

CRANFIELD UNIVERSITY

BAPTISTE CLUS

**Enhancing 3D Scan Quality and Automating Data
Processing for Equine Back Analysis in Saddle
Fitting Applications**

School of Aerospace, Transport and Manufacturing

MSc in Computational and Software Techniques in Engineering

Master of Science

Academic Year: 2024–2025

Supervisor: Dr. Peter Sherar

August 2025

ABSTRACT

Poor saddle fit can cause pain, muscle atrophy, and reduced performance in horses, while back shape changes over time make regular reassessment essential. To address this, Voltaire Group developed a mobile app using iPhone 3D scanning, but the current system struggles with scan accuracy, robustness, and extraction of meaningful geometric features. This research improves the reconstruction and analysis pipeline to enable more precise and reliable saddle fitting. The original workflow relied on surfel-based reconstruction followed by Poisson meshing, which often produced imprecision and inconsistencies due to low-resolution depth data. In contrast, the proposed workflow filters the raw depth data and applies a volumetric Truncated Signed Distance Function (TSDF) representation combined with marching cubes meshing. Comparative analyses were conducted between methods, with evaluation metrics focused on reconstruction precision relative to reference scans. In addition, a novel technique for detecting the dominant symmetry plane—tailored to horse back geometry—was introduced, along with new geometric measurements relevant to saddle fit. Results show that TSDF-based reconstruction increases precision by up to 30% while supporting real-time processing. The new pipeline also avoids the error accumulation observed in the previous point cloud-to-mesh workflow, ensuring consistent and reliable outcomes across configurations. The symmetry plane detection method proved highly effective, reducing orientation errors by an average of 2 cm. Overall, this work contributes a robust and precise 3D reconstruction pipeline for equine back scanning, supporting improved saddle fitting practices and demonstrating the potential of mobile 3D scanning in equestrian applications.

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my tutors, Aurélie Julienne and Laurent, for their guidance throughout my internship. I am especially thankful to Aurélie for her clarity in helping me to understand the context of the equine world, as well as her insights into the morphological analysis of the horse. I am equally grateful to Laurent for his technical perspectives, and his constructive feedbacks.

My thanks also go to the Voltaire Group for welcoming me warmly during this internship. I truly appreciated the efforts of everyone who took the time to introduce me to their work, explain the intricacies of saddle-making, and make me feel part of the team from my very first day.

Finally, I would like to extend my deep gratitude to Dr Peter Sherar for his continued support and guidance throughout the thesis process, and for taking the time to meet with me and answer my questions.

CONTENTS

1	Introduction	10
1.1	Background and Motivation	10
1.2	Aims and Objectives	10
1.3	Thesis Structure	11
2	Literature Review	12
2.1	Guided Depth Map Super-Resolution	12
2.2	3D Scan Generation from Multiple Views	15
2.3	Geometric Alignment and Surface Refinement in 3D Scans	16
2.3.1	Geometric Alignment	16
2.3.2	Surface Refinement	18
2.4	Automated Landmark Extraction and Measurement on 3D Surfaces	20
2.5	Predictive Fitting from 3D Morphological Data	21
3	Theoretical Background	23
3.1	Guided Depth Map Super-Resolution	23
3.2	3D Scan Generation from Multiple Views	26
3.2.1	Pinhole Camera Model	26
3.2.2	Iterative Closest Point	27
3.2.3	3D Models Representations and Generations	28
3.3	Geometric Alignment and Surface Refinement in 3D Scans	33
3.3.1	Surface Refinement	33
4	Methodology	36
4.1	3D Scan Enhancement	36
4.1.1	Current and Proposed Workflow	36
4.1.2	Evaluation Targets	38
4.1.3	Evaluation Metrics	40
4.2	Geometric Alignment and Surface Refinement in 3D Scans	43
4.2.1	Proposed Approach	43
4.2.2	Orientation Evaluation	47
4.3	Automated Landmark Extraction and Measurement on 3D Surfaces	48
5	Results & Discussion	51
5.1	3D Scan Enhancement	51
5.1.1	Brute and Real-Time Implementation Comparative	51
5.1.2	Frustum Model Dataset	53
5.1.3	Voxels Size Influence & TSDF Computational Time	58
5.1.4	Courbette Dataset	61
5.1.5	Horses Dataset	69
5.2	Geometric Alignment and Surface Refinement in 3D Scans	73
5.3	Automated Landmark Extraction and Measurement on 3D Surfaces	76
6	Further Work	78

LIST OF FIGURES

1	Visualization of Synthetic Image Artifacts: (a) Original Image, (b) Gradient Reversal, (c) Halo Artifact, (d) Ringing Artifact	12
2	Three-dimensional representation of the bilateral filtering process, illustrating the input image, the two filtering kernels, and the resulting output. Adapted from Durand and Dorsey [17]	13
3	Process to detect symmetry plane in partial 3D meshes using heat diffusion landmarks detection, adapted from Sipiran et al. [45]	17
4	Reconstruction of a horse mesh using spectral mesh processing. The parameter k denotes the number of spectral coefficients employed. Adapted from Levy and Zhang [61].	20
5	PointNet++ architecture for point cloud segmentation and classification. Adapted from PointNet++ [63].	21
6	Pinhole camera model expressed in the camera coordinate system. Adapted from Hartley and Zisserman [68].	26
7	Poisson reconstruction process in 2D. Adapted from Kazhdan et al. [35].	29
8	Concept and representation of the signed distance function (SDF) for the Stanford Bunny. Adapted from Park et al. [72].	30
9	Possible configurations in the original Marching Cubes algorithm.	32
10	Multi-level graph partitioning using the MeTiS algorithm [74].	35
11	Reconstruction workflows for (a) the current process and (b) the proposed process, with modifications and additions highlighted in red.	36
12	Ground truth of the evaluation targets: (a) Frustum, (b) <i>Courbette</i> , and (c) Horse. .	38
13	Measurements used for evaluating the quality of <i>Courbette</i> meshes.	42
14	Application of the heat diffusion filter from Sipiran et al. [45] to extract features on (a) Armadillo and (b) <i>Courbette</i> meshes.	44
15	Top-down views illustrating the stages of dominant symmetry plane detection for a 3D horse back model: (a) input model prior to orientation, (b) model segmented by its principal symmetry plane estimated via PCA, (c) mirrored version of the model, also segmented by its PCA-derived symmetry plane, (d) original model segmented by both symmetry planes after alignment of the mirrored model to the original, with the region between the planes highlighted in blue, and (e) final model segmented by the computed mean symmetry plane, representing the dominant symmetry plane of the horse back scan.	46
16	Architecture of the CNN used to extract the relative position of the reference point.	49
17	Visualization of extracted landmark points and measurements on a horse scan. . .	50
18	Example of RGB-D data from the <i>Courbette</i> dataset: (a) depth map (320×160) and (b) RGB guidance image (1280×720).	51
19	Comparison of computing times between the brute-force implementation and the real-time implementation using quantization for Joint Bilateral Upsampling (JBU) and Joint Unified Multi-Lateral (JUML) filters.	51
20	Visual comparison of the brute-force and real-time implementations of Joint Bilateral Upsampling and Joint Unified Multi-Lateral filters on depth maps [units: m].	52

21	Colour-coded representation of precision on frustum point clouds with threshold $\tau = 1$ mm: (a) current surfel-based scan, (b) unfiltered TSDF, (c) TSDF after bilateral filter, (d) TSDF after joint bilateral upsampling, (e) TSDF after unified multi-lateral filter and (f) TSDF after guided filter.	54
22	Precision plot for frustum point clouds	55
23	Zoom on TSDF scans: (a) unfiltered, (b) after bilateral filter, (c) after joint bilateral upsampling, (d) after unified multi-lateral filter and (e) after guided filter.	56
24	Impact of voxel size on computational time for TSDF RGB-D frame integration, point cloud generation, and mesh extraction.	58
25	Colour-coded representation of precision on <i>Courbette</i> meshes with threshold $\tau = 1$ mm at varying TSDF voxel size: (a) 1 mm, (b) 1.5 mm, (c) 2 mm, (d) 2.5 mm, (e) 3 mm, (f) 3.5 mm, (g) 4 mm, (h) 4.5 mm and (i) 5 mm	60
26	Colour-coded representation of precision on <i>Courbette</i> point clouds with threshold $\tau = 1$ mm: (a) current surfel-based scan, (b) unfiltered TSDF, (c) TSDF after bilateral filter, (d) TSDF after joint bilateral upsampling, (e) TSDF after unified multi-lateral filter and (f) TSDF after guided filter.	62
27	Colour-coded representation of precision on <i>Courbette</i> mesh vertices with threshold $\tau = 1$ mm: (a) current surfel-based scan, (b) unfiltered TSDF, (c) TSDF after bilateral filter, (d) TSDF after joint bilateral upsampling, (e) TSDF after unified multi-lateral filter and (f) TSDF after guided filter.	63
28	Precision plot for <i>Courbette</i> (a) point clouds, (b) mesh vertices	63
29	Scanning trajectories used for <i>Courbette</i> dataset	67
30	Colour-coded representation of precision using a new trajectory on <i>Courbette</i> mesh vertices with threshold $\tau = 1$ mm: (a) current surfel-based scan, (b) unfiltered TSDF, (c) TSDF after bilateral filter, (d) TSDF after joint bilateral upsampling, (e) TSDF after unified multi-lateral filter and (f) TSDF after guided filter.	68
31	Precision plot with new trajectory for <i>Courbette</i> (a) point cloud, (b) mesh vertices	68
32	Colour-coded representation of precision on horse mesh vertices with threshold $\tau = 2$ mm: (a) current surfel-based scan, (b) TSDF after bilateral filter, (c) TSDF after guided filter.	70
33	Precision plot for horse meshes	71
34	(a) Uniform colour with prominent withers, (b) uniform colour with prominent withers, also used to assess lighting conditions, (c) spotted horse to evaluate the influence on filters, (d) smooth withers.	72
35	(a) Left view, (b) Right view and (c) Top-down view of a <i>Courbette</i> mesh coloured based on symmetry plane	74
36	(a) Left view, (b) Right view and (c) Top-down view of a horse mesh coloured based on symmetry plane	75
37	Visualisation of key anatomical measurements extracted from the horse's back mesh: (a) withers height, (b) withers width, (c) withers width measured 8 cm below the highest point, (d) shoulder width, (e) withers angles, (f) shoulder angles, (g) angles at the straddle level, and (h) angles at the teats level.	76
38	Visualisation of the predicted reference point on (a) the backline and (b) the mesh	77

LIST OF TABLES

1	List of Notations from GDSR Used in This Thesis	23
2	Comparison of structural similarity and precision between the brute-force and real-time implementations for both filters [units: mm].	53
3	Precision metrics across different filtering methods for Frustum 3D scans [unit: mm].	55
4	Precision of planar surfaces — upper surface [unit: mm].	56
5	Precision of planar surfaces — combined sides (mean of sides 1 to 4) [unit: mm]. .	57
6	Mean errors for edge groups by filtering method [unit: mm].	57
7	Timing breakdown by voxel size	59
8	Precision metrics across different voxel sizes for <i>Courbette</i> 3D scans [unit: mm]. .	61
9	Precision metrics across different filtering methods for <i>Courbette</i> 3D point clouds [unit: mm].	64
10	Precision metrics across different filtering methods for <i>Courbette</i> 3D mesh vertices [unit: mm].	65
11	Roughness of each <i>Courbette</i> scan [unit: mm].	65
12	Errors in key measurements for each filtering method [unit: cm].	66
13	Precision metrics across different filtering methods for horse 3D mesh vertices [unit: mm].	71
14	Difference in x and z coordinates between the placed landmark points before and after orientation for each scan [unit: cm]	73

1 INTRODUCTION

1.1 Background and Motivation

Saddle fitting has become an important area of research in equine science due to its significant impact on horse performance, comfort, and morphological development. Given the diversity of horse breeds and individual anatomical variations, achieving a proper saddle fit is essential to ensure both the well-being of the horse and optimal performance.

