# Reconstruction of 3D objects from multiple scanners

*Efstathios Kyriazis*

Thesis submitted in partial fulfillment of the requirements for the

*Masters' of Science degree in Computer Science and Engineering*

University of Crete
School of Sciences and Engineering
Computer Science Department
Voutes University Campus, 700 13 Heraklion, Crete, Greece

Thesis Advisor: Prof. *Constantine Stephanidis*

This work has been performed at the University of Crete, School of Sciences and Engineering, Computer Science Department.

The work has been supported by the Foundation for Research and Technology - Hellas (FORTH), Institute of Computer Science (ICS).

University of Crete
Computer Science Department

**Reconstruction of 3D objects from multiple scanners**

Thesis submitted by
**Efstathios Kyriazis**
in partial fulfillment of the requirements for the
Masters' of Science degree in Computer Science

THESIS APPROVAL

Author:

Efstathios Kyriazis

Committee approvals:

Constantine Stephanidis
Professor, Thesis Supervisor

Antonios Argyros
Professor, Committee Member

Xenophon Zabulis
Principal Researcher, Committee Member

Departmental approval:

Polyvios Pratikakis
Associate Professor, Director of Graduate Studies

Heraklion, November 2022

# Ευχαριστίες

*στην οικογένεια, τους φίλους μου, και όσους με στήριξαν αυτά τα 2 χρόνια*

## Acknowledgements

Firstly, I would like to thank my master thesis supervisor, Professor Constantine Stephanidis for his support throughout my MSc studies. I would also like to acknowledge Dr. Xenophon Zabulis for his invaluable assistance and guidance which rendered me capable of successfully completing my master thesis.

Moreover, I want to thank the Department of Computer Science (CSD) of the University of Crete, and the Human Computer Interaction (HCI) Laboratory of the Institute of Computer Science (ICS) of the Foundation for Research and Technology - Hellas (FORTH) for supporting me and providing me the means and data to accomplish my research goals.

Ultimately, I would like to express my deepest gratitude to my parents, Aristeidis and Maria, who gave me their love, support and advice in order to pursue my objectives throughout my master studies.

# Reconstruction of 3D objects from multiple scanners


## Abstract

Today, multiple scanning modalities exist, which have different properties when it comes to the reconstruction of objects with geometrical accuracy and photometric fidelity. Some scanning modalities are better at reconstructing the texture and others are better at reconstructing the 3D geometry of the scanned surfaces. Also, the practical need for a supporting surface for the placement of the object to be digitized prevents the creation of geometrically complete object reconstructions from a single scan.

In this thesis, we present a fully automatic method for creating high-quality, complete reconstructions of 3D objects through the combination of geometrical and texture information from multiple scanning modalities and two scanning postures of the object of interest.

The focus of this work is placed on exploiting the geometric accuracy of 3D scanners with the photometric fidelity of photogrammetric methods. Emphasis is placed on texture continuity as it is, usually, the main problem in the appreciation of 3D model quality. The proposed approach uses numerical optimization to improve texture continuity across photographic views and partial 3D scans of the object.

We test the proposed method in the common task of scanning small-sized objects on a conventional scan table. We qualitatively evaluate the results of the proposed method in two data sets that we created for this purpose. The obtained results demonstrate that the proposed method is a simple-to-use solution to create high-quality scans using conventional means. With the use of a prescribed setup, the proposed method is automatic and alleviates the need for manual post-processing of the obtained 3D models.

# Ανακατασκευή 3Δ αντικειμένων από πολλαπλούς σαρωτές

## Περίληψη

Στις μέρες μας, υπάρχουν πολλαπλές μέθοδοι ψηφιοποίησης αντικειμένων οι οποίες παρουσιάζουν διαφορετικές ιδιότητες όσον αφορά τη γεωμετρική τους ακρίβεια και τη φωτομετρική τους πιστότητα. Ορισμένες τέτοιες μέθοδοι είναι καλύτερες στην ανακατασκευή της οπτικής υφής ενώ άλλες είναι καλύτερες στην ανακατασκευή της 3Δ γεωμετρίας των ψηφιοποιούμενων επιφανειών. Επιπροσθέτως, η πρακτική ανάγκη μιας υποστηρικτικής επιφάνειας για την τοποθέτηση του ψηφιοποιούμενου αντικείμενου, αποτρέπει την δημιουργία ολοκληρωμένων γεωμετρικά ανακατασκευών από μια μοναδική διαδικασία σάρωσης.

Σε αυτήν την εργασία, παρουσιάζουμε μια πλήρως αυτοματοποιήμενη μέθοδο κατασκευής υψηλής ποιότητας-ολοκληρωμένων 3Δ ανακατασκευών, συνδυάζοντας γεωμετρική πληροφορία και οπτική υφή από πολλαπλές σαρώσεις του αντικειμένου σε δύο διαφορετικές πόζες.

Η δουλειά μας εστιάζει στην αξιοποίηση της γεωμετρική ακρίβειας των 3Δ σαρωτών και της φωτομετρικής πιστότητας των μεθόδων φωτογραμμετριας. Δίνεται έμφαση στην συνέχεια της οπτικής υφής η οποία, συνήθως, αποτελεί το κύριο πρόβλημα στην εκτίμηση των 3Δ μοντέλων. Η προσέγγιση μας χρησιμοποίει αριθμητικές βελτιστοποιήσεις με σκοπό να βελτιώσει τη συνέχεια των φωτογραφικών όψεων και των μερικών 3Δ ψηφιοποιήσεων του αντικειμένου.

Ελέγχουμε τη μέθοδο μας στην κοινη διαδικασία ψηφιοποίησης μικρού μεγέθους αντικειμένων πάνω σε ένα συμβατικό τραπέζι ψηφιοποίησης. Αξιολογούμε ποιοτικά τα αποτελέσματα της μεθόδου μας σε δύο data sets τα οποία δημιουργήθηκαν γι' αυτό το σκοπό. Τα αποτελέσματά μας δείχνουν ότι η μέθοδος μας αποτελεί μια απλή λύση η οποία δημιουργεί υψηλής ποιότητας ανακατασκευές με τη χρήση συμβατικών μέσων. Μέσω μιας προκαθορισμένης διάταξης, η μέθοδος μας είναι πλήρως αυτόματη, περιορίζοντας την ανάγκη χειροκίνητης, μετέπειτα επεξεργασίας των παραγόμενων 3Δ μοντέλων.

# Contents

**Table of Contents** iii

**List of Tables** v

**List of Figures** viii

**1 Introduction** 1
- 1.1 Notation and definitions . . . 3
- 1.2 Abbreviations . . . 4

**2 Related work** 5
- 2.1 Engineering approaches . . . 5
- 2.2 View selection . . . 6
- 2.3 Seam concealment . . . 6
- 2.4 Known geometries . . . 7
- 2.5 Texture manipulation . . . 7
- 2.6 Complete scans . . . 7
- 2.7 This work . . . 8

**3 Method overview** 9

**4 Data acquisition** 11
- 4.1 Use of markers . . . 11
- 4.2 Digitisation environment and setup . . . 11
  - 4.2.1 Top view setup . . . 12
  - 4.2.2 Bottom view setup . . . 13
- 4.3 Acquisition of images . . . 14
- 4.4 Acquisition of 3D scans . . . 15
- 4.5 Outcome . . . 16

**5 Marker localization and floor removal** 17
- 5.1 Locating markers in photogrammetric reconstructions . . . 17
- 5.2 Locating markers in 3D scans . . . 18
- 5.3 Floor removal . . . 20

5.4 Outcome . . . . . 21

**6 Registration of 3D models** **23**
6.1 Registration of 3D scans with photogrammetric reconstructions . . 23
6.1.1 Registration methodology . . . . . 23
6.1.2 Accuracy issues . . . . . 25
6.2 Registration of partial 3D scans . . . . . 25
6.2.1 Registration methodology . . . . . 25
6.2.2 Accuracy issues . . . . . 27
6.3 Outcome . . . . . 27

**7 Creating a complete object model** **29**
7.1 Merging of 3D scans . . . . . 29
7.2 Associating M with the photogrammetric meshes . . . . . 30
7.3 Limitations in reconstruction M . . . . . 30
7.4 Outcome . . . . . 31

**8 Registration refinement** **33**
8.1 Motivation . . . . . 33
8.2 Rationale . . . . . 33
8.3 Optimization parameters . . . . . 34
8.4 Domain of parameters . . . . . 34
8.5 Hypothesis representation . . . . . 35
8.5.1 Scaling . . . . . 35
8.5.2 Rotation . . . . . 35
8.5.3 Translation . . . . . 36
8.5.4 Composite transform . . . . . 36
8.6 Objective function . . . . . 36
8.7 Computational improvements . . . . . 37
8.7.1 Computational cost . . . . . 37
8.7.2 Convergence robustness . . . . . 38
8.7.3 Refined registration error . . . . . 39
8.8 Outcome . . . . . 39

**9 Texture mapping** **41**
9.1 Assessing triangle visibility in photogrammetric views . . . . . 41
9.2 Texture source selection . . . . . 42
9.2.1 Normalized triangle angle . . . . . 42
9.2.2 Normalized triangle distance . . . . . 43
9.2.3 Mapping criterion . . . . . 43
9.3 Triangle texturing . . . . . 44
9.4 Outcome . . . . . 44

**10 Appearance optimization** **45**
10.1 Compacting texture clusters . . . 45
10.1.1 Smoothing border shape between texture clusters . . . 46
10.1.2 Discarding small area texture clusters . . . 47
10.1.3 Outcome . . . 48
10.2 The transition from top to bottom view setups . . . 48
10.2.1 Motivation . . . 49
10.2.2 Rationale . . . 49
10.2.3 Optimization parameters . . . 50
10.2.4 Domain of parameters . . . 50
10.2.5 Evaluating texture continuity in a texture cluster pair . . . 51
10.2.6 Quantifying the visual information of cluster pairs . . . 52
10.2.7 Objective function . . . 52
10.2.8 Outcome . . . 53
10.3 Texture blending . . . 53
10.3.1 Blending procedure . . . 53
10.3.2 Blending weight . . . 55
10.3.3 Outcome . . . 56
10.4 Outcome . . . 56

**11 Experimental evaluation** **57**
11.1 Dataset description . . . 57
11.1.1 Digitisation modalities . . . 57
11.1.2 Dataset information . . . 57
11.2 Performance evaluation . . . 60
11.2.1 Computation parameters . . . 61
11.2.2 Computational cost . . . 61
11.2.3 Discussion . . . 62
11.3 Qualitative evaluation . . . 63
11.3.1 Comparative results . . . 63
11.3.2 Results and discussion . . . 64

**12 Discussion and future work** **73**
12.1 Discussion . . . 73
12.2 Future work . . . 73
12.2.1 Quantitative evaluation . . . 73
12.2.2 Higher end scanners . . . 74
12.2.3 Multiple partial scans . . . 74

**Bibliography** **75**

# List of Tables

11.1 The 3D scan models details in both datasets. . . . . 60
11.2 The photogrammetric models details in both datasets. . . . . 60
11.3 The information of images captured for the photogrammetry for each dataset. . . . . 60
11.4 The execution time of each module for both datasets. . . . . 61
11.5 The approximate maximum memory used by each module for both datasets. . . . . 62

# List of Figures


1.1 Small inaccuracies in the application of texture give rise to well-pronounced visual effects. . . . . 2

3.1 An illustration of our method's steps as a flow chart. . . . . 10

4.1 The STag markers that we used. . . . . 12
4.2 An indicative appearance of the floor . . . . . 12
4.3 An indicative top view scene's layout. . . . . 13
4.4 An indicative bottom view scene's layout. . . . . 14
4.5 The described photo coverage of the object. . . . . 15

5.1 Estimating the 3D position of a marker in a photogrammetric mesh through the barycentric coordinates. . . . . 18
5.2 Estimating the 3D position of a marker in a 3D scan through the barycentric coordinates. . . . . 19
5.3 An indicative illustration of the selected triangles by the distance threshold (marked in red). The areas of the reconstruction that we not selected by the distance threshold are marked in blue. . . . . 21
5.4 An indicative illustration of the selected triangles by the angle threshold (marked in red). The areas of the reconstruction that we not selected by the angle threshold are marked in blue. . . . . 22
5.5 An indicative illustration of the filtering performed by the distance and angle threshold combined (left). The removal of outlier floor triangles (right). . . . . 22

6.1 The transforms $Q_t$ and $Q_b$ associate each 3D scan with the photogrammetric reconstruction of the same view. . . . . 24
6.2 The $P_{ceil}$ plane. . . . . 26
6.3 The projection of $\mathbf{b}_i$ points on $P_{ceil}$. . . . . 27
6.4 The transformations Q that interconnect the two views . . . . . 28

7.1 The cut performed by $P_{cut}$ plane. . . . . 30
7.2 The transformations $T_t$ and $T_b$ interconnect $M$ with the two photogrammetric meshes. . . . . 31

8.1 A template mask and the projection masks of two hypotheses $\mathbf{v}_1$ and $\mathbf{v}_2$. The mismatches between the template and the two hypotheses are highlighted with red and green color. . . . . 37
8.2 A cropped template mask. . . . . 38
8.3 A visualization of a cost mask. The red line indicates the border between the foreground and background pixels of the template mask. 40

10.1 An indicative image of texture clusters created from the texture mapping process. Each cluster is highlighted with a different color. 46
10.2 The border length between $C_\alpha$ and $C_\beta$ reduces after the reassignment of triangle $f_b$ to cluster $C_\alpha$. . . . . 47
10.3 Small clusters appearing between larger clusters. . . . . 48
10.4 The mapping of $S_k$ triangles to images $I_R$ and $I_S$. . . . . 50
10.5 The mapping of triangle $t$ to images $I_R$ and $I_S$. $\hat{\mathbf{p}}$ is estimated through the barycentric coordinates $\mathbf{b}$. . . . . 51
10.6 The triangle subset $S$ in cluster $C_1$. . . . . 54
10.7 The mapping of $t$ to images $I_1$ and $I_2$. $\hat{\mathbf{p}}$ is estimated through the barycentric coordinates $\mathbf{b}$. . . . . 54
10.8 The blending radius $R$ and the distance of a pixel $\mathbf{p}$ from the border. 55

11.1 The object captured in dataset 1 from different viewing perspectives. 58
11.2 The object captured in dataset 2 from different viewing perspectives. 59
11.3 A comparison of the projected $M$'s vertices on a photogrammetric image with the use of the initial and refined transforms $T_t$ and $T_b$. 65
11.4 A comparison of the texture mapping quality between the initial and the refined transforms $T_t$ and $T_b$. . . . . 66
11.5 A comparison between the initial and the compacted texture clusters 67
11.6 A comparison of texture continuity between neighboring top and bottom view clusters. between initial and refined alignment . . . . 68
11.7 A comparison between the original and blended texture. . . . . 68
11.8 The reconstructions of the two objects created by our method. The left column shows the ceramic teacup reconstruction. The right column shows the ceramic water cup reconstruction. . . . . 69
11.9 The difference at the thickness of the brims of the two cups. . . . . 70
11.10 The handle of teacup as reconstructed from the two 3D scans. . . . 71

# Chapter 1

# Introduction

This work targets the quality increment of object 3D reconstruction, in two ways. First, by increasing the completeness of the reconstruction, hindered by occlusions. Second, by combining modalities of different geometric and photometric accuracy, to get the best of each one.

These two problems are encountered systematically on the setup used to reconstruct a 3D object with high fidelity to its geometric structure and photometric appearance.

The first problem is encountered when placing the object of interest on a scan table, let us say a teacup. In this placement, its contact ("bottom") surface with the table cannot be imaged. Besides this, there are other regions of the cup that are practically difficult to image, such as regions below and at the inner side of the handle. Solutions including transparent tables hinder reconstruction accuracy, due to diffraction and color aberrations of the transparent medium. These solutions, as well as those that include hanging the object to increase digitization coverage, exhibit significant practical and safety problems, both for the artifact and the operator. This work investigates the possibility of placing the objects in multiple poses on the scan table, producing multiple scans, and combining them. The simplest case, that of two poses (the cup facing "upwards" and "downwards"), is considered in this work.

The second problem is encountered when choosing digitization modalities. Today, multiple scanning modalities exist, which have different properties when it comes to geometrical accuracy and photometric fidelity. The proliferation of CCD cameras has democratized photogrammetric and photorealistic reconstruction. Computational advances in SLAM automated the processes of systematically taking pictures and producing 3D models from them, without manual intervention. Nowadays, photogrammetry is aided by active illumination and SLAM takes advantage of accelerometers attached to the camera, or set of cameras. On the other hand, direct measurements provided by ToF sensors (i.e., laser scanners) provide more accurate geometrical reconstructions than photogrammetry, particularly in textureless surface regions. However, their photometric fidelity (texture quality)

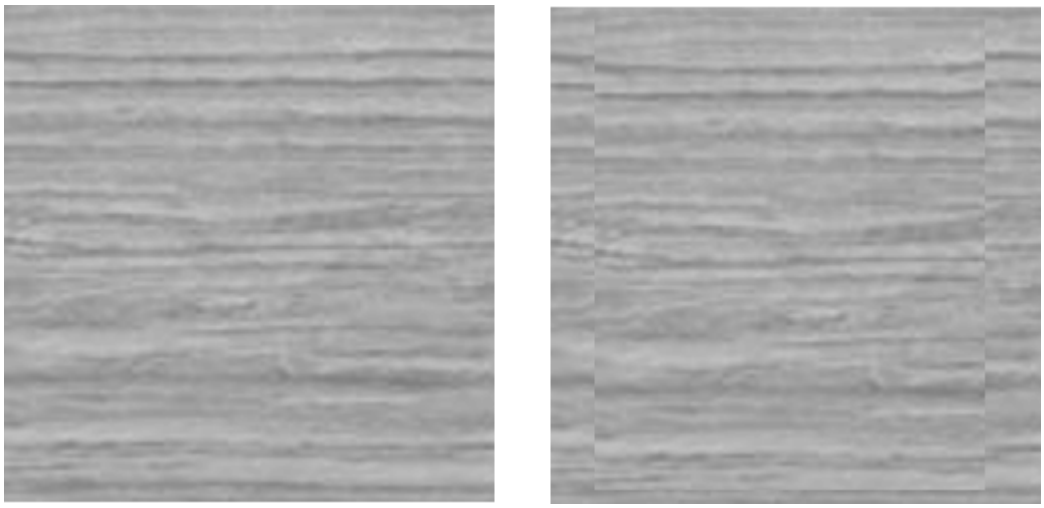


Figure 1.1: Small inaccuracies in the application of texture give rise to well-pronounced visual effects.

is lower than in photogrammetry. Moreover, in photogrammetry, one can directly use photographic filters (e.g., polariser) or multispectral imaging, to better capture the reflectance properties of the scanned surface.

The inaccuracies in the registration of photogrammetric "views" or partial scans are usually called "seams". The reason is the characteristic manifestation of these inaccuracies on the surface of the textured reconstruction and in particular at the boundary of regions textured from different images or scans. In the region of the boundary, discontinuities of texture are noticed. The human visual system is particularly apt in detecting and localizing such discontinuities. One can say it is even overzealous in doing so, given the illusionary perception of semi-formed contours as wholes (see [PM87] for a review), as in our case. As such, even small inaccuracies in texture mapping can lead to a dramatic loss of the appreciation, or quality, of the result. In Figure 1.1, this situation is illustrated by vertically displacing a texture segment by two pixels. This problem is similar to the same problem of stitching multiple images in mosaics, where care must be taken so that images are combined in a way to avoid seams that occur for the same reasons as above.

The inaccuracies due to errors in the reconstruction of object geometry lead to distorted texture appearance. Usually, such errors are less noticeable because if both structure and texture are of high resolution, the distortions are small. Moreover, shadows encoded in the texture compensate for structural errors, as the perception of shape from shading is a visual cue exploited by the human visual system. Nevertheless, reconstruction is not useful only for viewing and appreciating the reconstructed objects, but also for conserving them. As such, even though that such errors are less noticeable, they can be equally or more important, depending on the application.

The contributions of this work are the following.

1. Provides a method to create full 3D scans of objects digitized on a scan table, by combining two partial scans obtained by turning the object upside down on the scan table.

2. Provides a method to transfer the texture obtained from a photogrammetric reconstruction of an object to the geometry obtained from an active illumination or laser scan of the same object.

This work combines both methods to create high-quality scans of objects with (approximately) Lambertian (matte) surfaces.

## 1.1 Notation and definitions

In this work, the following concepts and notations are used.

**3D scanner**: We call a 3D scanner a device that uses active illumination or time-of-flight sensing to reconstruct (or scan) the 3D surfaces of a scene.

**3D triangle mesh**: A 3D triangle mesh is a set of connected 3D triangles which represent the geometry of a 3D object. In a textured mesh, each 3D triangle is mapped to a corresponding, 2D triangle. This 2D triangle contains the surface texture of the 3D triangle. The 2D triangles are usually stored in a "texture image" and encoded in the so-called, UV coordinate system.

**Markers**: A central point in this work is the use of markers. Four markers are placed on the scanning table prior to and throughout all imaging and scanning operations. The locations of these markers are constant.

**Photogrammetry procedure**: During photogrammetry, the scene is imaged from multiple views, indexed by $j$. At each view, an image $I_j$ is acquired. Each view $j$ is associated with a $3 \times 4$ projection matrix $P_j$. Thus, a 3D world point $\mathbf{p}$, represented in homogeneous coordinates, is projected in each $I_j$ as $\mathbf{u} = P_j \cdot \mathbf{p}$. Therefore, the image intensity at image location $\mathbf{u}$, in homogeneous coordinates, is given by the following equation $\mathbf{u} = I(P_j \cdot \mathbf{p})$.

**Reconstructions**: Both photogrammetry and 3D scanners provide one textured mesh (triangle mesh) per scanning operation. We refer to the meshes produced by 3D scanners as "3D scans". All scans cover the object and the scanning table with the placed markers. All scans are in their own coordinate system. Photogrammetric scans are up to scale, while scans obtained from 3D scanners are metric. All coordinate frames are related by a rotation and a translation transform. Photogrammetric scans also need a scaling transformation.

**Triangle barycentric coordinates**: In geometry, a barycentric coordinate system is a coordinate system in which the position of a point located in a triangle is specified by reference to the triangle's vertices. Barycentric coordinates can be used to express the position of any point located in the triangle with 3D vector. Let $t$ be a triangle and $\mathbf{p}$ a point in $t$. We denote the calculation of the barycentric coordinates $\mathbf{b}$ of point $\mathbf{p}$ with respect to $t$ as $\mathbf{b} = t(\mathbf{p})$. We also denote the

calculation of the point $\mathbf{p}$ by the application of barycentric coordinates $\mathbf{b}$ to triangle $t$ as $\mathbf{p} = t(\mathbf{b})$.

## 1.2 Abbreviations

Particle Swarm Optimization (PSO), Iterative Closest Point (ICP), Time of Flight (ToF), Degree Of Freedom (DOF), Digital Single-Lens Reflex camera (DSLR), Couple-Charged Device (CCD), Simultaneous Localization And Mapping (SLAM).

# Chapter 2

# Related work

In this section, we review works that we found of relevance and help in our work. A recent review of the quality of texture mapping in SfM-based approaches can be found in [WMG14]. The review below extends into more digitization modalities and methods.

## 2.1 Engineering approaches

Early approaches capitalized on engineering accuracy and meticulous calibration. In [NB95; MYA03], such an approach was proposed that employed a turntable-based photogrammetric 3D reconstruction method, which is more accurate than arbitrary-view photogrammetry. The increased accuracy was combined with a well-calibrated camera to capture the color appearance and the partial results were combined. Besides the significant and tedious labour involved in such approaches, another disadvantage is that they are susceptible to human measurement or mechanical errors, in the calibration process. Typically, such methods do exhibit noticeable errors and have been surpassed by more recent ones.

