing does require more computation (a few minutes per object, where 2000 points are sampled on each objects), we generally prefer it for our system. If this computation time is unacceptable, the greedy algorithm usually picks a near-optimal number of views, though the

views themselves may be sub-optimal. We also provide a clustering strategy to help improve efficiency, which will be discussed in Chapter 7.

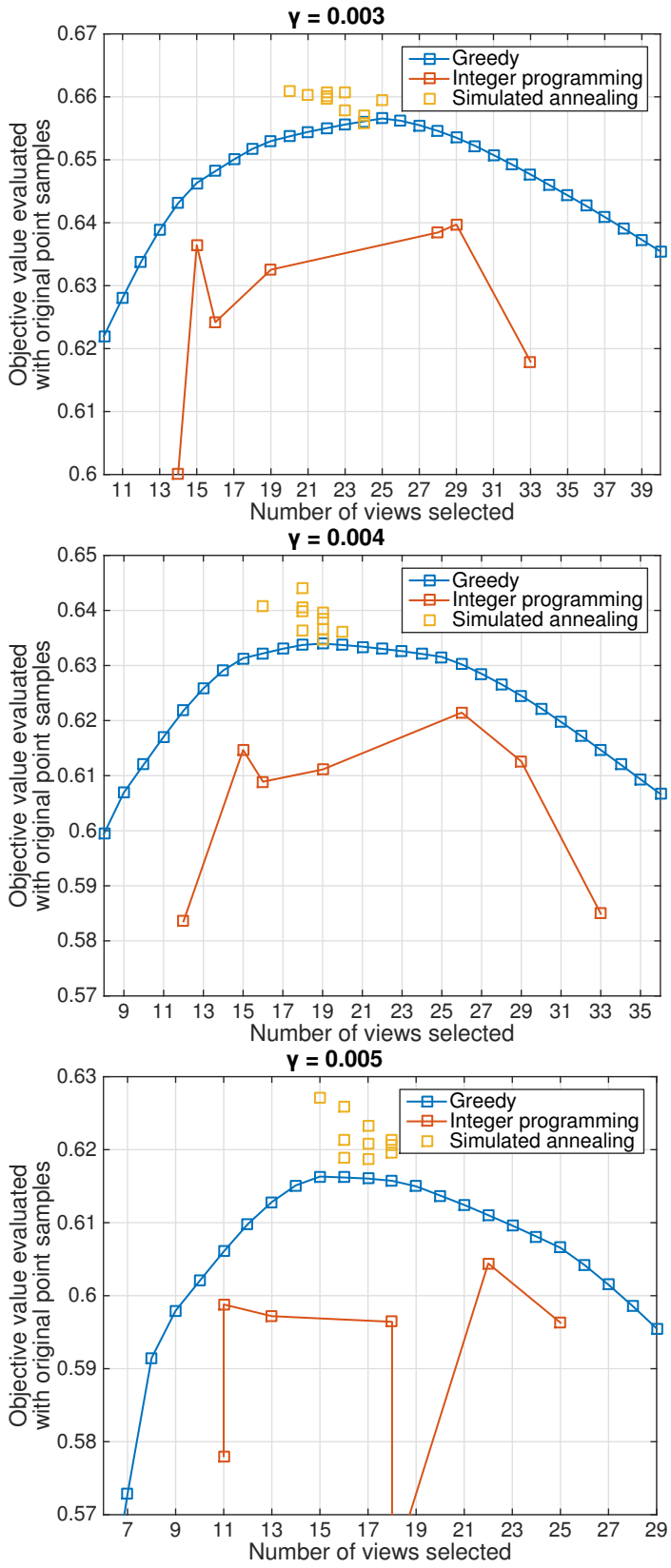


Figure 4.4: The optimal objective value achieved by different approaches with varying γ, with higher value indicating better overall view quality.

4.3.3 Path Finding

Once we have obtained a set of desired scanner positions for each object within the working volume, planning the optimal path among them is naturally formulated as the Traveling Salesman Problem (TSP) [34]. Between any pair of views, we compute a motion cost corresponding to the time taken by the positioning system to move between those views, taking into account that motion along multiple axes can happen simultaneously. We solve the TSP on a complete graph, where each node in the graph corresponds to a scanner pose (x, y, θ_z). For any pair of nodes (x_i, y_i, θ_{zi}) and (x_j, y_j, θ_{zj}), there is an edge between them, and the distance is defined as the travel time

$$\max\left\{\frac{|x_i - x_j|}{v_x}, \frac{|y_i - y_j|}{v_y}, \frac{\min\left\{|\theta_{zi} - \theta_{zj}|, 2\pi - |\theta_{zi} - \theta_{zj}|\right\}}{v_{\theta_z}}\right\},$$

where v_x, v_y, and v_{θ_z} respectively represent the motor speed along the linear axes and around the rotational axis. The problem can be solved by many efficient algorithms, e.g. the Christofides' algorithm [12], the Lin-Kernighan heuristic [36], and some branch-and-bound based method [13], We use the algorithm of Christofides [12] to obtain a $\frac{3}{2}$-approximated optimal path.

4.4 Joint Optimization

In the previous section, we decompose the objective function $\mathcal{E}$ into different parts and optimize for them independently. Such a simple and efficient approach provides a reasonable lower bound for the solution to our planning problem. In this section, we step by step introduce more dependence assumption among different parts in the objective function, and try to search for an optimal solution by jointly examining different parts constituting $\mathcal{E}$.

4.4.1 Joint View Selection for Multiple Objects

The design of our view quality metric $\mathcal{Q}$ takes into account the presence of multiple objects in the same scene, and is able to avoid views hurt by inter-object occlusion, as shown in Figure 4.3. There are views, on the other hand, which can actually benefit from the scanner being posed in a multi-object scene, and potentially see parts from different objects at the same time. To better leverage the multi-object scanning scenario, we study the case where each candidate view can contribute to any object in the scene. This is reflected in the view selection objective function as

$$\begin{aligned} \mathcal{E}_v &= \omega \cdot \mathcal{Q}(P, V^*) - \mathcal{s} \cdot |V^*| \\ &= \omega \cdot \mathcal{Q}(\bigcup_{i=1}^{n} P_i, \bigcup_{i=1}^{n} V_i^*) - \mathcal{s} \cdot \sum_{i=1}^{n} |V_i^*| \end{aligned} \tag{4.20}$$

Recall that previously $\mathcal{E}_v$ is decomposed by object in Equation 4.18:

$$\begin{aligned} \mathcal{E}_v &= \omega \cdot \sum_{i=1}^{n} (\mathcal{Q}(P_i, V_i^*) - \gamma \cdot |V_i^*|) \\ &= \sum_{i=1}^{n} (\omega \cdot \mathcal{Q}(P_i, V_i^*) - \mathcal{s} \cdot |V_i^*|) \end{aligned} \tag{4.21}$$

and it is independently maximized for each object, where $\forall v \in V_i^*$, v is not able to contribute to the view quality for a different object $\mathcal{Q}(P_j, V_j^*), (i \neq j)$.

Based on the study in Section 4.3.2, we choose to employ the simulated annealing based method described in Algorithm 1 to maximize $\mathcal{E}_v$ in Equation 4.20, jointly among all the objects. We compare the performance of this approach on view selection to that described in Section 4.3. Experiments are carried out focusing on an 8-object scene shown in Figure 4.5, starting with a single object, and gradually including more objects (one at a time). 200 oriented points are sampled on each object. For each scene with a given number of objects and a given value for ω, we run the experiment for 10 times using different random seeds for the simulated annealing.

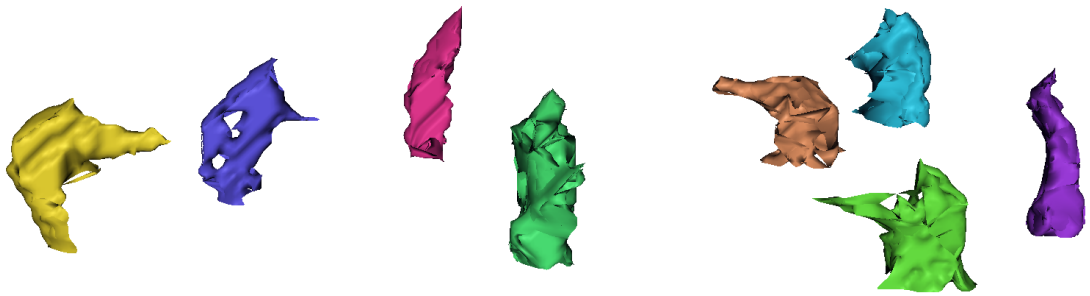


Figure 4.5: The approximate model of the 8 objects used for view planning.

In all the experiments we set $\mathit{s} = 37.5$ seconds, which is the average measured time needed for the scanner in our prototype system (Chapter 2) to scan from a single view. The constant ω is exposed as a user-adjustable parameter, which controls the amount of desired view quality.

We first compare the overall objective $\mathcal{E}_v$ obtained by both approaches. As the number of objects increases, no matter how ω is set, the joint optimization over all the objects gives $\mathcal{E}_v$ a boost, suggesting better solutions compared to the piecewise case, where the view selection objective is optimized over each object independently.

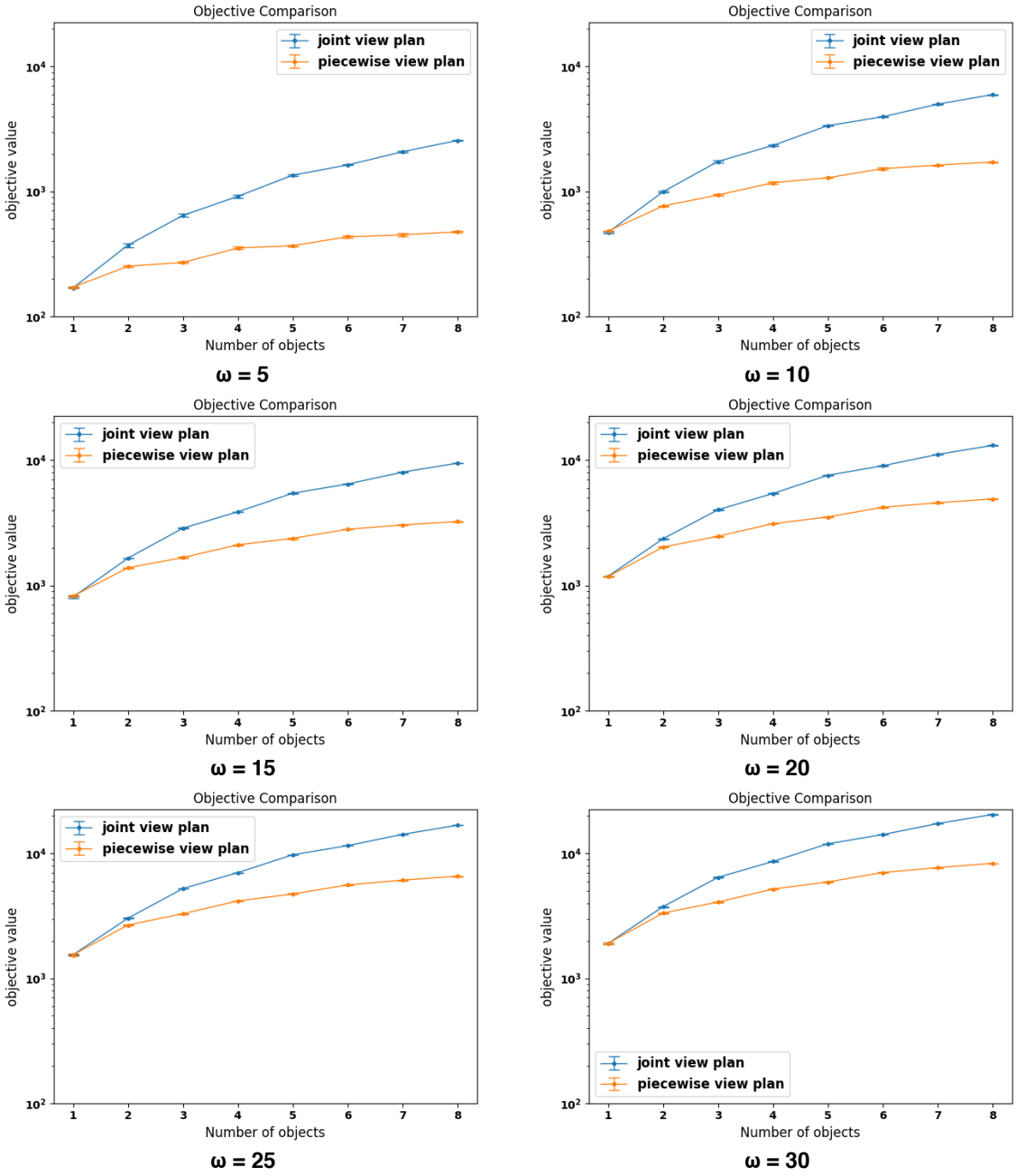


Figure 4.6: Comparison between the view selection objective value $\mathcal{E}_v$ in log space by optimizing for each object independently and by optimizing for all the objects together, with varying desired view quality parameter ω.

Such improvement is achieved by a combination of increase in view quality and decrease in number of selected views.