Bondi et al. [1] highlighted the crucial role of correct saddle fitting in enhancing horse performance and comfort. An ill-fitting saddle can restrict movement and cause discomfort, particularly when combined with uneven pressure distribution resulting from the rider's weight. This imbalance may lead to rider instability and increased physical strain on the horse. De Cocq et al. [2] emphasized the relationship between poor saddle fit and the development of back problems in horses. Using pressure measurement devices, they assessed saddle fit under the influence of rider weight and identified improper saddle fit as a contributing factor to back pain. Greve and Dyson [3] conducted a longitudinal study examining the influence of various factors—including seasonal changes and workload—on the horse's back dimensions. Their findings suggest that saddle fit—deemed as an important factor too—should be reassessed multiple times throughout the year to accommodate changes in the horse's morphology. Similarly, von Peinen et al. [4] demonstrated that an unsuitable saddle can hinder back development, induce muscle atrophy, and cause discomfort or pain. In another study, Greve et al. [5] explored the combined influence of saddle fitting, rider skill, and work quality. They concluded that saddle fit should be evaluated both before and after exercise to ensure an accurate fit, noting that saddle fitting remains largely empirical rather than grounded in objective scientific methods.

In an effort to bring a more data-driven approach to this process, Voltaire Group developed a mobile application that utilizes the StandardCyborg library in conjunction with the iPhone TrueDepth camera. The application captures RGB-D images to create 3D scans of the horse's back. This tool is primarily used by company sales representatives to assist in saddle fitting and to monitor morphological changes over time.

Despite this innovation, the current scanning process has limitations in terms of scan accuracy, post-processing robustness, and the extraction of useful geometric features from the 3D data. Therefore, the primary objective of this research is to improve the existing scanning pipeline, enhancing the precision and usability of the 3D reconstructions to support more accurate and reliable saddle fitting assessments.

1.2 Aims and Objectives

The primary aim of this research is to develop an enhanced 3D scanning with improved precision and extended functionality. To achieve this, the project is structured around the following key objectives:

1. **3D Scan Enhancement:** Evaluate the existing application's workflow and underlying technologies, propose an improved scanning workflow to enhance precision, and conduct a comparative analysis between the current and proposed systems.
2. **Post-Processing:** Develop techniques for refining the acquired 3D scans by removing artifacts and unwanted objects, and ensuring proper alignment and horizontality to facilitate accurate comparative analysis.

3. **Landmark Extraction and Measurements:** Design an automated pipeline for the identification and extraction of anatomical landmarks from the 3D scans to enable accurate measurement for saddle fitting and morphological analysis.
4. **Fitting Prediction:** Implement a method to predict the most appropriate saddle parts based on the extracted 3D scan data and corresponding morphological measurements.

1.3 Thesis Structure

The remainder of this thesis is organized as follows:

- **Chapter 2: Literature Review** – This chapter presents an overview of the existing research and foundational theories related to each of the main tasks of the project, highlighting key studies and identifying applications and gaps that this thesis aims to address.
- **Chapter 3: Theoretical Background** – This chapter presents the mathematical foundations and theoretical principles underlying the technologies utilized throughout the remainder of this work.
- **Chapter 4: Methodology** – This chapter describes the methods and techniques used for data collection, experiments and analysis, explaining the approach chosen.
- **Chapter 5: Results and Discussion** – This chapter presents the outcomes of the experiments, interpreting the results in light of the research objectives and comparing them with the previous and existing work.
- **Chapter 6: Future Work** – This chapter discusses the limitations of the work, and suggests directions for future research.
- **Chapter 7: Conclusion** – This chapter summarizes the main findings of the thesis and discusses its contributions to the problematic.

Each chapter is designed to build on the previous one, gradually leading the reader from a theoretical understanding to practical implementation and critical evaluation.

2 LITERATURE REVIEW

2.1 Guided Depth Map Super-Resolution

Depth information plays a critical role in numerous applications ranging from 3D reconstruction to autonomous driving systems. Among available range sensing technologies, Time-of-Flight (ToF) cameras [6] represent an active sensing approach that offers distinct advantages over passive methods like stereo reconstruction, particularly in real-time depth map generation. As demonstrated by Foix et al. [7], these cameras have found successful applications in mobile robotics, robotic manipulation tasks, and human recognition systems - with Apple's TrueDepth camera being a prominent commercial example.

Despite their advantages, ToF cameras face inherent limitations in spatial resolution compared to conventional RGB sensors, often producing noisy and imprecise depth measurements. This performance gap has motivated extensive research into Guided Depth Map Super-Resolution (GDSR) techniques, which enhance low-resolution depth maps using corresponding high-resolution RGB images as guidance. Early surveys by Eichhardt et al. [8] and Yang et al. [9] documented the initial progress in this field, but their coverage ends before recent methodological advances. The more comprehensive survey by Zhong et al. [10] provides an up-to-date taxonomy of GDSR approaches, categorizing them into three distinct categories: filtering-based methods as computationally efficient but prone to artifacts (Fig. 1); prior-based optimization methods that rely on handcrafted objective functions at higher computational cost and learning-based methods which achieve state-of-the-art performance but require extensive training data. This classification framework reflects the field's evolution from traditional image processing techniques to modern data-driven approaches.

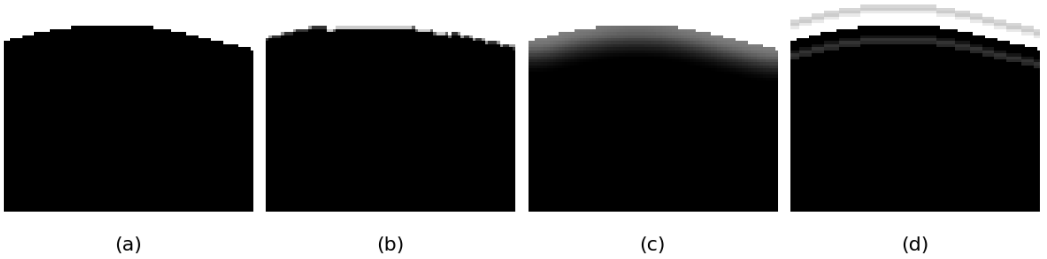


Figure 1. Visualization of Synthetic Image Artifacts: (a) Original Image, (b) Gradient Reversal, (c) Halo Artifact, (d) Ringing Artifact

Various filters have been developed with applications on Guided Depth Map Super-Resolution (GDSR), each with distinct characteristics. The Bilateral Filter (BF), introduced by Tomasi and Manduchi [11], represents a significant advancement over traditional Gaussian filters by preserving edges while smoothing images. This is achieved through dual kernel operations: a spatial kernel that weights contributions based on pixel proximity, and a range kernel that accounts for intensity variations (Fig. 2). However, BF's reliance on the same image for both kernels makes it susceptible to noise and inaccuracies inherent in depth data. Eisemann and Durand [12] addressed this limitation with the Joint Bilateral Filter (JBF), originally developed for flash photography enhancement but equally applicable to GDSR. JBF utilizes a high-resolution RGB image as guidance for the range kernel, leveraging its superior edge information. A fundamental constraint of JBF

is its requirement for input images to share identical resolutions. This motivated Kopf et al. [13] to develop Joint Bilateral Upsampling (JBU), which adapts the spatial kernel coordinates to match the HR RGB guidance image’s resolution, potentially using fractional pixel coordinates. Building upon JBU, Li et al. [14] introduced the Joint Trilateral Upsampling method, incorporating dual range kernels for both RGB and depth information to mitigate JBU’s texture copying artifacts. Further enhancements emerged through gradient-based modifications, including Garcia et al.’s Pixel Weighted Average Strategy (PWAS) [15] and Lo et al.’s Joint Trilateral Filter [16]. PWAS notably incorporates a credibility map derived from depth gradients to weight depth values.

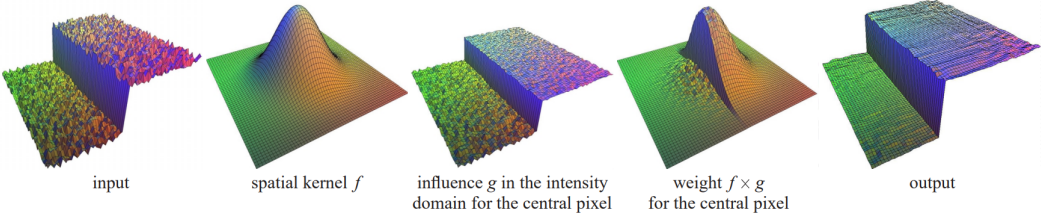


Figure 2. Three-dimensional representation of the bilateral filtering process, illustrating the input image, the two filtering kernels, and the resulting output. Adapted from Durand and Dorsey [17]

Despite these advancements, bilateral filters approaches face two persistent challenges: gradient reversal artifacts and computational intensity. He et al. [18] proposed an alternative solution through the Guided Filter (GF), which models the enhanced depth map as a linear transformation of the guidance image. While GF offers computational advantages, it can introduce halo artifacts near edges. This limitation prompted Li et al. [19] to develop the Weighted Guided Image Filter (WGIF), incorporating edge-aware weighting to preserve fine details.