More recently the problem was re-encountered in the combination of the geometrical structure obtained from ToF scanners with textures acquired from photographic sensors [WJP08; DY18; GFG12]. Although modern ToF scanners include photographic sensors, these are of inferior quality compared to dedicated photographic sensors (i.e., DSLR). Also, these are not optically configurable to potential lighting requirements and reflectance properties of the digitization target; i.e., in DSLR photography illumination and optical filters are conventionally used to obtain images of better quality. As such, texture mapping from images to ToF scans is still a common task where structural and photographic accuracy are both important, i.e. in the reconstruction of cultural heritage objects and monuments [Sta+19; Del+19].

## 2.2 View selection

The problem of selecting a texture source when "painting" a 3D model is encountered in a wide variety of tasks in the 3D digitization of surfaces. The works in [Gal+10; GDDA13; LI07; VS07], select a "best" for each triangle, based on the least angle criterion.

In this work, this criterion is entertained through the concept of "visibility". Furthermore, this work strives to gather information from all appearances of a physical point in the photographic data. In this work, texture maps are created for each view at the overlapping regions, after a local texture registration operation.

## 2.3 Seam concealment

Another avenue of research focuses on making seams less visible. Most approaches "blend" the transition from one texture source to another, in a direction perpendicular to their border. This strategy is analogous to the blending of images at their bordering regions when stitching them in a mosaic, e.g. [Gra+09]. In these cases, it is more effective because image mosaics require fewer DOFs in the registration of images and, in these cases, texture mismatches can be "covered up" using this technique.

Several works follow an approach utilized in the synthesis of image mosaics, which is to "blend" textures from different sources at the region of their border (seam). The simplest approach is followed, which is to weigh the influence of each texture source, proportionally to their distance from the border, e.g., [Roc+99; Roc+99; Cal+08]. Even in this case, linear blending is surpassed by the method in [BA83], which was originally used for image mosaics. In [Che+12], this method is used to reduce seam appearance.

Like this work, more recent works in image mosaics have identified the need for refining the alignment of images by global and local transforms to compensate for inaccuracies in camera pose estimation [SS02]. Though additional corrections were introduced by more recent methods (see below), the technique of blending borders (in one way or another) is common practice in all descendant methods to reduce the salience of residual registration errors.

The work in [LI07], accepts texture compatibility inaccuracies as unavoidable and strives to make them less visible. It differs from the methods above because it does more than not smooth texture near borders. Instead, it renegotiates borders so that they pass through homogeneous regions, which are less noticeable. In this way, texture inconsistencies are less harmful to the appreciation of the observer. Technically, the method is guided by image gradient and leads border redefinition towards regions where the magnitude of the image gradient is low.

In [Des+14], the aforementioned approach was improved by reducing the overall length of seems by a "geodesically growing" mesh segmentation that creates maps with as many neighboring triangles as possible. The proposed work is similar to

this one, in that it follows the principle of texturing as many as possible mesh triangles from the same texture source. We, however, extend this idea into also creating geodesically smooth borders between neighboring groups of triangles that receive texture from different sources.

## 2.4 Known geometries

In [Pag+15], color seams are suppressed by texture blending upon an approximate reconstruction of the 3D geometry of an object. The approximation of the object is coarse because it is provided by the visual hull reconstruction of the object. Due to the additional errors introduced by this coarse approximation of 3D geometry, seams have a spatially larger expression. The work in [Hua+20] is assisted by depth measurements of an RGB-D sensor and its corresponding depth measurements to better estimate the relative pose of RGB images to an object of known geometry.

The work in [Roc+99], relies on prior knowledge of the 3D geometry of the object and very accurate calibration of texture views. A "best" view is selected for each triangle, based on the least angle criterion. Texture maps are created for each view at the overlapping regions, after a local texture registration operation. The work in [GC09], extends this idea to map regular images onto the model and create superresolution texture maps.

## 2.5 Texture manipulation

Some methods use local texture transformations whose target is to align textures at bordering regions. In [Gal+10], a global optimization is used to generate compatible textures for adjacent triangles. The method searches over a set of local image transformations to compensate for geometric misalignment. The work in [Hua+20] trains an adversarial network, to generate sharp texture from misaligned images and synthesizes a new texture map.

## 2.6 Complete scans

A topic that is not very often encountered in the literature is the completeness of reconstruction. This problem is usually met in movable monuments and objects which can be potentially viewed from any viewpoint. Conventionally the task includes the combination of multiple partial reconstructions obtained by modulating the placement of the object so that every region of its surface is imaged, e.g. [Kan+22]. The combination of partial scans is usually manual and assisted by markers as well as the careful placement of the object. Such methods are extended in the proposed work by increasing the automation of combining partial scans.

## 2.7 This work

This work is based on engineering methods that use markers to robustly register models of different geometric and texture accuracy and keep the most accurate properties from each model. However, to increase the accuracy of matching a 3D scan and a 3D photogrammetric model, the proposed work refines the registration transfers that link 3D geometry with the photographs from which texture will be sampled (or collected). This refinement is based on the optimization of the compatibility of geometric appearance as indicated by the object's silhouette in images.

Moreover, this work uses markers and geometrical registration to increase automation in the combination of partial scans. The inclusion of a preliminary geometrical registration has the benefit of allowing for some affinity in object placement, thus simplifying the digitization setup.

To improve seams created in the combination of partial views this work optimizes texture continuity not by locally smearing or distorting textures but by refining the registration transforms that link partial views. To alleviate the presence of seams in the 3D reconstruction, a renegotiation of triangles among texture clusters is also proposed.

A representation quality that this work shares with [AFB19] is the per triangle point-based representation of intensities using barycentric coordinates. This work uses such a representation, in order to compare visual appearance in 3D rather than in the 3D texture map.

# Chapter 3

# Method overview

The reconstruction method we propose is composed of seven sequential steps. A brief description of these steps follows. An illustration of the method's steps is shown in Figure 3.1.

1. **Data acquisition**. In Chapter 4, we present guidelines for the collection of data. The setup of the scanning environment includes the use of markers. The guidelines refer to the placement of markers and the systematic acquisition of images and 3D scans, in a way that is compatible with the proposed method.

2. **Data preprocessing**. Chapter 5 deals with the 3D localization of markers and the effective removal of the "floor" upon which the object is placed for scanning in the scanning environment.

3. **Model registration**. In Chapter 6, we propose two procedures for the registration of 3D models, in the context of the proposed method. The first regards the registration of 3D scans with 3D photogrammetric models of the same scene. The second regards the registration of partial 3D scans of an object.

4. **3D geometry**. In Chapter 7 we create a complete model of our object by combining partial 3D scans.

5. **Registration refinement**. In Chapter 8, we propose an optimization framework for refining the accuracy of the registrations that we calculated in Chapter 6, thereby, enhancing reconstruction accuracy.

6. **Texture mapping**. In Chapter 9, we propose a criterion for mapping 3D triangles to photogrammetric images. This criterion increases the visual quality of the result, by selecting the view(s) from which the texture of each triangle of the reconstruction should be sampled.

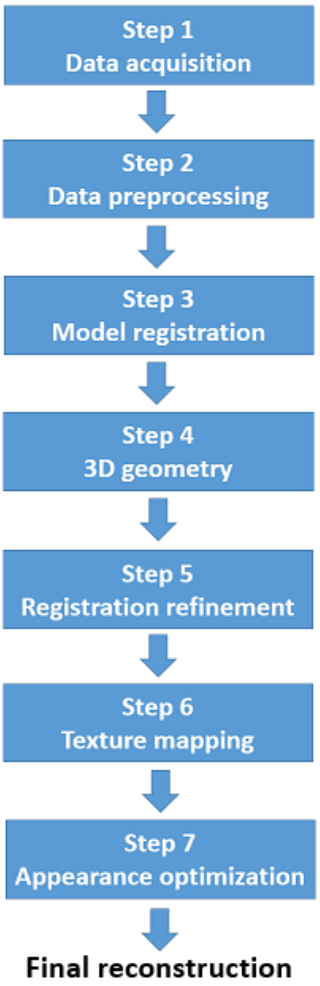



Figure 3.1: An illustration of our method's steps as a flow chart.

7. **Appearance optimization**. In Chapter 10, we present three additional procedures that contribute to the improvement of the overall texture quality of our object reconstruction.

# Chapter 4

# Data acquisition

As a first step, we need to acquire a working dataset that will be processed through the multiple stages of our reconstruction method. The dataset should include two photogrammetric reconstructions and two 3D scans of the input object. In the current section, we present some general guidelines for the effective capturing of the 3D models so that we create a compatible dataset for our method. We also present some good practices concerning the image capturing of the object so that we achieve a good coverage of its surfaces.

## 4.1 Use of markers

We first select some markers which will be set on the scene along with the object. These markers will help us with the registration of the resulting 3D models in the model registration phase (see Chapter 6). We use four markers. Each marker has to be unique and easily distinguishable from the others. The markers must be accompanied by a detection algorithm that should be able to track the centers of the markers in a given image and return their 2D image coordinates. We refer to the selected markers' entities as $\mathcal{M}_i$ and to the 3D coordinates of their centers on the scene as $\mathbf{m}_i$, where $i \in [0, 3]$ and denotes the index of the marker.

In this work, we used the STag markers [BTA19]. However, our method is transparent to the selection of markers and, thus, other markers can be also employed provided that they meet the previously stated requirements (e.g., QR codes).

## 4.2 Digitisation environment and setup

For capturing, it is necessary that we select a solid planar surface on which we can place the object with stability. This surface, or scan table, will constitute a part of our scene along with the object and the markers. The ideal surface should be smooth and horizontal (i.e. not inclined), allowing the object to be steadily set on its top. The area of the surface has to be adequately large to fit the object

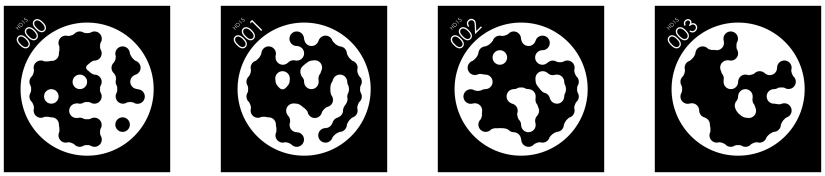



Figure 4.1: The STag markers that we used.

and the four markers without their placement resulting in occlusions. It is highly recommended that the scan table's surface has a rich visual texture (i.e. contains much visual information) as this attribute will significantly help the photogrammetry algorithm to produce high-quality reconstructions. For simplicity, we refer to the scan table's surface as the "floor". An indicative appearance of the floor is shown in Figure 4.2.

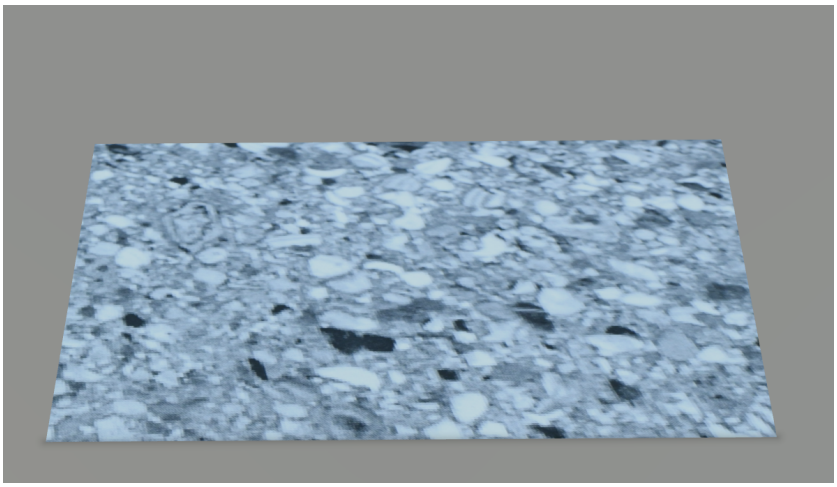


Figure 4.2: An indicative appearance of the floor

### 4.2.1 Top view setup

Initially, we have to organize the scene for the capturing of the top view of the object. We begin the setup with the placement of the object at the center of the floor in a standing position. The four selected markers should be set on the floor perimetrically of the object, in a way that their centers $\mathbf{m}_i$ constitute the vertices of a hypothetical rectangle. The markers' positions should be properly adjusted so that the object lies at the center of the formed rectangle. Each pair of consecutively indexed markers $(\mathbf{m}_i, \mathbf{m}_{i+1})$ represents an edge of the rectangle. Thus, the edges $(\mathbf{m}_0, \mathbf{m}_1)$, $(\mathbf{m}_2, \mathbf{m}_3)$ and $(\mathbf{m}_1, \mathbf{m}_2)$, $(\mathbf{m}_3, \mathbf{m}_0)$ should be parallel and equally-sized. The rectangle's size is not strictly defined however, it is recommended that the markers are placed in a relatively close proximity with regard to the object.

In this arrangement (formed by the markers and the object), we can begin the capturing of the top view of our object both as a 3D scan and as a photogrammetric reconstruction. Even though the described setup is an important requirement for our method's function, there is no need for extremely high precision concerning the placement of the object and the markers on the scene. An indicative layout of the top view scene is shown in Figure 4.3.

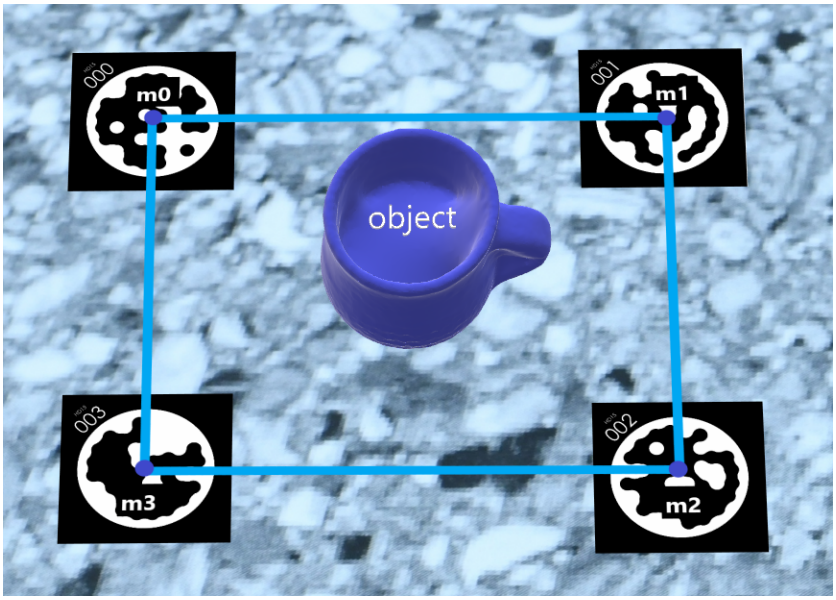



Figure 4.3: An indicative top view scene's layout.

### 4.2.2 Bottom view setup

After the completion of the top view capturing, we should continue with the capturing of the bottom view of the object. This extra procedure takes place so that we cover the areas that were previously occluded by the floor in the object's top-view pose.

The object is already placed on the scene at the center of the markers' defined rectangle. Therefore, we have to rotate it upside down (i.e. 180°) so that the bottom is now visible. This rotation should not be random but has to be performed in reference to one of the axes defined by the edges of our hypothetical rectangle. As a convention, we select to perform the rotation around the axis defined by the edge $(\mathbf{m}_1, \mathbf{m}_2)$. Despite its rotation, the object must preserve its position at the rectangle's center when it is set back on the floor. Throughout the whole procedure the markers must maintain the same position on the floor.

In this scene configuration, we can proceed to the capturing of the bottom view of our object as a 3D scan and as a photogrammetry reconstruction. Equally with the top view setup, the precision of the described object displacement should not

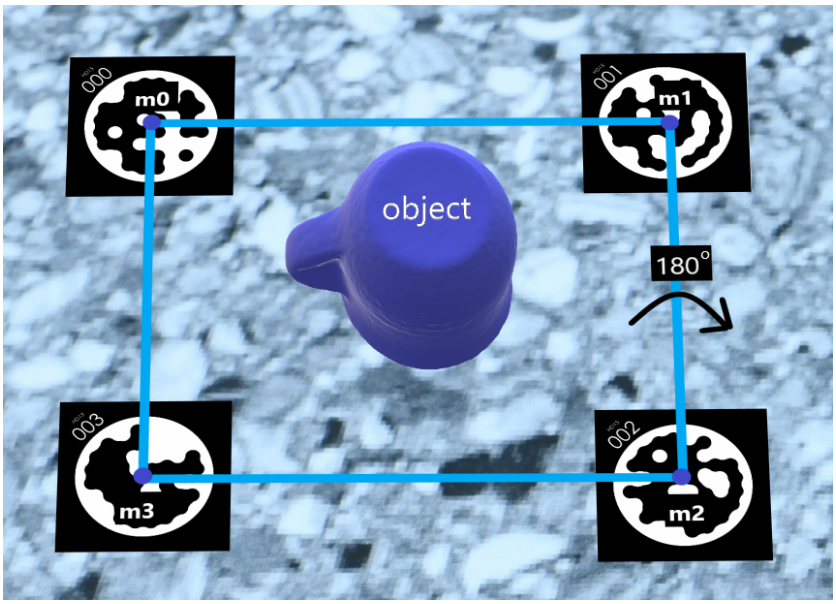



Figure 4.4: An indicative bottom view scene's layout.

necessarily be too strict. An indicative layout of the bottom view scene is shown in Figure 4.4.

## 4.3 Acquisition of images

Theoretically speaking, the creation of a photogrammetric reconstruction of a scene requires the taking of a series of overlapping images $I_j$ that depict the scene from various viewpoints. The overlapping between neighboring images will help the photogrammetry algorithm to accurately capture the geometric details of the object on the created reconstruction. Thus, it is necessary that we provide images that offer complete coverage of the scene while maintaining adequately sufficient overlap between them ($> 50\%$). For the effective imaging of the scene, it is recommended that we split the photography procedure into two capturing height levels: a) an intermediate and b) a top height level.

For the capturing of the intermediate level photographs, the camera should be placed in a slightly elevated position with respect to the scene, so that the center of focus is the middle regions of the object and to a lesser degree the object's top part. The viewing direction of the camera should be set in a way that favors the visibility of object areas that will be less distinct in the opposite object's placement (owing to the rotation). Our intention is to accurately capture the entire scene's area including the object and the markers lying on the floor. So, we have to adjust the camera's distance from the scene accordingly so that a significant portion of the floor is also covered in each image's view. In this camera configuration, we should start taking photos in a circle around the object until we reach again the starting

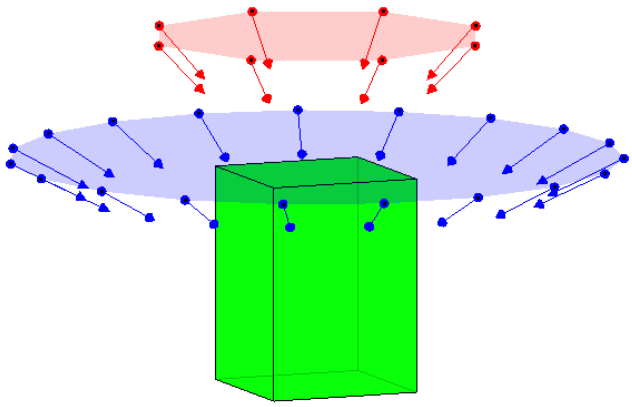


Figure 4.5: The described photo coverage of the object.

position. The rotation step of the camera should be determined depending on the geometric complexity of the input object. Our goal is to provide adequate but not unnecessarily large overlapping regions with excessive information repetition.

With the middle photos captured, we should repeat the same procedure with the camera at a greater height to acquire images that provide a good visibility of the object's topmost surfaces. As the upper part of the object in each scene will be invisible from the opposite view (will be occluded by the floor), the camera's direction should be properly adjusted to provide clear views of the object's upper surfaces. The rotation step can be significantly higher (i.e fewer photos required) as the upper surface will be generally smaller in comparison with the object's middle parts and possibly geometrically simpler.

We note the top and the bottom view photogrammetric reconstructions as $P_T$ and $P_B$ respectively. The described photography procedure is illustrated in Figure 4.5.

## 4.4 Acquisition of 3D scans

The 3D scan reconstructions of the object can come from any available 3D scanning technique that provides an accurate model of the object's geometry. So, we focus on the expected output reconstruction's traits instead of the method-related details.

As in the photogrammetric reconstructions, we need to take two separate 3D scans of the scene for both object placements. The resulting 3D geometries will constitute the base for the creation of a representative reconstruction for our object. Consequently, we need to ensure that we get high-quality 3D scans as any possible geometric defect may propagate to the final model. The 3D scans must be textured and reconstruct the entire scene including the markers on the floor. It is also

required that the markers are represented as compact structures in the texture file (not as scattered parts) enabling their detection by the marker detection algorithm in the texture map.

We note the top and the bottom view scans as $S_T$ and $S_B$ respectively.

## 4.5 Outcome

As an outcome of this step, we get a dataset comprised of two 3D scans $S_T$ and $S_B$ and two photogrammetric reconstructions $P_T$ and $P_B$. Each photogrammetric mesh is accompanied by its photogrammetric images $I_j$ and their associated projection matrices $P_j$. Using $P_j$ we can map the meshes' triangles to $I_j$ images.

# Chapter 5

# Marker localization and floor removal

By completing the dataset acquisition phase, we should have available two photogrammetric and two 3D scan reconstructions, one for each view of the object (top or bottom). These reconstructions will represent the object in a certain pose along with the surrounding markers that are placed on the floor. In this section, we estimate the 3D positions of the markers on the two types of reconstructions in our dataset. Finally, we present a methodology for the removal of the floor from our reconstructions with a view to maintaining only the object's structure in each reconstruction.

## 5.1 Locating markers in photogrammetric reconstructions

The images $I_j$ of a photogrammetric reconstruction depict the scene from multiple views and provide complete coverage of the scene. Therefore, the markers will be visible from at least one photogrammetric image. This property enables us to search for the markers in images $I_j$. We note the detection and localization of marker $\mathcal{M}_i$ in image $I_j$ (through the marker detection algorithm) as $\mathbf{m}_{i,j}$.

In order to associate the 2D locations of image $I_j$ with world coordinates, we use its projection matrix $P_j$. Through $P_j$, the 3D triangles of the photogrammetric reconstruction can be mapped to triangular regions in $I_j$. Thus, the 3D triangle $t$ that corresponds to the marker $\mathcal{M}_i$ will have its projection $\tau$ on $I_j$ contain the marker's 2D coordinate $\mathbf{m}_{i,j}$. It is possible that the projections of more than one 3D triangle contain $\mathbf{m}_{i,j}$. This usually indicates that the marker is partially occluded by the object in the image though, the marker detection algorithm manages to detect it. In this case, we should try to locate the marker in another image where the marker is clearly visible. In this way, we avoid ambiguities concerning the selection of the 3D triangle that corresponds to $\mathcal{M}_i$.