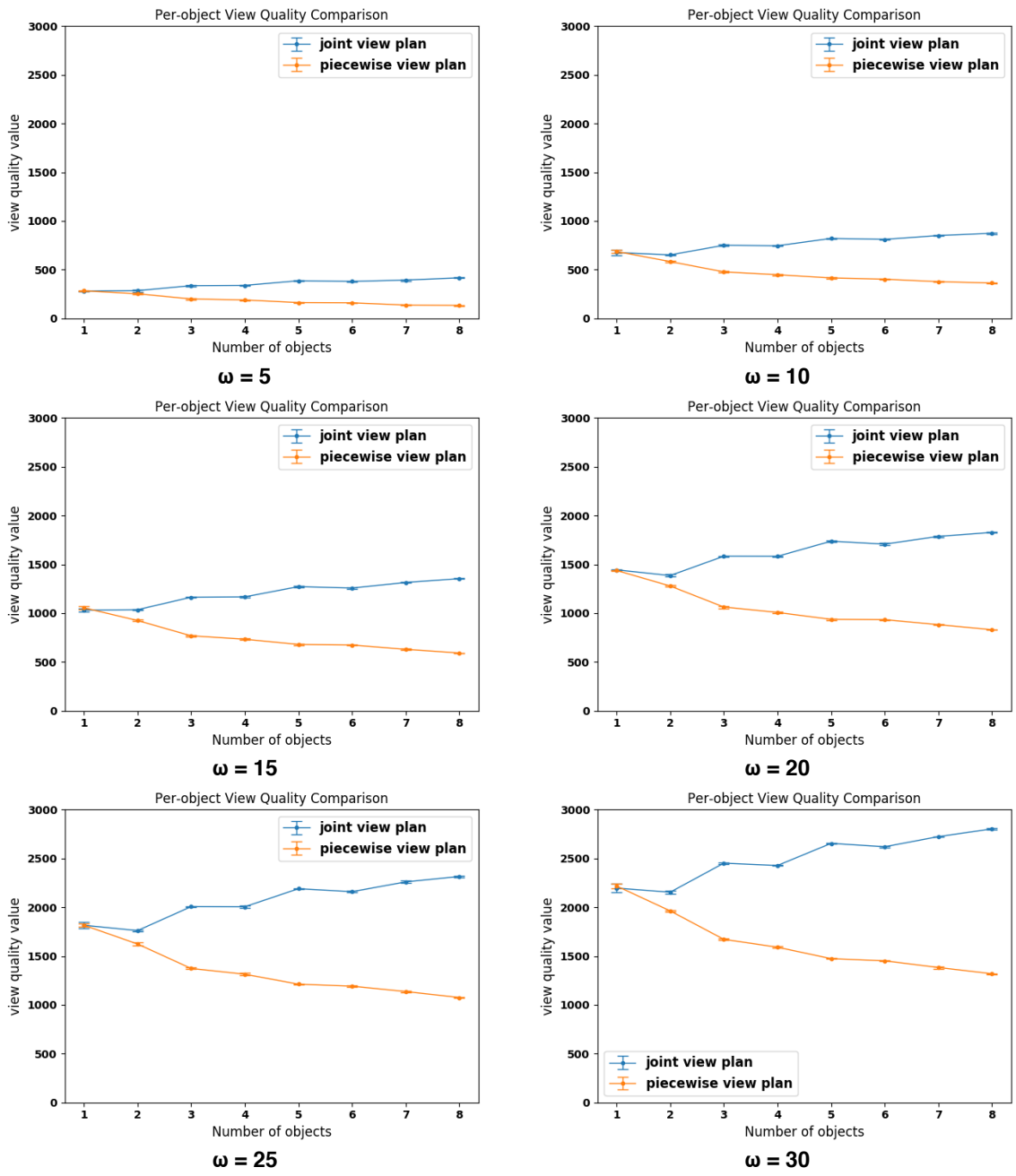


Figure 4.7: Comparison between the average view quality achieved by optimizing for each object independently and by optimizing for all the objects together, with varying desired view quality parameter ω.

Plots in Figure 4.7 demonstrate how the average per-object view quality $\mathcal{Q}$ is improved by the joint optimization. Notice that the similar "staricase" pattern shown in the six plots implies certain objects arrangement in the test scene, as shown in

Figure 4.5. This can be eliminated if sufficient experiments are conducted on different multi-object scenes. As the user-controlled parameter ω increases, the view quality achieved for each object also increases, suggesting more surface areas represented by the point samples P are "seen" more clearly.

The joint optimization allows any candidate view $v \in V^*$ to "see" any object in the scene, which makes it possible for the algorithm to select views where point samples from more than one objects can be "seen". This potentially reduces the necessary number of views for achieving the same amount of view quality. Figure 4.8 shows how the number of selected views is affected by different approaches to optimizing the view selection objective $\mathcal{E}_v$. The decrease in number of selected views achieved by the joint optimization becomes more and more visible as ω increases. The error bars on each curve in the plot suggest the variance in number of views selected in different runs. As we use simulated annealing, a randomized algorithm, to optimize for the objective, it is likely for it to yield different solutions with similar objective values.

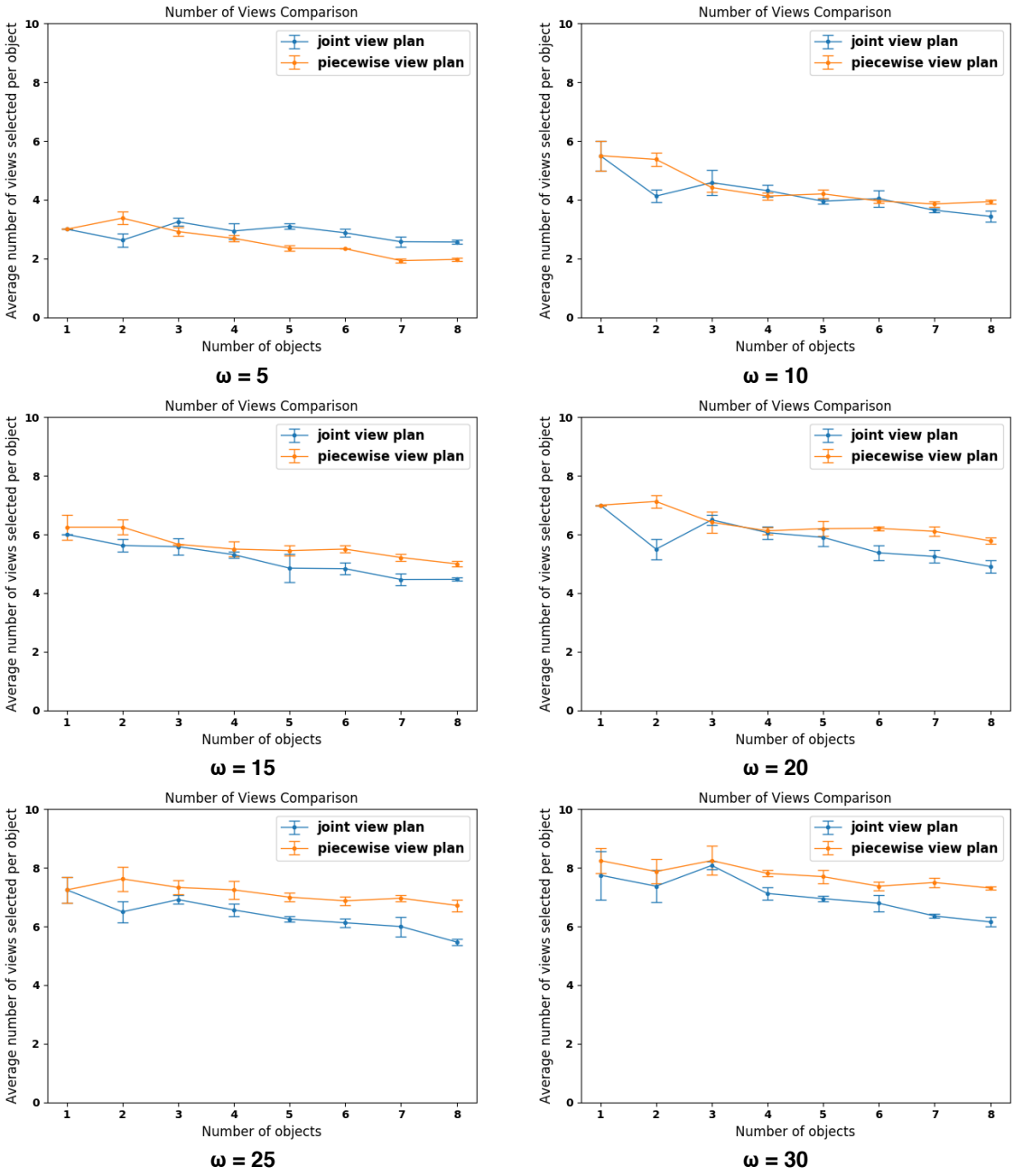


Figure 4.8: Comparison between the per-object number of views selected by optimizing for each object independently and by optimizing for all the objects together, with varying desired view quality parameter ω.

Figure 4.9 shows the computational time consumption for both approaches. Optimizing the view selection objective $\mathcal{E}_v$ jointly over all the objects can actually achieve some reduction in runtime.

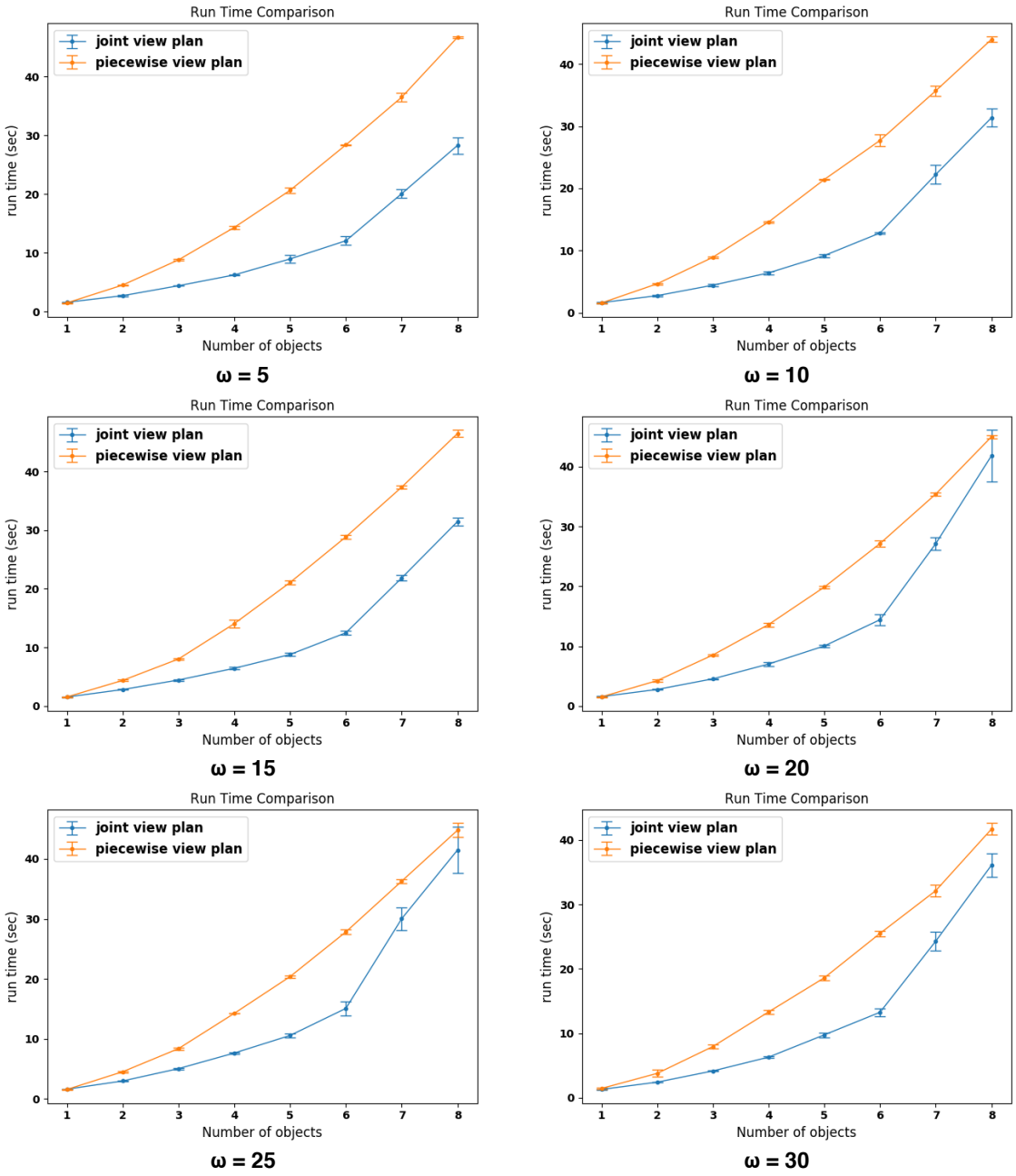


Figure 4.9: Comparison on the CPU time between optimizing for each object independently and by optimizing for all the objects together, with varying desired view quality parameter ω.

4.4.2 Joint View and Path Planning

Let's revisit the objective function introduced in Equation 4.4 for the overall planning problem:

$$\mathcal{E}(P, V^*) = \omega \cdot \mathcal{Q}(P, V^*) - \mathcal{s} \cdot |V^*| - \mathcal{L}(V^*) \qquad \text{(4.4 revisited)}$$

the path length $\mathcal{L}(V^*)$ is defined as the total time needed for the positioning system to move the scanner and traverse all the view configurations in V^*. Its value will definitely depend on the selection of V^*. In the previous sections, however, the view selection does not take into account the path length that might be yielded by given set of views V^*, and $\mathcal{L}(V^*)$ only gets separately optimized for after V^* has been determined. Such decomposition of the objective makes the problem easier to solve, but potentially it might yield suboptimal result for the overall objective $\mathcal{E}$.

In this section, we examine the objective $\mathcal{E}$ as a whole. We again employ the simulated annealing based method described in Algorithm 1, with an adaption of evaluating ΔE according to Equation 4.4, instead of Equation 4.18, in each iteration. For each configuration of $\vec{X}_t$, $\mathcal{L}$ is computed among the corresponding V^* as the shortest path length by Christofides' algorithm [12].

We compare the performance of the proposed joint view and path planning approach to the sequential approach. In the sequential approach, the view selection and the path finding are independently computed, but the view selection objective $\mathcal{E}_v$ is jointly computed among all the views, as described in Section 4.4.1. This is referred by label "view only joint plan" in the subsequent plots for quantitative analysis, whereas the "grand unified" method proposed in this section is referred by label "joint path and view plan".

As shown by Figure 4.10, since path length $\mathcal{L}$ is also evaluated during the optimization for view selection, the "joint path and view plan" always yields a shorter path in average. The "view only joint plan" computes $\mathcal{L}$ only once at the end of

the optimization, after the views are selected. Occasionally, it might obtain a shorter path, but the large variance shown in Figure 4.10 suggests its instability.