While Guided Filter and its variants are often preferred for real-time applications due to their $O(N)$ complexity (compared to the $O(Nr^2)$ complexity of brute-force bilateral filtering), significant research has focused on accelerating bilateral filters. Early work by Weiss [20] reduced the filtering time to $O(\log r)$ using a hierarchical histogram approach, enabling efficient median and bilateral filtering. Paris and Durand [21] introduced a signal-processing-based approximation that leverages subsampling in a higher-dimensional space, achieving faster performance as the filter kernel size increases. A major breakthrough came from Porikli [22], who demonstrated that three specific types of bilateral filters (box spatial with arbitrary range, arbitrary spatial with polynomial range, and arbitrary spatial with Gaussian range) could be computed in constant time $O(1)$. However, this method suffers from two limitations: a non-unified framework and reduced precision due to quantization artifacts. Durand and Dorsey [17] proposed an acceleration strategy based on quantization and piecewise linear interpolation. Their approach quantizes both the input image intensities and the range function, pre-computing linear filters for discrete intensity levels and interpolating between them. Building on this, Yang et al. [23] improved accuracy by quantizing only the range function while preserving the original image intensities. Their method introduces Principle Bilateral Filtered Image Components (PBFICs)—precomputed filtered images for quantized intensity levels—and obtains the final output through linear interpolation between the two nearest PBFICs. This avoids input quantization artifacts and achieves higher precision with lower memory overhead. This quantization framework has been successfully extended to bilateral filter variants such as joint bilateral filtering (JBF) and joint bilateral upsampling (JBU). Garcia et

al. [24, 25] leveraged piecewise linear interpolation to generalize these concepts, developing their Unified-Multilateral Filter (UML) - a real-time application of their PWAS filter. Their work demonstrates how the quantization approach can be adapted to maintain efficiency while handling more complex filtering scenarios.

Optimization-based methods represent a distinct approach to Guided Depth Map Super-Resolution, employing mathematically rigorous frameworks to enhance depth maps through carefully designed objective functions. While some works served as a basis for innovation, the next paragraphs rather focus on presenting optimizations leveraging different techniques.

Among these, Markov Random Field (MRF) formulations have demonstrated particular effectiveness in addressing the GDSR problem. The foundational work by Diebel and Thrun [26] pioneered the application of MRFs to depth super-resolution, establishing a comprehensive probabilistic framework. Their approach defines the MRF through five interconnected node types: range measurement nodes capturing raw sensor data, image pixel nodes from the high-resolution guidance image, reconstructed range value nodes aligned with image pixels, and specialized nodes for depth discontinuities and image gradients. This sophisticated network enables the integration of multiple information sources through two key potential functions: a measurement potential that minimizes discrepancies between known range values and their reconstructed counterparts, and a depth smoothness potential that regularizes adjacent nodes in the reconstructed depth field. Crucially, the model incorporates weighting factors that establish probabilistic connections between the depth reconstruction and the guidance image layer, allowing for edge-aware depth enhancement while mitigating texture copying artifacts. Building upon this foundation, Zuo et al. [27] introduced an advanced MRF formulation specifically designed to address texture copying and edge blurring artifacts. Their method quantifies the inconsistency between depth maps and guidance images through a bi-directional edge map quality assessment mechanism. This innovative approach evaluates discrepancies between binary edge maps derived from interpolated depth data and those extracted from the RGB guidance image, enabling explicit correction of common artifacts in MRF-based depth super-resolution. The resulting framework maintains the probabilistic advantages of MRF modelling while significantly improving edge preservation and texture discrimination capabilities.

Jiang et al. [28] proposed a depth super-resolution method that combines two ideas. First, in the transform domain, it uses a Sparse Coefficient-based Auto-Regressive model to capture similarities between image patches, guided by both depth and colour information. Second, in the spatial domain, it applies a Weighted Multi-Directional Total Variation prior to preserve edges in many directions while reducing noise and artifacts. The method is solved using an iterative optimization process that gradually improves the high-resolution depth map by alternating between enforcing data consistency and applying the two regularization steps. A step-by-step upsampling strategy is used for large scale factors. By combining non-local similarity modelling with adaptive edge preservation, it achieves strong results on both synthetic and real RGB-D data. However, it is computationally heavier than simple filtering methods and can struggle when patch matching is inaccurate or when filling large missing regions.

Ferstl et al. [29] proposed a guided depth upsampling using anisotropic Total Generalized Variation (TGV). The optimization problem is formulated as an energy function to be solved in order to obtain a high-resolution depth map. The function is composed of a data term, which keeps the solution close to the measured depth values, and a regularization term using TGV. Total Variation (TV) is a method for removing noise, thereby smoothing the depth values while preserving the edge structure, by penalising the gradients (first derivative) of the image. TGV builds

on this by penalising higher-order derivatives, thus ensuring piecewise-affine smoothness rather than piecewise-constant regions (staircase effects) as achieved by TV. In addition, an anisotropic diffusion tensor is computed from the high-resolution RGB image to serve as guidance. The problem is then reformulated as a convex optimization problem and solved using a gradient descent algorithm. The main issues discussed concern the influence of the depth measurements, which can, for instance, result in displaced surfaces in the final HR depth map.

Attention is now focused on learning-based methods and their application to the GDSR problem. Unlike the previously discussed unsupervised filtering and optimization-based approaches, learning-based methods offer a data-driven, supervised framework for addressing guided super-resolution by training neural networks.

The Deformable Kernel Network (DKN) proposed by Kim et al. [30] draws on concepts from both filtering-based methods and learning-based approaches. It employs a CNN to adaptively learn a set of sparsely selected neighbours and the corresponding interpolation weights, followed by an explicit image filter to generate the final prediction. However such method either suffer from limited model interpretability when directly regressing the target or depend on a simple image filter that cannot fully leverage the guide image. To correct these issues, Tang et al. [31] proposed the Joint Implicit Function (JIIF). Starting with a high-resolution (HR) guide image and a low-resolution (LR) input image, two separate encoders generate corresponding sets of latent codes. The JIIF decoder is then queried at a specific coordinate in the HR space to estimate the pixel value at that location. The resulting prediction is computed as a weighted average of the four closest coordinates in the LR space, similar to traditional image interpolation. However, both the interpolation weights and values are learned through a deep implicit function rather than being fixed.

While most GDSR techniques evaluate filtered depth maps against ground truth, their use in 3D reconstruction is often mentioned only briefly, with limited assessment of their impact on the quality of reconstructed 3D scans. This work addresses that gap by directly evaluating how GDSR affects 3D reconstruction outcomes, providing a more application-oriented perspective on GDSR performance.

2.2 3D Scan Generation from Multiple Views

Building on the previous discussion of guided depth map super-resolution and its importance for enhancing RGB-D data quality in downstream applications, we now focus on how such data can be utilized to generate 3D scans from multiple RGB-D images and views.

A basic approach for 3D reconstruction involves projecting depth maps at each frame using the camera’s intrinsic parameters and registering the resulting point clouds. However, this method heavily depends on accurate alignment - including proper initial pose estimation - and often produces noisy, imprecise reconstructions. Registration typically employs Iterative Closest Point (ICP) [32], which iteratively minimizes point-to-point or point-to-plane distances between successive scans. Variants like Colored ICP [33] incorporate color information to improve correspondence matching, while feature-based methods such as Fast Point Feature Histograms (FPFH) [34] use local geometric descriptors for robust alignment. The registered points can be converted to meshes using Poisson Reconstruction [35], which solves an implicit surface equation to create watertight models, or Marching Cubes [36], which extracts polygonal meshes from volumetric data by interpolating surface crossings through a 3D grid.

For practical applications, particularly with low-cost RGB-D sensors and real-time systems,

volumetric approaches have become predominant [37]. The Truncated Signed Distance Function (TSDF) [38] represents one of the most successful techniques, maintaining a volumetric grid where each voxel stores the signed distance to the nearest surface within a truncated range. This formulation naturally handles noise and merges observations from multiple views. More efficient implementations like Hash-based TSDF [39] use spatial hashing to dynamically allocate memory only near observed surfaces, enabling larger-scale reconstructions. These methods require robust pose estimation, typically obtained through RGB-D odometry or feature matching (using descriptors like ORB or SIFT), before final surface extraction via adapted Marching Cubes.

Alternative surfel-based approaches include Dense Surfel Mapping [40], SurfelMeshing [41], and ElasticFusion [42], which provide different implementations of surface element representations for 3D reconstruction. Surfel-based methods represent surfaces using oriented discs with attributes such as position, normal, colour, and radius, which allows for efficient updates and fusion. These approaches can be computationally less demanding compared to volumetric TSDF methods, particularly when dealing with large-scale or dynamic scenes. ElasticFusion, for instance, demonstrates real-time dense SLAM without relying on a pose graph by leveraging frame-to-model tracking that jointly minimizes photometric and geometric error against a surfel-based map. It introduces non-rigid loop closures through a deformation graph to maintain global consistency. This enables continuous dense reconstruction with loop closure by deforming the active model to align with previously mapped regions, producing globally consistent surfel maps even in challenging environments with loopy camera trajectories.

2.3 Geometric Alignment and Surface Refinement in 3D Scans

In these sections, potential techniques for improving the re-orientation of the obtained meshes are discussed, so that the length, width, and height are aligned as closely as possible with the canonical axes for future analysis. Additionally, a variety of filters are described, along with their advantages and disadvantages, to smooth the 3D model while preserving its shape, features (even if difficult to define), and precision.

2.3.1 Geometric Alignment

A general approach based on Principal Component Analysis (PCA) can be used to determine the principal axes of a point cloud or mesh using its vertices. For orientation purposes, this method allows aligning the principal axes of the scan with canonical axes. However, the effectiveness of this approach relies heavily on the symmetry of the object, which may not be suitable or repeatable for certain cases, such as the horse scan considered here.