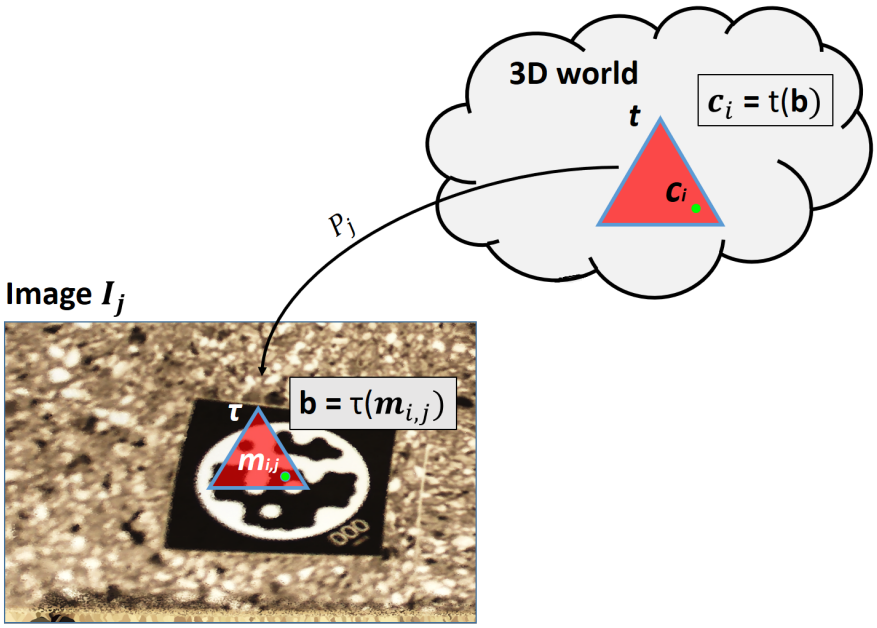



Figure 5.1: Estimating the 3D position of a marker in a photogrammetric mesh through the barycentric coordinates.

The 2D coordinate $\mathbf{m}_{i,j}$ is inside the projection triangle $\tau$. So, its relative position with respect to $\tau$ can be determined through the calculation of the barycentric coordinates $\mathbf{b}$ of $\mathbf{m}_{i,j}$. With the application of $\mathbf{b}$ to the 3D triangle $t$, we can acquire an estimation of the corresponding 3D coordinate $\mathbf{c_i}$ of $\mathbf{m}_{i,j}$. The described procedure is illustrated in Figure 5.1.

We should note here that the projective transformation does not preserve the ratio of the triangles' edges. As a result, the determination of $\mathbf{c_i}$ through the barycentric coordinates constitutes an approximation. Though, owing to the small area of 3D triangles of the photogrammetric reconstructions, localisation's precision will not be significantly affected.

By applying the above procedure for all markers in a photogrammetric reconstruction, we acquire an estimation of the markers' 3D locations. We note the reconstructed positions of the markers' centers of $P_T$ and $P_B$ as $\mathcal{U}_\mathbf{i}$ and $\mathcal{B}_\mathbf{i}$ respectively, where $i \in [0, 3]$ and represents the index of the marker.

## 5.2 Locating markers in 3D scans

As formulated in the 3D scan acquisition guidelines (in Chapter 4.4), the markers in a 3D scan's texture file must be represented as compact structures. This enables us to locate the markers in the texture file's image. This process will give us the image coordinates of the markers' centers in the texture file represented as 2D vectors $[x, y]^T$.

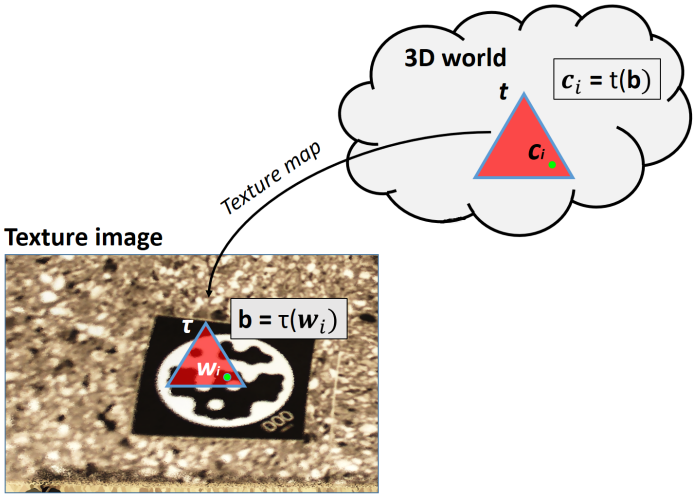



Figure 5.2: Estimating the 3D position of a marker in a 3D scan through the barycentric coordinates.

As our 3D scans are textured meshes, a 2D texture triangle will be assigned to each scan's triangle, providing a mapping of texture from the texture file. The three vertices that define each texture triangle will be in UV coordinates. Hence, it is necessary that we convert the 2D coordinates of the detected markers from image to UV coordinates so that the markers' positions are directly comparable to the texture triangles. The conversion from image to UV coordinates can be performed with the following procedure:

$$u = x \,/\, W$$
$$v = 1 - (y \,/\, H)$$

where $W, H$ are the width and the height of the texture file's image respectively.

With the markers' coordinates and the texture triangles in the same reference system, we can identify the 3D triangle $t$ that corresponds to marker $\mathcal{M}_i$ as its assigned texture triangle $\tau$ will contain the UV coordinate $\mathbf{w}_i$ of $\mathcal{M}_i$'s center.

Similarly to the previous section, we can estimate the corresponding 3D position $\mathbf{c}_i$ of $\mathbf{w}_i$ in the 3D scan, using the barycentric coordinate $\mathbf{b}$ of $\mathbf{w}_i$ with respect to $\tau$. This will give us a good approximation of $\mathbf{c}_i$. The described procedure is illustrated in Figure 5.2.

With the described methodology, we locate the markers' 3D positions in both 3D scans. We denote the reconstructed positions of the marker of $S_T$ and $S_B$ as $\mathbf{u}_i$ and $\mathbf{b}_i$ respectively, where $i$ represent the index of the marker.

## 5.3 Floor removal

With the 3D positions of the markers known, the floors in the reconstructions do not anymore provide any useful information. On the contrary, their presence may possibly add some extra challenge to our effort to register the 3D models in Chapter 6. Consequently, we have to remove the floor from all the reconstructions in our dataset. The methodology that we follow is the same for both reconstruction types and is described below.

The floor of each reconstruction is a smooth and planar surface. Therefore, it can be perceived as a 3D plane in the 3D coordinate space. From 3D geometry we know that a plane is determined by two factors: (a) a point lying on the plane, (b) a plane's normal (i.e a vector perpendicular to the plane). For the floor of an object's reconstruction, these parameters can be obtained with the use of the 3D coordinates of its markers, which are in essence four points lying on the floor's plane. In specific, we trivially have a point on the floor's plane (any of the four coordinates) and the plane's normal is computed as the following cross-product:

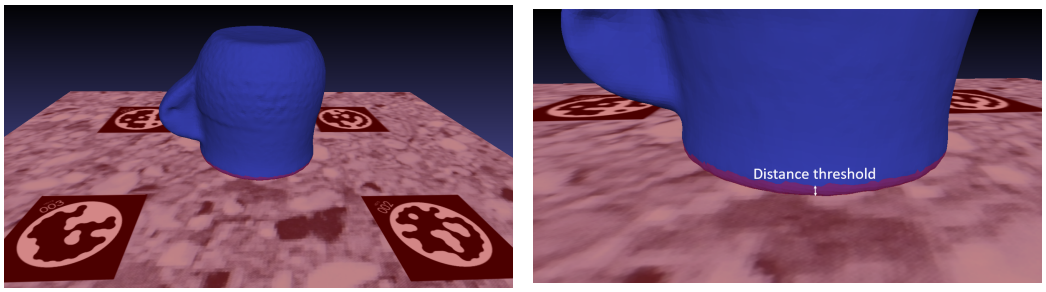
$$\mathbf{n} = (\mathbf{p}_2 - \mathbf{p}_0) \times (\mathbf{p}_1 - \mathbf{p}_0), \tag{5.1}$$

where $\mathbf{p}_0$, $\mathbf{p}_1$, $\mathbf{p}_2$ are three different points on the plane that can be replaced by any three markers' 3D coordinates.

The 3D triangles that belong to the floor, would ideally lie in the floor's plane. However, due to possible noise and errors during the 3D capturing procedure, it is impossible for all these floor triangles to lie on the exact same plane. Furthermore, the calculation of the floor's plane parameters is based on the estimated markers' positions. So any error with their 3D coordinates will negatively affect the precision of the plane's parameters. Therefore, it is required that we employ an approximate criterion for the identification of the floor triangles on a scene's reconstruction.

A first observation is that all candidate floor triangles should be relatively close to the estimated floor plane. This means that their centroids' distance from the floor plane should not be longer than a small distance threshold. As the scales of different reconstructions can vary, the distance threshold should be set with a value proportional to the size of the working mesh (i.e we cannot set a fixed value). Thus, we select to set it as the 3% of the reconstruction's height. The height of a mesh can be defined as the maximum distance between its floor's plane and the vertices of the mesh.

Within the defined distance threshold there should be included all the floor's triangles but there will also be some triangles that belong to the object as shown in Figure 5.3. This phenomenon occurs as the distance threshold is not too strictly defined, and thus, some triangles that are close to the floor are also included in the selection. In order to separate the object and floor triangles, we consider the direction of the 3D triangles. Likewise the floor, each 3D triangle defines its own plane. A triangle's normal can be computed with the use of function (5.1) with $\mathbf{p}_0$, $\mathbf{p}_1$, $\mathbf{p}_2$ now being the 3D vertices of the current triangle. Given two planes




(a) Zoom out. (b) Zoom in.

Figure 5.3: An indicative illustration of the selected triangles by the distance threshold (marked in red). The areas of the reconstruction that we not selected by the distance threshold are marked in blue.

$P_1$, $P_2$ with normals $\mathbf{n}_1$, $\mathbf{n}_2$, the dihedral angle between $P_1$, $P_2$ is the acute angle $\theta$ such that:

$$cos(\theta) = \frac{|\mathbf{n}_1 \cdot \mathbf{n}_2|}{\|\mathbf{n}_1\|\|\mathbf{n}_2\|}, \tag{5.2}$$

and denotes the direction difference between the two planes. The 3D triangles which correspond to the reconstruction's floor are expected to have a plane direction comparable to the floor's. As a result, the dihedral angle formed between their planes and the floor should be relatively small. So, we can set an angle difference threshold to distinct the floor's triangles from the ones that belong to the object. A safe angle threshold value would be 15°.

By combining the distance and the angle threshold we can filter out the majority of the floor triangles while maintaining the object structure barely intact. However, there is a possibility that some floor triangles "survive" the filtering performed by the two thresholds (Figure 5.5a). This can happen due to large errors during the 3D capturing of certain floor triangles, resulting in a high deviation of these triangles' direction from the actual floor plane's direction (higher than our angle threshold). Although, these triangles will be few and sparse on the floor's surface. So, we can choose to maintain the largest connected component of the filtered reconstruction, which will essentially be our object. In this way, we can preserve the object structure while excluding any remaining floor outliers (Figure 5.5b).

## 5.4 Outcome

With the completion of the above steps, we acquire an estimation of the markers' 3D positions for each object reconstruction in our dataset. Additionally, we accomplish a complete removal of the floor from all reconstructions. Henceforth, we consider that all reconstructions in our dataset are floor-less.

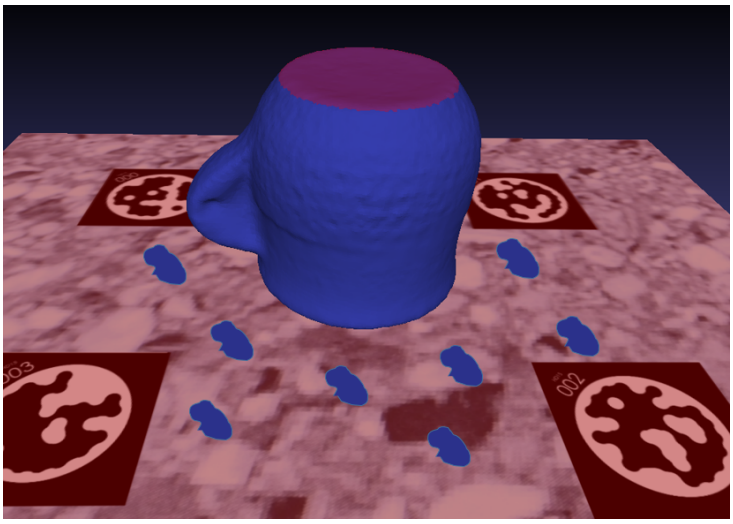
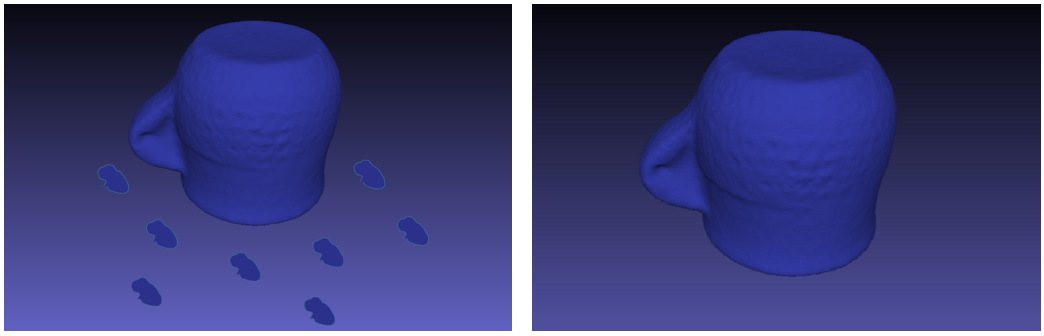


Figure 5.4: An indicative illustration of the selected triangles by the angle threshold (marked in red). The areas of the reconstruction that we not selected by the angle threshold are marked in blue.



(a) Distance and angle threshold.

(b) Largest connected component.

Figure 5.5: An indicative illustration of the filtering performed by the distance and angle threshold combined (left). The removal of outlier floor triangles (right).

# Chapter 6

# Registration of 3D models

The four object models in our dataset constitute independent entities that have been created from separate scanning procedures. Therefore, they will be in a different position, rotation, and possibly scale in the 3D world. In the context of our reconstruction method, it is necessary that we combine geometric and texture information from all of our dataset's 3D models. To this end, we need to acquire an association between the different object's reconstructions. We achieve this association through the registration of the 3D models. Specifically, we are interested in two types of registration: a) the registration of the 3D scans with the photogrammetric reconstructions of the same view and b) the registration of the two 3D scans.

## 6.1 Registration of 3D scans with photogrammetric reconstructions

Initially, we perform the registration of 3D scans with their corresponding photogrammetric reconstructions of the same view. The purpose of this registration is to bring the 3D scans in the same pose with the photogrammetric meshes so that we can transfer the texture from the photogrammetric reconstructions to the 3D scans. In this task, we have two scenes corresponding to the upward and downward placement of the object on the floor. Therefore, the described procedure will be repeated for both reconstruction pairs $S_T$, $P_T$ and $S_B$, $P_B$. We register a 3D scan and a photogrammetric reconstruction of the same scene as follows.

### 6.1.1 Registration methodology

First, we utilize the 3D coordinates of the markers located in Chapter 5. Since the two reconstructions model the same scene, there is a one-to-one correspondence between their reconstructed markers. Thus, each pair of coordinates with the same index ( ($\mathbf{u}_i$, $\mathcal{U}_i$) for the top view scene or ($\mathbf{b}_i$, $\mathcal{B}_i$) for the bottom view

scene, $i \in [0, 3]$) will constitute a 3D-to-3D point correspondence between the two reconstructions.

The resulting, four, correspondences are enough to proceed to the registration of the two object models. To do so, we employ the Kabsch–Umeyama registration algorithm [Ume91], [Kab76], [Kab78]. Given the four correspondences as an input, the algorithm will yield three registration parameters: (a) a scaling factor $s$, (b) a $3 \times 3$ rotation matrix $R$, and (c) a $3 \times 1$ translation vector $t$. These parameters are combined as $s \cdot [\, R \mid t \,]$ to produce a $3 \times 4$ transformation matrix $T_M$, which constitutes the initial registration transform that aligns the 3D scan with the photogrammetric reconstruction.

However, registration transform $T_M$ is derived solely from the, four, point correspondences. Thus, any inaccuracy in the localization of the markers will have a negative impact on registration accuracy. To enhance our registration result, we capitalize on the geometric similarity of the two object reconstructions.

With the initial registration $T_M$, the two models will be in approximately the same scale, location and orientation. Therefore, they are adequately aligned as input to the ICP algorithm [BM92]. ICP will utilize the shape of the two reconstructions to minimize the distance between their point clouds. The points of the photogrammetric reconstruction will be regarded as the reference point cloud. The 3D scan will be the source point cloud which will be processed to match the reference. ICP returns a $3 \times 4$ transformation matrix $T_C$, which we use as a correction to the initial registration.

The combination of the $T_M$ and $T_C$ gives a composite transform that finely registers the 3D scan with the photogrammetric reconstruction. We denote the composite transforms associating reconstruction pairs $S_T$, $P_T$ and $S_B$, $P_B$ as $Q_t$ and $Q_b$, respectively. The transforms are illustrated in Figure 6.1.

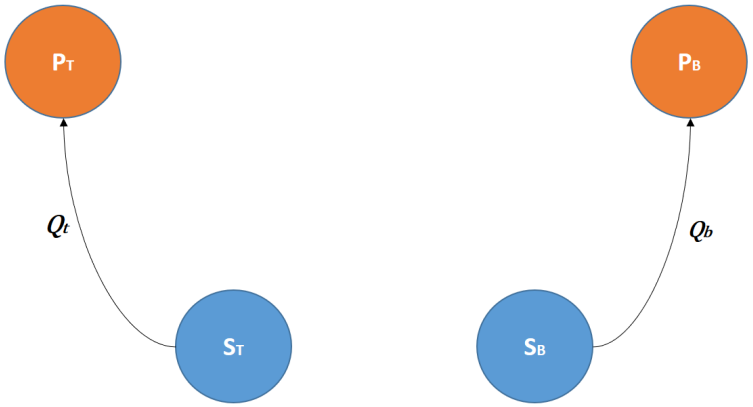



Figure 6.1: The transforms $Q_t$ and $Q_b$ associate each 3D scan with the photogrammetric reconstruction of the same view.

### 6.1.2 Accuracy issues

The 3D scans and their corresponding photogrammetric meshes are initially in a different scale. The matching of the scales between the two reconstructions in our registration methodology is accomplished through the $T_M$ registration component. However, due to possible errors in the localization of markers, the scaling can also present slight defects. Additionally, the geometry of photogrammetric reconstructions is less accurate than that of 3D scans. As a result, the shape inconsistency between the in-process models can add some extra challenge to ICP to provide a precise registration which reduces the correction effectiveness of $T_C$ component. These two problems can negatively affect the accuracy of $Q_t$ and $Q_b$ transforms.

## 6.2 Registration of partial 3D scans

In Chapter 6.1, we acquired a way of interconnection between the reconstructions of the same scene. Nevertheless, we still lack an association between the opposite scenes of our object. This renders impossible the combination of geometric and texture information from reconstructions of different views. To this end, we proceed to the registration of the $S_B$ with $S_T$.

### 6.2.1 Registration methodology

The alignment logic that we use is similar to Chapter's 6.1. We begin the registration process using the information provided by the reconstructions' markers. However, in this case, there is no direct correspondence between the markers of the two 3D scans. This is owing to the fact that the two 3D scans model different scenes. As a result, the relative position of the object with respect to the markers in each scene will be different.

To overcome this issue, we perform a transformation of $\mathbf{b}_i$ marker points in a way that the transformed points $\mathbf{b}_i^T$ correspond to $\mathbf{u}_i$. In reference to $S_B$'s scene, the 3D points that correspond to $\mathbf{u}_i$ will lie in a hypothetical plane $P_{ceil}$ which is tangent to the topmost part of $S_B$ and is parallel to its floor. Essentially, the $P_{ceil}$ plane replicates the $S_T$'s floor in the bottom view's scene. $P_{ceil}$ is defined using the normal of $S_B$'s floor and the vertex of $S_B$ which has the maximum distance from the $S_B$'s floor. The $P_{ceil}$ plane is illustrated in Figure 6.2.

The definition of $P_{ceil}$ limits significantly the search space for the corresponding points of the $\mathbf{u}_i$, but we still need extra information to specify their positions. The answer to this problem is provided by the rectangular layout of the markers (see Chapter 4.2). Between the two scenes, the only part modified was the object which underwent a 180° rotation. Contrarily, the markers were left intact on the floor preserving the exact same arrangement. Therefore, the points that correspond to $\mathbf{u}_i$ will form a rectangle on $P_{ceil}$ identical to the one formed by the $\mathbf{b}_i$ points.

Additionally, given that the object maintains its position at the center of the marker's rectangle in both scenes, we can deduce that the the corresponding points

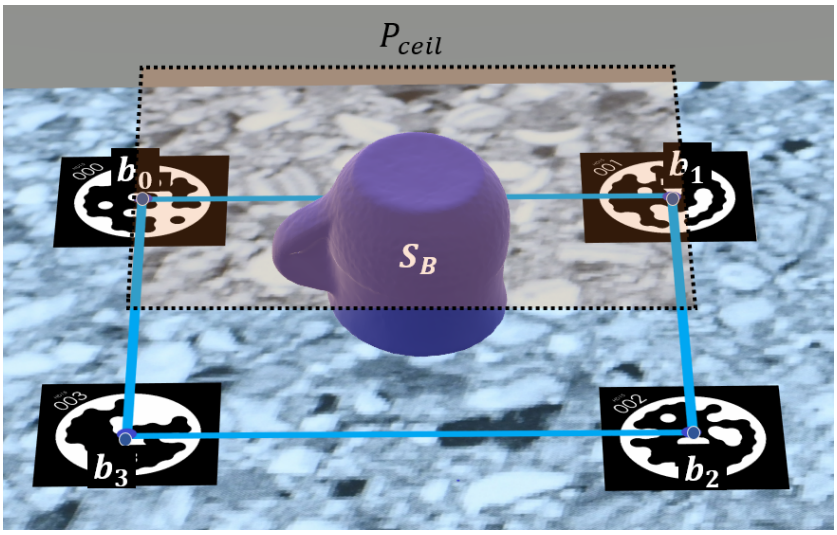



Figure 6.2: The $P_{ceil}$ plane.

of $\mathbf{u}_i$ won't be any shifted with regard to $\mathbf{b}_i$ points. As a result, they will constitute a direct mirroring of $\mathbf{b}_i$ points on the $P_{ceil}$ plane. This mirroring is fundamentally the perpendicular projection of the $\mathbf{b}_i$ on $P_{ceil}$. In this way, we transform $\mathbf{b}_i$ to $\mathbf{b}_i^T$. $\mathbf{b}_i^T$ point are shown in Figure 6.3

Points $\mathbf{u}_i$ and $\mathbf{b}_i^T$ are matching in the two 3D scan reconstructions. However, their correspondence is not one-by-one, meaning that markers with the same index don't actually match. This occurs due to the rotation of the object around the marker edge $\mathbf{m}_1 - \mathbf{m}_2$ which leads to a swap of correspondences between markers of opposite sides with respect to $\mathbf{m}_1 - \mathbf{m}_2$ edge. Thus, the four correspondences are $(\mathbf{b}_0^T, \mathbf{u}_1)$, $(\mathbf{b}_1^T, \mathbf{u}_0)$, $(\mathbf{b}_2^T, \mathbf{u}_3)$ and $(\mathbf{b}_3^T, \mathbf{u}_2)$. Using the Kabsch–Umeyama algorithm with these correspondences as an input, we can find the transform $T_M$ that roughly registers our 3D scan models. In this case, the scaling factor $s$ is not necessary as the two scans are already in the same scale.

For the calculation of the $T_M$ transform, we relied to a great extent on the accurate formation of the rectangular layout of the markers and the object during the acquisition of the dataset. Hence, depending on the level of deviation from the target arrangement, it is expected that the registration of $T_M$ will be analogously worse. To compensate for the possible alignment errors of $T_M$, we use the ICP algorithm on the roughly registered meshes. ICP will produce a transform $T_C$ that corrects the $T_M$.

The combination of the $T_M$ and $T_C$ forms a composite transform that registers $S_B$ to $S_T$. We denote this composite transform as $Q$ and its inverse as $Q^{-1}$. An overview of the calculated transforms is shown in Figure 6.4.