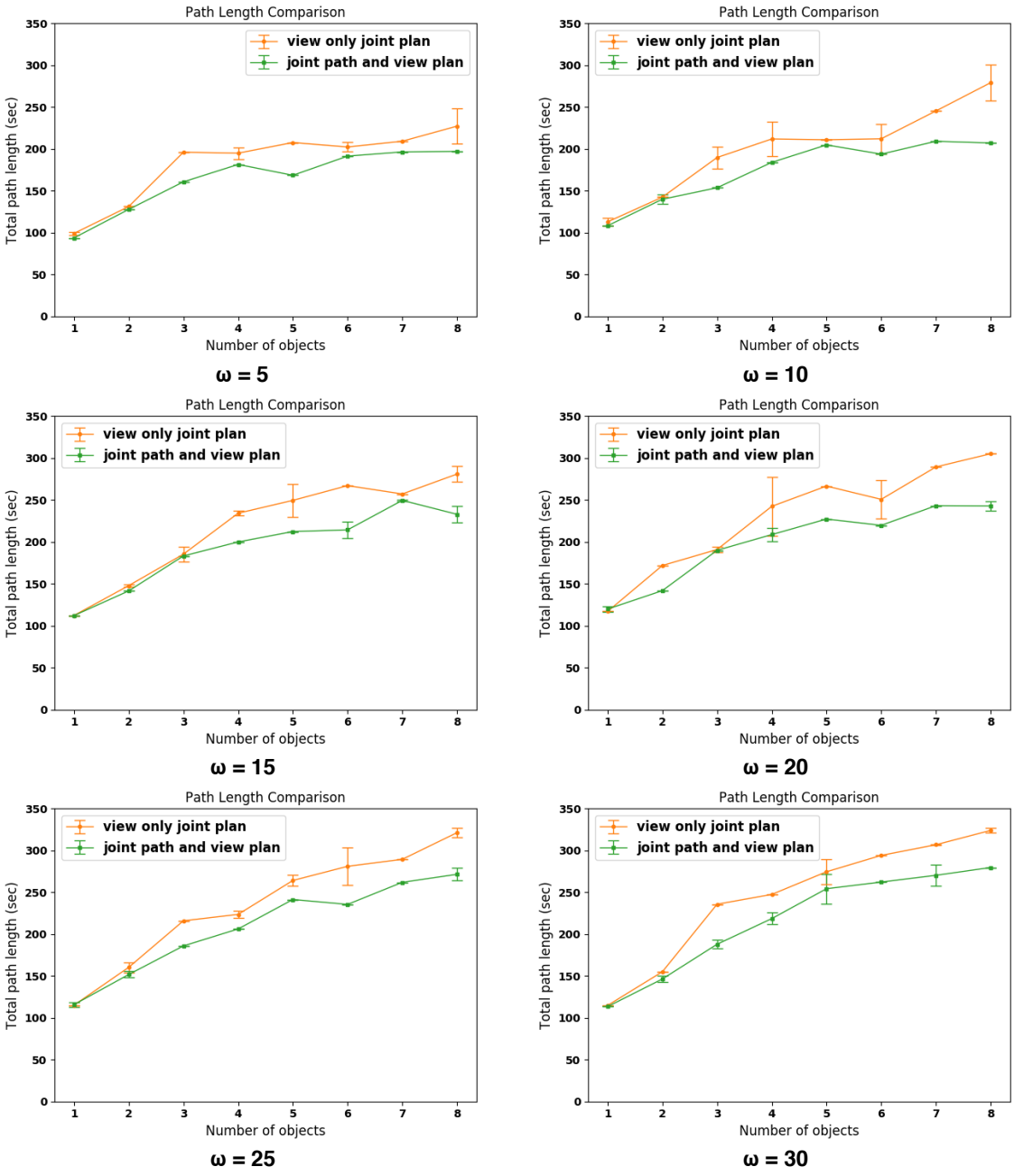


Figure 4.10: Comparison between path length $\mathcal{L}$ achieved by jointly and sequentially optimizing $\mathcal{E}_v$ and $\mathcal{E}_p$, with varying desired view quality parameter ω.

Figure 4.11, 4.12, and 4.13 demonstrate that, on the surface, the two approaches produce similar result in the overall objective, the average per-object view quality, as well as the number views selected.

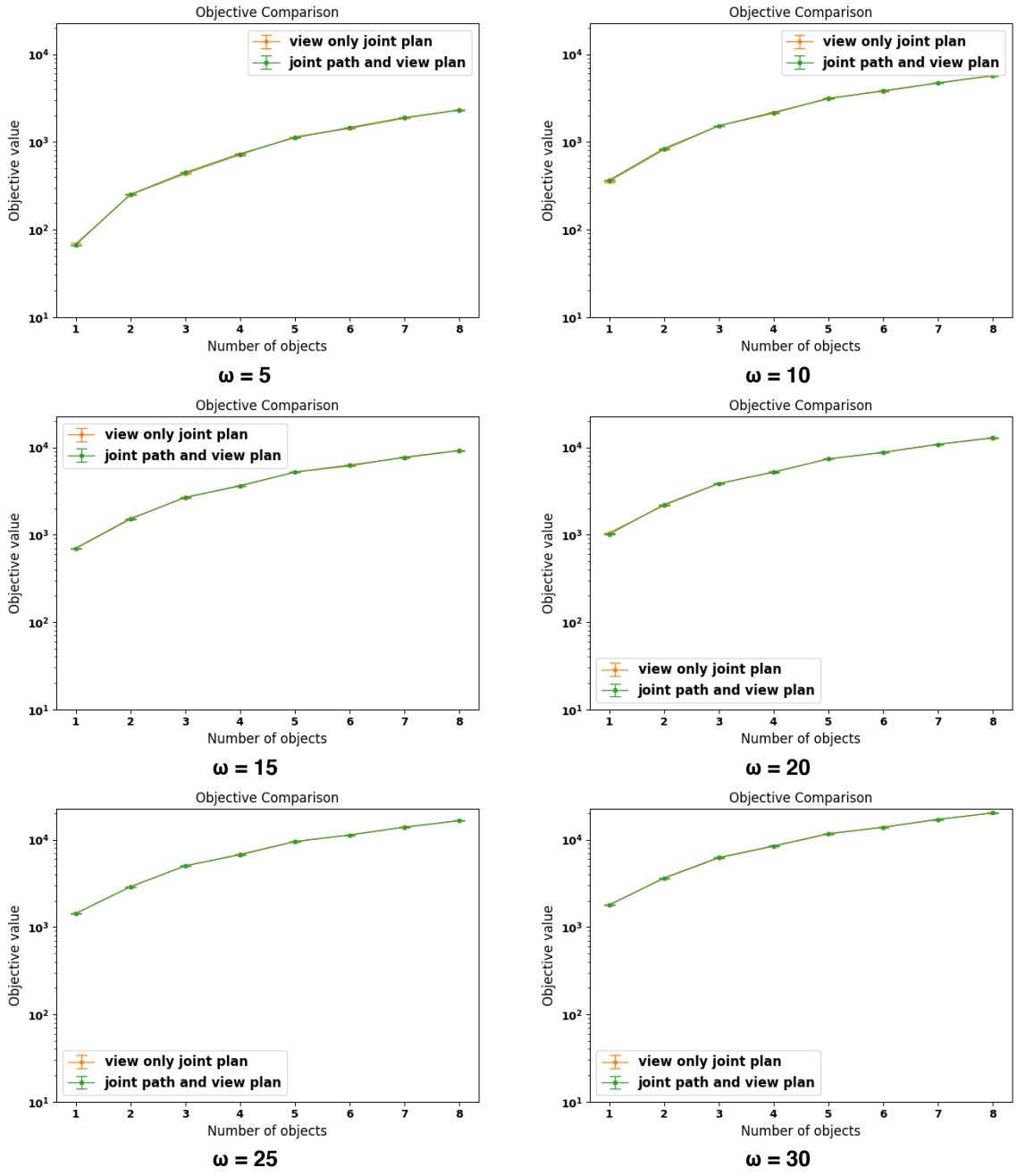


Figure 4.11: Comparison between jointly and sequentially optimizing $\mathcal{E}_v$ and $\mathcal{E}_p$ from the overall planning objective $\mathcal{E} = \mathcal{E}_v + \mathcal{E}_p$ in log space, with varying desired view quality parameter ω.

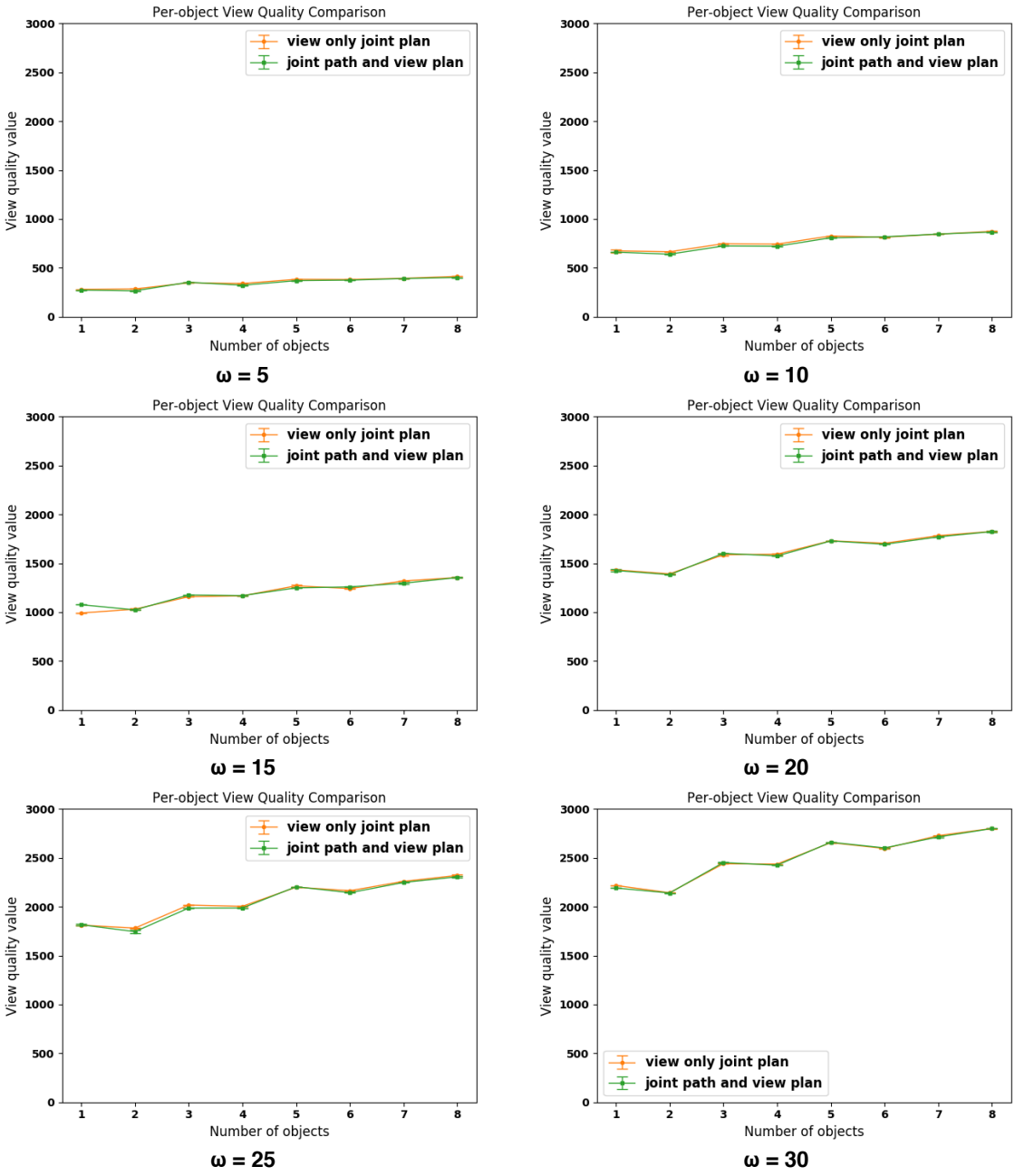


Figure 4.12: Comparison between the average per-object view quality $\omega \cdot \mathcal{Q}$ achieved by jointly and sequentially optimizing $\mathcal{E}_v$ and $\mathcal{E}_p$, with varying desired view quality parameter ω.

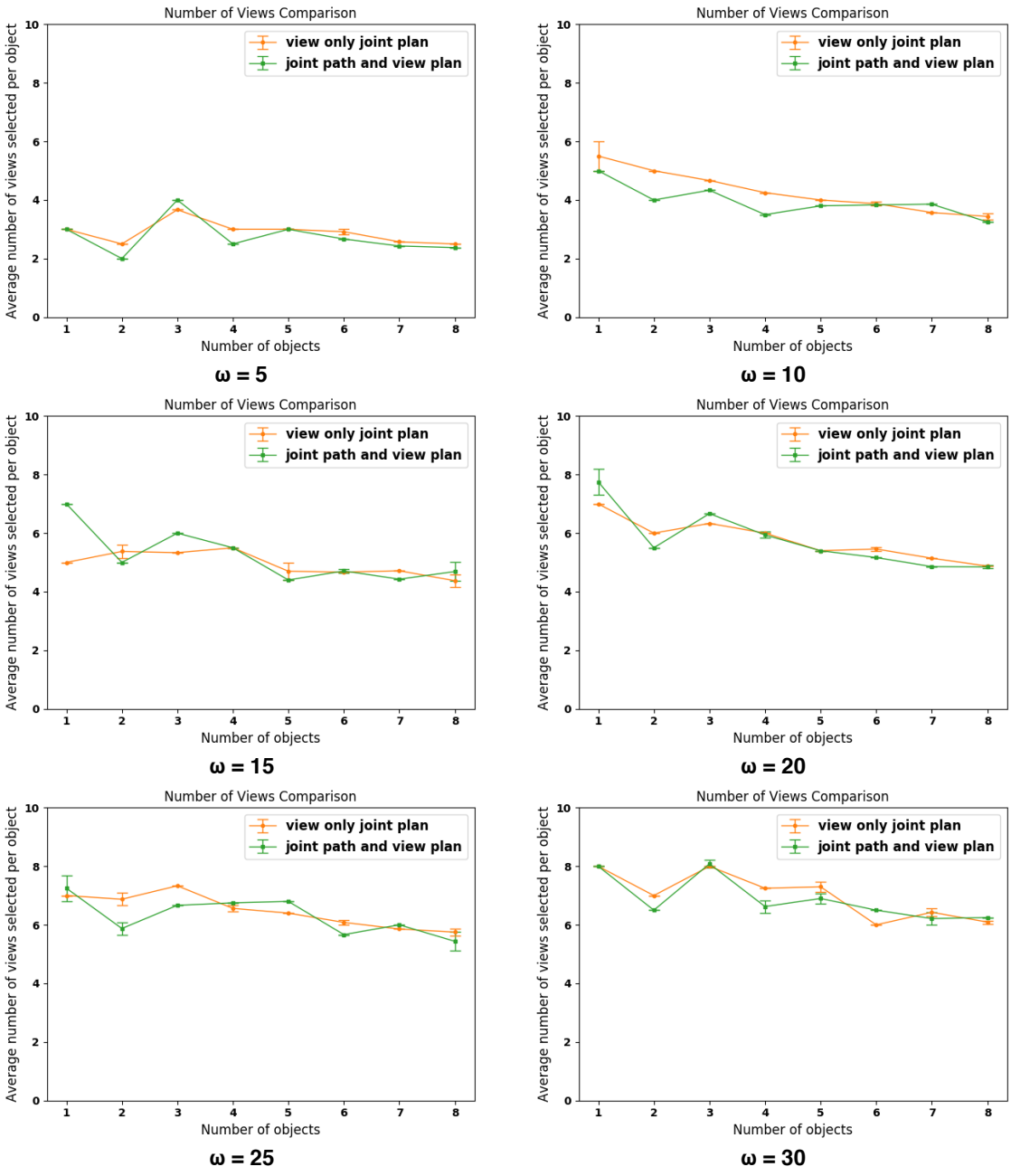


Figure 4.13: Comparison between the number of views selected by jointly and sequentially optimizing $\mathcal{E}_v$ and $\mathcal{E}_p$, with varying desired view quality parameter ω.