The problem of finding the correct orientation and horizontality can therefore be reformulated as the problem of identifying the symmetry plane of the scan. PCA can assist in this task by using one of its components as the normal vector of the plane and the centroid of the point cloud as a point on the plane. Nonetheless, this method remains dependent on the PCA results and may not always provide robust outcomes. To address this limitation, He et al. [43] proposed a dynamic approach that focuses on identifying the dominant plane rather than strictly the symmetry plane. In this method, a candidate plane is defined using a PCA component as its normal, and the mirrored model is generated accordingly. The distances between corresponding points on either side of the plane are then computed. These distances are used to iteratively update the weights, with smaller distances contributing more to the assessment of symmetry. However, this approach may be limited in certain cases. For instance, if the meshes are partial, the correspondences may not be

sufficiently meaningful, and if the surface is smooth, the correspondences may lack significance. Consequently, alternative techniques can be considered, particularly those that utilize detected landmarks on the models. These correspondences can then be employed to evaluate and select an appropriate symmetry plane.

In their work, Hosoki et al. [44] project the point cloud onto an image plane and centre it, after which the region of interest containing the face is detected using HOG features. Since their study focuses on faces, the facial features are computed through repeated regression to fit the face and subsequently mapped back into 3D space. A mirrored version of the point cloud is then created, and both the original and mirrored point clouds are first aligned using a global registration based on FPFH features, followed by a local registration using ICP. Finally, the features in both aligned point clouds are utilized to determine a normal vector and a centroid, which together define the symmetry plane. This approach is particularly effective for faces due to their specific geometry and distinct features. This approach appears as interesting for the mirroring and plane computation from symmetric planes. However the detection of features is specific to faces and difficultly transposable to horse scans.

Sipiran et al. [45] proposed a two-step approach to determine the symmetry plane for partial 3D meshes. First, local features are extracted using heat diffusion within local domains, after which potential symmetry planes are generated. A voting-based strategy is then employed to select the most suitable plane. The use of heat diffusion is related to the Heat Kernel Signature (HKS) method, which identifies landmarks on meshes or point clouds [46]. Areas exhibiting extreme HKS values typically correspond to regions of high curvature. This approach is particularly useful for partial meshes, where the symmetry plane does not necessarily pass through the centroid. In their work, Sipiran et al. used compact, watertight meshes, which may not be applicable to the horse scans considered here.

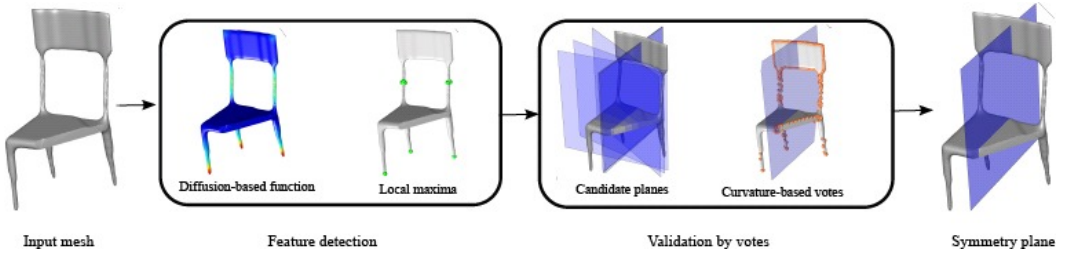


Figure 3. Process to detect symmetry plane in partial 3D meshes using heat diffusion landmarks detection, adapted from Sipiran et al. [45]

Mitra et al. [47] proposed a method for identifying symmetry in 3D models. Signatures are computed at points on the 3D model based on principal curvatures. These signatures are then used to match points in a subset of the model to points in the larger point cloud. For each matched pair, a 7-dimensional vector is calculated, incorporating both the curvature information and the transformation between the two points. Finally, clustering is performed on these transformation vectors, with larger clusters indicating matches that are more relevant from a symmetry perspective. The two best matched features are used to create a symmetry plane as they define the normal of the plane and the plane crossing the midpoint of the vector between the features. However, this method mainly based on curvatures which may not be suitable for horse scans where those are

not significant enough to be defined as features, and even to be use for matching.

Thus, it is necessary to determine whether a simple method exists to identify the symmetry plane of the scan. However, existing approaches fall into different categories, each with limitations. Some methods are heavily dependent on heuristics, while others require feature computation that may not be suitable, as they rely on local variations or curvatures to define distinctive signatures. For smooth surfaces, such as a horse’s back, these features are often not indicative of symmetry. Therefore, there is a need to develop an approach that addresses this gap and provides a reliable solution specifically for the case of a horse’s back.

2.3.2 Surface Refinement

The first step is to segment the mesh by retaining only the main structure while removing artefacts or other extraneous objects. Common clustering techniques, such as DBSCAN, can be employed for this purpose. For small, isolated mesh fragments that are far from the main structure, a Statistical Outlier Removal (SOR) filter can be applied, which rejects points based on the number of neighbours within a given distance. Finally, an effective approach is to represent the mesh as a graph composed of nodes and edges and retain only the largest connected component.

Meshes are widely used for various computational purposes but often suffer from non-smooth surfaces, exhibiting staircase effects or noisy data. To address this problem, extensive research has explored a range of smoothing techniques, including isotropic and anisotropic methods, filter-based approaches, optimization-based methods, and data-driven methods, as reviewed by Wang et al. [48].

Isotropic methods treat the mesh uniformly without considering its geometric features. These filters iteratively update vertex coordinates based on their neighbours. The simplest example is the average filter, which updates a vertex position by computing the mean of its neighbours within a specified window. While straightforward, this approach does not adapt to local geometry or mesh resolution, as all neighbours are weighted equally. To improve upon this, the Laplacian filter proposed by Vollmer et al. [49] applies a distance-based weighting to neighbours and includes a scalar factor to control the strength of the smoothing. However, both the average and Laplacian filters suffer from shrinkage due to their iterative nature.

The Taubin filter [50] addresses this issue by alternately applying Gaussian smoothing with positive and negative factors, effectively performing low-pass filtering to remove high-curvature variations without causing shrinkage. Additionally, Badri et al. [51] adapted the Laplacian filter by introducing a kernel function based on the gradient of neighbouring vertices to mitigate shrinkage while preserving geometric features.

Filter-based methods are largely derived from their image processing counterparts and can achieve excellent results, making them among the most commonly used techniques for mesh denoising. These filters primarily rely on vertex normals to update positions. Yagou et al. [52] proposed implementations of mean and median filters for meshes, while Jones et al. [53] and Fleishman et al. [54] developed versions of the bilateral filter adapted to meshes.

In Jones et al.’s method, the new position of a vertex is estimated based on predictions from its neighbouring triangles. Each triangle defines a plane via its normal, and the vertex is projected onto the planes of neighbouring triangles. To account for noise in the normals and mesh, two steps are applied: first weighting the predictions based on the distance to the original vertex, and

second mollifying the normals to reduce sensitivity to noise.

Zhang et al. [55] proposed a guided mesh normal filter. Although the name suggests a relation to the guided filter in GDSR, this method is also a bilateral-filter-like approach applied to meshes. Guidance normals are used to iteratively update vertex positions. For each triangle, multiple patches are evaluated, and a metric assessing the range of normals is computed. The patch with the smallest range is selected, as it corresponds to a planar region, avoiding the inclusion of edges in the averaging that could lead to over-smoothing. The guidance normal is then calculated as the area-weighted mean of the normals. This method provides an efficient, edge-aware denoising technique, though it may be less effective for inherently smooth meshes.

Finally, other techniques include the Non-Local Means approach by Yoshizawa et al. [56], which denoises meshes based on a similarity measure where vertices with similar curvature contribute more significantly to the averaging process.

While these filter-based approaches can be effective for mesh denoising, their advantage over isotropic methods such as the Taubin filter may be limited, as they are primarily useful for noisy meshes with sharp edges. Their relevance for smooth surfaces, such as a horse’s back, is therefore questionable. An alternative approach is spectral mesh processing, also known as graph spectral processing [57]. Spectral methods applied to mesh processing rely on eigen-decomposition, where eigenvalues, eigenvectors, or eigenspace projections are obtained from the graph structure of the mesh. These methods have been used for a variety of applications, including mesh compression [58], watermarking [48], and denoising.

Liu et al. [59] present the main process for point clouds, which can be extended and is particularly suitable for meshes. The goal is to treat vertex coordinates as signals and filter the high frequencies responsible for noise. The first step is to construct the graph Laplacian matrix, typically computed from the adjacency data of the mesh. An eigen-decomposition is then performed to obtain the eigenvalues and eigenvectors. In graph signal processing, the eigenvalues correspond to frequencies, with smaller values representing lower frequencies. Vertex coordinates are projected into the spectral domain using a Graph Fourier Transform (GFT), leveraging the eigenvectors, to represent the signal structure. Filtering is performed in this domain using a low-pass filter, followed by an Inverse Graph Fourier Transform (IGFT) to update the vertex coordinates. The level of smoothing is controlled by the number of retained eigenvalues: retaining all eigenvalues reconstructs the original mesh, while reducing the number of eigenvalues oversimplifies the mesh, potentially losing geometric details, as illustrated in Fig. 4. The challenge is to determine a threshold that effectively reduces noise while preserving the accuracy of the scan.

However, this process can be computationally expensive for large meshes, as the Singular Value Decomposition (SVD) used to compute the eigenvalues and eigenvectors has a time complexity of $O(n^3)$, where n is the number of vertices. To address this limitation, Arvanitis et al. [60] proposed a modified approach using partitioning, reducing the time complexity to $O(nc^2)$, where $c \ll n$ is the number of preserved spectral components. The mesh is first partitioned into sub-meshes using MeTiS multi-level graph partitioning. Eigenvectors for each sub-mesh are then computed via Orthogonal Iterations (OI), using the eigenvectors of the previous sub-mesh as the initial approximation. This method assumes geometric consistency between sub-meshes, significantly reducing computation time and making spectral mesh processing feasible for near real-time applications.

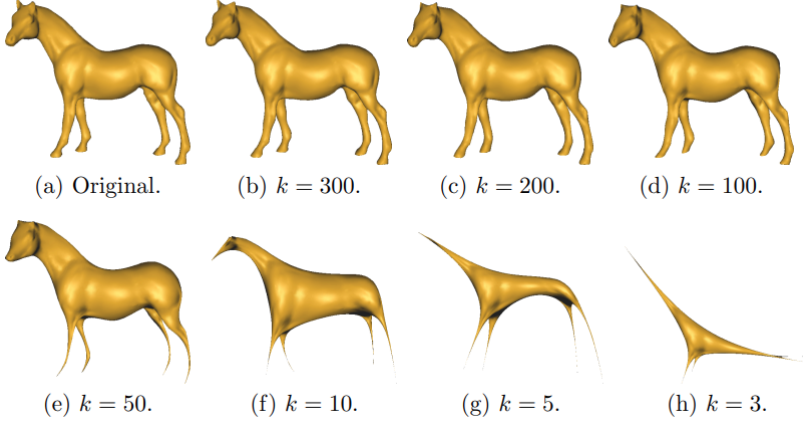


Figure 4. Reconstruction of a horse mesh using spectral mesh processing. The parameter k denotes the number of spectral coefficients employed. Adapted from Levy and Zhang [61].