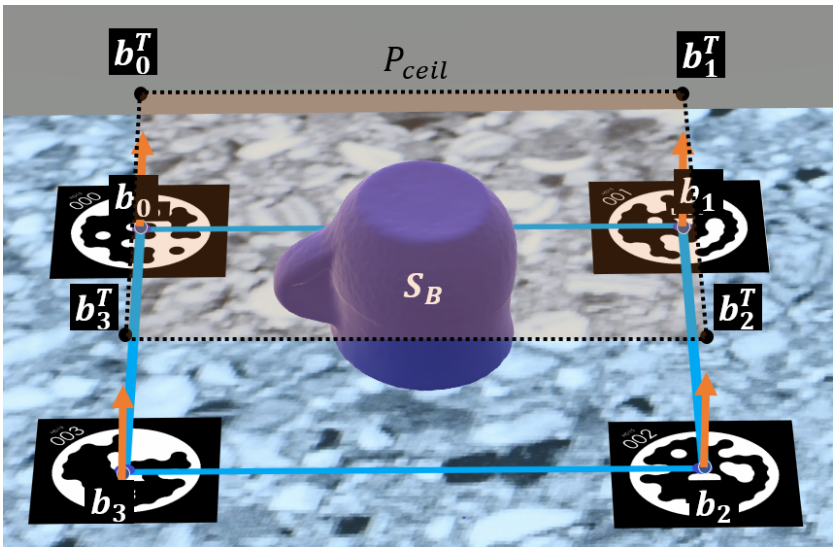



Figure 6.3: The projection of $\mathbf{b}_i$ points on $P_{ceil}$.

### 6.2.2 Accuracy issues

The 3D scans $S_T$ and $S_B$, model the object in two different poses. Consequently, the main structure of the two scans will be similar, but there will also be object areas that are missing in each scan, i.e., due to the floor of the scan table. So, there will exist surfaces in the two 3D scan reconstructions that are considered as outliers in the registration's context. ICP is robust to a small proportion of outliers, but it is impossible that the final registration is completely unaffected. This can reduce the accuracy of the correction $T_C$. Thus, the registration $Q$ is also prone to inaccuracies.

## 6.3 Outcome

From this step, we acquire two transforms $Q_t$ and $Q_b$ that register the 3D scans with the photogrammetric meshes of the same view. We also acquire a transform $Q$ that registers the two 3D scans. These transforms enable us to interconnect the reference systems of all reconstructions in our dataset.

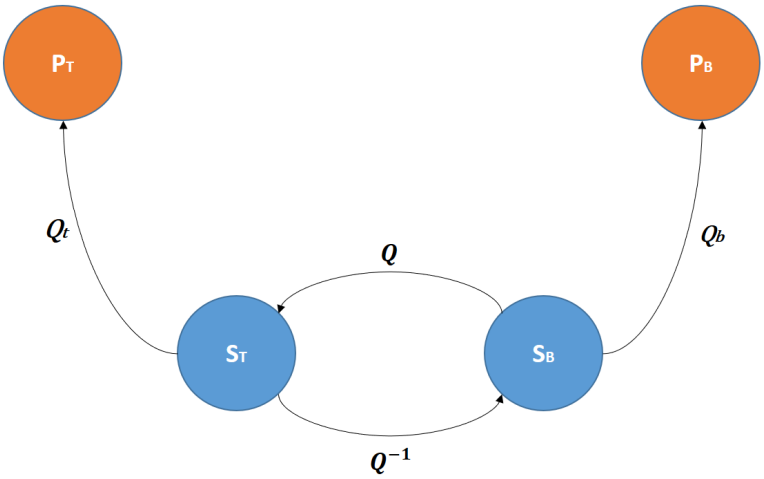



Figure 6.4: The transformations Q that interconnect the two views

# Chapter 7

# Creating a complete object model

The 3D scans in our dataset provide an accurate reconstruction of the object's geometry. This renders them an ideal source for the 3D structure of our final reconstruction. However, the floor of the scene prevents the creation of a complete reconstruction of the object from a single scan. In both object placements, either the top or the bottom part will be occluded by the floor. Nevertheless, the concealed area is different in the two object's views. This enables us to combine parts from both 3D scans $S_T$ and $S_B$ to create a complete reconstruction of the object.

## 7.1 Merging of 3D scans

The $S_T$ scan reconstructs completely the object's top part while the $S_B$ the object's bottom part. To this end, we wish to bring the two models in the same pose. We accomplish this through the transformation $Q$ calculated in Chapter 6.1.

Let 3D point $\mathbf{c}$ be the centroid of $S_T$. The centroid $\mathbf{c}$ and the $S_T$ floor's normal define a plane $P_{cut}$, that is parallel to the floor, and divides the 3D world into two volumes that we call "half-spaces". We note the half-space that includes the $S_T$'s marker points $\mathbf{u}_i$ as $H_b$ and its complement as $H_t$. We filter out the $S_T$'s triangles that have at least one point that belongs to $H_b$. We repeat the same for $S_B$ but now we filter out the triangles that have at least one point that belongs to $H_t$. The described procedure is illustrated in Figure 7.1.

The aforementioned triangle filtering will give us two object components, one from each scan. Their combination will yield the full reconstruction of the object. As the two 3D scans are registered, the two components will constitute the natural continuation of one another that is interrupted by a middle "cut". We compensate for this "cut" by performing a zipping operation between the two components. Zipping can be accomplished through a surface reconstruction method such as Screen Poisson [KH13] or Ball Pivoting [Ber+99].

The zip created between the two components (from the surface reconstruction

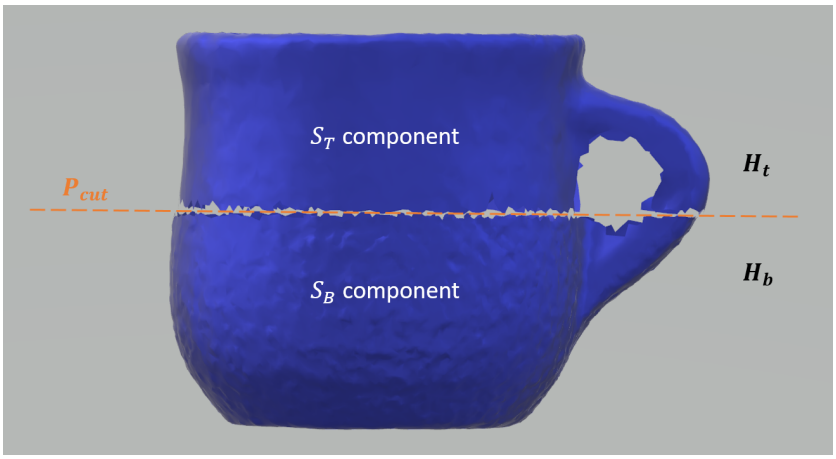



Figure 7.1: The cut performed by $P_{cut}$ plane.

method) may be visible due to small alignment errors during the registration of $S_T$ and $S_B$ by $Q$ (see Chapter 6.2.2). Thus, we perform a Laplacian smoothing [Nea+06] across the boundary defined by $P_{cut}$, to conceal the seam. In this way, we get the final reconstruction of the object, which we denote as $M$.

## 7.2 Associating M with the photogrammetric meshes

The procedure described in Chapter 7, creates the complete model of the object $M$ in the reference system of $S_T$. Therefore, $M$ can be directly registered to $P_T$ through the $Q_t$ transform. However, there is no direct association of $M$ with $P_B$. To perform this interconnection we combine $Q^{-1}$ and $Q_b$. In this way, we use the $S_B$'s reference system as an intermediate node so that we can register $M$ to $P_B$ through $Q_b$. Henceforth, we refer to the registration transforms of $M$ with the two photogrammetric meshes $P_T$ and $P_B$ as $T_t$ and $T_b$, respectively. The model $M$ is registered with the two photogrammetric meshes as shown in Figure 7.2.

## 7.3 Limitations in reconstruction M

The geometric quality of reconstruction $M$ is dependent on the registration accuracy provided by transform $Q$. To this end, we require that $Q$ finely registers the two 3D scans. In case of gross inaccuracies in $Q$, our method fails to create a representative reconstruction of the object. However, we are able to recover from small pose inconsistencies between the two scans through the Laplacian smoothing performed at the seam of the two components.

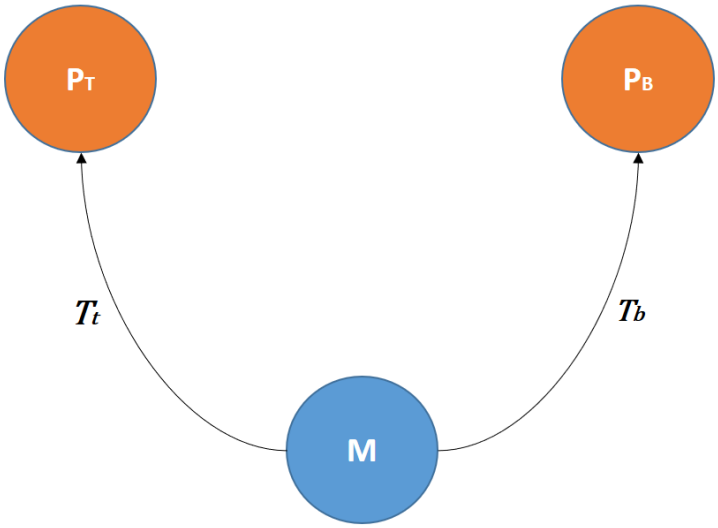



Figure 7.2: The transformations $T_t$ and $T_b$ interconnect $M$ with the two photogrammetric meshes.

## 7.4 Outcome

From this step we acquire a complete geometric reconstruction $M$ of our object. We also get an association of $M$ with the two photogrammetric meshes through $T_t$ and $T_b$ registration transforms.

# Chapter 8

# Registration refinement

In Chapter 7, we created a complete reconstruction $M$ of the object which we associated with the two photogrammetric reconstructions $P_T$ and $P_B$ through $T_t$ and $T_b$, respectively. The accuracy of these two transforms is of great importance for our method, as they essentially provide a texture mapping of the triangles of $M$ to the photogrammetric images. In this section, we refine $T_t$ and $T_b$ with a view to improving the overall quality of the texture mapping procedure.

## 8.1 Motivation

The $T_t$ and $T_b$ transforms are built by the $Q_t$, $Q_b$, $Q$ (see Chapter 7.2). As discussed in Chapters 6.1.2 and 6.2.2, these transforms are prone to inaccuracies owing to geometric inconsistencies of the registered meshes and errors in the localization of the markers.

Lack of accuracy in the texture mapping transforms leads to misplaced texture on the reconstructed surface. Sometimes the error can be so gross that object triangles are textured with texture belonging to the background. This problem affects negatively the quality of the texture. Therefore, we would like to diminish it.

## 8.2 Rationale

To improve the registration of $M$ with $P_T$ and $P_B$, we capitalize on the projection of $M$ on the images of each photogrammetric reconstruction. We will refine $T_t$ or $T_b$ performing the following operation, once for each.

For simplicity, the remainder of this section is agnostic if applied to the top or bottom setup of the object. In both cases, we refer to images from photogrammetry as $I_j$, the photogrammetric reconstruction as $P$, and the transform that registers $M$ to the reference frame of photogrammetry as $T$. In Chapter 8.8, we denote the names of each individual result.

Initially, we will create binary masks, $B_j^P$, and $B_j^M$, of equal size to images $I_j$ and initialize their values to zero (0). Then, we register $M$ to $P$ through $T$. In each view $j$, we use $P_j$ to project $M$'s triangles on $I_j$. We detect the pixels occupied by this projection and set the corresponding pixels of $B_j^M$ to one (1). This creates binary masks $B_j^M$. In the same way, we also project the photogrammetric reconstruction, $P$, on images $I_j$, to create $B_j^P$.

Ideally, masks $B_j^M$ and $B_j^P$ would be identical. If this was true for all the photogrammetry images, it would mean that we have achieved perfect compatibility between the projection matrices and model $M$. So the texture mapping between the $M$'s triangles and images $I_j$ would be flawless.

We follow this line of thought and search for a transform that updates $T$ so that the similarity between corresponding masks $B_j^M$ and $B_j^P$ is maximized. We treat this as an optimization problem that we solve through PSO.

## 8.3 Optimization parameters

To achieve better compatibility of $M$ with projection matrices $P_j$, we need to find a new transform that refines the current registration of $M$ with the photogrammetric mesh through $T$.

This transform should include a translation, a rotation, and a scaling of the object so that we can reduce any remaining alignment errors from the previous registration. The 3D transform will have 9 DOFs, 3 for the translation, 3 for the rotation, and 3 for anisotropic scaling.

Selecting one degree of freedom in scaling assumes that we perform a uniform scaling to the object. However, our experiments show that higher flexibility in the scaling of $M$ contributes to better registration results. Hence, we choose to use three independent scaling factors with respect to the different axes instead of uniform scaling. So, our problem will have 9 DOFs, and a candidate solution vector $\mathbf{v}$ will have the following form: $[s_x, s_y, s_z, r_x, r_y, r_z, x, y, z]$.

## 8.4 Domain of parameters

The 9 DOFs define a 9D search space in which we will search for an optimal solution. However, a search space of this dimensionality would hinder PSO convergence to an acceptable solution fast and reliably. Thus, we have to try to restrict the search space as much as possible. To accomplish this, we take into consideration the initial registration ($T$) of $M$ with $P$. Despite the small alignment errors of $T$, it provides a close matching of $M$ with $P$. Thus, the transform that we are seeking can be perceived as a refinement of $T$.

In this context, we restrict our search space depending on how much we trust our initial registration. The scaling parameters $s_x$, $s_y$, and $s_z$ should take values near 1 as we want a slight modulation of the initial scaling. The rotation and

translation parameters $r_x$, $r_y$, $r_z$ and $x$, $y$, $z$ will take values within a small margin around 0.

We note special attention to the fact that the scaling and the rotation are performed "in place" (with respect to the centroid of $M$). Otherwise, the transformation results in an extra shifting of the object from its initial position compromising the translation bound that we previously set.

## 8.5 Hypothesis representation

The PSO algorithm tests candidate solutions as "hypotheses". Each hypothesis is quantitatively evaluated and the result of this evaluation is used by the algorithm. We represent each candidate solution using a 9D vector. Candidate solution vector $\mathbf{v}$ includes three main transformation components: a) a scale component $S$, b) a rotation component $R$, and c) a translation component $T$.

### 8.5.1 Scaling

A scaling matrix $S$ with parameters $[s_x,\ s_y,\ s_z]$ and a custom center of scaling with coordinates $c = [c_x,\ c_y,\ c_z]$ can be constructed as the following $3 \times 4$ matrix:

$$S = \begin{bmatrix} s_x & 0 & 0 & b_x - s_x \cdot c_x \\ 0 & s_y & 0 & b_y - s_y \cdot c_y \\ 0 & 0 & s_z & b_z - s_z \cdot c_z \end{bmatrix}$$

### 8.5.2 Rotation

A rotation matrix with parameters $[r_x,\ r_y,\ r_z]$ can be constructed by three independent rotation components $R_x$, $R_y$, $R_z$ as the following matrix product:

$$R = R_x \cdot R_y \cdot R_z =$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(r_x) & -sin(r_x) \\ 0 & sin(r_x) & cos(r_x) \end{bmatrix} \begin{bmatrix} cos(r_y) & 0 & sin(r_y) \\ 0 & 1 & 0 \\ -sin(r_y) & 0 & cos(r_y) \end{bmatrix} \begin{bmatrix} cos(r_z) & -sin(r_z) & 0 \\ sin(r_z) & cos(r_z) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The resulting $3 \times 3$ matrix $R$ contains the rotation information defined by the rotation parameters with regard to the axes origin. To enforce a custom center of rotation with coordinates $[c_x,\ c_y,\ c_z]$ for the combined rotations we augment the $R$ matrix with fourth column to form a $3 \times 4$ matrix as follows:

$$R = \left[\begin{array}{ccc|c} R_{11} & R_{12} & R_{13} & c_x - R_{11} \cdot c_x - R_{12} \cdot c_y - R_{13} \cdot c_z \\ R_{21} & R_{22} & R_{23} & c_y - R_{21} \cdot c_x - R_{22} \cdot c_y - R_{23} \cdot c_z \\ R_{31} & R_{32} & R_{33} & c_z - R_{31} \cdot c_x - R_{32} \cdot c_y - R_{33} \cdot c_z \end{array}\right]$$

### 8.5.3 Translation

The translation matrix $T$ with parameters $[x, y, z]$ can be constructed as:

$$T = \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \end{bmatrix}$$

### 8.5.4 Composite transform

The matrices $S, R, T$ are all $3\times4$. In order to combine them into one transformation matrix we need to calculate their product. To do so, we need to augment them with the vector [0 0 0 1] as the last row so that they all become homogeneous transform, $4 \times 4$, matrices. In this way, we get the composite transform $T(\mathbf{v})$ that is defined by a transformation vector $\mathbf{v}$ as follows.

$$T(\mathbf{v}) = \begin{bmatrix} & T & & \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} & R & & \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} & S & & \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

## 8.6 Objective function

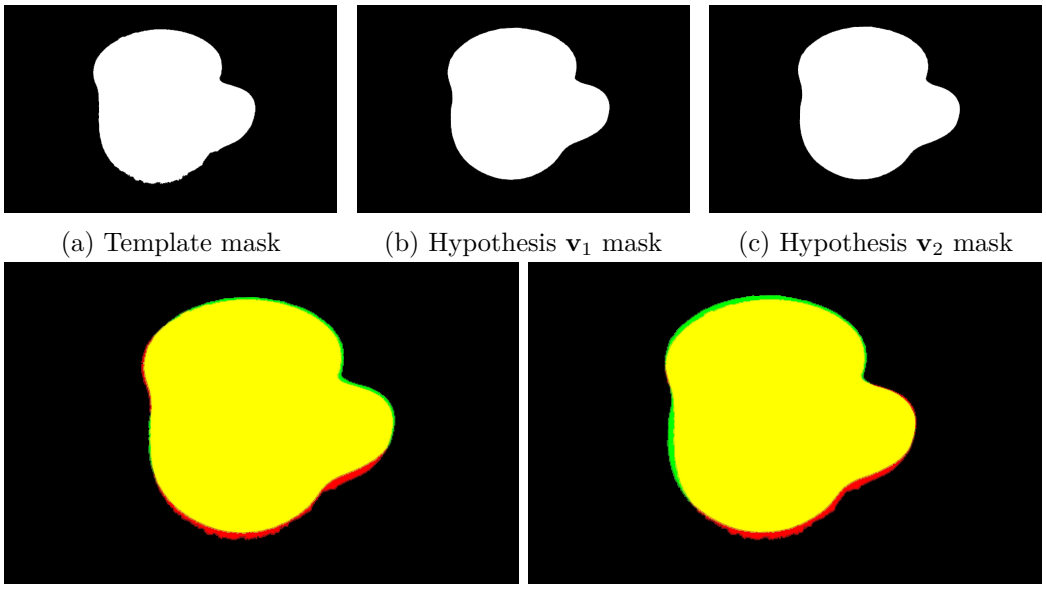
The PSO will need to compare multiple candidate solution vectors until it converges to an optimal result. Thus, we need a standard criterion, or an "objective function", for evaluating the quality of the transform associated with the state $\mathbf{v}$ of a particle. In our case, the objective function measures the dissimilarity of corresponding masks as follows.

As described earlier in Chapter 8.2, to project a photogrammetric mesh or $M$ on a given image, we construct a binary mask of equal size to the image. In this mask, the pixels covered by a projected triangle are set to 1 (foreground pixels) while the rest are set to 0 (background pixels). In this way, we obtain a collection of template masks, one for each $I_j$. We denote the masks produced by projecting $P$ on $I_j$s as $B_j^P$. We denote the masks produced by projecting hypothesis $\mathbf{v}$ on $I_j$s as $B_{\mathbf{v},j}^M$.

For each object setup (top or bottom), we quantify the registration error $E$ of transform $T(\mathbf{v})$ for a view $j$ as the mean of the pixel mismatches detected between the $B_{\mathbf{v},j}^M$ and $B_j^P$ as:

$$E(j, \mathbf{v}) = \frac{1}{H_j W_j} \sum_{r=1}^{H_j} \sum_{c=1}^{W_j} B_{\mathbf{v},j}^M(r, c) \oplus B_j^P(r, c)$$

where $H_j$ and $W_j$ are the height and width of $I_j$, respectively, and $\oplus$ denotes the XOR (not XNOR) logical operator. In the context of this function, we interpret logical values (true, false) as numerical values 1 and 0, respectively. A visual example of the comparison of a template mask with two projection masks of $M$ is shown in Figure 8.1.



(a) Template mask (b) Hypothesis $\mathbf{v}_1$ mask (c) Hypothesis $\mathbf{v}_2$ mask

(d) Mismatches in template and $\mathbf{v}_1$ masks (e) Mismatches in template and $\mathbf{v}_2$ masks

Figure 8.1: A template mask and the projection masks of two hypotheses $\mathbf{v}_1$ and $\mathbf{v}_2$. The mismatches between the template and the two hypotheses are highlighted with red and green color.

Ultimately, we calculate the total score of the objective function as the average registration error from all object views as:

$$Err(\mathbf{v}) = \frac{1}{N}\sum_{j=1}^{N} E(j, \mathbf{v})$$

where $N$ is the total number of views. We aim for the minimization of the this function.

## 8.7 Computational improvements

### 8.7.1 Computational cost

The constraints applying to a transformation vector's parameters allow PSO to examine small displacements of $M$ from its initial registration with $P$. So, the possible mismatches between an $M$'s projection mask $B^M_{\mathbf{v},j}$ and its corresponding template mask $B^P_j$ are expected to be in a small area around the foreground pixels of $B^P_j$. This observation enables us to exclude from consideration the remote background pixels from each template mask that would otherwise add some redundant overhead to the calculation of the transformation cost.

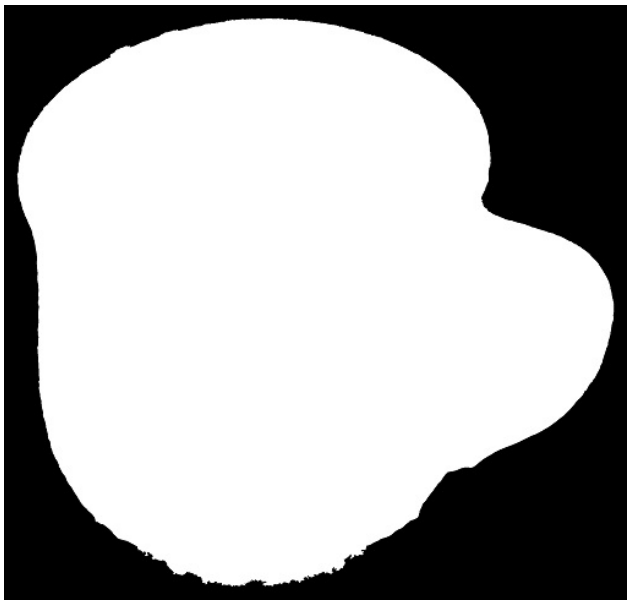


Figure 8.2: A cropped template mask.

To specify the comparison area for each template mask $B_j^P$ we calculate the bounding box $b_j$ of its foreground pixels. This bounding box is defined as quadruple $(b_j.x, b_j.y, b_j.h, b_j.w)$ where $(b_j.x, b_j.y)$ is the coordinate of the top-left of its corners and $(b_j.h, b_j.w)$ its respective height and width. The resulting bounding box should be extended towards all directions by a specific margin $m$ to include some surrounding area in the background. Thus, the area of interest $b_j$ is formulated as $(b_j.x - m, b_j.y - m, b_j.h + 2m, b_j.w + 2m)$. We select to set $m$ as $0.04 \cdot max(b_j.h, b_j.w)$.