In order to take a closer look at the performance comparison between the two approaches, we evaluate the difference in the objective

$$\begin{aligned}\Delta\mathcal{E} &= \mathcal{E}^{\text{jnt}} - \mathcal{E}^{\text{seq}} \\ &= (\omega \cdot \mathcal{Q}^{\text{jnt}} - \mathcal{S}^{\text{jnt}} - \mathcal{L}^{\text{jnt}}) - (\omega \cdot \mathcal{Q}^{\text{seq}} - \mathcal{S}^{\text{seq}} - \mathcal{L}^{\text{seq}}) \\ &= \omega \cdot (\mathcal{Q}^{\text{jnt}} - \mathcal{Q}^{\text{seq}}) + (\mathcal{S}^{\text{seq}} - \mathcal{S}^{\text{jnt}}) + (\mathcal{L}^{\text{seq}} - \mathcal{L}^{\text{jnt}}) \\ &= \Delta\mathcal{Q} + \Delta\mathcal{S} + \Delta\mathcal{L} \end{aligned} \tag{4.22}$$

The superscript "jnt" refers to the joint view and path approach, whereas "seq" refers to the sequential view and path approach. Notice that positive $\Delta\mathcal{Q}$, $\Delta\mathcal{S}$, and $\Delta\mathcal{L}$ indicate better performance by the joint approach. $\Delta\mathcal{S}$ and $\Delta\mathcal{L}$ represent the saved scanning time and travel time, respectively.

Figure 4.14 shows a breakdown in $\Delta\mathcal{E}$ achieved by the joint approach over the sequential one, where the the upward pointing (positive) bars denote improvement in different terms, and the downward pointing (negative) bars denote the worsening. The absence of the negative pink travel time worsening bars suggests that the joint approach always achieves at least as short (good) path length as the sequential approach. Although the improvement in path length is often achieved sacrificing the view quality, it still leads to an increase in the overall objective $\mathcal{E}$ for about 70% of the time, among all the experiments we run. It is also worth noticing that, the changes in view quality (purple) and scan time (olive) usually appear to be opposite, reflecting the fact that fewer views tends to yield lower view quality.

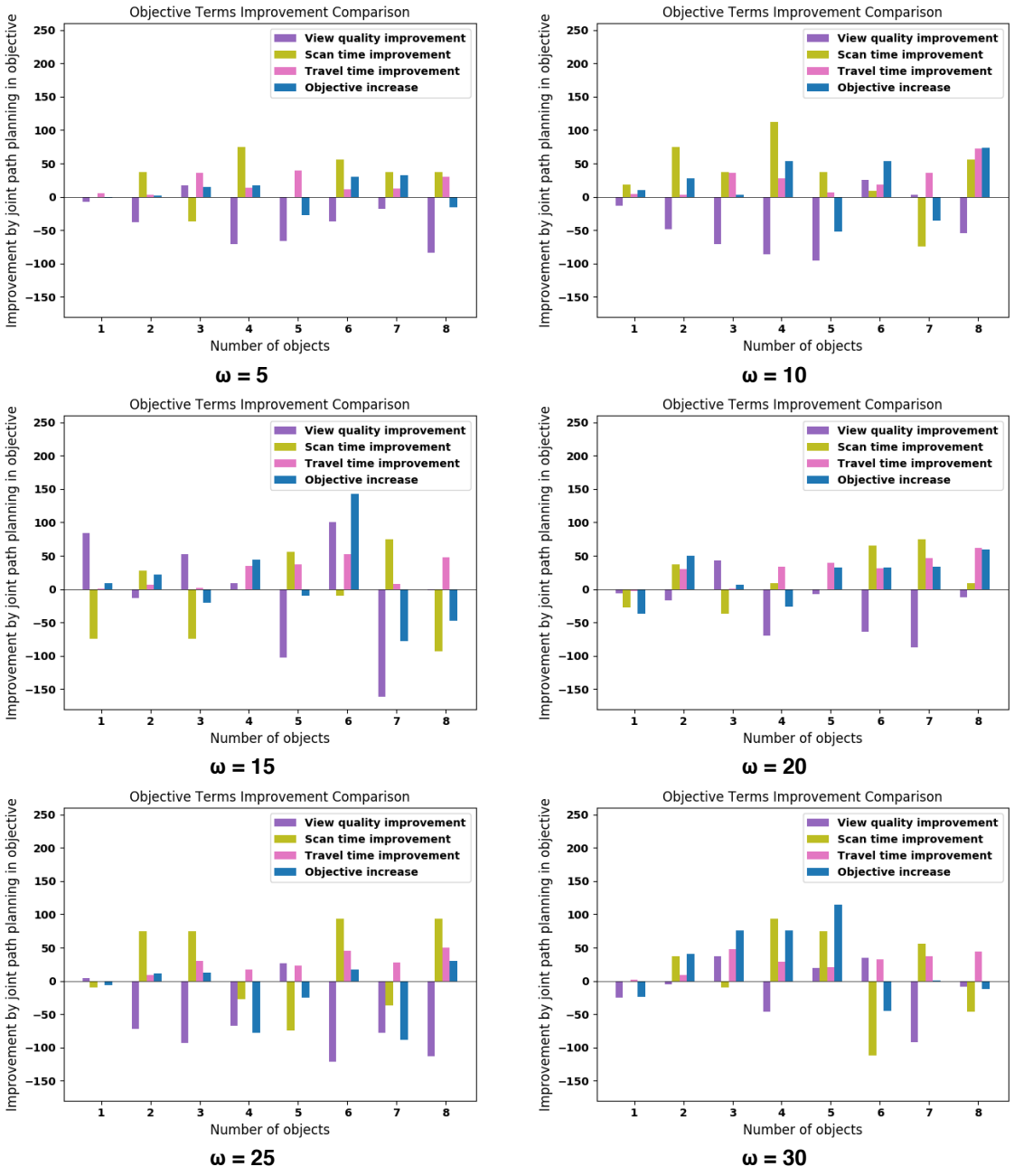


Figure 4.14: Comparison on the breakdown in the objective change introduced by the joint optimization, with varying desired view quality parameter ω.

The scanning time $\mathcal{S} = \mathfrak{s} \cdot |V^*|$, determined by the number of views selected, appears to be fluctuating, due to the randomness in the simulated annealing. Notice that the coexistence of positive pink bar and negative olive bar suggests that, even in the cases where the scanning time worsens because of more views being selected,

the algorithm is still able to produce a shorter path. The visualization in Figure 4.15 gives an example of how the two paths compare to each other. The joint approach is able to obtain a path that takes 296 seconds for the scanner to travel across 57 views, whereas the sequential approach yields a 336-second-long path with 54 views. In the joint view and path optimization, to maintain a short path among all the views, when new views are introduced, the path length term $-\mathcal{L}$ in the objective encourages the algorithm to select views "along the way" to avoid the increase in path length.

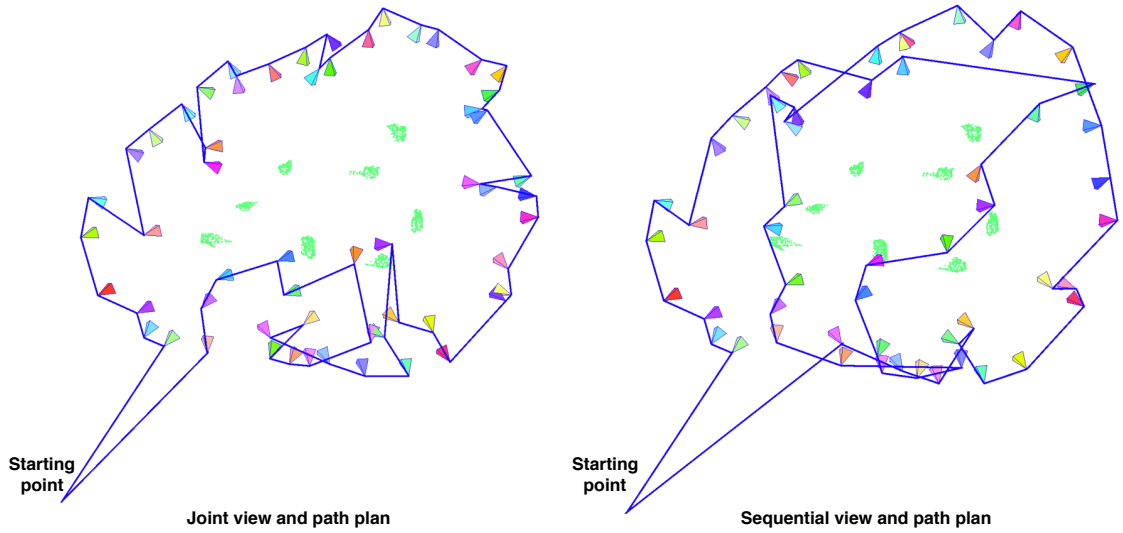


Figure 4.15: Visualization of a path (296 seconds long, 57 views) obtained by the joint approach (left) and a path (336 seconds long, 54 views) obtained by the sequential approach (right).

While the joint view and path plan does achieve some improvement on the efficiency of the acquisition system, the current simulated annealing based algorithm takes much longer to run, compared to the sequential approach, where the view selection optimization does not depend on the resulting path length. Figure 4.16(left) shows the CPU time comparison between the two approaches. We also compare the decrease in the acquisition system run time (including the time for the scanner to both scan and move) to the increase in the computational time introduced by the joint op-

timization, as shown in Figure 4.16(right), and find that the end-to-end system time great increases due to the huge extra computational time.

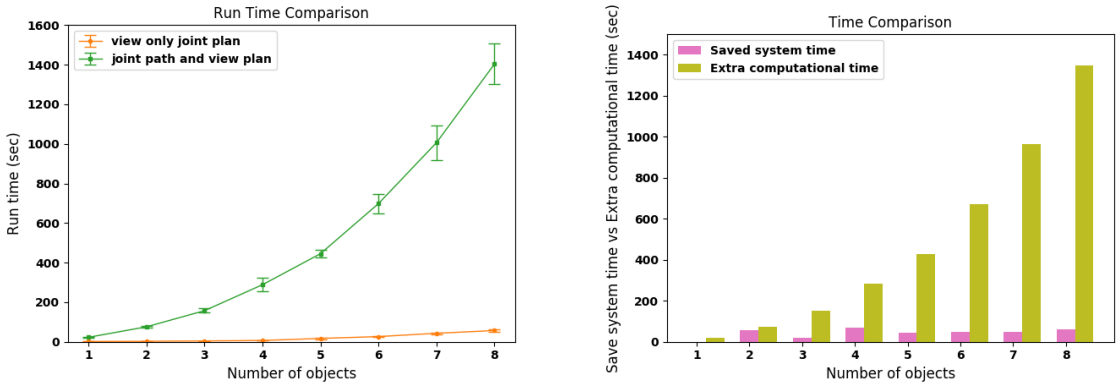


Figure 4.16: Comparison on the CPU run time between the joint and the sequential approaches (left) and comparison between the saved runtime for the acquisition system and the extra CPU time introduced by using joint view and path planning (right).

In conclusion, our study in Section 4.4.1 and 4.4.2 shows that, it's always worth doing the joint optimization over all objects for view selection, as it achieves better objective and costs less computational time at the same time. Jointly planning the view and path together, on the other hand, can potentially achieve shorter path and better overall objective. However, using the current simulated annealing based method takes much longer in computation, and can not directly apply in practice. Although more sophisticated optimization tools can be employed to close the gap between the acquisition system run time and the computational time for planning, our preliminary experiment suggests that it should only expect a small improvement in the planning objective.

Chapter 5

System Evaluation

"All experimentation is criticism."

Sir Peter B. Medawar

We have implemented the sequential view and path planning in the prototype 3D acquisition system we built. Experiments are conducted to evaluate the view planning, path planning, as well as the system as a whole. In each case, we present results for scanning time and quality, comparing our system to possible simpler implementations. We also demonstrate that our system is capable of scanning a variety of 3D objects with different geometry. The structured-light scanner in our system can achieve a 0.1 mm resolution.

5.1 View Planning Evaluation

To demonstrate that our view plan improves the *combination* of scan time and quality, we compare the acquisition results based on our view planning algorithm to those from a naive strategy commonly adopted by previous work, namely placing views uniformly around an object's centroid, at a fixed radius.

Efficiency. We demonstrate the improvement in efficiency due to introducing view planning into the acquisition pipeline on two test scenes (on both sides) with four objects, each object having different shape and size. We run our view planning algorithm to obtain the view schedules for all the objects. Then we compare to a naive strategy with a fixed number of views, spaced equally around the centroid of each object, with the number of views set equal to either the fewest or most views associated with any object in our view-planning result from each scene.

Table 5.1: Comparison of total time for our view planning vs. naive strategies employing a fixed number of views per object.

	min fixed	max fixed	adaptive
total number of scans	44	64	52
planning time (min)	2×10^{-5}	2×10^{-5}	3.20
total scan time (min)	27.28	40.30	32.63
total travel time[1](min)	7.97	11.27	8.83
total time (min)	35.25	51.57	44.66
avg. time per object (min)	8.06	12.89	11.17

[1] Note that in Table 5.1 the total travel time includes a five second pause per scan, for vibration damping before capture, while in Table 5.2 the total travel time refers to the amount of time the positioning system spent on moving only.