2.4 Automated Landmark Extraction and Measurement on 3D Surfaces

The challenge of automatic landmark detection on 3D meshes has motivated the use of neural networks capable of handling irregular geometric data. Various approaches exist to extract precise landmarks from a mesh by operating at different dimensional levels. Since the most critical points on horse meshes are located along the backline, one natural approach is to leverage the distribution of backline points. By analysing the local and global variations in height, landmarks could be detected using a Multi-Layer Perceptron (MLP) or Convolutional Neural Network (CNN), with the resulting positions mapped to 3D coordinates.

The use of a CNN can be extended to 2D images, detecting landmarks on image planes and projecting them into 3D using RGB-D scanning data. Multiple views can be combined to triangulate the 3D positions for increased accuracy. RGB images captured during scanning can provide top-down views of the horse, facilitating annotation for training, although results remain sensitive to scanning conditions such as lighting. Synthetic images from side views could also be used to highlight the profile and backline, but annotation would be more challenging.

Masood and Qi [62] demonstrated the use of RGB-D data and CNNs to first detect 2D features and then localize them in 3D, in the context of hand acupuncture points. Their architecture combines two VGGNet networks to independently learn features from RGB and depth data, which are then fused to project points into 3D. A limitation of their approach is the requirement for well-controlled scanning conditions, which is particularly challenging when scanning horses compared to human hands.

Finally, landmarks can be detected directly on the mesh, avoiding potential information loss from lower-dimensional representations. However, the unordered nature of vertices and edges makes this task challenging. Several neural network architectures originally designed for segmentation or classification can be adapted for landmark detection:

- **PointNet++** [63] uses a hierarchical architecture to learn geometric features directly from point clouds through a sampling and grouping strategy (Fig. 5). Segmentation is performed

by assigning per-point scores and interpolating to recover the original data, while classification uses an MLP. Its ability to capture both local and global context makes it a strong candidate for landmark extraction.

- **Graph Convolutional Networks (GCNs)** [64] generalize convolution to graph structures, propagating information across vertices based on adjacency. Graph-based neural networks (GNNs) naturally exploit mesh connectivity, making them suitable for landmark detection.
- **Graph Attention Networks (GATs)** [65] enhance GCNs by introducing attention mechanisms, allowing adaptive weighting of neighbor aggregation.
- **MeshCNN** [66] defines convolution and pooling operations directly on mesh edges, effectively capturing local shape details.
- **SpiralNet++** [67] leverages spiral sequences around vertices to impose a consistent neighbor ordering, enabling convolution-like operations on meshes while preserving local geometry.

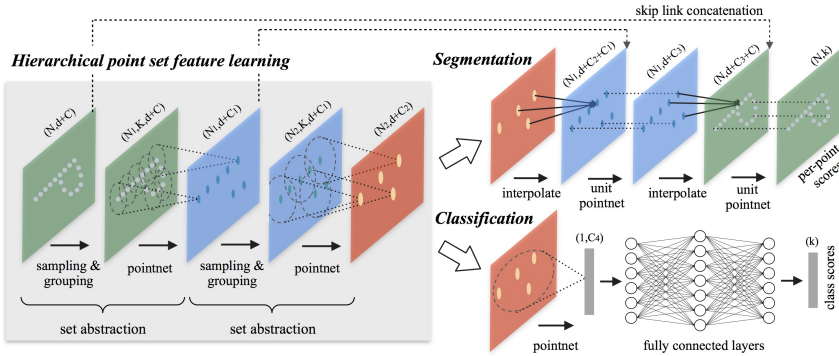


Figure 5. PointNet++ architecture for point cloud segmentation and classification. Adapted from PointNet++ [63].

Therefore, the aim of this work is to determine the most appropriate dimensional level for learning, such that landmark extraction remains precise while maintaining a simple and effective approach for the intended purpose.

2.5 Predictive Fitting from 3D Morphological Data

In the previous section, the process of extracting landmarks was discussed, which can then be used to compute measurements. This section focuses on how these measurements can be leveraged to predict saddle fit and assist sales representatives. In the context of saddle fitting, this involves mapping anatomical landmarks and shape descriptors of the horse’s back into predictions about which saddle components, such as tree width or panel shape, are most suitable. Given the large variety of possible saddle combinations, predictive models are particularly useful for efficiently identifying the best fit without extensive trial and error.

A variety of machine learning algorithms can be applied to establish this mapping, each offering different trade-offs in terms of interpretability, flexibility, and performance. Classical machine

learning models are particularly suitable for structured feature sets derived from landmarks and measurements. Decision trees and random forests provide an interpretable approach, allowing practitioners to understand which features most strongly influence saddle fit predictions. This interpretability is valuable for explaining recommendations to customers or identifying key anatomical factors. Support vector machines are effective when measurement-based features are high dimensional, as they can accurately separate different fit categories by finding optimal boundaries between them. Gradient boosting methods, such as XGBoost or LightGBM, often outperform simpler tree-based models for tabular morphological data because they handle non-linear interactions between measurements and capture complex dependencies, improving prediction accuracy across diverse horse anatomies.

Neural networks offer a different approach by directly learning complex mappings from input measurements to fit predictions. Fully connected networks, in particular, can model intricate relationships between anatomical features and saddle specifications. While they typically require larger datasets and may be less interpretable than classical models, they can capture subtle patterns that simpler models might miss, providing highly accurate predictions. By using these algorithms, it becomes possible to efficiently navigate the large combination of saddle components, supporting informed saddle selection and enhancing the sales process with data-driven recommendations.

3 THEORETICAL BACKGROUND

This section provides the theoretical foundation for the key concepts introduced and applied throughout this work.

3.1 Guided Depth Map Super-Resolution

This section details the theoretical foundation of filters used in Guided Depth Map Super-Resolution (GDSR). These filters combine spatial priors and image-based guidance to enhance a low-resolution depth map D using a high-resolution RGB image I . While the literature review addressed their conceptual motivation and practical roles, here we formalise the mathematical expressions and highlight the influence of key terms. The notation used throughout is summarised in Tab. 1.

Table 1. List of Notations from GDSR Used in This Thesis

Notation	Description
$D \in \mathbb{R}^{h \times w}$	Low-resolution depth map of dimensions $h \times w$
$D^f \in \mathbb{R}^{H \times W}$	High-resolution (refined) depth map of dimensions $H \times W$
$I \in \mathbb{R}^{H \times W}$	High-resolution RGB image of dimensions $H \times W$
$D_{p_\downarrow} \in \mathbb{R}$	Depth value at pixel $p_\downarrow$ in the low-resolution depth map D
$D_p \in \mathbb{R}$	Depth value at pixel p in the high-resolution depth map D^f
$I_p \in \mathbb{N}^3$	RGB colour vector at pixel p in the image I

A core operation in depth refinement is the *Bilateral Filter*, which computes a weighted average of neighbouring depth values, where weights depend both on spatial proximity and depth similarity. This is expressed as:

$$D_p^f = \frac{\sum_{q \in \Omega} f_s(p, q) \cdot f_r(D_p, D_q) \cdot D_q}{\sum_{q \in \Omega} f_s(p, q) \cdot f_r(D_p, D_q)} \quad (1)$$

The functions f_s and f_r are typically Gaussian kernels that control, respectively, the spatial and range influence of neighbouring pixels q on the centre pixel p :

$$f_s(p, q) = \exp\left(-\frac{\|p - q\|^2}{2\sigma_s^2}\right), \quad f_r(D_p, D_q) = \exp\left(-\frac{|D_p - D_q|^2}{2\sigma_r^2}\right) \quad (2)$$

The spatial kernel f_s enforces locality by assigning larger weights to pixels closer to p , controlled by the spatial standard deviation σ_s . The range kernel f_r , on the other hand, promotes edge preservation: if $|D_p - D_q|$ is large (e.g., across a depth discontinuity), then f_r becomes small, suppressing the influence of dissimilar pixels. The strength of this suppression is governed by σ_r : a small σ_r leads to sharper preservation of edges but less smoothing, while a large σ_r allows more aggressive averaging, potentially blurring depth boundaries.

To incorporate structural cues from the RGB image, the *Joint Bilateral Filter* replaces the range term with one defined over colour differences:

$$D_p^f = \frac{\sum_{q \in \Omega} f_s(p, q) \cdot f_r(I_p, I_q) \cdot D_q}{\sum_{q \in \Omega} f_s(p, q) \cdot f_r(I_p, I_q)} \quad (3)$$

This allows the high-resolution guidance image I to steer the filtering process, assuming that intensity edges in I are correlated with depth discontinuities in D .

For upsampling, *Joint Bilateral Upsampling* adapts the same formulation but evaluates the spatial term in the low-resolution domain and the range term in the high-resolution domain:

$$D_p^f = \frac{\sum_{q \in \Omega} f_s(p_{\downarrow}, q_{\downarrow}) \cdot f_r(I_p, I_q) \cdot D_{q_{\uparrow}}}{\sum_{q \in \Omega} f_s(p_{\downarrow}, q_{\downarrow}) \cdot f_r(I_p, I_q)} \quad (4)$$

Here, $p_{\downarrow}$ denotes the corresponding location of pixel p in the low-resolution map. As described by Garcia et al. [24], the kernel widths are often adapted: the spatial variance is tied to the upsampling ratio, while the range variance can depend on image gradient statistics.

As an extension, *Unified Multi-Lateral* filter includes a confidence measure using a *credibility map* Q to weigh the reliability of depth values:

$$D_p^f = \frac{\sum_{q \in \Omega} f_s(p_{\downarrow}, q_{\downarrow}) \cdot f_r(I_p, I_q) \cdot Q_{q_{\uparrow}} \cdot D_{q_{\uparrow}}}{\sum_{q \in \Omega} f_s(p_{\downarrow}, q_{\downarrow}) \cdot f_r(I_p, I_q)} \quad (5)$$

The credibility at each pixel is derived from the gradient magnitude of the depth map:

$$Q_{p_{\downarrow}} = \exp\left(-\frac{|\nabla D_{p_{\downarrow}}|^2}{2\sigma_q^2}\right) \quad (6)$$

This formulation penalises high-gradient regions (e.g., noise or sharp transitions), reducing their influence during refinement.