We crop all the template masks $B_j^P$ and we keep only the image region defined by bounding box $b_j$. Furthermore, each subsequent projection mask $B_{\mathbf{v},j}^M$ of a transformation vector $\mathbf{v}$ should also be cropped with $b_j$ in order to be comparable to $B_j^P$. In this way, we effectively reduce the comparison operations needed for the cost calculation and the memory footprint of the template masks. A cropped mask is shown in Figure 8.2.

### 8.7.2 Convergence robustness

With the way that we have defined our problem's objective function, every pixel mismatch spotted between two corresponding projection masks, equally contributes to the determination of the total cost of the tested transformation vector $\mathbf{v}$. Whereas, this cost calculation approach does not have any logical defect, in practice it is not so effective regarding the ease of convergence it provides.

During the first few iterations, the PSO particles will significantly approach the function's global minimum as many mask pixel mismatches will be corrected from

the initial steps towards the minimum. However, the more the pixel mismatches decrease the less the tendency of the particles to move towards the global minimum will become as the cost reduction gain will gradually shrink. Consequently, this function's unwanted property may prevent us from approaching the global minimum to the extent that we want.

To solve this issue, we introduce the notion of the cost mask $C_j$ for every template mask $B_j^P$. A cost mask $C_j$ will allow us to apply different penalties to pixel mismatches located at different regions of a template mask $B_j^P$. The idea is that the pixels which are in a small range around the border between the foreground and the background of a template mask, are the most critical to match as they determine the shape of the object's figure. Thus, if this zone of pixels is correctly matched then the rest pixels should essentially follow.

So, we penalize mismatches in a radius of $R$ around the border and less the other mismatches. Specifically, within the area defined by $R$, we enforce a linearly decreasing penalty as we move away from the border to promote good coherence to the border. The values of the pixel $(r, c)$ of a cost mask $C_j$ are defined as follows:

$$C_j(r,c) = \begin{cases} H + s \cdot \frac{R - d(r,c)}{R}, & \text{if } d(r,c) \leq R \\ L & \text{otherwise} \end{cases}$$

where $d(r, c)$ is the Euclidean distance of pixel $(r, c)$ from the border. $L$ is a fixed penalty value for pixels mismatches outside radius $R$. $H$ is the minimum penalty that a pixel inside $R$ can have that is increased up to a scalar s as the pixels approach the border. It must hold that $L < H$. We empirically set these values as $R = 40$, $L = 0.25$, $H = 3$ and $s = 2$. A visualization of a cost mask is illustrated in Figure 8.3.

### 8.7.3 Refined registration error

Considering the previous two performance upgrades, the refined registration error of a transform $T(\mathbf{v})$ for a view $j$ is formulated as below:

$$E(j, \mathbf{v}) = \frac{1}{(b_j.h) \cdot (b_j.w)} \sum_{r=1}^{b_j.h} \sum_{c=1}^{b_j.w} (B_{\mathbf{v},j}^M(r,c) \oplus B_j^P(r,c)) \cdot C_j(r,c)$$

## 8.8 Outcome

The outcome of this step is two refinements transforms for $T_t$ and $T_b$. The respective refinements are incorporated into $T_t$ and $T_b$ to update their registration accuracy. Henceforth, we denote as $T_t$ and $T_b$ the refined versions of these transforms.

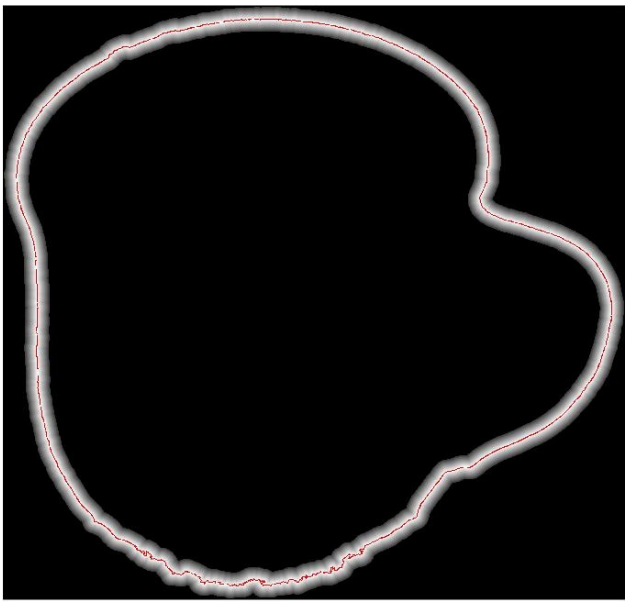


Figure 8.3: A visualization of a cost mask. The red line indicates the border between the foreground and background pixels of the template mask.

# Chapter 9

# Texture mapping

The refined transforms $T_t$ and $T_b$ provide an accurate registration of $M$ with the two photogrammetric meshes $P_T$ and $P_B$. Essentially, these registrations provide compatibility between $M$ and the projection matrices of the two photogrammetric reconstructions. This enables us to map $M$'s triangles to photogrammetric images through their projection matrices. In this section, we capitalize on this property to texture our complete object model $M$ from the photogrammetric images.

## 9.1 Assessing triangle visibility in photogrammetric views

Our first task is to find out which triangle is visible in which image or images, or otherwise, assess the triangles' visibility.

To effectively map $M$'s triangles to the photogrammetric images, it is required that we examine the triangles' visibility from the different photogrammetric views. From any given view, a 3D triangle can be fully visible, partially visible, or invisible. To assess the visibility of triangles with regard to a photogrammetric image $I_j$, we employ the z-buffering algorithm. Using the projection matrix $P_j$ and the mesh $M$ registered to the photogrammetric reconstruction of view $j$, z-buffering yields two matrices: a) a depth map $D$, and b) an index map $I$. These matrices are equally sized with $I_j$. Pixel values of $D$ yield the distance of $M$'s surface imaged at each pixel from the camera center. Pixel values of $I$ indicate the id of the triangle imaged at that pixel.

Thus, for a triangle $t_i$ with id $i$, its projection $\tau_i$ on the image $I_j$ (though $P_j$) will contain a number of pixels. By examining the values of these pixels on the index map $I$ we define the $t_i$'s visibility $V_{i,j}$ in image $I_j$ as the following ratio:

$$V_{i,j} = \frac{number\,of\,pixels\,with\,id = i\,in\,\tau_i}{total\,number\,of\,pixels\,in\,\tau_i}$$

The domain of this "visibility ratio" is $[0, 1]$. It will hold that $V = 1$ for fully visible triangles, $0 < V < 1$ for partly visible triangles, and $V = 0$ for invisible

triangles. In this way, we can acquire a visibility estimation for all $M$'s triangles in each photogrammetric view.

We should note the special case where the projection $\tau_i$ of a triangle $t_i$ on an image $I_j$ contains no pixels. In this case, $V$ cannot be defined as the denominator of the above ratio will be zero. This possibly indicates that either $t_i$ is a degenerate triangle or that $t_i$ is captured with a steep angle in the image $I_j$ so its projection $\tau_i$ is degenerate. To estimate the visibility of a triangle $t_i$ in this situation, we consider the 4 neighboring pixels of $\tau_i$'s centroid $c$ in depth map $D$. These values will indicate the distance of $M$'s surfaces that are imaged from neighboring pixels. If these distances are comparable the distance of $c$ from $I_j$'s camera center, then we can assume that $t_i$ is visible from $I_j$ and $V_{i,j} = 1$.

## 9.2 Texture source selection

The photogrammetric images provide a complete coverage of the object's areas. Thus, for each triangle of $M$, there will exist at least on image where the triangle is fully visible ($V = 1$). The photogrammetric images that provide non-zero visibility for a given triangle, constitute its candidate texture sources. We wish to select the best candidate texture source for each triangle in $M$. So, it required that we formulate a standard criterion for evaluating the view quality provided to a triangle by an image. To this end, we introduce two metrics: (a) the normalized triangle angle $\hat{a}$, and (b) the normalized triangle distance $\hat{d}$.

### 9.2.1 Normalized triangle angle

In each photogrammetric view $j$, the visible triangles of $M$ appear in a certain angle with respect to the camera. The less that angle is, the less the distortion of the imaging of these triangles will be in the respective image. Ideally, we would like that we have a direct image view for each triangle (i.e zero triangle-camera angle). However, this is practically impossible as it would entail that we have a prohibitively large number of photogrammetric images. Hence, to minimize the texture distortion we will look for the camera view which captures a triangle $t_i$ with the least angle.

To evaluate the angle between a camera $c_j$ and a visible triangle $t_i$, we calculate the dihedral angle $\theta_{i,j}$ formed between $I_j$'s plane and the plane of $t_i$. The normal of $t_i$'s plane is calculated with the use of function (5.1) with $\mathbf{p}_0$, $\mathbf{p}_1$, $\mathbf{p}_2$ being the vertices of the $t_i$. The normal of $I_j$'s plane is essentially the direction of the $c_j$'s principal axis which can be derived from the projection matrix $P_j$. Using these normals we can calculate $\theta_{i,j}$ though the definition of dihedral angle in function (5.2).

The $\theta_{i,j}$'s domain is $[0°, 90°]$. We normalize $\theta_{i,j}$ to $[0, 1]$ by dividing it with $90°$. In this way, we get the normalized angle $\hat{a}_{i,j}$ which constitutes a measurement of $t_i$'s imaging distortion in image $I_j$.

### 9.2.2 Normalized triangle distance

Each photogrammetric image offers a targeted view to a specific area of the object. Thus, for a given image $I_j$, we consider that the triangles which are closer to the camera, tend to have a better imaging. From this perspective, we can compare the Euclidean distance of visible triangles from camera $c_j$'s center to assess their proximity in $I_j$. In this way, we can specify the relative distance of triangles from the camera.

Nevertheless, the distance of the camera from object does not necessarily remain constant in the different image captures. Therefore, there can exist images that depict the object from a slightly closer ranges with respect to the others. As a result, we can not use the raw distances to compare the relative distance of a triangle in two neighboring images.

To overcome this issue, we employ a normalized version of the camera-triangle distance which we calculate as follows. Let a triangle $t_i$ of $M$ and its Euclidean distance $d_{i,j}$ from the camera $c_j$. By examining all the triangles of $M$ we can extract the minimum and the maximum distance that a triangle can have from $c_j$. We note these distances as $d^j_{min}$ and $d^j_{max}$, respectively. We define the normalized distance $\hat{d}$ of $t_i$ from $c_i$ as the following ratio:

$$\hat{d}_{i,j} = \frac{d_{i,j} - d^j_{min}}{d^j_{max} - d^j_{min}}$$

The subtraction of $d^j_{min}$ from the distance $d_{i,j}$ essentially eliminates the camera's distance contribution to the determination of a triangle's distance. This allows the comparison between triangles' distances from neighboring views. The denominator of the ratio enforces a $[0, 1]$ domain. The normalized distance $\hat{d}_{i,j}$ denotes $t_i$'s proximity to the camera $c_j$ in comparison with the rest triangles of $M$.

### 9.2.3 Mapping criterion

The $\hat{a}$ and $\hat{d}$ constitute two different metrics for the evaluation of the viewing quality provided by an image to a triangle. Normalized angle $\hat{a}$ examines the angle between a triangle and the camera to assess the triangle's imaging distortion. On the other hand, $\hat{d}$ considers the relative distance of triangles from a camera to promote close-to-camera triangles in each view.

In the context of our method, we want combine the two metrics to create a composite criterion for the selection of the best candidate image for each triangle. The two metrics have the same domain. Thus, we formulate the mapping criterion as the weighted average of the two metrics denoted as follows:

$$\mathcal{Q}_{i,j} = w_a \cdot \hat{a}_{i,j} + w_d \cdot \hat{d}_{i,j},$$

where $w_a$ and $w_d$ are two weight scalars that determine the contribution of each

metric. In our implementation we set $w_a = 0.6$ and $w_d = 0.4$. However, other weight values can also be used for a different combination of the two metrics.

The minimization $\mathcal{Q}$ provides the assignment of $M$'s triangles to photogrammetric images. More specifically, given a triangle $t_i$ and a subset of the photogrammetric views where $t_i$ is fully visible ($V_{i,j} = 1$), we select the best candidate view $j_i$ as the triangle's texture source as follows:

$$j_i = \arg\min_j \mathcal{Q}_{i,j}$$

By repeating this process for each triangle of $M$ we get an optimal assignment of all triangles to photogrammetric images.

## 9.3 Triangle texturing

Given a triangle $t_i$ and a selected photogrammetric view $j_i$, we want to texture $t_i$ from the corresponding photogrammetric image $I_{j_i}$. Let $P_{j_i}$ be $I_{j_i}$'s projection matrix and $T_{j_i}$ the registration transform that registers $M$ with the photogrammetric mesh of view $j_i$. We transform $t_i$ through $T_{j_i}$ and project it to $I_{j_i}$ through $P_{j_i}$. This yields image triangle $\tau_i$ which corresponds to $t_i$. The vertices of $\tau_i$ are in image coordinates. We convert them to UV coordinates to acquire the texture mapping of $t_i$ in image $I_{j_i}$. By repeating this procedure for all triangles of $M$, we perform the texturing of $M$ from the different photogrammetric images.

## 9.4 Outcome

With the completion of this step, we get a mapping of texture for each triangle of $M$ from a photogrammetric image that optimally images the triangle.

# Chapter 10

# Appearance optimization

The texture mapping of triangle's of $M$ yields regions of connected triangles that are textured by the same photogrammetric image. We refer to the connected regions "painted" by texture from the same image as texture clusters $C_i$ of the reconstruction $M$. In essence, the texture clusters constitute a partition of the $M$'s triangles into subsets of locally connected triangles that share the same texture source. A photogrammetric image may be associated with multiple texture clusters.

From the previous definition, it follows that the triangles of a cluster will be textured by neighboring regions in the corresponding photogrammetric image. Thus, it is expected that the texture within a cluster will be continuous and visually coherent. However, the same does not necessarily apply to the bordering regions of adjacent texture clusters, which receive texture from different images. Consequently, the texture consistency in these regions is dependent on the accuracy of the texture mapping in the two images.

Another problem concerning the intersection of clusters is related to the lighting conditions throughout the image acquisition task. Variability of these conditions can lead to noticeable photometric differences between neighboring texture clusters.

In this section, we present methodologies that contribute to the reduction of the texture inconsistencies between neighboring texture clusters.

## 10.1 Compacting texture clusters

Before we proceed to tackle the texture quality problems between clusters, we will reduce the extent of the problem by performing a triangle reassignment between the existing clusters. As the current clusters constitute an optimal distribution of the triangles to images (through the mapping criterion $\mathcal{Q}$), we will proceed to mild reassignments that do not significantly affect the basic structure of the clusters.

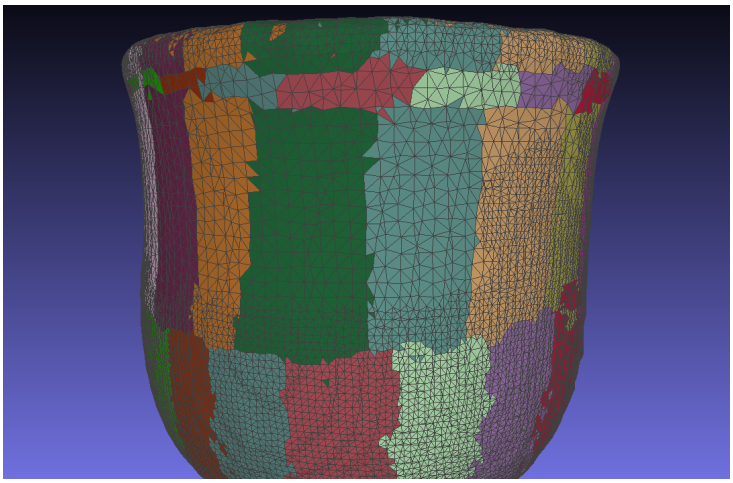


Figure 10.1: An indicative image of texture clusters created from the texture mapping process. Each cluster is highlighted with a different color.

### 10.1.1 Smoothing border shape between texture clusters

The shapes of the texture clusters are solely determined by the criterion $\mathcal{Q}$ which is responsible for the assignment of the 3D triangles to the photogrammetric images. Therefore, we do not have any direct control over the formation of the texture clusters which may acquire irregular shapes and rough borders. This phenomenon is undesirable for our clusters as the rougher their borders are, the larger the intermingling of cluster boundaries would be, and the larger the expression of texture discontinuities ("seams").

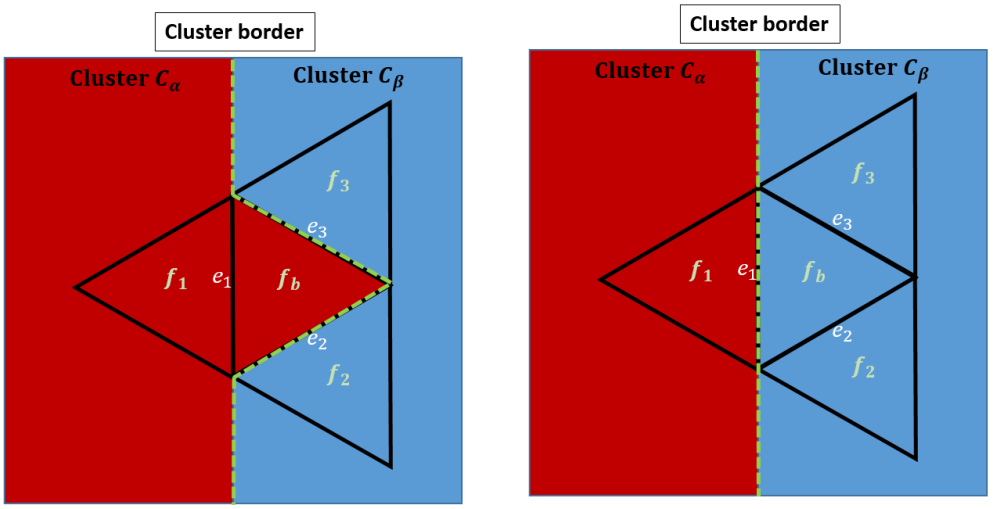
The proposed smoothing regards the shape of the boundary of clusters. It is performed in a discrete fashion, by reallocating $M$'s triangles among texture clusters. The purpose of this reallocation is to reduce the roughness (or "fractal dimension") of the boundary, which is equivalent to increasing the compactness of the cluster's shape.

As $M$ is a mesh of triangles, each of its triangles will have three adjacent triangles (neighbors) with which it shares one common edge. Considering this relation between triangles, we will proceed to the definition of two notions:

1. **Border triangles**: We define as border triangles of $M$, the triangles that have at least one neighbor that belongs to a different texture cluster.

2. **Strong cluster connectivity**: We say that a triangle has strong connectivity with its assigned texture cluster $C_i$, if and only if it has at least two neighbors that belong to $C_i$.

For the purpose of border smoothing, we will enforce strong cluster connectivity to the border triangles of $M$. For instance, let a border triangle $f_b$ with edges $e_1$, $e_2$, $e_3$, and its three neighboring triangles $f_1$, $f_2$, and $f_3$. $f_b$ and $f_1$ belong to




(a) Before reassignment. (b) After reassignment.

Figure 10.2: The border length between $C_\alpha$ and $C_\beta$ reduces after the reassignment of triangle $f_b$ to cluster $C_\alpha$.

cluster $C_\alpha$ while $f_2$ and $f_3$ belong to a different cluster $C_\beta$. The contribution of triangle $f_b$ to the total length of the border between $C_\alpha$ and $C_\beta$ will be $|e_2| + |e_3|$. However, if we reassign $f_b$ to cluster $C_\beta$ (with which it has strong connectivity) the border length contribution of $f_b$ will now be $|e_1|$. From triangle inequality, it holds that $|e_1| \leq |e_2| + |e_3|$, thus the border length will decrease. The above example is illustrated in Figure 10.2.

The reduction of the border length leads to a subsequent smoothing of the border between two clusters. So, we should iterate through all $M$'s triangles and reassign border triangles that are not strongly connected with their cluster. With the reassignments, there will occur new border triangles. Hence, we have to re-iterate through $M$'s triangles to ensure that no non-strongly border triangle remains. When all border triangles are strongly connected with their cluster the smoothing process ends. We should note here that a reassignment of triangle to a neighboring cluster should be performed only if the triangle is visible from its new cluster's image ($V = 1$).

### 10.1.2 Discarding small area texture clusters

With the completion of the border smoothing step, the remaining texture clusters will have more compact boundary structures. However, among the texture clusters, there will also exist some small area clusters that add extra interactions between texture clusters.

Considering the view criterion $\mathcal{Q}$, the current texture clusters define an optimal assignment of $M$'s triangles to photogrammetry images. On the other hand, it is

Figure 10.3: Small clusters appearing between larger clusters.

not worth having an extra cluster that will possibly offer a slight view advantage in comparison to the incompatibility it will introduce with its neighboring clusters. Thus, we set a minimum area threshold that a certain texture cluster must cover to give rise to a cluster. We select to set the cluster threshold as a fraction of the total area of our object reconstruction $M$. We set this threshold as the $\frac{1}{400}$ of the total object's area. If a cluster's area is less than this threshold, the cluster is disassembled and its triangles are redistributed to the neighboring clusters.

The triangle redistribution policy that we adopt, aims for the equal expansion of the neighboring clusters to the region of a discarded texture cluster, let $C_d$. To preserve the connectivity of the clusters' triangles we will begin the reassignment from the border triangles of $C_d$. Each border triangle will have at least one neighbor that belongs to a different texture cluster. These clusters will constitute the candidate new clusters for the current border triangle. In the case of multiple candidate clusters, the assignment will be determined by the minimization of criterion $\mathcal{Q}$. With the redistribution of the border triangles to neighbor clusters, new borders will arise to which we will apply the same operation. The procedure is repeated until no triangle is left in $C_d$.

### 10.1.3 Outcome

With completion of this step, we get an update of the texture cluster shapes which acquire a more compact structure.

## 10.2 The transition from top to bottom view setups

The object surfaces covered by two neighboring texture clusters are imaged (receive texture) from different photogrammetric views. Thus, the texture consistency at their border depends on if their triangles are mapped to matching regions in

the two photogrammetric images. For clusters textured by images of the same setup, the projection matrices' compatibility is provided by the photogrammetry algorithm, typically by virtue of a pertinent bundle adjustment refinement. In the case of neighboring texture clusters textured from opposite-view setups though, the mapping of textures is accomplished through the $T_t$ and $T_b$ transformation matrices.

### 10.2.1 Motivation

In our registration approach in Chapter 8, the registration of $M$ with the two photogrammetric meshes is treated as two independent optimization problems which are solved by PSO. PSO is not guaranteed to always converge to a globally optimal solution but may converge to a near-optimal result. Thus, the resulting transformations $T_t$ and $T_b$ will only provide an approximate registration of $M$ with the two photogrammetric meshes.

More specifically, for the effective interconnection of the opposite object's views, it required that the $T_t$ and $T_b$ transforms are consistent in the way that they approach their respective photogrammetric meshes' pose. As a result, any small deviation in the relative pose of $M$ with the two photogrammetric reconstructions can lead to inconsistent mappings of the 3D triangles to the image regions of the two image sets. The problem is magnified when the object's structure presents rotational symmetries which promote even more serious pose inconsistencies.

These phenomena can compromise the continuity between opposite-view texture clusters.

### 10.2.2 Rationale

To deal with the discontinuity of opposite-view texture clusters, we consider a setup view of the object as the reference and the other as the source. Specifically, the selection of the reference view is determined by selecting the setup view (top or bottom) for which PSO yielded the smallest registration error in Chapter 8. With the reference view set, we will search for a transform that refines $T_S$, or otherwise the registration transform of the opposite setup view.