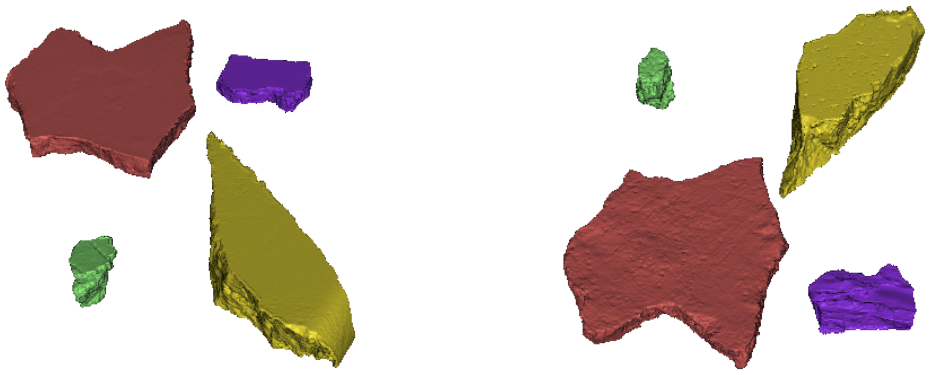


Figure 5.1: Front (left) and back (right) side of the reconstructed models of four objects scanned simultaneously with adaptive view planning. Models with the same color correspond to each other.

Table 5.1 summarizes the acquisition time for each of these scenarios. Note that we achieve an improvement over the naive plan with the maximum number of views, with

no penalty (or even improvement) in the quality of the acquired data. Of course, our view planning strategy is not as fast as the naive method with the minimum number of views, but it is less prone to missing areas of the surface or ending up with low data quality.

Figure 5.1 shows the final reconstructed models of the objects on both sides from the acquisition with the adaptive view plan.

Coverage. Unlike the naive approach, which equally distributes a fixed number of views around an object, the simulated annealing based view plan adaptively selects the number of views for each object. Therefore, an acquisition with view planning usually yields better coverage, especially for non-convex objects. We show a comparison on the scans of an object acquired in the last 4-object experiment. Figure 5.2 shows a closeup to the aligned raw scans from naive methods and our adaptive view plan (white regions indicate missing data). The scans acquired from the naive method with five views are missing data from both the tip of a sharp corner and a deep concavity. Even with the same number of views equally spaced around the object, the naive method with nine views is still missing some data at the tip of the sharp corner. With the neighborhood aggregation improvement added to our objective function, the view plan optimization places views around sharp corners to increase *meaningful overlap*.

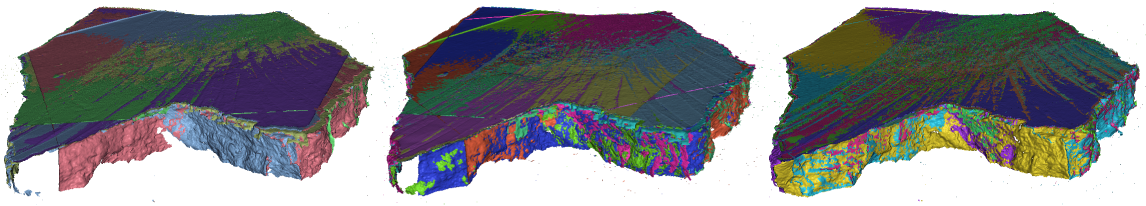
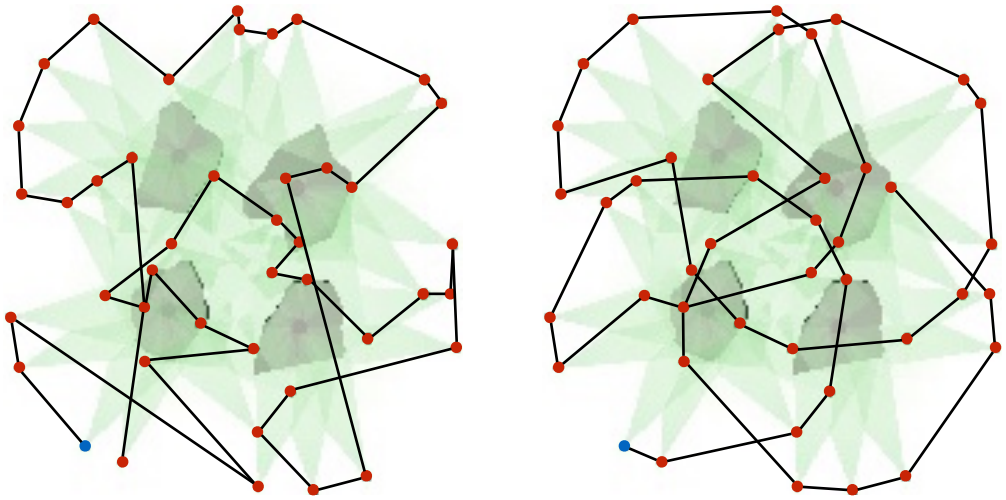


Figure 5.2: We compare the scans obtained using our view planning (right) to those acquired with a naive method employing five (left) or nine (middle) views, equally spaced around the centroid of the object. Our view planning result also selects nine views in this case.

5.2 Path Planning Evaluation

Given a set of views for multiple objects produced by the view planning stage, we compare our TSP-based path planning strategy to a naive algorithm. For the latter, we use the views selected for each object in sequence, always beginning the scanning of an object from the nearest view to the last one in the previous object. Figure 5.3 shows the results of the two different strategies for a simple scene with four objects.



(a) TSP-based path planning (b) Naive path planning

Figure 5.3: Comparison of our TSP-based path planning (a) and naive path planning (b). The former reduces motion time by approximately 15%.

Table 5.2: Comparison of total distances and times for our TSP-based path planning vs. a naive path planning strategy.

	our path plan	naive path plan
number of scans	44	44
planning time (μs)	579	19
total translation distance (m)	5.30	6.05
total rotation distance (deg)	2330	2690
total travel time (s)	228	268

Table 5.2 compares the travel distances and times for the two strategies. Notice that the TSP-based strategy achieves an improvement in travel time of 15%, even

with as few as four objects. For more objects, we have observed even greater savings in travel time, with small increases in computation time.

5.3 System-Level Evaluation

We compare the performance of our system to another acquisition system which employs the same structured light scanner, but uses a turntable as a simple positioning system. The turntable system adopts the naive view planning strategy described above, which uniformly samples a fixed number of views around the table center. We assume that the turntable system chooses the average number of views for the objects planned by our algorithm as its fixed number of views. Therefore the scanning time per view of both systems should be approximately the same.

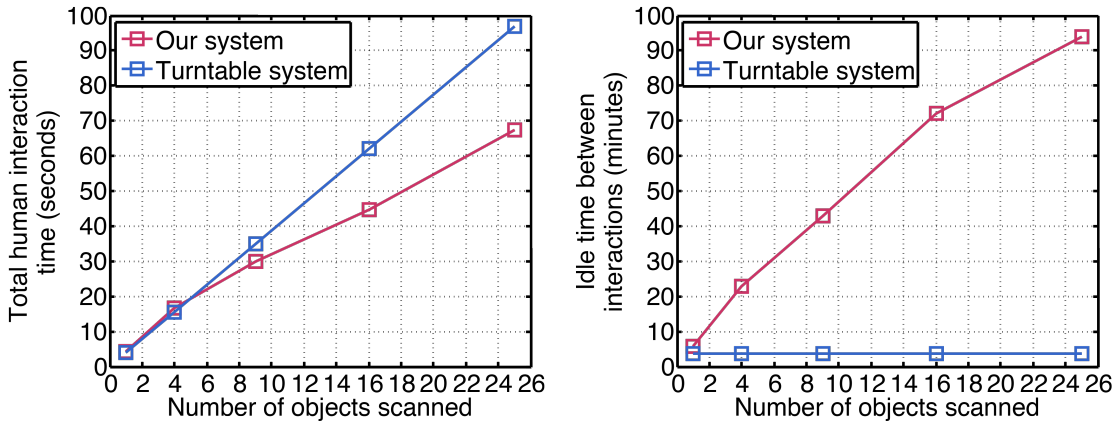


Figure 5.4: The two plots show the total amount of time a human interacted with the scanning system (left) and the total amount of idle time when the human did not need to attend the system between two adjacent interactions (right) on scanning batches of fresco fragments using both our system and the turntable system. Our system required less interaction time overall and afforded far larger gaps between interactions during which the operator was free to do other work undisturbed.

Figure 5.4 presents comparative statistics from scanning batches of fragments using both systems, illustrating how the total human interaction time and the idle time between interactions scale up with increasing number of objects scanned. In this

set of experiments we use fresco fragments as test objects, which are an important category of objects in archaeological digitization applications.

Scalability. The interaction time for the turntable system is linear in the number of objects, because for each single object the average operating time stays the same. On the other hand, the total interaction time for our system grows (slightly) sub-linearly, indicating that our system makes large scale acquisition tasks more efficient.

More important is the comparison of (human-operator) idle time. This shows a significant advantage of our system over the turntable system in that the user is free from tending to our system for long periods of time. This is essential for practicality in a scaled-up scanning scenario. In the case of fresco fragments, the idle time is actually half of the entire acquisition time, because each fragment is flipped once during the acquisition to obtain data from both sides. This leads to only two interactions with the system, while the turntable system requires constant flipping and replacing of objects. Notice that the sub-linear scalability of the idle time is mostly due to the travel time, as a result of our path planning.

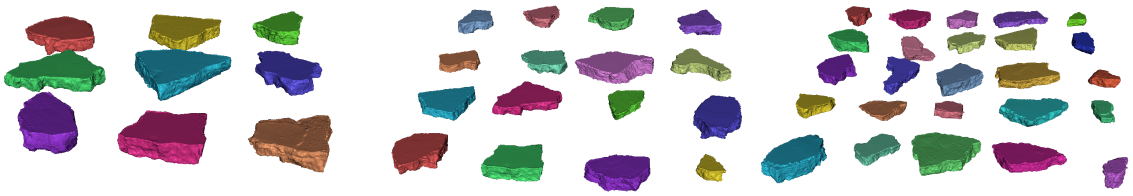
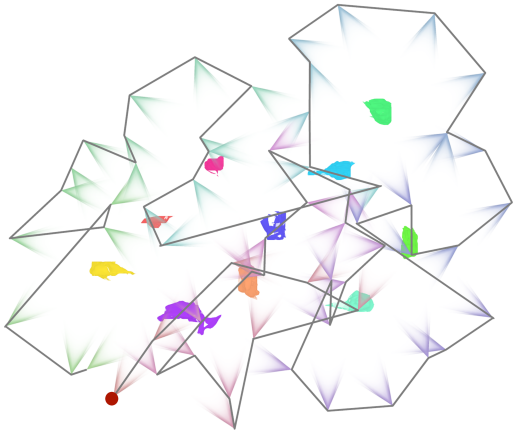


Figure 5.5: Reconstructed models of increasingly-large sets of fresco fragments, as used in our scalability experiment. The batch size is, from left to right, 9, 16, and 25.

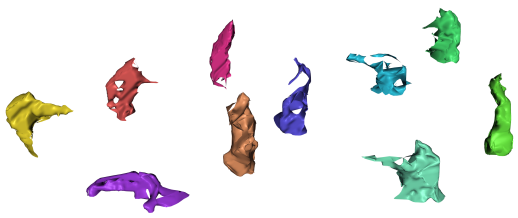
Quality. Figure 5.5 shows the reconstructed models for arrangements of 9, 16, and 25 objects scanned using our acquisition system. Based on manual inspection of the resulting 3D models, our system achieves at least comparable surface coverage over the objects being scanned, as compared to a turntable-based system.

5.4 Object Variety Evaluation

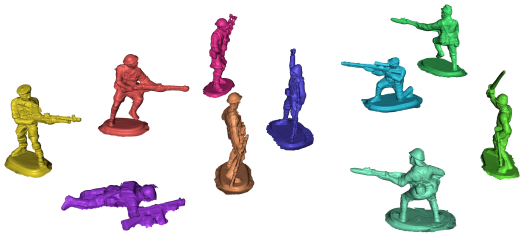
We demonstrate the capability of our system to scan a variety of 3D objects in addition to the fresco fragments.



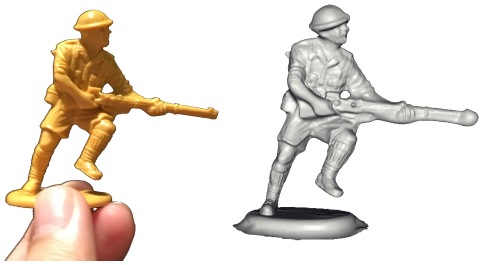
(a) Planned views and path.



(b) Approximate object models.



(c) Reconstructed models.



(d) Close-up visualization.

Figure 5.6: We scan a scene with ten toy soldiers based on views (a) planned from approximate, silhouette-carved models (b), and reconstruct high resolution 3D models (c). Each toy soldier is about 5 cm tall, and our system reconstructs sub-millimeter geometric detail (d).

Multiple small 3D objects. Figure 2.2 and Figure 5.6 together show an example of scanning a scene with ten toy soldiers. Each soldier is represented in the same color in Figures 5.6a, 5.6b and 5.6c. The entire scene is scanned from the views visualized in Figure 5.6a, and the scanner travels along the path planning result, where the color changes in camera view visualization correspond to the order along the path.

As demonstrated in Figure 5.6b, the approximate models acquired from our silhouette carving-based scene exploration capture sufficient meaningful detail, as opposed to the models obtained from consumer depth sensors such as the Kinect, which tend to have more random noise and miss detail. Our structured-light scanner achieves a resolution of 0.1 millimeter and captures an abundance of detail on the toy soldiers, as illustrated by the closeup visualization in Figure 5.6d. We note the importance of capturing structures such as the long barrel of the weapon in the exploration phase: this is used in the subsequent view-planning stage to place the necessary scanner positions to capture this tricky area. The reconstructed results suggest the potential of our system to be used for large-scale capture of stop-motion animation.

Large object. Figure 5.7 shows a reconstructed model acquired from our scanning system compared to the real figurine. The angel figurine is about 20 centimeters tall and has complicated self-occlusion. Scanning it from a single height would yield a large amount of missing data. Thus, we augment our prototype system with a platform that we can manually raise and lower. We restrict the number of heights (to three for this experiment), and calibrate them, allowing all of the scan positions to still be planned using the same view-planning algorithm. The final model is reconstructed by combining all of the scans, using a pipeline essentially identical to that for the single-height case.