Another important filter is the *Guided Filter*, which assumes a local linear relationship between the guidance image I and the refined depth map D^f within a window $\mathcal{N}_k$:

$$D_p^f = a_k I_p + b_k, \quad \forall k \in \mathcal{N}_k \quad (7)$$

The coefficients a_k and b_k are estimated via least squares:

$$a_k = \frac{\frac{1}{|\mathcal{N}|} \sum_{p \in \mathcal{N}_k} D_p I_p - \bar{D}_k \bar{I}_k}{\sigma_k^2 + \gamma}, \quad b_k = \bar{D}_k - a_k \bar{I}_k \quad (8)$$

Where σ_k^2 is the variance of the guidance image I within $\mathcal{N}_k$, and γ is a regularisation parameter. The final estimate for D_p^f is computed as the average of overlapping windows:

$$D_p^f = \bar{a}_p I_p + \bar{b}_p, \quad \text{with} \quad \bar{a}_p = \frac{1}{|\mathcal{N}|} \sum_{k \in \mathcal{N}_p} a_k, \quad \bar{b}_p = \frac{1}{|\mathcal{N}|} \sum_{k \in \mathcal{N}_p} b_k \quad (9)$$

This filter is particularly efficient and edge-aware, as the local linear model naturally adapts to intensity changes in I .

To enable real-time filtering, the bilateral filter can be approximated via quantisation. The depth range is divided into N discrete bins such as $k \in \{L_0, \dots, L_{N-1}\}$. For each bin k , the filter constructs weighted and modulated terms:

$$W_k(q) = f_r(k, D_q), \quad J_k(q) = W_k(q) \cdot D_q \quad (10)$$

Then, a spatial filter mentioned as Principle Bilateral Filtered Image Component (PBFIC) in Sec. 2 is applied:

$$J_k^B(p) = \frac{\sum_{q \in \Omega} f_s(p, q) \cdot J_k(q)}{\sum_{q \in \Omega} f_s(p, q) \cdot W_k(q)}$$

Finally, the refined value D_p^f is obtained via linear interpolation between adjacent bins L_k and L_{k+1} such that $D_p \in [L_k, L_{k+1}]$:

$$D_p^f = (L_{k+1} - D_p)J_k^B(p) + (D_p - L_k)J_{k+1}^B(p)$$

This approximation greatly reduces computation by avoiding per-pixel evaluations of the full filter, at the cost of small quantisation error. The same technique can be adapted to joint filtering by binning RGB or gray intensity values rather than depth.

3.2 3D Scan Generation from Multiple Views

3.2.1 Pinhole Camera Model

The pinhole camera model is the most commonly used camera model and provides a good approximation of the behaviour of most cameras, as illustrated in Fig. 6.

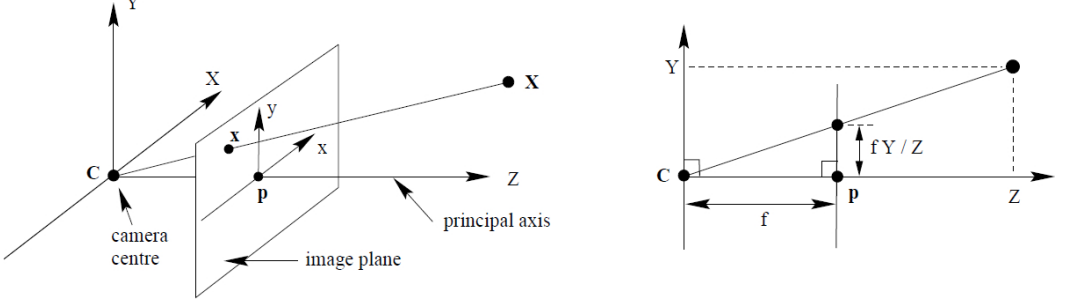


Figure 6. Pinhole camera model expressed in the camera coordinate system. Adapted from Hartley and Zisserman [68].

Using this model, the relationship between a 3D point and its corresponding 2D point on the image plane can be described in several steps. First, it is important to note that the use of homogeneous coordinates is convenient and can be derived from Euclidean coordinates. For a homogeneous 3D point such as $\tilde{\mathbf{X}} \sim [\tilde{X} \ \tilde{Y} \ \tilde{Z} \ \tilde{W}]^T$, $\tilde{\mathbf{X}}$ is related to its Euclidean equivalent $\mathbf{X} = [X \ Y \ Z]^T$ by $\mathbf{X} = [\tilde{X}/\tilde{W} \ \tilde{Y}/\tilde{W} \ \tilde{Z}/\tilde{W}]^T$. Therefore, the homogeneous coordinate of a 3D point can be expressed as $\tilde{\mathbf{X}} \sim [X \ Y \ Z \ 1]^T$.

The first step is to establish the relationship between world coordinates and camera coordinates. A 3D point $\tilde{\mathbf{X}} \sim [X \ Y \ Z \ 1]^T$ in the world coordinate system is associated with a point in the camera coordinate system $\tilde{\mathbf{X}}_c \sim [X_c \ Y_c \ Z_c \ 1]^T$ as

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} \sim \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (11)$$

where

$$\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathbb{SE}(3) \quad (12)$$

is the extrinsic parameters defined by a 4×4 matrix, composed of a 3×3 rotation matrix $\mathbf{R} \in \mathbb{SO}_3$ and a 3×1 translation vector $\mathbf{t} \in \mathbb{R}^3$. This extrinsic matrix specifies the camera pose in the world coordinate system and is widely used in various 3D reconstruction applications. To map a point from the camera coordinate system back to the world coordinates, the inverse of the extrinsic matrix is applied.

Next, we consider the relationship between a 3D point in the camera coordinate system, $\tilde{\mathbf{X}}_c \sim [X_c \ Y_c \ Z_c \ 1]^T$, and its corresponding 2D point on the image plane, $\tilde{\mathbf{x}} \sim [x \ y \ 1]^T$. As illustrated in Fig. 6, the coordinates x and y can be expressed as

$$x = f \frac{X_c}{Z_c}, \quad y = f \frac{Y_c}{Z_c}, \quad (13)$$

where f denotes the focal length of the optical system.

Finally, we consider the relationship between a 2D point on the image plane, $\tilde{\mathbf{x}} \sim [x \ y \ 1]^\top$, and its corresponding pixel coordinates, $\tilde{\mathbf{u}} \sim [u \ v \ 1]^\top$, which can be expressed as

$$\tilde{\mathbf{u}} \sim \mathbf{K} \tilde{\mathbf{x}} \quad (14)$$

Here, $\mathbf{K}$ is the intrinsic matrix that defines the optical properties of the camera and is expressed as

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \quad (15)$$

where f_x and f_y are the focal lengths along the x and y axes, typically very similar, and expressed in units of pixel dimensions. The parameters c_x and c_y define the principal point, which is the intersection of the optical axis with the image plane.

All of these steps can be combined to express the relationship between a homogeneous 3D point in world coordinates, $\tilde{\mathbf{X}}$, and its corresponding pixel position in a 2D image, $\tilde{\mathbf{u}}$, as

$$\tilde{\mathbf{u}} \sim \mathbf{P} \tilde{\mathbf{X}}, \quad (16)$$

where $\mathbf{P}$ is the projection matrix that combines the intrinsic and extrinsic parameters:

$$\mathbf{P} \sim \mathbf{K} [\mathbf{R} \ \mathbf{t}]. \quad (17)$$

3.2.2 Iterative Closest Point

Iterative Closest Point (ICP) is a widely used algorithm for performing geometric registration between two point clouds. The algorithm takes as input two point clouds along with an initial transformation and outputs a transformation matrix that minimizes the distances between the two point clouds. In most cases, the initial transformation is the 4×4 identity matrix, but ICP can also be used to refine a rough registration obtained from another method.

The algorithm operates iteratively, either until convergence or until a predefined number of iterations is reached. In each iteration, the first step is to find the set of closest correspondences between the points $\mathbf{p}$ in a target point cloud $\mathbf{P}$ and the points $\mathbf{q}$ in a source point cloud $\mathbf{Q}$, given the current transformation $\mathbf{T}$. The second step is to compute the optimal transformation $\mathbf{T}$ that minimizes an objective function $E(\mathbf{T})$.

One common variant is the Point-to-Point ICP, where the objective function is defined as

$$E(\mathbf{T}) = \sum_{(\mathbf{p}, \mathbf{q}) \in \mathcal{K}} \|\mathbf{p} - \mathbf{T}\mathbf{q}\|^2. \quad (18)$$

Another variant, Point-to-Plane ICP, instead measures the distance from a source point $\mathbf{q}$ to the tangent plane at the corresponding target point $\mathbf{p}$, with normal $\mathbf{n}_p$. Its objective function is

$$E(\mathbf{T}) = \sum_{(\mathbf{p}, \mathbf{q}) \in \mathcal{K}} ((\mathbf{p} - \mathbf{T}\mathbf{q}) \cdot \mathbf{n}_{\mathbf{p}})^2. \quad (19)$$

The choice between these two variants depends on the application. Point-to-Plane ICP is generally computationally faster, as it converges in fewer iterations [69, 70]. Li et al. [71] further demonstrate the fast convergence of Point-to-Plane ICP while highlighting scenarios where each variant may be preferable. Point-to-Point ICP is more robust for poor initial guesses and large misalignments but is more sensitive to noise. Conversely, Point-to-Plane ICP requires a better initial alignment but is less sensitive to noise, making it suitable when precision is required or computational speed is a priority.

A practical strategy is to first perform a coarse registration with a larger correspondence threshold to handle substantial misalignment, followed by a refinement using ICP with a tighter threshold for more accurate alignment.

3.2.3 3D Models Representations and Generations

This subsection focus on common representations for 3D geometries and algorithms to obtain a dense representation with mesh.

Point Cloud Representation A point cloud can be defined as an unordered set—which can pose challenges in some applications—of 3D coordinates:

$$\mathbf{P} = \{\mathbf{p}_i \in \mathbb{R}^3 \mid i = 1, \dots, N\}, \quad (20)$$

where each point $\mathbf{p}_i = (x_i, y_i, z_i)$ may also be associated with additional attributes, such as color (r_i, g_i, b_i) or normals $\mathbf{n}_i$, which can be used in algorithms like the previously described Point-to-Plane ICP. Point clouds can be acquired using various methods, including LiDAR devices, stereo vision, or structured light.