In this context, we consider each pair $p_k$ of neighboring texture clusters that receive texture from opposite setup views. A pair $p_k$ will have a cluster $C_R$ that is textured from a reference view's image $I_R$ and a cluster $C_S$ that is textured by an opposite view image $I_S$. Among $p_k$'s triangles, there will exist a subset $S_k$ of triangles that are fully visible ($V = 1$) from both images $I_R$ and $I_S$. Thus, we can get the mapping of $S_k$ triangles in both images $I_R$ and $I_S$ (see Figure 10.4) through the registration transforms $T_t$ and $T_b$ and the projection matrices that correspond to the images. In this way, we get the imaging of $S_k$ triangles in both images. If that imaging is consistent in both images it means that there is texture continuity between the two clusters $C_R$ and $C_S$.

We capitalize on this property to specify the transform $T_S$ which achieves

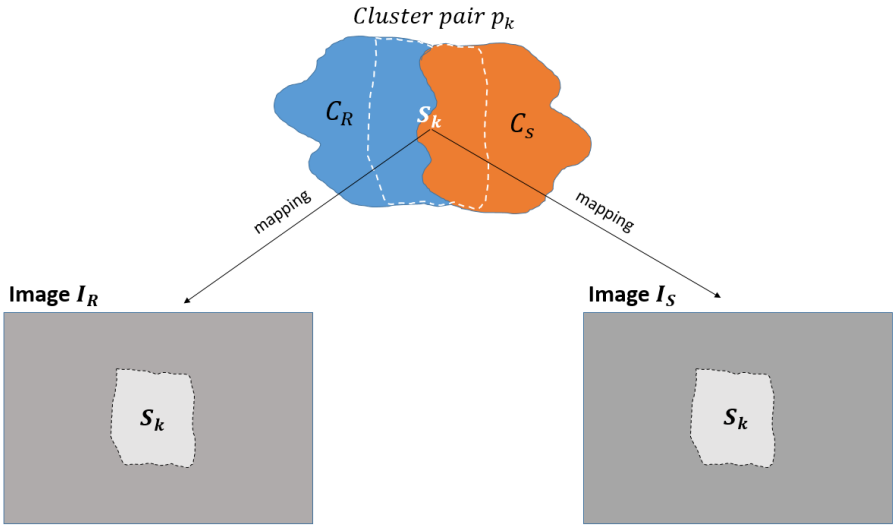



Figure 10.4: The mapping of $S_k$ triangles to images $I_R$ and $I_S$.

continuity between opposite view cluster pairs. We cast the search of this transform as an optimization problem that is also solved by PSO.

### 10.2.3 Optimization parameters

The transform that we are looking for, will be an update to the refined transform that we calculated in Chapter 8 or, in other words, a second refinement. Hence, the pose correction will concern the location, rotation, and scaling of $M$ in this setup view. As in Chapter 8, the candidate solution vectors will have the following form: $[s_x, s_y, s_z, r_x, r_y, r_z, x, y, z]$. The transformation parameters define a 9D search space. To retrieve the transformation defined by a solution vector $\mathbf{v}$ we use the methodology presented in Chapter 8.5.

### 10.2.4 Domain of parameters

Despite the small registration errors, the transforms $T_t$ and $T_b$ already provide a good registration of $M$ with the photogrammetric meshes. Thus, the expected deviation of $M$ from the target pose will be rather small. This enables us to bind the parameter values. The scaling parameters $s_x$, $s_y$, $s_z$ should take values near 1 and the rotation and translation parameters $r_x$, $r_y$, $r_z$ and $x$, $y$, $z$ will take values around 0. In the case of rotational symmetry of the object, it is recommended that the rotation parameters' domain is set looser so that we are able to recover from the direr rotational inconsistencies.

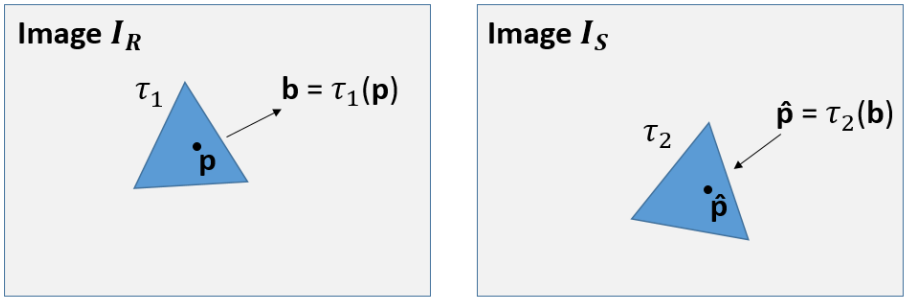



Figure 10.5: The mapping of triangle $t$ to images $I_R$ and $I_S$. $\hat{\mathbf{p}}$ is estimated through the barycentric coordinates $\mathbf{b}$.

### 10.2.5 Evaluating texture continuity in a texture cluster pair

For each cluster pair $p_k$, the mapping of triangles $S_k$ to image $I_R$ cover a sub-region of $I_R$. We can describe this sub-region with a binary mask $M_k$ of equal size with $I_R$. The pixels of $M_k$ that are covered by $S_k$ triangles are set to (1) and denote its foreground pixels. The area defined by $M_k$ in $I_R$ constitutes a template for the imaging of $S_k$ triangles in image $I_S$. Thus, we can construct a template image $I_T^k$ as the sub-image of $I_R$ that is cropped by the bounding box $b_k$ of the foreground pixels of $M_k$.

Ideally, we would like that the imaging of the $S_k$ triangles through a correction transform $T(\mathbf{v})$ in the image $I_S$, is highly correlated with the template image $I_T^k$. This would mean that the candidate transformation vector $\mathbf{v}$ achieves continuity between the intersecting clusters of $p_k$. Nevertheless, images $I_R$ and $I_S$ illustrate the object from different viewing perspectives. Thus, the shape of the mapped $S_k$ triangles in these images will be different. Therefore, we cannot directly (pixel-by-pixel) compare the imaging of $S_k$ in $I_R$ and $I_S$.

To overcome this issue we construct an image $I_S^k$ of equal size with $I_R^k$ and we fill its pixels with the corresponding intensities of $I_S$ as follows.

Let a 3D triangle $t$ of $S_k$. Triangle $t$ will have two 2D mappings $\tau_R$, $\tau_S$ in $I_R$ and $I_S$ respectively. The 2D triangle $\tau_R$ contains a number of integer pixel coordinates. For each contained pixel $\mathbf{p}$, we calculate its barycentric coordinates $\mathbf{b}$ with respect to $\tau_R$. By applying $\mathbf{b}$ to the triangle $\tau_S$ we get the corresponding point $\hat{\mathbf{p}}$ in $I_S$ image. This process is illustrated in Figure 10.5. The point $\hat{\mathbf{p}}$ will not necessarily have integer coordinates so it cannot be directly related to a pixel intensity in $I_S$. Thus, we perform bilinear interpolation to acquire the corresponding intensity $I_S(\hat{\mathbf{p}})$. The image formation equation is $I_S^k(\mathbf{p}) = I_S(\hat{\mathbf{p}})$

By repeating the above procedure for all triangles in $S_k$, the $I_S^k$ image will constitute the equivalent image of $I_S$ in comparable form with $I_R$. We denote the construction of the equivalent image $I_S^k$ for a cluster pair $p_k$ and a given transformation vector $\mathbf{v}$ as $E(p_k, T(\mathbf{v}))$.

We assess the texture continuity for a given transformation vector $\mathbf{v}$ by comparing $I_T^k$ and $I_S^k$ in pixels determined by the foreground pixels of $M_k$ mask. We select as the comparison method of the two images the normalized cross-correlation. The normalized version will compensate for any possible photometric differences that can be observed between the images of the two clusters. We quantify the texture continuity of a cluster pair $p_k$ for a given transformation vector $\mathbf{v}$ as:

$$C(\mathbf{v}, k) = ncross(I_T^k,\ E(p_k, T(\mathbf{v})),\ M_k)$$

where *ncross* denotes normalized cross-correlation operation and $M_k$ mask determines the pixels to be compared in the two images. The optimization goal is the maximization of the above score.

### 10.2.6 Quantifying the visual information of cluster pairs

In qualitative terms, the cluster pairs are not of equal significance. The reason is that some will contain homogeneous textures and others will contain rich textures. When registration is inaccurate, the latter yields an asymmetrical loss of visual quality of the result because they are more salient to the human observer.

To represent this qualitative property we use the notion of "visual information" and quantify it as the intensity variance of the pertinent texture. Specifically, we assess the "visual information" of a cluster pair $p_k$ as the variance $v_k$ of its template image $I_T^k$ at the pixels marked by the foreground pixels of mask $M_k$. The calculated variance constitutes a measure of the inhomogeneity of the texture associated with $p_k$ and denotes the importance of $p_k$.

### 10.2.7 Objective function

The $C(\mathbf{v}, k)$ quantifies the texture continuity achieved by a transformation vector $\mathbf{v}$ on a given cluster pair $p_k$. To formulate the total evaluation of $\mathbf{v}$, we need to combine the texture continuity score from all the available cluster pairs $p_k$. However, the contribution of each pair is not equal. As discussed in Chapter 10.2.6, the cluster pairs with higher "visual information" are the most critical to match. This information is encoded in the variance $v_k$ of each cluster pair $p_k$. Thus, we choose to consider only the pairs $p_k$ whose variance is above a variance threshold $V$. We set $V = 5$.

We formulate the objective function as the average of texture continuity scores $C(\mathbf{v}, k)$ of cluster pairs $p_k$ weighted by their respective variances $v_k$. noted as below:

$$C_t(\mathbf{v}) = \frac{1}{\sum_{k=1}^{N} v_k} \cdot \sum_{k=1}^{N} v_k \cdot C(\mathbf{v}, k)$$

### 10.2.8 Outcome

The outcome of this optimization step is a refinement transform for $T_S$. With the refined version of $T_S$ we update the texture mapping of $M$'s triangles that correspond to clusters of the source view. In this way, we achieve a continuous texture mapping between the texture clusters of $M$.

## 10.3 Texture blending

The registration refinement performed in Chapter 10.2, rectifies the texture discontinuities between clusters that are textured from images acquired from opposite views. However, despite the geometric alignment, the neighboring clusters will still present issues that concern their photometric consistency. To address these problem, we perform blending to the areas of neighboring clusters that are visible from both clusters' images.

### 10.3.1 Blending procedure

The triangles of $M$ receive texture from the photogrammetric images. For blending purposes, we create a copy of each photogrammetric images. The intensities that will occur from the blending of pixels of each cluster will replace the original intensities in the copy of the cluster's image. Hence, these copies will constitute the new texture sources for the triangles of $M$ after the blending.

The blending methodology that we propose includes a separate blending process for each texture cluster. A texture cluster will be blended with respect to all of its neighboring clusters. Let $C_1$ and $C_2$ be two neighboring clusters and $I_1$ and $I_2$ their respective images. We blend the texture of $C_1$ with respect to $C_2$ as follows.

The blending process will take place to the subset $S$ of $C_1$'s triangles that are fully visible ($V = 1$) in $I_2$ image.

Let $t$ be a triangle in $S$. Triangle $t$ is mapped to $I_1$ and $I_2$ as two image triangles $\tau_1$ and $\tau_2$ respectively. The triangle $\tau_1$ will contain a number of pixels in image $I_1$. For each such pixel $\mathbf{p}$ with image intensity $I_1(\mathbf{p})$, we calculate its barycentric coordinates $\mathbf{b}$ with regard to $\tau_1$. Then, we apply $\mathbf{b}$ to triangle $\tau_2$. This essentially yields $\mathbf{p}$'s corresponding point in image $I_2$. Let that point be $\hat{\mathbf{p}}$. This process is shown in Figure 10.7.

$\hat{\mathbf{p}}$ may have decimal coordinates. To associate it with an image intensity in $I_2$, we perform bilinear interpolation to its coordinates. In this way, we acquire the corresponding image intensity $I_2(\hat{\mathbf{p}})$. Finally, we calculate the blended image intensity $I_1^b(\mathbf{p})$ at $\mathbf{p}$ as the following weighted average:

$$I_1^b(\mathbf{p}) = \alpha_{\mathbf{p}} \cdot I_1(\mathbf{p}) + (1 - \alpha_{\mathbf{p}}) \cdot I_2(\hat{\mathbf{p}}),$$

where $\alpha_p$ is a weighting coefficient that corresponds to pixel $\mathbf{p}$ and measures the contribution of the two image intensities in the blended intensity. The value of

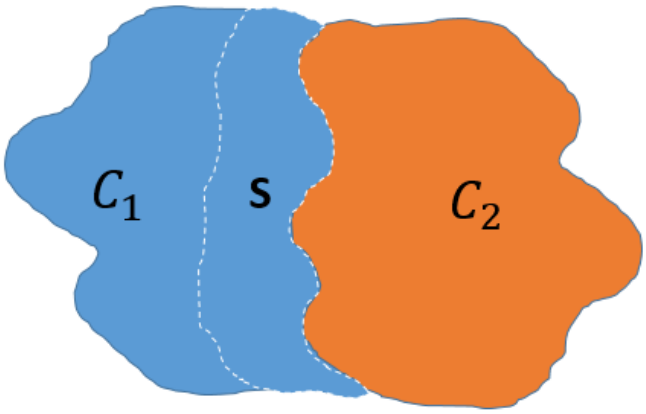



Figure 10.6: The triangle subset $S$ in cluster $C_1$.

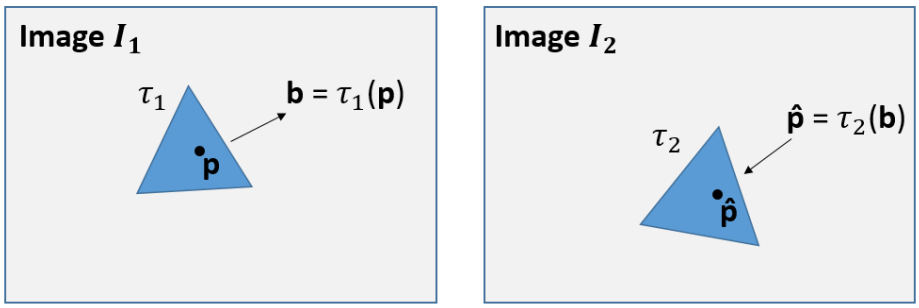



Figure 10.7: The mapping of $t$ to images $I_1$ and $I_2$. $\hat{\mathbf{p}}$ is estimated through the barycentric coordinates $\mathbf{b}$.

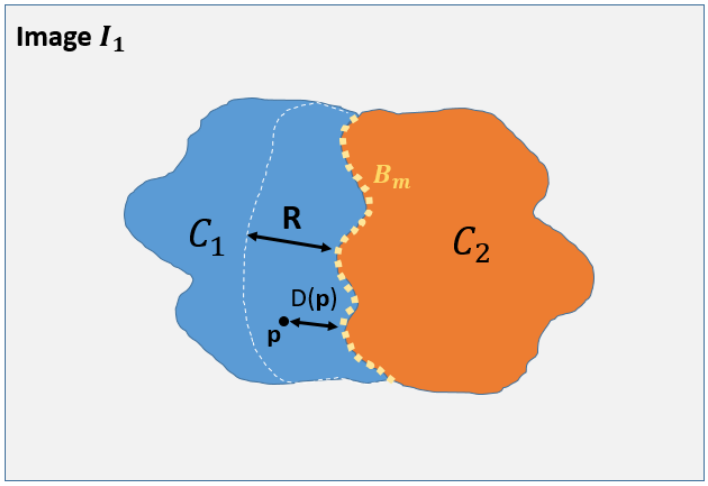



Figure 10.8: The blending radius $R$ and the distance of a pixel $\mathbf{p}$ from the border.

this weight is determined in Chapter 10.3.2. The intensity $I_1^b(\mathbf{p})$ is stored at the copy of image $I_1$.

By applying the aforementioned procedure for all triangles in $S$, we blend $C_1$'s texture with respect to its neighboring cluster $C_2$. The blending procedure will continue for the rest neighbors of $C_1$. In this way, we accomplish the complete blending of $C_1$. The blending process finishes when all texture clusters are processed.

### 10.3.2 Blending weight

The purpose of blending is to create a smooth visual transition from the in-process cluster $C_1$ to $C_2$. To accomplish this, we need to properly adjust the coefficient $\alpha_p$ for each processed pixel $\mathbf{p}$ in $I_1$. The farthest a pixel is from the clusters' border the higher the contribution of $I_1(\mathbf{p})$ should be in the blended intensity and, thus, the value of $\alpha_p$.

To this end, we define the border $B$ between texture clusters $C_1$ and $C_2$ as the set of common vertices of their border triangles. We map the vertices $B$ to $I_1$ and we denote the mapped border vertices as $B_m$. Moreover, we define the distance $D(\mathbf{p})$ of a pixel $\mathbf{p}$ from border $B_m$ as the minimum (Euclidean) distance between $\mathbf{p}$ and the border points $B_m$.

The mapping of triangles $S$ to $I_1$ specify the blending area in $I_1$. Thus, we define the blending radius $R$ as the maximum distance between the vertices of the mapped triangles of $S$ and the border $B_m$. We define the relative distance of a pixel $\mathbf{p}$ from $B_m$ as the ratio $\frac{D(\mathbf{p})}{R}$. This ratio's domain is $[0, 1]$.

The goal of blending is to enforce a linear variation of the blending weight from 0.5 to 1 as we move away from the border. Therefore, for a given pixel $\mathbf{p}$ we

calculate the weighting coefficient $\alpha_{\mathbf{p}}$ as:

$$\alpha_{\mathbf{p}} = 1 - 0.5 \cdot (1 - \frac{D(\mathbf{p})}{R})$$

Pixels which are close to $B_m$ will take weights near 0.5 which is translated to equal contribution of the corresponding intensities of $I_1$ and $I_2$ to the blended intensity. Contrarily, pixels far from the border will take weights near 1 which leads to an exclusive influence of the blended intensity from $I_1$.

### 10.3.3 Outcome

The blending step yields new photogrammetric images which contain the blended intensities of texture clusters' triangles. Using these images as texture sources of $M$ we accomplish a balancing of the photometric differences between neighboring clusters.

## 10.4 Outcome

The previous three steps optimize the appearance of $M$ by correcting texture discontinuities and photometric differences between neighboring texture clusters. This constitutes the final step of our method and returns the final reconstruction of the object.

# Chapter 11

# Experimental evaluation

In this section, we perform an experimental evaluation of our reconstruction method with the use of two datasets. Initially, we present results regarding the performance of our method running on the two datasets. Next, we provide comparative results to assess the contribution of each method's step to the overall improvement of the texture quality. Finally, we present the output reconstructions created by our method.

## 11.1 Dataset description

For the experimental evaluation of our method, we acquired two datasets from two ceramic objects. During the acquisition of datasets, the guidelines presented in Chapter 4 were followed. Dataset 1 contains the reconstructions of a ceramic teacup shown in Figure 11.1. Dataset 2 captures a simple ceramic water cup shown in Figure 11.2.

### 11.1.1 Digitisation modalities

The equipment that we used for the capturing of both objects was the same. In specific, for the capturing of the 3D scans, we utilized the Faro Freestyle style scanning modality. It is a handheld 3D scanner based on trinocular stereo, imperceptible active illumination, and inertial motion estimation. Regarding the photogrammetry, the input images used for the creation of each reconstruction were taken using a Canon DSRL camera at the resolution of $8256 \times 5504$ pixels.

### 11.1.2 Dataset information

The 3D capturing of the two objects yielded eight reconstructions, two 3D scans and two photogrammetric models, for each object entity. Table 11.1 and table 11.2 summarize the information of the resulting 3D scans and photogrammetric meshes of both datasets. Specifically, each table reports the number of vertices and triangles of the created meshes and their total size in memory (in MB). Table 11.3

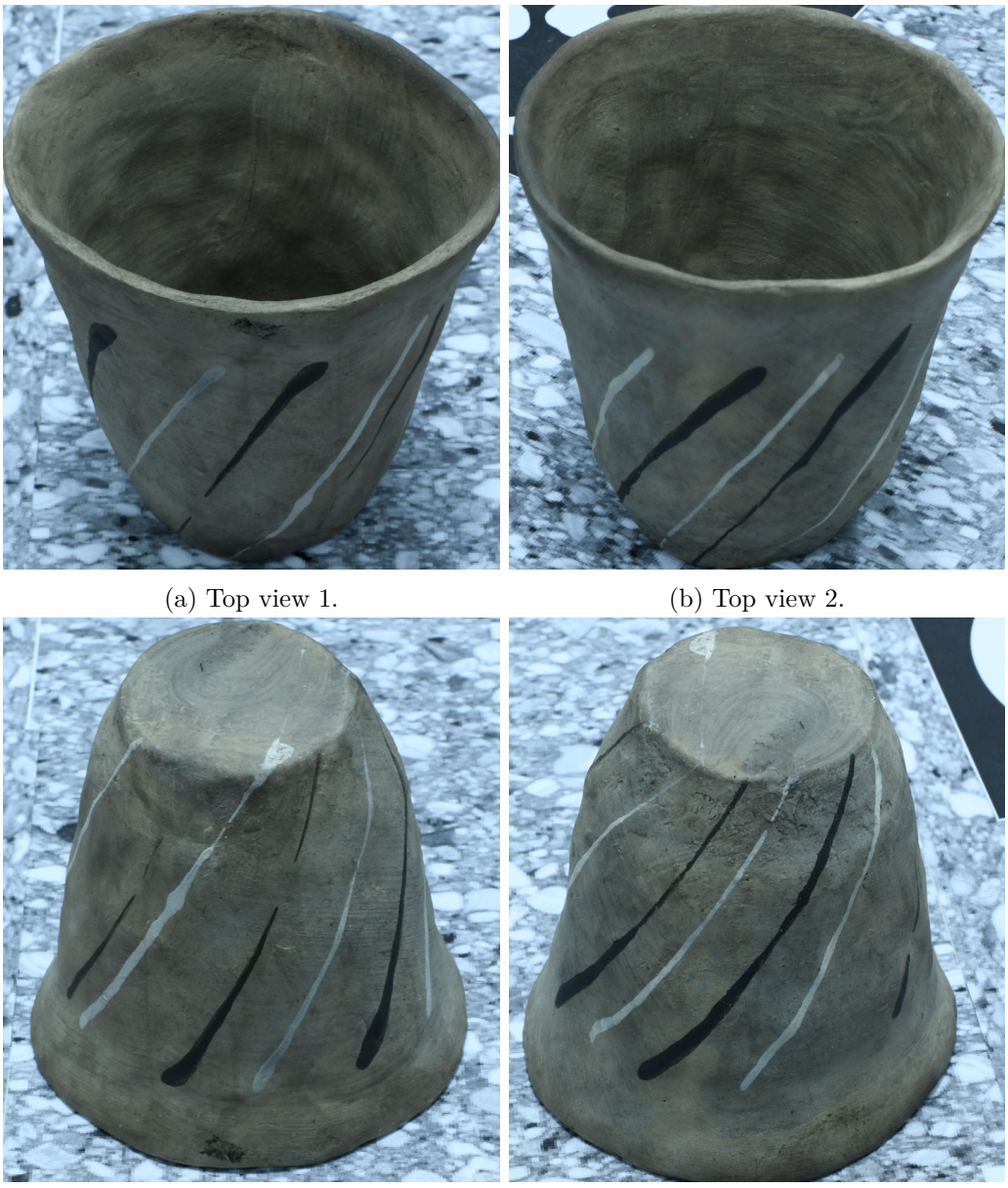
(a) Top view 1.

(b) Top view 2.

(c) Bottom view 1.

(d) Bottom view 2.

Figure 11.1: The object captured in dataset 1 from different viewing perspectives.



(a) Top view 1.

(b) Top view 2.