Figure 5.7: A 20 cm tall angel figurine (left) is scanned and reconstructed (right) using our system, with the object platform adjusted to three heights.

Figure 5.8: Two differently sized soldiers (left) are scanned together and reconstructed (right) with our system.

Objects with different scales. Our system is capable of simultaneously scanning objects with different scales thanks to the adaptive view planning. Figure 5.8 shows

two soldiers at different sizes and their reconstructed scanned models. As introduced in Section 2.3, candidate views are sampled based on an elliptical cylinder fit to each approximate object model. In this case, the candidate views for the larger soldier span a much wider range compared to those for the smaller soldier. This allows greater flexibility in the scale of objects being scanned at the same time, compared to a turntable scanning system in which the candidate views are always sampled on a circle with fixed radius.

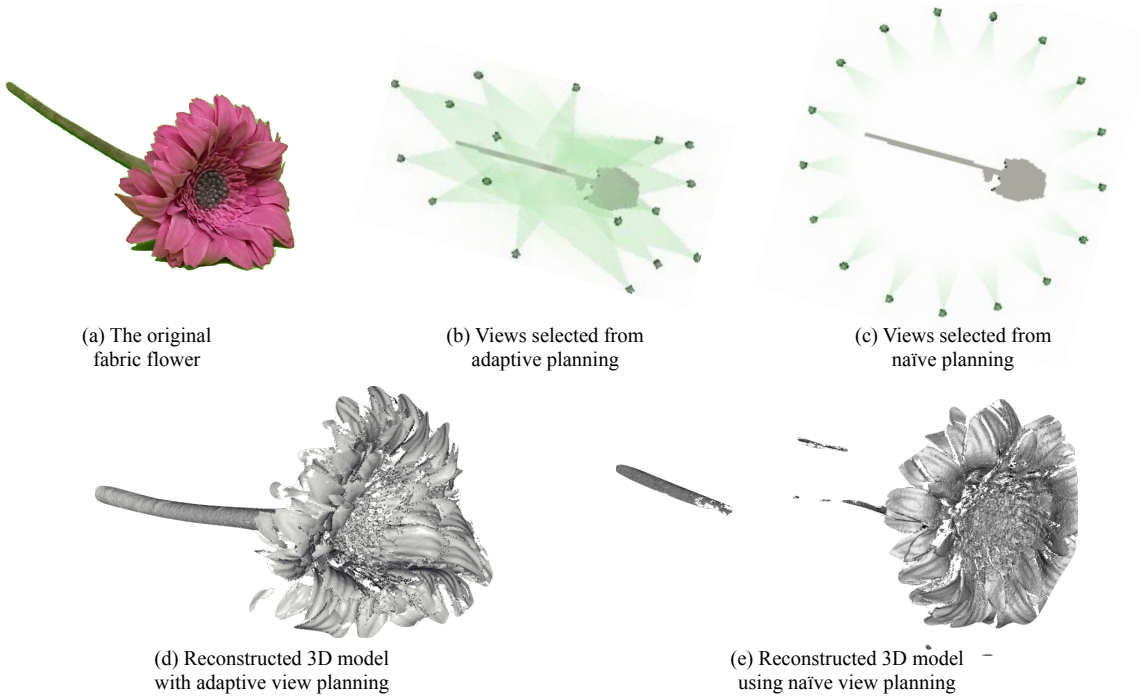


Figure 5.9: A flower with a long, thin stem (a) is scanned and reconstructed (d) with our system. Elongated objects such as this are a worst case for a turntable-based system with equally-spaced views (c).

Long, thin object. Figure 5.9 demonstrates that our system is able to handle extreme geometry such as the flower with its long, thin stem. Due to the fact that we compute candidate views based on an elliptical cylinder fit to the approximate object model, as shown in Figure 5.9b, it is easy for our system to focus on areas such as the stem and the backs of the flower's petals (see Figure 5.9d for reconstruction). A turntable scanning system that places views uniformly around the object at a fixed

radius (Figure 5.9c) is likely to yield very poor surface coverage of this flower (see Figure 5.9e for reconstruction).

Cultural heritage. Figure 5.10 shows a reconstructed model of a reproduction cuneiform tablet, which along with fresco fragments forms another important category of objects in archaeological digitization. The inscriptions on the tablet are clearly captured by our high-fidelity structured-light scanner, and with the scalability of our system we believe it would be easy to digitize these artifacts en masse with little human interaction, thus setting archaeologists and conservators free from tedious tasks.

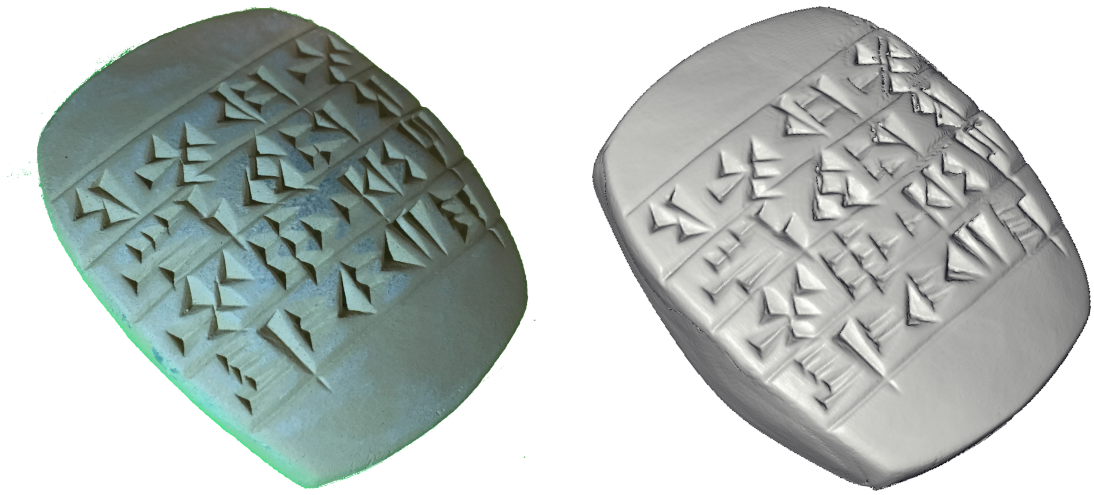


Figure 5.10: An $8 \times 6 \times 3$ cm reproduction cuneiform tablet (left) is scanned and reconstructed (right) with our system.

Chapter 6

Addressing Surface Inaccessibility

"In other words, because it's relatively impossible, it's possible."

Douglas Adams

Surface inaccessibility can become a major limitation pertaining to existing automated 3D acquisition systems, due to a combined effect of both the complexity of shape geometry and the degrees of freedom of the scanner motion capped by engineering capability. Figure 6.1 shows an example of failure in reconstructing certain detail in the object surface. The 3D model is reconstructed from the data captured using our prototype 3D acquisition system. Since the scanner in our system is facing downward, as shown in Figure 2.2c, inherently it is not able to capture the geometry information below the soldier's hand. Similarly, the bottom of the object is not accessible to the any scanner view, either. To ameliorate this, we explore solutions from different aspects, including improving the system hardware design, and adapting the view planning algorithm.

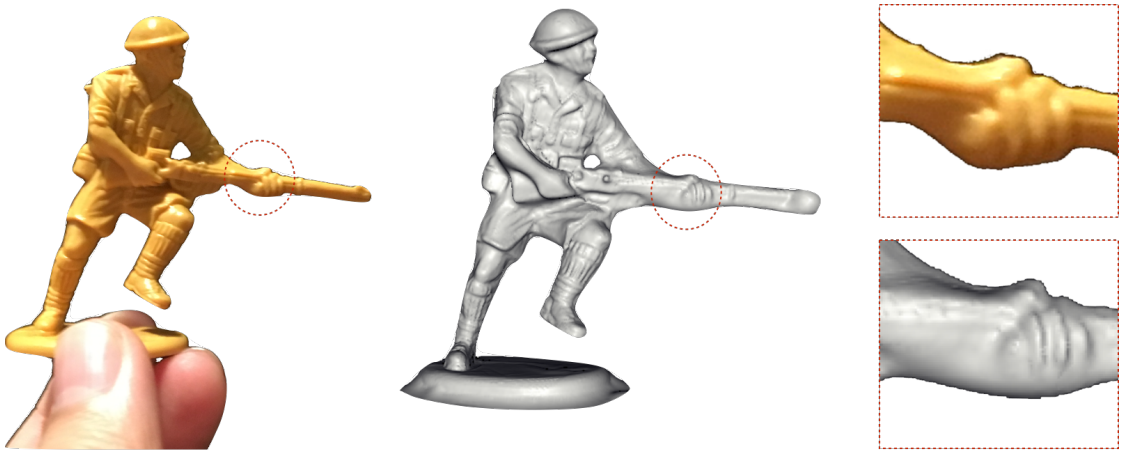


Figure 6.1: Detail in certain region cannot be perfectly reconstructed due to missing data caused by surface inaccessibility.

6.1 Augmented Hardware Design

In the current setup of our prototype system, the objects are placed on a scanning platform covered by a black cloth. In order to obtain a complete surface model of the objects, at least one flipping is required from the user, to reveal the hidden surface region. Occlusion at the physical touching region is in fact unavoidable in current platform based acquisition systems. This is also true to human hand-held or robotic-arm-held systems. One simple augmentation to the current platform based system design can be replacing the scanning platform with transparent material, and incorporating from below an additional scanner, together with the positioning system. Figure 6.2 renders an imagined design of the augmented system. Hypothetically, this solution can entirely avoid the user flipping interaction.

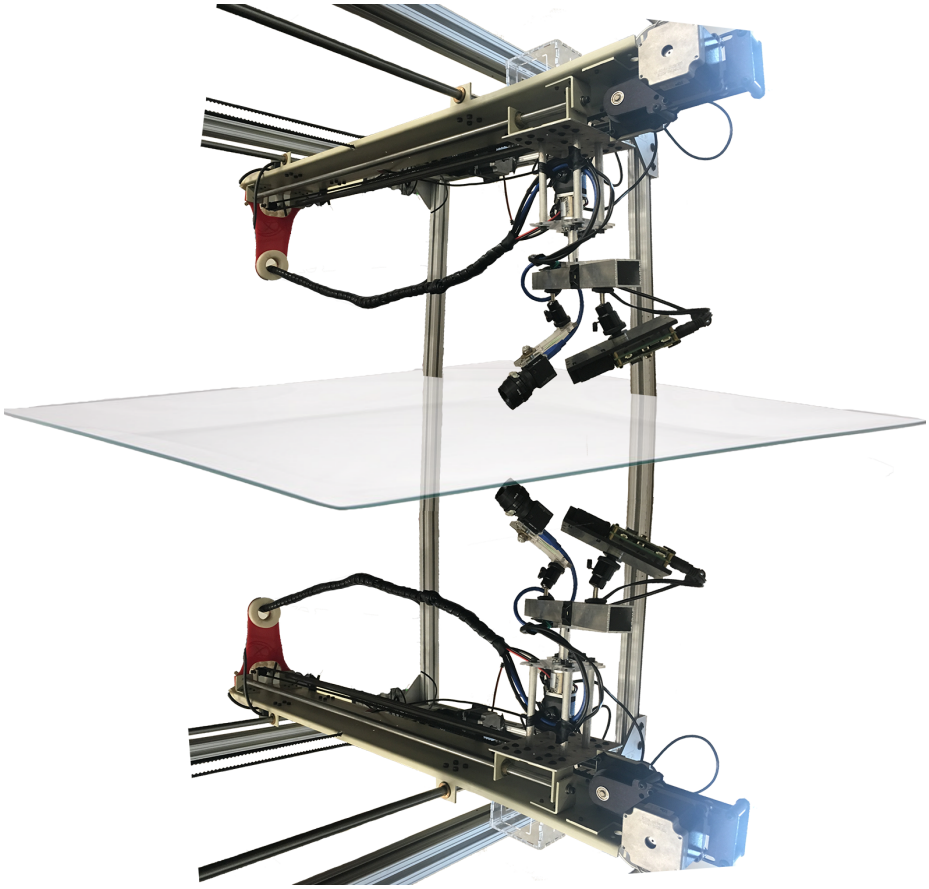


Figure 6.2: An imagined augmentation to the current acquisition system design, with a transparent scanning platform and a coupled scanner from below.

We run experiments to test the feasibility of this technique. We first place an object on the platform of our existing system, and scan the revealed surface normally. Then we flip the object, and place a sheet of transparent material on top of it, mimicking the effect of placing the object on a transparent platform, but only upside down. The second round of scan is performed by capturing the other side of the object through the transparent material. Data for both sides are integrated into 3D models and nicely merged together, as shown in Figure 6.3

This suggests that the proposed new design potentially makes it possible to remove the user flipping interaction, making the acquisition more automatic in practice.

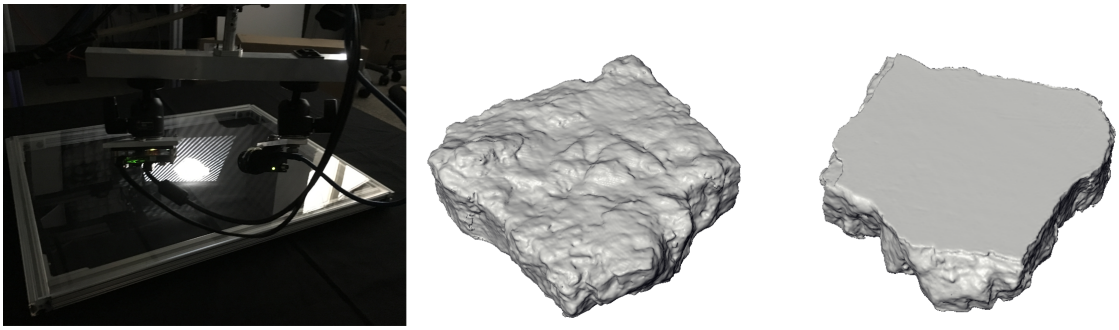


Figure 6.3: We scan the front side of an object using our system as usual but the back side through a sheet of transparent thin acrylic (left) and obtained the final models nicely reconstructed for both the front (right) and back (middle) side, which are then merged together.