Although point clouds are easy to generate and provide a sparse geometric representation of a scene, they lack explicit surface connectivity, which can limit their applicability for certain operations. Therefore, more structured surface representations are often required.

Surfel Representation A surfel—short for surface element—extends the concept of a point cloud by representing each point as a small oriented disk in 3D space:

$$s_i = (\mathbf{p}_i, \mathbf{n}_i, r_i, w_i, c_i, t_i), \quad (21)$$

where

- $\mathbf{p}_i$ is the 3D position of the centre of the disk,
- $\mathbf{n}_i$ is the surface normal of the disk,
- r_i is the surfel radius, which can be defined as the distance from the centre point to its neighbours,
- w_i is a weight, often based on confidence or the number of observations, used to update the properties of the element from multiple views,

- c_i is the RGB colour of the disk for texture and visualization, and
- t_i is a timestamp, primarily useful for updating or trimming surfels in dynamic mapping or SLAM.

Surfel-based representations capture both local orientation and appearance, enabling real-time incremental updates without requiring full mesh re-computation.

Poisson Reconstruction Poisson surface reconstruction converts an oriented point cloud (points with normals) into a watertight surface by solving a volumetric partial differential equation. It's based on the observation that the gradient of the indicator function $\chi(\mathbf{x})$ —a function that is 1 inside the object and 0 outside—should match a vector field $\mathbf{V}$ constructed from the oriented points' normals. It can be expressed mathematically as

$$\nabla \chi \approx \mathbf{V} \quad (22)$$

where

- $\mathbf{V}(\mathbf{x})$ is a 3D vector field containing surface normals at the surface points and zero elsewhere.
- Normals are interpreted as a flow pointing outward from the surface.

Taking the divergence of both sides gives the Poisson equation:

$$\nabla \cdot \nabla \chi = \nabla \cdot \mathbf{V} \quad (23)$$

The reconstruction process (Fig. 7) begins with normal orientation, ensuring that all normals are consistently directed outwards. Next, the oriented points are embedded into a volumetric grid to construct a vector field, where each voxel containing a sample is assigned the corresponding normal vector. A Poisson equation is then solved for χ over the grid, producing a scalar field whose zero-level set approximates the surface boundary. Finally, the surface defined by $\chi(\mathbf{x}) = \tau$ (typically with $\tau = 0.5$) is extracted using a modified version of the Marching Cubes algorithm.

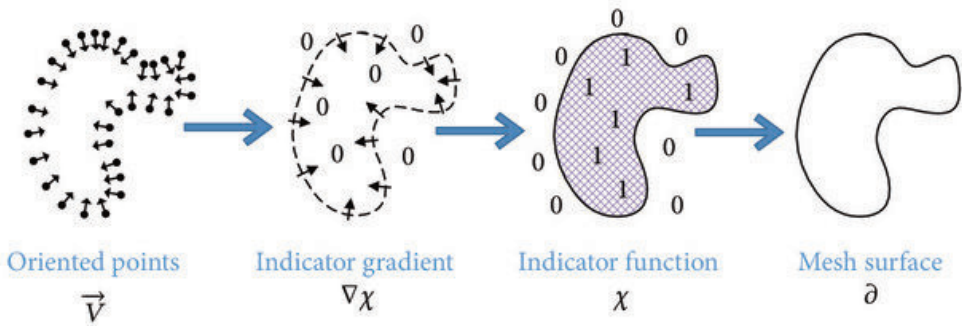


Figure 7. Poisson reconstruction process in 2D. Adapted from Kazhdan et al. [35].

Truncated Signed Distance Function (TSDF) This algorithm is used to first create a volumetric representation and then, using an additional algorithm such as Marching Cubes, extract a 3D mesh from range images (depth maps). The first step is the computation of a Signed Distance Function (SDF). In this approach, the space is discretized into a grid composed of voxels (volume elements). The goal is to assign to each voxel at a 3D position $\mathbf{x}$ a signed distance to the closest surface, with the sign defined as follows:

$$d(\mathbf{x}) = \begin{cases} + \text{ distance} & \text{if outside surface ,} \\ - \text{ distance} & \text{if inside surface .} \end{cases} \quad (24)$$

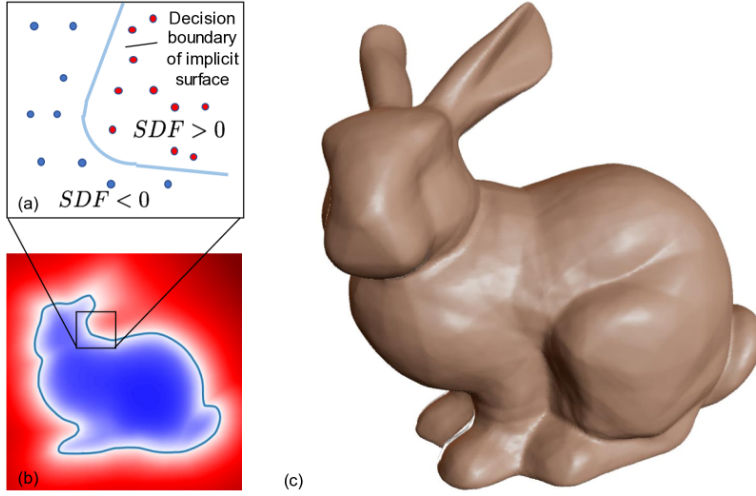


Figure 8. Concept and representation of the signed distance function (SDF) for the Stanford Bunny. Adapted from Park et al. [72].

To compute the signed distance $d(\mathbf{x})$ at a given voxel, we consider its centre in homogeneous world coordinates $\mathbf{X} = [X \ Y \ Z \ 1]^T$. The centre point is then transformed into the camera coordinate system using the camera pose (extrinsic matrix) to obtain $\mathbf{X}_c = [X_c \ Y_c \ Z_c \ 1]^T$. For the computation of the TSDF, the camera poses must be known, either provided by the device or estimated using algorithms such as ICP or RGB-D odometry. Using the intrinsic matrix $\mathbf{K}$, the centre is projected onto the image plane, and the corresponding depth measurement z_m is retrieved from the pixel coordinates (u, v) . The signed distance is then computed as the difference between the depth of the projected voxel centre and the measured value.

$$d(\mathbf{x}) = z_m - Z_c \quad (25)$$

Therefore, if $Z_c < z_m$, the voxel lies in front of the surface, i.e., outside the object, and $d_i > 0$. Conversely, if $Z_c > z_m$, the voxel lies behind the surface, i.e., inside the object, and $d_i < 0$. The signed distance value is then clipped to the range $[-\mu, \mu]$ to reduce the influence of noise.

However, in practice multiples depth map are combined and the signed distances are updates. For the i -th range image:

- $d_i(\mathbf{x})$ is the signed distance function from that image
- $w_i(\mathbf{x})$ is a weight function that decreases with distance from the observed surface point and is zero behind the surface.

The cumulative signed distance function $D(\mathbf{x})$ and cumulative weight $W(\mathbf{x})$ are computed as

$$D(\mathbf{x}) = \frac{\sum_i w_i(\mathbf{x})d_i(\mathbf{x})}{\sum_i w_i(\mathbf{x})} \quad (26)$$

Marching Cubes The Marching Cubes algorithm can extract a mesh from a scalar field, such as a TSDF, by identifying where the field equals a chosen isovalue τ . The isovalue represents the surface boundary — for a TSDF, this is typically zero, corresponding to the zero-crossing between the interior and exterior of the object. The algorithm proceeds as follows:

1. Divide the volume into a grid of voxels, each defined by its corner values $v_0, v_1, \dots, v_7$. The initial volumetric representation is the TSDF, which is why it can be used to extract a mesh.
2. Classify each corner as inside if $v_i < \tau$ or outside if $v_i > \tau$. For a TSDF, the corner values correspond to the stored voxel signed distances.
3. For each cube, compute an 8-bit index with one bit per corner. Although there are $2^8 = 256$ possible configurations, symmetries reduce this to the 15 unique cases shown in Fig. 9.
4. Use a lookup table to map each index to a triangle pattern.
5. If the scalar changes value along an edge, rather than taking the midpoint, the surface crossing can be determined by interpolation, such as

$$\mathbf{p} = \mathbf{p}_a + \frac{\tau - v_a}{v_b - v_a}(\mathbf{p}_b - \mathbf{p}_a) \quad (27)$$

where $\mathbf{p}_a$ and $\mathbf{p}_b$ are the points that form the edge and v_a and v_b their respective values

6. Connect the interpolated points to form the mesh

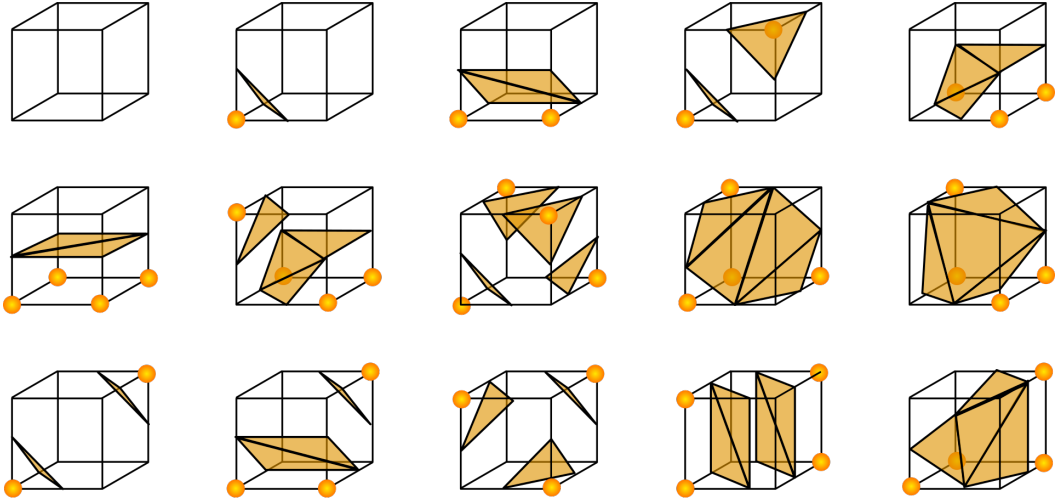


Figure 9. Possible configurations in the original Marching Cubes algorithm.