(c) Bottom view 1.

(d) Bottom view 2.

Figure 11.2: The object captured in dataset 2 from different viewing perspectives.

summarizes the information of the images captured for the purposes of the photogrammetry in both datasets. We report the number of images taken from each scene (top or bottom), their resolution and the total size of the images (in MB).

| | View | Vertices | Triangles | Mesh size (in MB) |
|---|---|---|---|---|
| **Dataset 1** | Top | 163879 | 324995 | 28.7 |
| | Bottom | 163732 | 325000 | 25.6 |
| **Dataset 2** | Top | 163531 | 325003 | 36.1 |
| | Bottom | 163526 | 324989 | 35.9 |

Table 11.1: The 3D scan models details in both datasets.

| | View | Vertices | Triangles | Mesh size (in MB) |
|---|---|---|---|---|
| **Dataset 1** | Top | 2428385 | 4977229 | 365.1 |
| | Bottom | 2496046 | 4999999 | 364.0 |
| **Dataset 2** | Top | 2469043 | 5000000 | 365.2 |
| | Bottom | 2495246 | 5000000 | 368.1 |

Table 11.2: The photogrammetric models details in both datasets.

| | View | Images | Resolution | Total size (in MB) |
|---|---|---|---|---|
| **Dataset 1** | Top | 21 | 8256×5504 | 2866 |
| | Bottom | 20 | 8256×5504 | 2730 |
| **Dataset 2** | Top | 21 | 8256×5504 | 2866 |
| | Bottom | 21 | 8256×5504 | 2866 |

Table 11.3: The information of images captured for the photogrammetry for each dataset.

## 11.2 Performance evaluation

To evaluate the computational performance of our reconstruction method, we proceeded to the execution of our method for both ceramic objects' datasets. From these executions, we took measurements regarding the total execution time and the memory footprint of each method's step. Specifically, the performance analysis that we present is split into the following modules:

1. **Marker localization**: Implements steps described in Chapter 5.
2. **Model registration**: Implements steps described in Chapter 6.
3. **Model creation**: Implements steps described in Chapter 7.

| | Execution time (min) | |
|---|---|---|
| | **Dataset 1** | **Dataset 2** |
| **Marker localization** | 4.05 | 4.21 |
| **Model registration** | 1.18 | 1.03 |
| **Model creation** | 0.31 | 0.29 |
| **Registration refinement** | 109.8 | 113.6 |
| **Texture Mapping** | 1.03 | 0.59 |
| **Cluster processing** | 2.15 | 2.18 |
| **Registration refinement 2** | 22.03 | 14.48 |
| **Blending** | 11.41 | 9.18 |

Table 11.4: The execution time of each module for both datasets.

4. **Registration refinement**: Includes two executions of the optimization process described in Chapter 8 (one execution for each view).

5. **Texture mapping**: Implements steps described in Chapter 9.

6. **Cluster processing**: Implements steps described in Chapter 10.1.

7. **Registration refinement 2**: Implements steps described in Chapter 10.2.

8. **Blending**: Implements steps described in Chapter 10.3.

### 11.2.1 Computation parameters

In the context of our method, there are two types of optimization problems (see Chapter 8 and 10.2). Each problem is described by an objective function which is optimized through PSO. The effectiveness of the optimization performed by PSO as well as its running time is determined by the number of particles and iterations. In our experiments, we empirically set the iterations and particles of PSO to 25 and 50, respectively, for both optimization problems.

### 11.2.2 Computational cost

The coding language that we used for the implementation of each module is C++. All the experiments were conducted on a system equipped with an i7-7700HQ processor and 8GB of RAM. Table 11.4 summarizes the total execution time (in minutes) of each module. Table 11.5 presents an approximation of the maximum memory required for each module. This calculation essentially indicates the maximum volume of data that was required to be simultaneously loaded in the processes' memory throughout the execution of each module.

| | ≈ Max memory (MB) | |
|---|---|---|
| | **Dataset 1** | **Dataset 2** |
| **Marker localization** | 501 | 500 |
| **Model registration** | 193 | 201 |
| **Model creation** | 54 | 72 |
| **Registration refinement** | 431 | 453 |
| **Texture Mapping** | 29 | 33 |
| **Cluster processing** | 35 | 38 |
| **Registration refinement 2** | 661 | 328 |
| **Blending** | 301 | 309 |

Table 11.5: The approximate maximum memory used by each module for both datasets.

### 11.2.3 Discussion

By observing the Tables 11.4 and 11.5 we deduce that the most computationally expensive modules are "Marker localization", "Registration refinement 1 and 2" and "Blending". We proceed to a brief analysis of each one.

The "Marker localization" module has a relatively fast execution time in both datasets. However, it has one of the biggest impacts on the memory. This owing to the fact that the reconstructions' meshes need to be loaded in memory for the localization of the 3D positions of the markers and the removal the floors. Consequently, in the case of photogrammetric meshes, which are large inputs, the expected memory usage is high. With the execution of this module, the size of all reconstructions (including photogrammetric meshes) is significantly reduced due to the removal of the floors' triangles. So, the loading of reconstructions in memory becomes less heavy in the next modules.

The "Registration refinement 1" module takes up to 70% of the total execution time for both datasets. That is explained by the fact that this module includes two optimization procedures which require the calculation of multiple projection masks for each candidate solution. Specifically, for a single optimization process running on 50 particles, 25 iterations and 21 images, will need to create a total of $(25 \cdot 50 \cdot 21) = 26.250$ projection masks. The creation of a projection mask is an expensive process as it requires an iteration through all triangles of $M$. As a result, the total execution time of this module is high. The memory consumption in this module is determined by the template and the cost masks which need to be saved in memory for their comparison with the projection masks of candidate solutions. This means that we need 21 binary masks and cost masks in the memory of equal size with the photogrammetric images. However, due to the mask cropping step (Chapter 8.7.1) the memory footprint of this module is significantly reduced.

In "Registration refinement 2" module, we observe a variation between the execution time and the memory required between the two datasets. This occurs due to the different number of cluster pairs that are considered for the alignment of

the texture of the two views. In dataset 1, 18 pairs are considered while in dataset 2 only 9. The total execution time of this module is determined by the construction of the images $I_k^S$ for each candidate solution. This requires the consideration of a small subset of $M$'s triangles in contrast to the previous registration module where all $M$'s triangles were considered. So, this optimization process is generally faster. Concerning the memory usage, this process needs to have in memory the template images and their corresponding masks. Thus, depending on the cluster pairs' number considered the memory usage will be higher.

The "Blending" module processes each pair of neighboring texture clusters. Thus, its execution time depends on the number of neighboring cluster pairs that exist in $M$. Throughout the whole blending procedure, only two photogrammetric images need to be loaded simultaneously in memory. So, the maximum memory required is fixed.

## 11.3 Qualitative evaluation

### 11.3.1 Comparative results

In this section, we assess the contribution of each step of our method to the improvement of the texture quality. Specifically, we consider four steps of our method. The registration refinement is described in Chapter 8 and the three view optimization steps are proposed in Chapter 10. To this end, we provide targeted views of the resulting reconstructions in areas where texture defects are initially detected to highlight the update provided by each step.

First, we examine the contribution of the refinement performed to the registrations $T_t$ and $T_b$ in Chapter 8. Figure 11.3 shows the projection of $M$'s vertices on a photogrammetric image. The left column illustrates the projection of the vertices with the use of unrefined transforms. The right column illustrates the same projection but with the refined versions of these transforms. As it occurs, the unrefined transforms manage to roughly approach the figure of the object. However, apparent inaccuracies can be spotted with some border vertices been projected to background areas (Figures 11.3c and 11.3g) or some others been projected to inner parts of the object (see Figures 11.3a and 11.3e). On the other hand, the refined transforms offer a clear upgrade to the initial transforms achieving a better coherence to the boundaries of the object and, thus, providing an overall better quality of texture mapping.

We also present figures that show the effect of the refinement of $T_t$ and $T_b$ to the quality of the texture mapped on $M$'s triangles. In the left column of Figure 11.4 we can see the texture mapping performed with the initial registrations while in the right column we can the texture mapping performed with the refined transforms. In Figure 11.4a we can observe some texture artifacts created on the texture due to registration misalignment which disappear in the refined version in Figure 11.4b. Furthermore, in Figure 11.4c we observe some small misalignment between the neighboring texture clusters with this phenomenon becoming even more serious

in Figure 11.4e. With the transformation refinement, the above observations are significantly limited (Figure 11.4f) or even disappear (Figure 11.4d).

Second, we examine the texture cluster compacting step that we describe in Chapter 10.1. The left column of Figure 11.5 shows the texture mapping performed to $M$. The right column shows a colored version of texture clusters so that we have a better perspective of the clusters' shape. The first and the third row illustrate the initial triangle assignment to texture clusters for both objects. We can observe that the visible clusters present irregular shapes and small holes which are translated to extra defects in the $M$'s texture. The second and the fourth row present the refined assignment of triangles to clusters. The shape of clusters has become more compact resulting in less cluster interactions and, thus, texture inconsistencies.

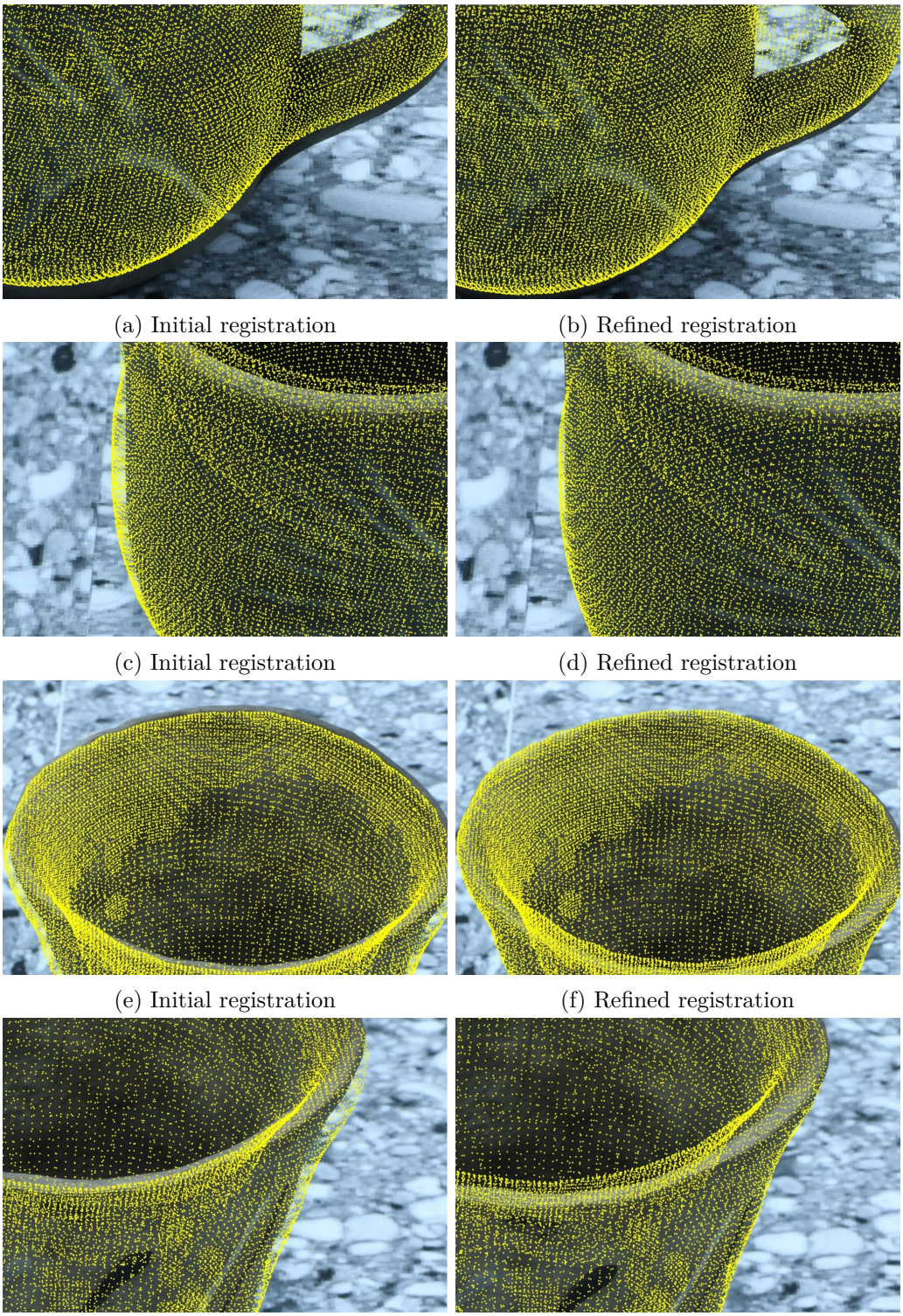
Next, we examine the contribution of the second registration refinement which targets the alignment of the texture clusters of opposite views (Chapter 10.2). The Figure 11.6 illustrates the texture mapping on areas of the two objects that are textured from images of opposite views. In the left column we can see the texture mapping before the registration refinement. As shown in Figures 11.6a and 11.6c, we observe incontinuities in the border between these clusters. With the refined alignment, we manage to enforce texture continuity to these areas (see Figures 11.6b and 11.6d).

Finally, we present the contribution of blending to the improvement of the texture quality. Figure 11.7 shows the texture clusters before blending (left column) and after blending (right column). The original texture appears noticeable photometric differences resulting to the appearance of seams between the neighboring texture clusters (see Figures 11.7a and 11.7c). With the blending step the seams are smoothed and the photometric inconsistencies are eliminated without a significant blurring of the overall texture.

### 11.3.2 Results and discussion

In this section, we present the output reconstructions of the two objects as returned from our method. Figure 11.8 shows four different perspectives of the output reconstructions of both ceramic objects.

By observing the images in Figure 11.8 we can see that the quality of texture for both reconstructions is relatively good. Specifically, the texture at the middle areas (Figures 11.8a-11.8d) of the two reconstructions, seams to be continuous and optically coherent despite the rich optical information encoded in these areas. This means that the method has achieved a fine registration of neighboring photogrammetric view. The same applies to the bottom areas of the two objects. Nevertheless, some texture discontinuities are observed at top areas of the objects and more precisely at the brim of the two cups. This texture issue is owing to geometric inaccuracies of the 3D scans. As shown in Figure 11.9 the brims of the cups are thinner in reality in comparison with the their reconstructed shape. Consequently, this thickness error from the 3D scanner leads to misaligned texture at these areas in the resulting reconstructions.



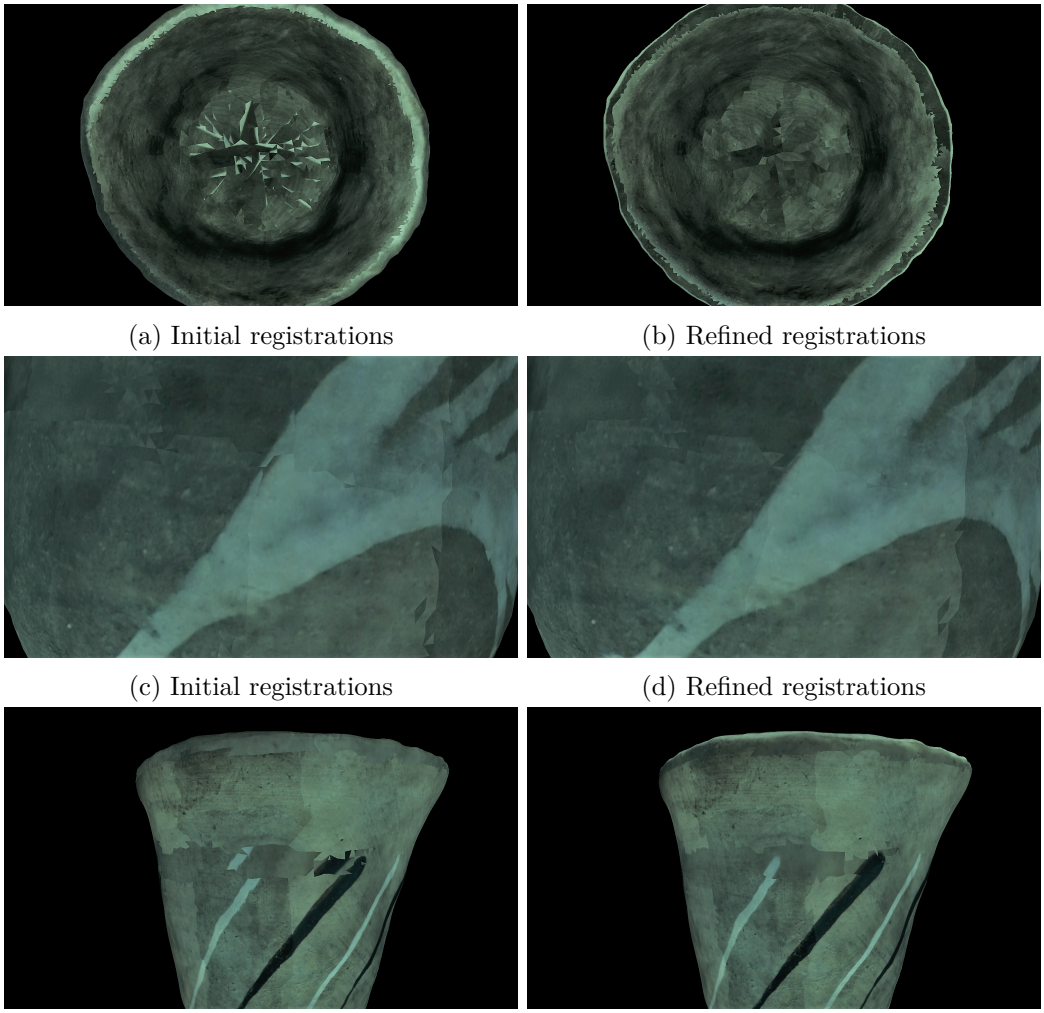
(a) Initial registration (b) Refined registration

(c) Initial registration (d) Refined registration

(e) Initial registration (f) Refined registration

(g) Initial registration (h) Refined registration

Figure 11.3: A comparison of the projected $M$'s vertices on a photogrammetric image with the use of the initial and refined transforms $T_t$ and $T_b$.

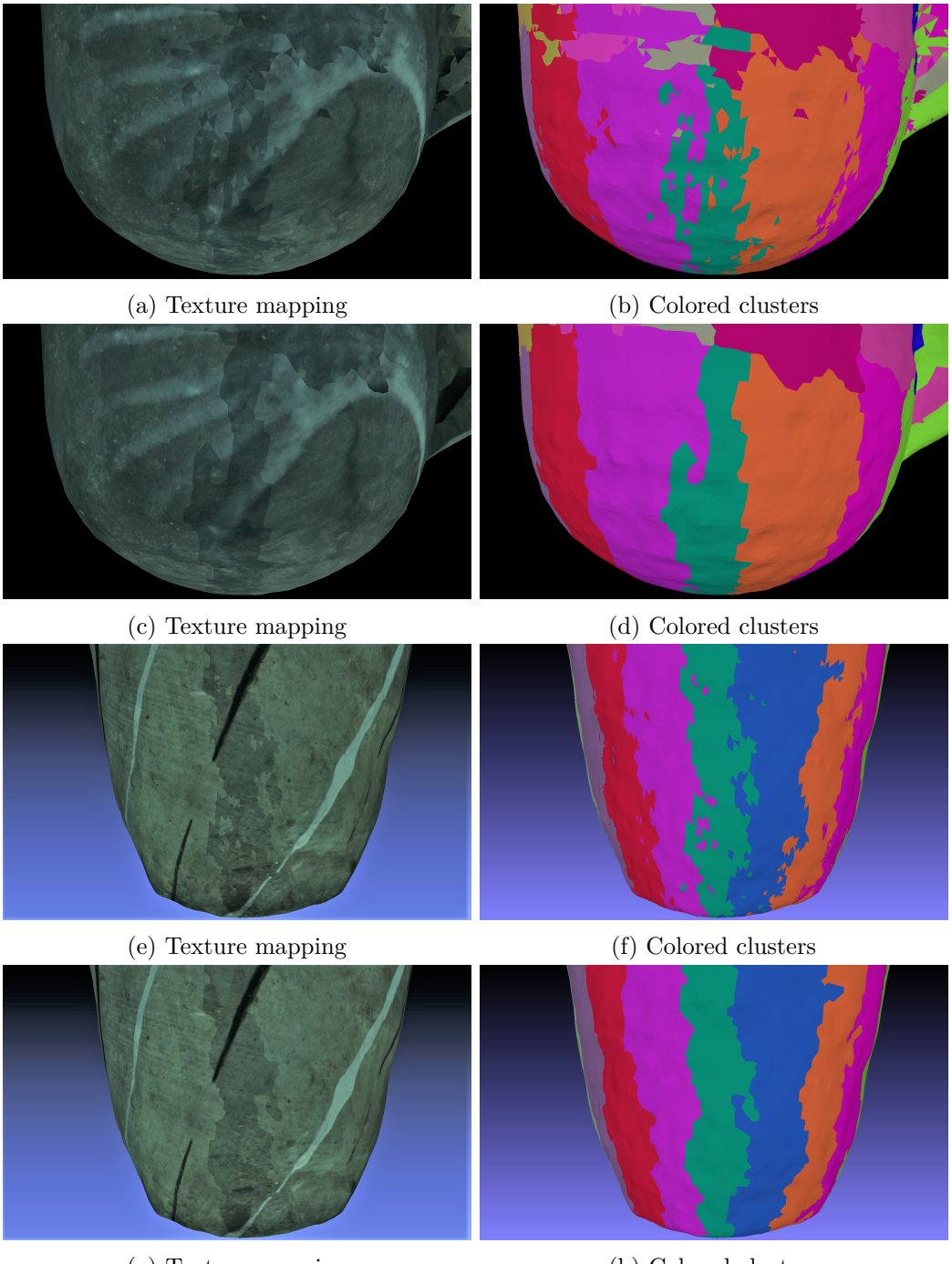


(a) Initial registrations (b) Refined registrations

(c) Initial registrations (d) Refined registrations

(e) Initial registrations (f) Refined registrations

Figure 11.4: A comparison of the texture mapping quality between the initial and the refined transforms $T_t$ and $T_b$.