6.2 Iterative View Planning

As aforementioned in the problem statement (Section 1.2), obtaining a complete object surface model requires altering the relative pose (view) between the object and the scanner. The candidate view set of our prototype system is limited by its motion flexibility. Section 6.1 above explores solution by doubling the motion flexibility of the scanner, so that the candidate view set V extends. The complement of increasing the engineering complexity to the system, is to re-orient the object. The flipping operation involved in our current system is a straightforward example of re-orienting the object. In this section, we study an extension of the view planning discussed in Chapter 4, leveraging the information we already have about the object surface, to provide guided re-orientation of the object, assuming re-orientation is necessary.

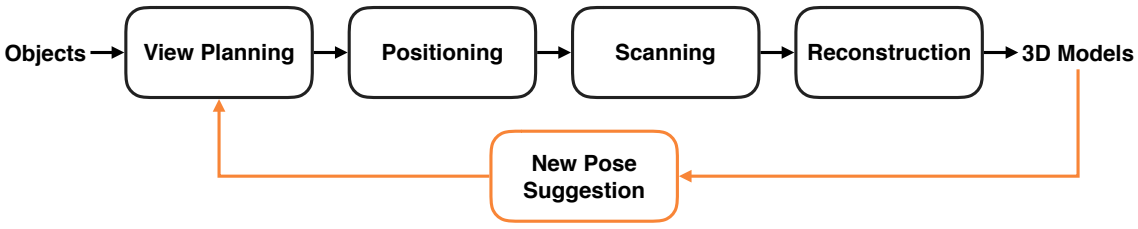


Figure 6.4: An iterative 3D acquisition pipeline.

Figure 6.4 shows how the original 3D acquisition pipeline is adapted. The idea is to feed the information from the reconstructed 3D model back to the pipeline to make the acquisition process iterative. We evaluate the output 3D model from the original end-to-end pipeline to identify the regions with poor reconstruction, and based on such information suggest new poses for re-orientating the object, which will hopefully yield better views for the scanner to capture data from.

6.2.1 New Pose Suggestion

To re-orient an object, one key thing is to identify the possible poses that the object can stably rest upon, given a flat supporting surface. A method similar to that from [22] is adopted to estimate the stable poses.

Algorithm 2 Stable Pose Suggestion

Input: Reconstructed 3D triangular mesh model $\mathcal{M}$
  Compute the volume centroid M for $\mathcal{M}$
  Compute convex hull $\mathcal{C}$ for $\mathcal{M}$
  Simplify $\mathcal{C}$ by merging co-planar faces
  for every face $f \in \mathcal{C}$ **do**
    Compute the projection P_M of M in the plane defined by f
    if P_M is inside polygon f **then**
      Compute $\mathcal{T}_f$ that transforms vector $\overrightarrow{P_M M}$ to be upright
    end if
  end for
Output: Set of transforms $\{\mathcal{T}_f\}$

The volume centroid enclosed by the triangular mesh surface is computed by (1) decomposing the volume into tetrahedra by the triangle faces and a random point, and (2) accumulating the tetrahedra centroids weighted by their corresponding signed volumes. The sign of each tetrahedron is defined by the whether the selected random point is on the same side where the face normal is pointing to.

Figure 6.5 shows an example of the reconstructed 3D model and its convex hull. We use CGAL [55] to compute the 3D convex hull.

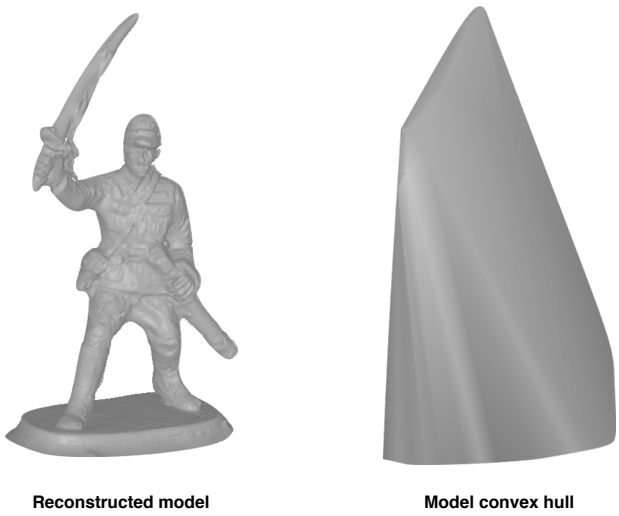


Figure 6.5: The reconstructed 3D model (left) and its convex hull (right.

By applying the output transforms from Algorithm 2 to the originally reconstructed 3D model, it can be re-oriented into different stable poses, as shown in Figure 6.6.

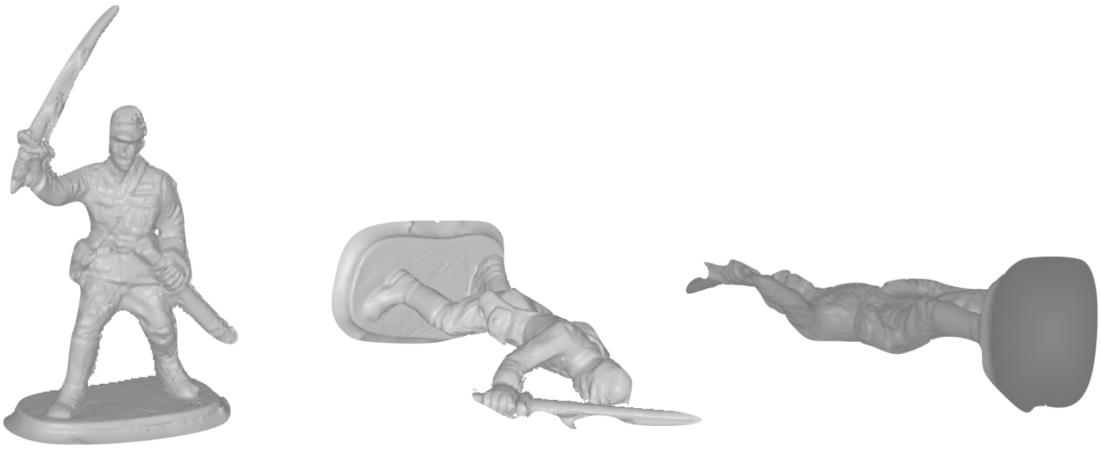


Figure 6.6: The reconstructed 3D model (left) re-oriented at different stable poses (middle and right).

6.2.2 View Quality Re-evaluation

The original view planning utilizes the approximate models obtain from the scene exploration as a prior to evaluate view quality. Let's revisit the original view quality

measurement defined in Chapter 4,

$$\mathcal{Q}(P, V^*) = \sum_{p \in P} q(p, \beta_1(p)), \qquad \text{(4.9 revisited)}$$

where P represents the set of point samples from the surface of the approximate model. After running the entire acquisition pipeline once, much more information about the object surface is obtained, and therefore this can be leveraged by planning in the subsequent iteration.

The view quality is re-evaluated on the reconstructed model once it is obtained. Figure 6.7 shows the view quality visualization on the object before (left) and after (right) the scanning and reconstruction. The point samples from both the approximate model and the reconstructed model are examined in the pose where the object is scanned.

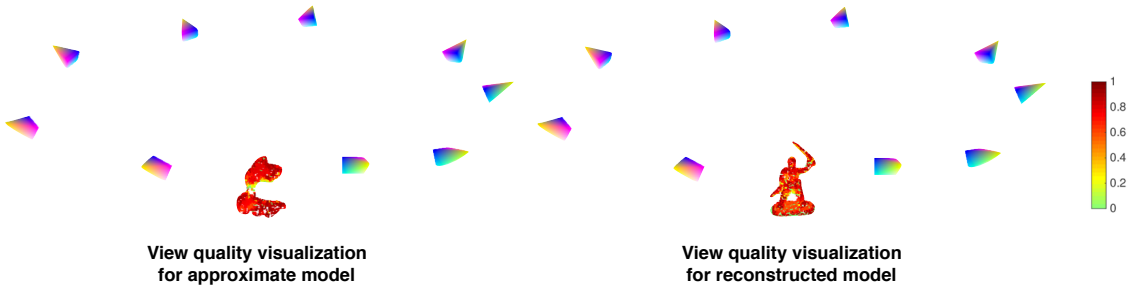


Figure 6.7: The view quality visualization on point samples representing the object before (left) and after (right), where red stands for good view quality and green for bad.

Notice that starting from the second iteration, the planning should be "incremental", i.e., it ought to focus on selecting views so that regions that are poorly scanned or not scanned from previous iteration are "seen" well this time. Since a large portion of the object surface is well covered by planning from previous iteration, there is no need to repetitively scan those already nicely reconstructed regions. Objective wise, instead of maximizing the sum of view quality over all point samples, we maximize

the increase in view quality only over the points with low view quality. We generalize the view quality measurement $\mathcal{Q}$ from Equation 4.9 so that it becomes a function of the iteration number t, $(t = 1, 2, ...)$, corresponding to different poses in each iteration the object is re-oriented to:

$$\mathcal{Q}^{(t)}(P^{(t-1)}, V^{*(t)}) = \sum_{p \in \tilde{P}^{(t-1)}} q^{(t)}(p, \beta_1^{(t)}(p)) - q^{(t-1)}(p, \beta_1^{(0...t-1)}(p)) \tag{6.1}$$

$P^{(t-1)}$ denotes the set of oriented points sampled from the reconstructed surface model from previous iteration $t-1$. $\tilde{P}^{(t-1)} = \{p \ : \ p \in P^{(t-1)};$ and $q^{(t-1)}(p, \beta_1^{(0...t-1)}(p)) < \zeta\}$ is the set of low confidence points in $P^{(t-1)}$, where the best view quality of these points is lower than a threshold ζ. $V^{*(t)}$ denotes the set of selected views in iteration t. $q^{(t)}(p, \beta_1^{(t)}(p))$ is the best view quality for p achieved by the selected views $V^{*(t)}$ in the current iteration, whereas $q^{(t-1)}(p, \beta_1^{(0...t-1)}(p)$ is the best view quality for p achieved by $\bigcup_{\tau=0}^{t-1} V^{*(\tau)}$, views selected over all the previous iterations. Notice that when $t = 1$, this generalized formulation of $\mathcal{Q}$ degenerates to Equation 4.9, since initially all the point samples from the approximate model $P^{(0)}$ have zero view quality and $\tilde{P}^{(0)} = P^{(0)}$. Similarly, the generalization can apply to the view quality design with overlap-aware heuristics (Equation 4.11 and 4.14).

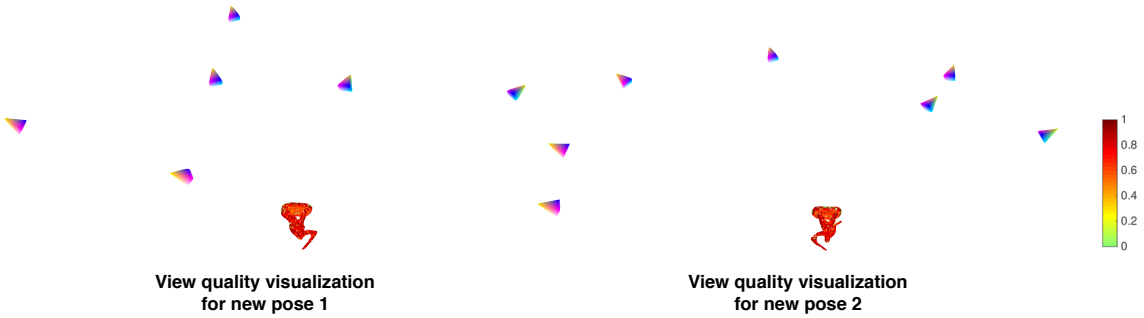


Figure 6.8: The view quality visualization on point sampled from the reconstructed model which is oriented into two new stable poses suggested by Algorithm 2, where red stands for good view quality and green for bad.

Figure 6.8 shows the selected view for the two suggested new poses, by optimizing the planning objective with the incremental view quality measurement in Equation 6.1 plugged in. As suggested in Figure 6.7 (right), the soldier stands upright in the first iteration of scanning, and therefore the bottom surface is entirely missing. This is captured by the planning in the subsequent iteration, where more views are placed to "look at" the bottom when the soldier lies down (Figure 6.8). Figure 6.9 shows that the quality of reconstruction is improved from iteration 1 to 2, where previously missing region as well as more detail are captured due to the re-orientation and incremental view planning in iteration 2. This suggests the potential of our proposed approach in refining the final reconstruction after even more iterations.

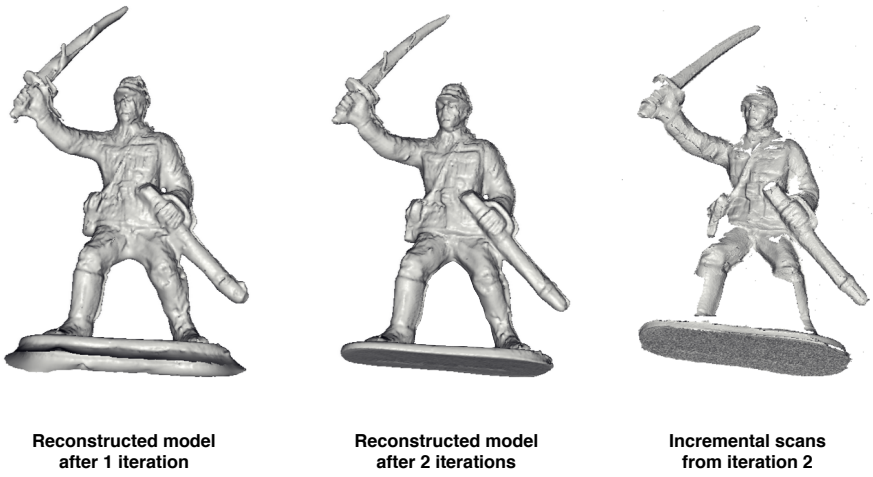


Figure 6.9: Side by side comparison among the reconstruction after 1 iteration (left), 2 iteration (middle), and the merged data acquired from iteration 2 (right).

Chapter 7

Conclusion, Discussion, and Future Work

"I open at the close."

J. K. Rowling

In this dissertation, we study the problem of practical 3D acquisition for large numbers of objects.

We propose a scalable prototype that automates the 3D acquisition of multiple objects with novel view and path planning algorithms, using a high-accuracy structured-light based scanner, and a calibrated motorized positioning system. Our system significantly reduces the per-object human interaction time associated with 3D acquisition, which should lead to the broader use of 3D scanning in a variety of fields.