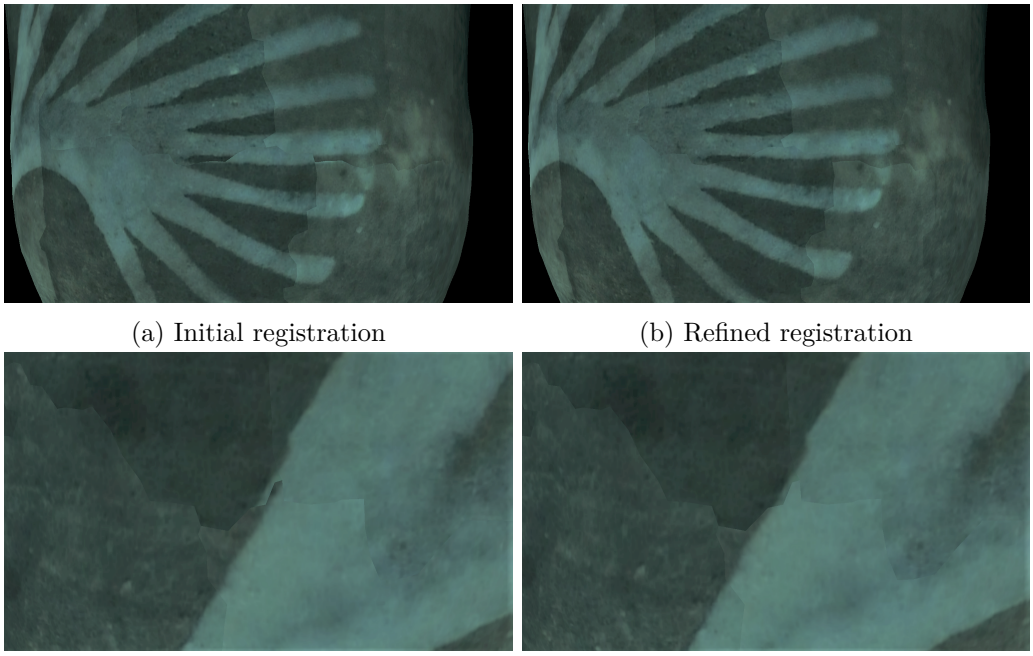
(a) Texture mapping (b) Colored clusters

(c) Texture mapping (d) Colored clusters

(e) Texture mapping (f) Colored clusters

(g) Texture mapping (h) Colored clusters

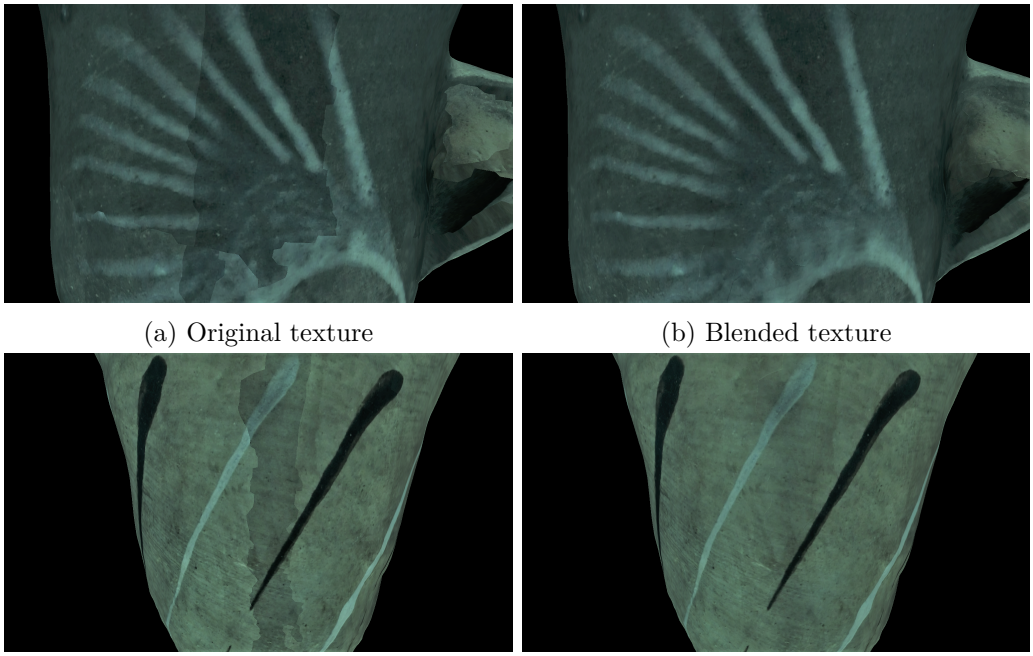
Figure 11.5: A comparison between the initial and the compacted texture clusters



(a) Initial registration (b) Refined registration

(c) Initial registration (d) Refined registration

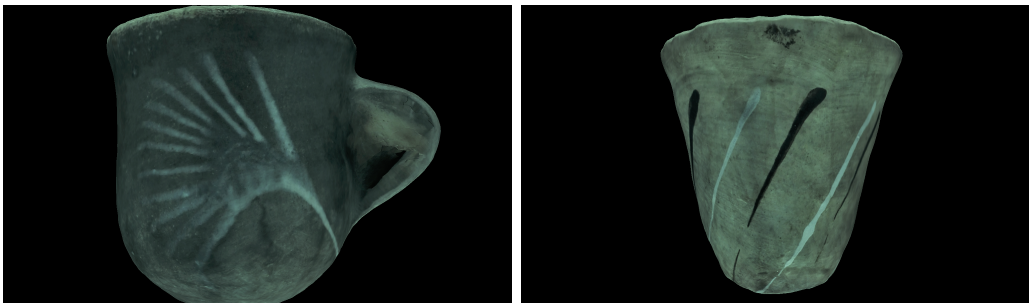
Figure 11.6: A comparison of texture continuity between neighboring top and bottom view clusters. between initial and refined alignment



(a) Original texture (b) Blended texture

(c) Original texture (d) Blended texture

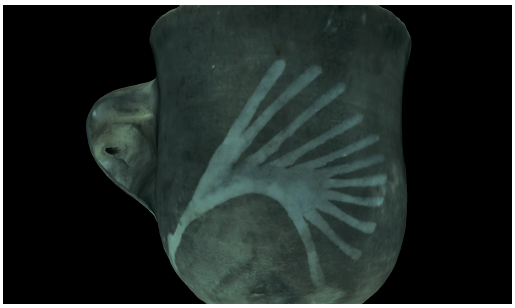
Figure 11.7: A comparison between the original and blended texture.



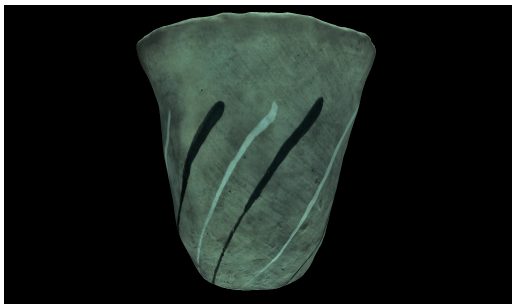
(a) Teacup middle part 1



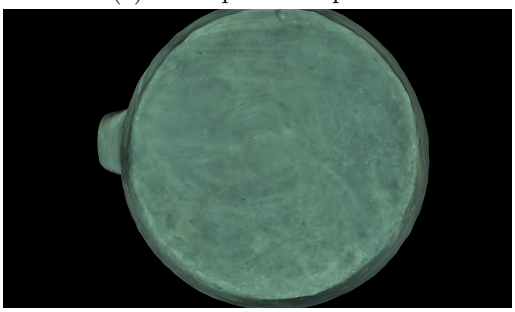
(b) Water cup middle 1



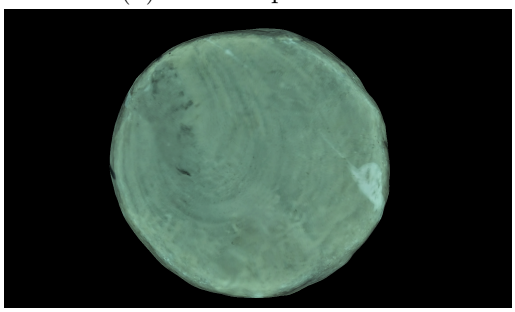
(c) Teacup middle part 2



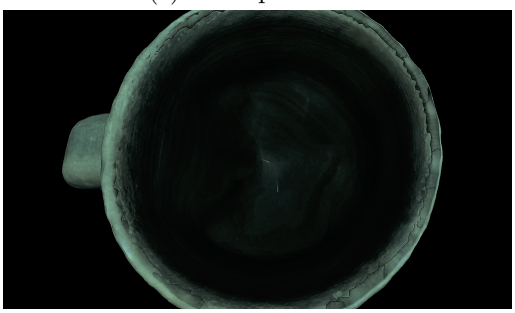
(d) Water cup middle 2



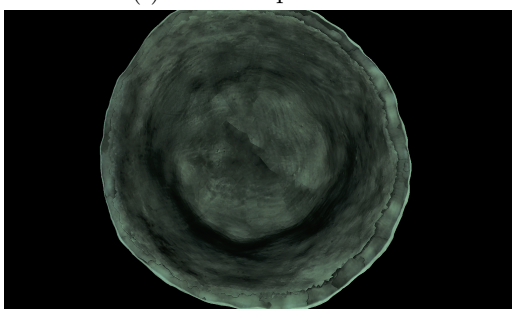
(e) Teacup bottom



(f) Water cup bottom

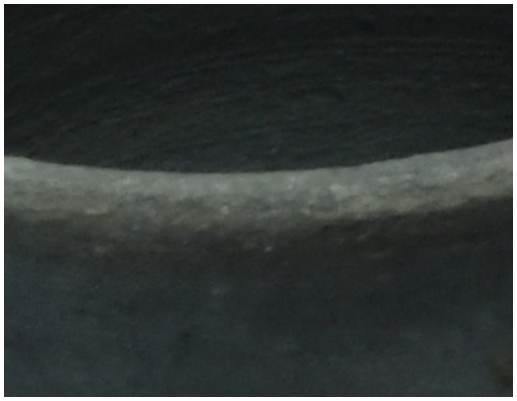


(g) Teacup top

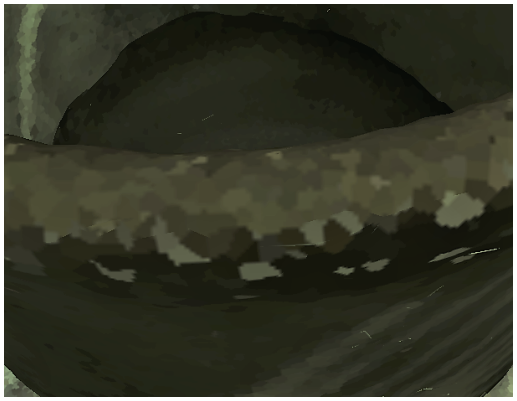


(h) Water cup top

Figure 11.8: The reconstructions of the two objects created by our method. The left column shows the ceramic teacup reconstruction. The right column shows the ceramic water cup reconstruction.

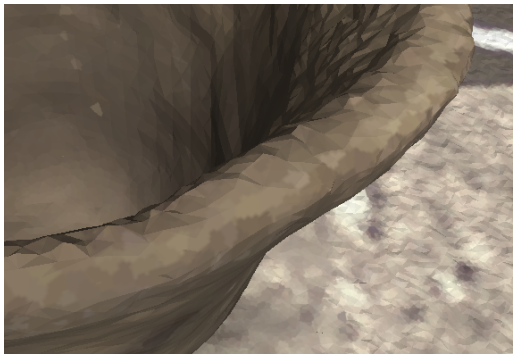


(a) Original brim thickness.



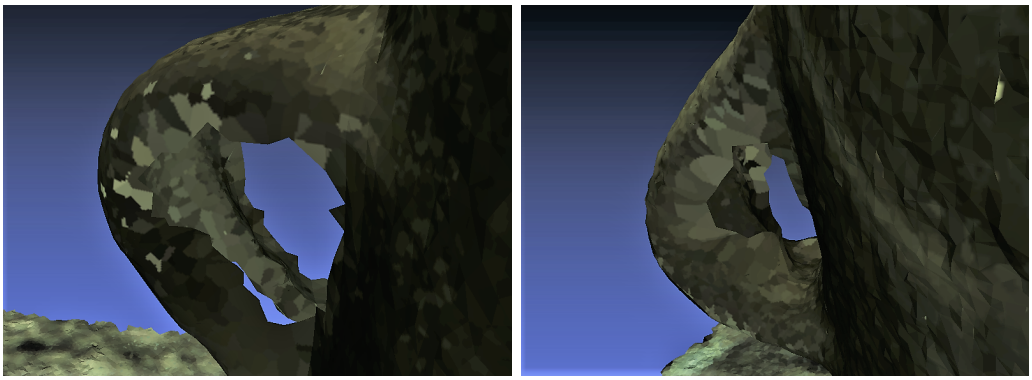
(b) Reconstructed brim thickness.

(c) Original brim thickness.



(d) Reconstructed brim thickness.

Figure 11.9: The difference at the thickness of the brims of the two cups.



(a) Top view scan. (b) Bottom view scan.

Figure 11.10: The handle of teacup as reconstructed from the two 3D scans.

Regarding the geometry, the water cup seams to be reconstructed realistically in the resulting mesh. On the other hand, one noticeable defect is observed at the handle of the ceramic teacup. Specifically, the hollow region inside the teacup's handle is filled in the reconstructed mesh which leads to a deformation of the original shape of the object. However, this problem is due to incomplete scanning of the teacup's handle from the 3D scanners. As shown in Figure 11.10 the handle is incompletely reconstructed in the 3D scans. As a result, the surface reconstruction algorithm used for the zipping in Chapter 7 treats these incomplete areas as a cut which fills with triangles. Consequently, the handle is covered in the final reconstruction.

Overall, our reconstruction method performs well in the tested datasets. The texture reconstruction seams to be accurate and optically sound with no apparent discontinuities and excessive blurring. Concerning the geometric quality of the resulting models the main issue is observed in the teacup object. However, the inconsistencies in its geometry are mainly a repercussion of the inaccuracies of the 3D scanner during the acquisition of 3D scans.

# Chapter 12

# Discussion and future work

## 12.1 Discussion

In this work, we presented a practical method to improve the quality of the 3D models we obtain using conventional and modest digitisation tools in a usage scenario that is very common when scanning objects in the laboratory. In particular, we discuss two frequent issues. First, that of having to scan the object in multiple placements (setups) to have complete digitisation of its surface. Second, of having digitisation modalities that of different advantages, in terms of geometry and texture reconstruction.

The motivation for this work targets the visual quality of these modes, particularly texture discontinuities, which have an important negative effect on the appreciation of the digitisation of these objects, particularly when they are works of art. In this quest, we explored the refinement of the geometric parameters that come into play, when combining partial scans and texture sources.

In this way, we acknowledge prior work in the "blending" texture seams as the last step in the combination of texture sources. The proposed work comes a step earlier than texture blending, in that it refines the overall registration of geometric models. Due to the improved registration, the residual errors that have to be concealed by texture blending are smaller in size.

## 12.2 Future work

### 12.2.1 Quantitative evaluation

For the complete evaluation of our reconstruction method, it is necessary that we provide a quantitative analysis of the result accuracy, in terms of geometry and texture reconstruction.

Geometrical evaluation usually entails the experimentation with 3D printed models, where the digital blueprints will be used as the ground truth. Depending on the scale of objects, this task entails the consideration of the printer's accuracy.

There are two problems to be considered. The first is that we are not primarily interested in measuring the accuracy of the 3D scanner or the photogrammetry method. These are already measured by their manufacturers and authors, respectively. Moreover, conventional 3D printing does not yet include the painting of texture upon 3D printed objects. To measure texture continuity the printed objects would have to be handpainted in a measurable way to obtain ground truth for texture. Reference points should be identified and manually measured with specific tools of the purpose, e.g., a caliper.

Photometric (i.e., color) evaluation of surface reconstruction is also demanding. In addition to known geometry, it requires ground truth regarding scene illumination and regarding the reflectance properties of the scanned surfaces. As such it would require the use of at least an optical absorption spectrometer (an optical emission spectrometer would be required if we are uncertain of the illumination properties). Moreover, some very common materials, such as marble, have non-Lambertian properties and exhibit translucency as well.

To this end, we need to study different metrics which quantify the precision of the object's geometry and the quality of texture of our method's outputs in comparison with the ground truth models.

### 12.2.2 Higher end scanners

This work is motivated by the need to create high-quality models, with conventional means, and with cost-efficiency. The IMU-assisted, active illumination 3D scanner used in this work has become a commodity in higher-end mobile devices. Still, inaccuracies particularly in the thickness of reconstructed objects are observed.

Undoubtedly, the method would benefit from the use of higher-end scanners, such as laser scanners. Though long-range (tens of meters) laser scanning is becoming an, still expensive, commodity, smaller-scale laser scanning is still challenging. The reason is that it is mainly used in industrial scanners along with specific turntable and/or robotic apparatuses. A similarity of this work to them is that they also require a data collection protocol to be fully automatic.

### 12.2.3 Multiple partial scans

The objects in the world can have varying shapes and amounts of geometric information on their surfaces. From this perspective, the scanning of an object in just two placements (upward and downward) may not be enough for the complete reconstruction of its geometrical details. Thus, our method could be extended to provide support for multiple object poses and incorporate information from more partial scans to the final reconstruction.

# Bibliography


[AFB19] Matthieu Armando, Jean-Sébastien Franco, and Edmond Boyer. "Adaptive Mesh Texture for Multi-View Appearance Modeling". In: *International Conference on 3D Vision (3DV)*. 2019, pp. 700–708. DOI: 10.1109/3DV.2019.00082.

[BA83] P. Burt and E. Adelson. "A Multiresolution Spline with Application to Image Mosaics". In: *ACM Trans. Graph.* 2.4 (1983), 217–236. ISSN: 0730-0301. DOI: 10.1145/245.247.

[Ber+99] F. Bernardini et al. "The ball-pivoting algorithm for surface reconstruction". In: *IEEE Transactions on Visualization and Computer Graphics* 5.4 (1999), pp. 349–359. DOI: 10.1109/2945.817351.

[BM92] P.J. Besl and Neil D. McKay. "A method for registration of 3-D shapes". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14.2 (1992), pp. 239–256. DOI: 10.1109/34.121791.

[BTA19] Burak Benligiray, Cihan Topal, and Cuneyt Akinlar. "STag: A stable fiducial marker system". In: *Image and Vision Computing* 89 (2019), pp. 158–169. ISSN: 0262-8856. DOI: 10.1016/j.imavis.2019.06.007.

[Cal+08] M. Callieri et al. "Masked photo blending: Mapping dense photographic data set on high-resolution sampled 3D models". In: *Computers and Graphics* 32.4 (2008), pp. 464–473. ISSN: 0097-8493. DOI: 10.1016/j.cag.2008.05.004.

[Che+12] Z. Chen et al. "3D Texture Mapping in Multi-view Reconstruction". In: *Advances in Visual Computing*. Berlin, Heidelberg: Springer, 2012, pp. 359–371. DOI: 10.1007/978-3-642-33179-4_35.

[Del+19] Ekaterini T. Delegou et al. "A Multidisciplinary Approach for Historic Buildings Diagnosis: The Case Study of the Kaisariani Monastery". In: *Heritage* 2.2 (2019), pp. 1211–1232. ISSN: 2571-9408. DOI: 10.3390/heritage2020079.

[Des+14] Arnaud Dessein et al. "Seamless texture stitching on a 3D mesh by Poisson blending in patches". In: *2014 IEEE International Conference on Image Processing (ICIP)*. 2014, pp. 2031–2035. DOI: 10.1109/ICIP.2014.7025407.

[DY18] C. Dostal and K. Yamafune. "Photogrammetric texture mapping: A method for increasing the Fidelity of 3D models of cultural heritage materials". In: *Journal of Archaeological Science: Reports* 18 (2018), pp. 430–436. ISSN: 2352-409X. DOI: 10.1016/j.jasrep.2018.01.024.

[Gal+10] R. Gal et al. "Seamless Montage for Texturing Models". In: *Computer Graphics Forum* 29 (2010). ISSN: 1467-8659. DOI: 10.1111/j.1467-8659.2009.01617.x.

[GC09] B. Goldluecke and D. Cremers. "Superresolution texture maps for multiview reconstruction". In: *IEEE International Conference on Computer Vision*. 2009, pp. 1677–1684. DOI: 10.1109/ICCV.2009.5459378.

[GDDA13] Ignacio Garcia-Dorado, Ilke Demir, and Daniel G Aliaga. "Automatic urban modeling using volumetric reconstruction with surface graph cuts". In: *Computers and Graphics* 37.7 (2013), pp. 896–910. ISSN: 0097-8493. DOI: https://doi.org/10.1016/j.cag.2013.07.003. URL: https://www.sciencedirect.com/science/article/pii/S0097849313001131.

[GFG12] F. Gabellone, I. Ferrari, and F. Giuri. "A Quick Method for the Texture Mapping of Meshes Acquired by Laser Scanner". In: *Geoinformatics FCE CTU* 9 (2012), pp. 17–26. DOI: 10.14311/gi.9.2.

[Gra+09] Nuno Gracias et al. "Fast image blending using watersheds and graph cuts". In: *Image and Vision Computing* 27.5 (2009), pp. 597–607. ISSN: 0262-8856. DOI: https://doi.org/10.1016/j.imavis.2008.04.014.

[Hua+20] J. Huang et al. "Adversarial Texture Optimization from RGB-D Scans". In: *CoRR* abs/2003.08400 (2020).

[Kab76] Wolfgang Kabsch. "A solution for the best rotation to relate two sets of vectors". In: *Acta Crystallographica Section A* 32 (1976), pp. 922–923.

[Kab78] Wolfgang Kabsch. "A discussion of the solution for the best rotation to relate two sets of vectors". In: *Acta Crystallographica Section A* 34 (1978), pp. 827–828.

[Kan+22] A. Kaneda et al. "A proposal of a new automated method for SfM/MVS 3D reconstruction through comparisons of 3D data by SfM/MVS and handheld laser scanners". In: *PLoS ONE* 17.7 (2022), e0270660. DOI: 10.1371/journal.pone.0270660.

[KH13] Michael Kazhdan and Hugues Hoppe. "Screened Poisson Surface Reconstruction". In: *ACM Trans. Graph.* 32.3 (2013). ISSN: 0730-0301. DOI: 10.1145/2487228.2487237. URL: https://doi.org/10.1145/2487228.2487237.

[LI07] Victor Lempitsky and Denis Ivanov. "Seamless Mosaicing of Image-Based Texture Maps". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2007, pp. 1–6. DOI: 10.1109/CVPR.2007.383078.

[MYA03] Adem Yasar Mülayim, Ulas Yilmaz, and Volkan Atalay. "Silhouette-based 3-D model reconstruction from multiple images". In: *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society* 33 4 (2003), pp. 582–591.

[NB95] W. Niem and H. Broszio. "Mapping Texture From Multiple Camera Views Onto 3D-Object Models For Computer Animation". In: *Proceedings of the International Workshop on Stereoscopic and Three-Dimensional Imaging*. 1995, pp. 99–105.

[Nea+06] Andrew Nealen et al. "Laplacian mesh optimization". In: Jan. 2006, pp. 381–389. DOI: 10.1145/1174429.1174494.

[Pag+15] R. Pagés et al. "Seamless, Static Multi-Texturing of 3D Meshes". In: *Computer Graphics Forum* 34.1 (2015), pp. 228–238. DOI: 10.1111/cgf.12508.

[PM87] S. Petry and G. Meyer. *The Perception of Illusory Contours*. Springer, 1987. ISBN: 9780387965185.

[Roc+99] C. Rocchini et al. "Multiple Texture Stitching and Blending on 3D Objects". In: *Rendering Techniques' 99*. Jan. 1999, pp. 119–130.

[SS02] Heung-Yeung Shum and Richard Szeliski. "Correction to Construction of Panoramic Image Mosaics with Global and Local Alignment". In: *International Journal of Computer Vision* 48.2 (2002), pp. 151–152.

[Sta+19] M. Stampouloglou et al. "3D DOCUMENTATION AND VIRTUAL ARCHAEOLOGICAL RESTORATION OF MACEDONIAN TOMBS". In: *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLII-2/W11 (May 2019), pp. 1073–1080. DOI: 10.5194/isprs-archives-XLII-2-W11-1073-2019.

[Ume91] S. Umeyama. "Least-squares estimation of transformation parameters between two point patterns". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13.4 (1991), pp. 376–380. DOI: 10.1109/34.88573.

[VS07] Luiz Velho and Jonas Sossai. "Projective texture atlas construction for 3D photography". In: *The Visual Computer* 23 (2007), pp. 621–629.

[WJP08] F. Walkowski, R. Johnston, and N. Price. “Texture Mapping for the FastSCAN hand-held laser scanner”. In: *International Conference Image and Vision Computing*. Nov. 2008. DOI: 10.1109/IVCNZ.2008.4762078.

[WMG14] Michael Waechter, Nils Moehrle, and Michael Goesele. “Let There Be Color! Large-Scale Texturing of 3D Reconstructions”. In: *European Conference on Computer Vision (ECCV)*. Ed. by David Fleet et al. Cham: Springer International Publishing, 2014, pp. 836–850. ISBN: 978-3-319-10602-1.