We analyze the view and planning problem for multi-object 3D scanning. We propose an objective function in consideration of both accuracy and efficiency for the scanning. We study and compare different approaches to optimize for the objective function, and conclude that sequentially solving the view planning and the path planning problem is a reasonable strategy to adopt in practice. Jointly solving the

view selection problem over multiple objects can further expedite the process with higher planning quality. Jointly solving the view selection and path finding problems, on the other hand, can only achieve small increment in the objective, while requires more sophisticated optimization strategies to reduce computational time, before it can apply to a practical scenario.

Furthermore, we study different approaches to refine our multi-object 3D acquisition system. We propose two solutions, including improving the hardware design for the system, and refining the view planning by making it iterative. This exploration brings up new research topics, leading to fully automated high-quality 3D acquisition for large numbers of objects.

Generalization. Currently our system focuses on acquiring a surface geometry model. It would be interesting to generalize the view planning to support appearance acquisition as well. This would involve augmenting our current view quality function with a new term representing the expected response of a point sample to controlled illumination, which would evaluate whether a given view is also good for photometric capture.

Scan registration. Registration is always a required part of the post process in a standard acquisition pipeline. Scanning multiple objects at a time provides more global information for registering scans for the same scene, compared to scanning with a single object system. However, our current prototype requires flipping the objects to scan their under-sides, which in fact creates a new scene. There is no easy way of aligning the front side to the back side globally. The strategy we adopt now is to perform global alignment within the front side scene and the back side scene to obtain models integrated for both sides, and then to segment out each object to perform the back-to-front alignment independently. One possible future direction is

to explore global back-to-front registration algorithms that automatically account for the user interaction of flipping each fragment.

Hardware limitations. The quality of our initial scan alignments is limited by the precision of the scan-head's motor control. While the existing initial estimates of alignment are usually sufficient for automatic registration using ICP, inaccurate initial poses complicate both automatic segmentation and registration of flat (and otherwise underconstrained) objects such as fresco fragments. Adding encoders to the motors to precisely read off their positions would lead to greater robustness in post-processing.

Our prototype system is also limited in the motion ability of the scan head, since we only have three automatic degrees of freedom in our positioning system. Chapter 6 has discussed preliminary solutions for scanning objects with significant self-occlusion.

By using two scissor-jack lifting platforms, we have demonstrated the possibility of introducing an additional degree of freedom (vertical translation) with the system still calibrated, and we believe it would be simple to motorize the axes of the lifting platform. Because the view planning algorithm supports arbitrary scan-head motion, a more complex gantry design can use the same planner to scan a more diverse group of objects at once.

Planning Trade-off. While model based view planning and non-model based view planning appear to be the two extremes of the solution spectrum to the problem, our study on making the view planning iterative in Section 6.2 creates a bridge between the two ends. Given any unknown object to be scanned, model based approaches assume a priori knowledge about the object - this makes the problem easier to solve, but the quality of the final reconstructed model is limited by the prior model. The non-model based view planning, on the other hand, always looks for the next best view based on the scans captured so far, which does make it more accurate to decide

where to scan next, but the entire process tend to become much more time- and view- consuming. Our iterative approach is essentially doing a "next best set of views" planning. We believe this to be a rich field for future research to study the trade-off between the two categories of approaches.

Bibliography

[1] 3dr integrating with dji drones. https://3dr.com/blog/3dr-dji-enterprise-atlas/. Accessed: 2018-04-04.

[2] Artec3d. https://www.artec3d.com/. Accessed: 2018-03-12.

[3] Google street view. https://en.wikipedia.org/wiki/Google_Street_View#Data_capturing_equipment. Accessed: 2018-03-12.

[4] Edward H. Adelson and John Y. A. Wang. Single lens stereo with a plenoptic camera. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(2):99–106, February 1992.

[5] Sema Berkiten, Xinyi Fan, and Szymon Rusinkiewicz. Merge2-3D: Combining multiple normal maps with 3D surfaces. *Proc. Int. Conf. 3D Vision (3DV)*, pages 440–447, December 2014.

[6] Fausto Bernardini and Holly Rushmeier. The 3D model acquisition pipeline. *Computer Graphics Forum*, 21(2):149–172, 2002.

[7] Fausto Bernardini, Holly Rushmeier, Ioana M. Martin, Joshua Mittleman, and Gabriel Taubin. Building a digital model of Michelangelo's Florentine Pietà. *IEEE Computer Graphics and Applications*, 22:59–67, 2002.

[8] Kevin W. Bowyer and Charles R. Dyer. Aspect graphs: An introduction and survey of recent results. In *Proc. SPIE: Close-Range Photogrammetry Meets Machine Vision*, volume 1395, pages 200 – 208, 1990.

[9] Benedict J. Brown, Corey Toler-Franklin, Diego Nehab, Michael Burns, David Dobkin, Andreas Vlachopoulos, Christos Doumas, Szymon Rusinkiewicz, and Tim Weyrich. A system for high-volume acquisition and matching of fresco fragments: Reassembling Theran wall paintings. *ACM Trans. Graphics (Proc. SIGGRAPH)*, 27(3), 2008.

[10] Shengyong Chen, Youfu Li, and Ngai Ming Kwok. Active vision in robotic systems: A survey of recent developments. *Int. J. Robotics Research*, 30(11):1343–1377, 2011.

[11] Peng Cheng, James F. Keller, and Vijay Kumar. Time-optimal UAV trajectory planning for 3D urban structure coverage. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, pages 2750–2757, 2008.

[12] Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical Report 388, Graduate School of Industrial Administration, Carnegie Mellon University, 1976.

[13] William J Cook. *In Pursuit of the Traveling Salesman: Mathematics at the Limits of Computation*. Princeton University Press, 2011.

[14] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proc. ACM SIGGRAPH*, pages 303–312, 1996.

[15] Brian Lee Curless. *New Methods for Surface Reconstruction from Range Images*. PhD thesis, Stanford, CA, USA, 1998. UMI Order No. GAX98-10106.

[16] David H Douglas and Thomas K Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 10(2):112–122, 1973. doi:10.3138/FM57-6770-U75U-7727.

[17] B. Englot and F. Hover. Inspection planning for sensor coverage of 3D marine structures. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, pages 4412–4417, 2010.

[18] Xinyi Fan, Linguang Zhang, Benedict Brown, and Szymon Rusinkiewicz. Automated view and path planning for scalable multi-object 3D scanning. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 35(6), November 2016.

[19] Olivier Faugeras, Bernard Hotz, Herv Mathieu, Thierry Viville, Zhengyou Zhang, Pascal Fua, Eric Thron, Laurent Moll, Grard Berry, Jean Vuillemin, Patrice Bertin, and Catherine Proy. Real time correlation-based stereo: algorithm, implementations and applications. 1993.

[20] Uriel Feige. A threshold of Ln N for approximating set cover. *J. ACM*, 45(4):634–652, 1998.

[21] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[22] Hongbo Fu, Daniel Cohen-Or, Gideon Dror, and Alla Sheffer. Upright orientation of man-made objects. *ACM Trans. Graph.*, 27(3):42:1–42:7, August 2008.

[23] Jose-Joel Gonzalez-Barbosa, Teresa García-Ramírez, Joaquín Salas, Juan-Bautista Hurtado-Ramos, and José-de-Jesús Rico-Jiménez. Optimal camera placement for total coverage. In *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, pages 3672–3676, 2009.

[24] Inc. Gurobi Optimization. Gurobi optimizer reference manual. http://www.gurobi.com, 2015.

[25] Miles Hansard, Seungkyu Lee, Ouk Choi, and Radu Horaud. *Time-of-Flight Cameras: Principles, Methods and Applications*. Springer Publishing Company, Incorporated, 2012.

[26] R.J.(Ed.) Hocken and P.H.(Ed.) Pereira. *Coordinate Measuring Machines and Systems, Second Edition*. Manufacturing Engineering and Materials Processing. CRC Press, 2016.

[27] Larry Hornbeck. Digital Micromirror Device, US Patent No. 5,061,049, Inducted in 2009.

[28] Niels Joubert, Mike Roberts, Anh Truong, Floraine Berthouzoz, and Pat Hanrahan. An interactive tool for designing quadrotor camera shots. *ACM Trans. Graph.*, 34(6):238:1–238:11, 2015.

[29] Richard M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum, 1972.

[30] Michael Kazhdan and Hugues Hoppe. Screened Poisson surface reconstruction. *ACM Trans. Graph.*, 32(3):29:1–29:13, 2013.

[31] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[32] S. Kriegel, M. Brucker, Z. C. Marton, T. Bodenmller, and M. Suppa. Combining object modeling and recognition for active scene exploration. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, pages 2384–2391, 2013.

[33] Aldo Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Trans. PAMI*, 16(2):150–162, 1994.

[34] Eugene L Lawler, Jan Karel Lenstra, AHG Rinnooy Kan, and David B Shmoys. *The traveling salesman problem: a guided tour of combinatorial optimization*, volume 3. Wiley, 1985.

[35] Marc Levoy, Kari Pulli, Brian Curless, Szymon Rusinkiewicz, David Koller, Lucas Pereira, Matt Ginzton, Sean Anderson, James Davis, Jeremy Ginsberg, Jonathan Shade, and Duane Fulk. The Digital Michelangelo Project: 3D scanning of large statues. In *Proc. ACM SIGGRAPH*, pages 131–144, 2000.

[36] S. Lin and B. W. Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, 21(2):498–516, 1973.

[37] Y. Matsumoto, H. Terasaki, K. Sugimoto, and T. Arakawa. A portable three-dimensional digitizer. In *Proceedings. International Conference on Recent Advances in 3-D Digital Imaging and Modeling (Cat. No.97TB100134)*, pages 197–204, May 1997.

[38] Wojciech Matusik, Chris Buehler, Ramesh Raskar, Steven J. Gortler, and Leonard McMillan. Image-based visual hulls. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 369–374, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[39] Ren Ng. *Digital Light Field Photography*. PhD thesis, Stanford, CA, USA, 2006. AAI3219345.

[40] Edwin Olson. AprilTag: A robust and flexible visual fiducial system. In *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, pages 3400–3407, 2011.

[41] Besl P.J. Active optical range imaging sensors. *Sanz J.L.C. (eds) Advances in Machine Vision. Springer Series in Perception Engineering*, 1989.

[42] J.L Posdamer and M.D Altschuler. Surface measurement by space-encoded projected beam systems. *Computer Graphics and Image Processing*, 18(1):1 – 17, 1982.

[43] Mike Roberts and Pat Hanrahan. Generating dynamically feasible trajectories for quadrotor cameras. *ACM Transactions on Graphics (SIGGRAPH 2016)*, 35(4), 2016.

[44] Guziec A Rushmeier H., Taubin G. Applying shape from lighting variation to bump map capture. 1997.

[45] Szymon Rusinkiewicz, Olaf Hall-Holt, and Marc Levoy. Real-time 3D model acquisition. *ACM Trans. Graph.*, 21(3):438–446, 2002.

[46] Szymon Rusinkiewicz and Marc Levoy. Qsplat: A multiresolution point rendering system for large meshes. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 343–352, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[47] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the ICP algorithm. In *Proc. 3D Digital Imaging and Modeling (3DIM)*, pages 145–152, 2001.

[48] Joaquim Salvi, Jordi Pagès, and Joan Batlle. Pattern codification strategies in structured light systems. *Pattern Recognition*, 37:827–849, 2004.

[49] William R. Scott, Gerhard Roth, and Jean-François Rivest. View planning for automated three-dimensional object reconstruction and inspection. *ACM Computing Surveys*, 35(1):64–96, 2003.

[50] William R. Scott, William R. Scott Yz, Gerhard Roth, and Jean-Franis Rivest. View planning as a set covering problem. Technical Report 44892, NRC Canada, 2001.

[51] 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, June 2006.

[52] Michael Tao, Pratul Srinivasa, Sunil Hadap, Szymon Rusinkiewicz, Jitendra Malik, and Ravi Ramamoorthi. Shape estimation from shading, defocus, and correspondence using light-field angular coherence. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 39(3):546–560, March 2017.

[53] Glenn H. Tarbox and Susan N. Gottschlich. Planning for complete sensor coverage in inspection. *Computer Vision and Image Understanding*, 61(1):84–111, 1995.

[54] C.J. Taylor. Implementing high resolution structured light by exploiting projector blur. In *Proc. IEEE Workshop on Applications of Computer Vision (WACV)*, pages 9–16, 2012.

[55] The CGAL Project. *CGAL User and Reference Manual*. CGAL Editorial Board, 4.11.1 edition, 2018.

[56] R.Y. Tsai. An efficient and accurate camera calibration technique for 3d machine vision. *Proc. of Comp. Vis. Patt. Recog.*, pages 364–374, 1986.

[57] Jorge Urrutia. Art gallery and illumination problems. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*. Elsevier, 2000.

[58] Pere-Pau Vázquez, Miquel Feixas, Mateu Sbert, and Wolfgang Heidrich. Viewpoint selection using viewpoint entropy. In *Proc. Vision Modeling and Visualization (VMV)*, pages 273–280, 2001.

[59] Pengpeng Wang, R. Krishnamurti, and K. Gupta. View planning problem with combined view and traveling cost. In *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, pages 711–716, 2007.

[60] T. Weise, T. Wismer, B. Leibe, , and L. Van Gool. In-hand scanning with online loop closure. In *Proc. 3D Digital Imaging and Modeling (3DIM)*, 2009.

[61] Shihao Wu, Wei Sun, Pinxin Long, Hui Huang, Daniel Cohen-Or, Minglun Gong, Oliver Deussen, and Baoquan Chen. Quality-driven Poisson-guided autoscanning. *ACM Trans. Graph.*, 33(6):203:1–203:12, 2014.

[62] Kai Xu, Hui Huang, Yifei Shi, Hao Li, Pinxin Long, Jianong Caichen, Wei Sun, and Baoquan Chen. Autoscanning for coupled scene reconstruction and proactive object analysis. *ACM Trans. Graph.*, 34(6):177:1–177:14, 2015.

[63] Feilong Yan, Andrei Sharf, Wenzhen Lin, Hui Huang, and Baoquan Chen. Proactive 3D scanning of inaccessible parts. *ACM Trans. Graph.*, 33(4):157:1–157:8, 2014.

[64] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11):1330–1334, 2000.

[65] Jian Zhao, Ruriko Yoshida, Sen ching Samson Cheung, and David Haws. Approximate techniques in solving optimal camera placement problems. *Int. J. Distributed Sensor Networks*, (Article ID 241913), 2013.