3.3 Geometric Alignment and Surface Refinement in 3D Scans

3.3.1 Surface Refinement

Direct Laplacian filtering is a technique used to smooth or adjust values in a dataset based on their relationship with neighbouring points. In this approach, the updated value v_i at a given location i is updated with weighted contribution from its neighbouring values, scaled by a parameter λ . The general formulation is given by

$$v_i = v_i \cdot \lambda \sum_{n \in N} w_n v_n - v_i \quad (28)$$

where:

- v_i is the value at the i -th vertex.
- N denotes the set of neighbouring vertices around i .
- w_n is the weighting factor associated with the n -th neighbour, determining its influence.
- λ is a scaling parameter controlling the overall effect of the Laplacian operation.

This formulation enables the redistribution of values based on local spatial relationships, effectively capturing variations in the dataset while applying controlled smoothing.

While the direct Laplacian filtering approach is straightforward to implement, it often suffers from undesirable side effects such as mesh shrinkage. An alternative is spectral mesh processing, which operates in the spectral domain defined by the eigen-decomposition of the mesh Laplacian. This allows for more controlled editing of the mesh geometry by manipulating the frequency components of the graph signals associated with the mesh vertices.

A mesh can be represented as a graph $\mathcal{M} = \{\mathcal{V}, \mathcal{E}, \mathbf{A}\}$, where:

- $\mathcal{V}$ is the set of vertices,
- $\mathcal{E}$ is the set of edges,
- $\mathbf{A}$ is the adjacency matrix.

The coordinates of each vertex $v_i \in \mathcal{V}$ are interpreted as *graph signals*. In graph theory, the Laplacian matrix is defined as:

$$L = D - A \quad (29)$$

where D is a diagonal degree matrix with entries $d_{ii} = \sum_{j=1}^n a_{ij}$.

The Laplacian can be eigen-decomposed as:

$$L = Q \Lambda Q^\top \quad (30)$$

where Q contains the eigenvectors and Λ is a diagonal matrix containing the eigenvalues in ascending order.

For a graph signal $s \in \mathbb{R}^3$, the Graph Fourier Transform (GFT) is defined as:

$$\hat{s} = Q^\top s \quad (31)$$

and the inverse as:

$$s' = Q\hat{s}_f \quad (32)$$

The Graph Fourier Transform (GFT) is used to project the signals defined by the vertex coordinates into the spectral domain. In this domain, the spectral representation of the coordinates can be analysed, and filtering can be applied to $\hat{s}$, for example by setting the last n values to zero to implement a low-pass filter. The Inverse Graph Fourier Transform (IGFT) is then used to retrieve the filtered vertex coordinates s' .

Although spectral processing offers better control and avoids shrinkage effects, the eigen-decomposition of large Laplacian matrices is computationally expensive. In practice, a Singular Value Decomposition (SVD) formulation is often used, but its cost is prohibitive for large meshes. To mitigate this, the mesh can be partitioned into smaller sub-meshes, making the spectral computations tractable.

The MeTiS algorithm is widely used for mesh and graph partitioning. It operates in three stages:

1. **Coarsening** the graph into progressively smaller graphs. One common method for graph coarsening is Heavy Edge Matching (HEM). HEM reduces the size of a graph by iteratively merging pairs of nodes connected by high-weight edges. The process works by visiting nodes in random order and attempting to match each with an unmatched neighbour along the heaviest edge. Once a match is made, both nodes are removed from the unmatched set, and the process continues until no more pairs can be formed. The resulting set of matched edges defines a smaller, coarser graph. The original mesh $\mathcal{M}_0$ is iteratively reduced to $\mathcal{M}_N$, where at stage N the number of nodes is significantly smaller. A typical stopping condition is when the number of nodes n in $\mathcal{M}_i$ is less than $k \times 10$, where k is the number of target partitions.
2. **Partitioning** the smallest (coarsest) graph. Several algorithms exist for this step, as presented by Elsner [73], but a greedy graph growing algorithm can be used since refinement will occur later. The process starts with a pseudo-peripheral node (far from most other nodes) as the starting vertex and adds it to the first partition. For each partition, the algorithm repeats the same process until the partition reaches the target size. Among all unmarked vertices adjacent to the current partition, choose the one with the fewest unmarked neighbours. If unmarked vertices remain, select one adjacent to the current partition with the fewest unmarked neighbours as the new starting vertex.
3. **Uncoarsening** and refining the partitions. During the uncoarsening process, partitioning algorithms can be applied to refine the partitions. The Kernighan-Lin algorithm is commonly used here, swapping nodes at neighbouring regions to minimize the number of edges between partitions.

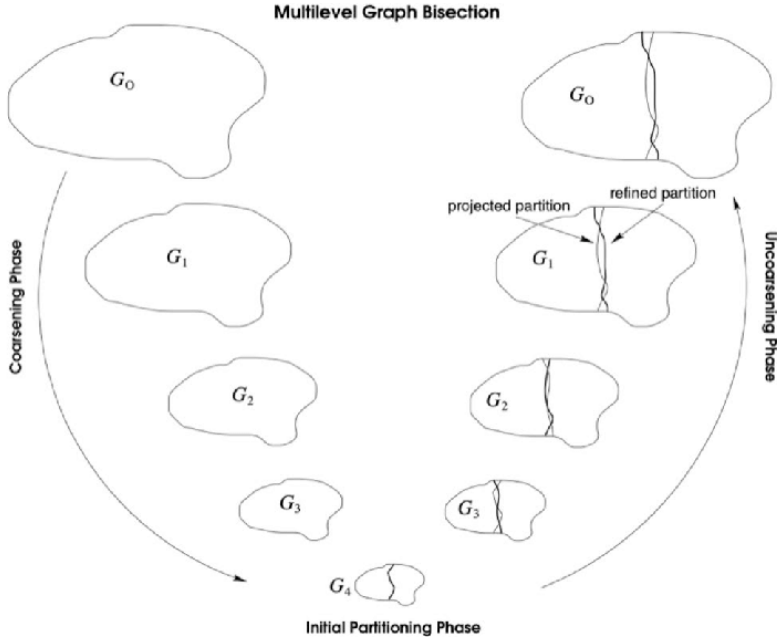


Figure 10. Multi-level graph partitioning using the MeTiS algorithm [74].

To efficiently compute the eigenvectors of submeshes, Orthogonal Iteration (OI) is employed. OI is an iterative procedure that approximates the eigenvectors of a symmetric matrix by repeatedly multiplying a subspace by the matrix and orthonormalizing the result. While the eigenvectors of the first submesh are computed using a full SVD, the resulting vectors can serve as an initialization for OI on subsequent submeshes, exploiting the coherence between the spectral components of adjacent submeshes. This initialization accelerates convergence, as starting from a subspace close to the target eigenvectors leads to a rapid solution. Once the eigenvectors are obtained for all the submeshes, the vertices positions of the mesh are updated as expressed for the brute implementation. In the case of overlap (as the partitions need to be the same size of OI), a weighted average is done based on the number of neighbours the vertex has in each submesh.

4 METHODOLOGY

4.1 3D Scan Enhancement

4.1.1 Current and Proposed Workflow

This subsection compares the current 3D reconstruction workflow, based on the StandardCyborg library, with a proposed alternative leveraging a volumetric representation through Truncated Signed Distance Function (TSDF) fusion. Both approaches aim to build real-time 3D scans from RGB-D data, but differ significantly in terms of data structure, precision, and extensibility. The overall workflows can be seen in Fig. 11 below.

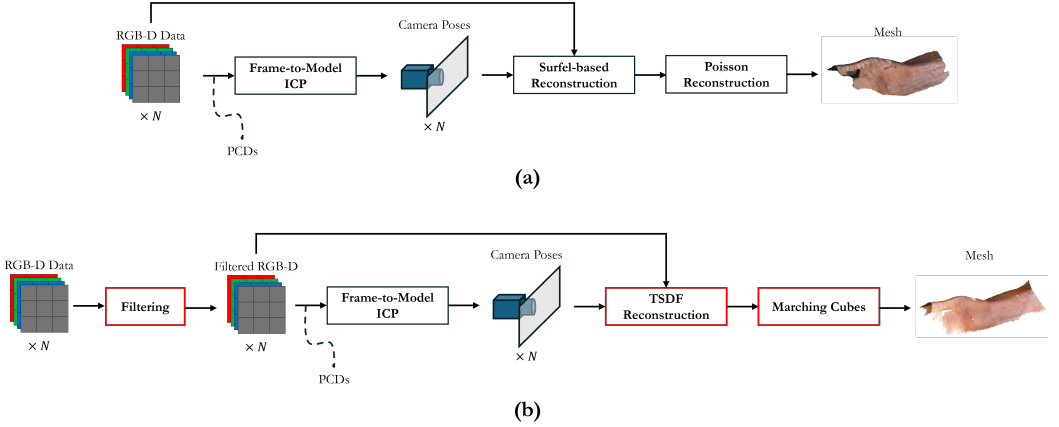


Figure 11. Reconstruction workflows for (a) the current process and (b) the proposed process, with modifications and additions highlighted in red.

Current Workflow (StandardCyborg Surfel-Based Reconstruction) The workflow implemented by StandardCyborg, as described in Sec. 3, adopts a surfel-based representation. To estimate the poses, a Frame-to-Model ICP approach is employed, meaning that a new frame point cloud is aligned with an updated model constructed from the previous frames rather than just the previous frame. In this process, the depth map is first unprojected into world coordinates to generate a point cloud. Both the model’s and the frame’s point clouds are then downsampled in order to improve computational efficiency during the search and matching of correspondences. An initial guess for the transformation is obtained by applying the previous extrinsic matrix, which is subsequently refined using ICP. This procedure avoids the use of an identity matrix as an initial guess, which would be inadequate in cases involving large motions.

To ensure accuracy, the quality of the ICP alignment is evaluated by monitoring the convergence of the algorithm, while the precision of the pose estimation is further assessed by examining both linear and angular velocities. Once the pose has been determined, the 3D scan is constructed by performing surfel fusion over multiple frames. At this stage, a lookup table is created to map pixels to surfel indices, enabling efficient integration of new data. Each frame is assimilated only after validating the depth information, using both a predefined threshold and the angle of incidence as criteria. When a corresponding surfel is found within close proximity, the new information is merged with it; otherwise, a new surfel is created. The weighting assigned to each surfel is determined based on the incidence angle, allowing the system to prioritise more reliable observations.

Over time, surfel maintenance is performed in order to preserve the quality of the reconstruction. This involves culling low-confidence surfels according to their weights and lifetimes, with the latter being decremented at each frame. Once the surfel representation is complete, a Poisson surface reconstruction is applied to obtain a mesh, after which UV unwrapping is used to generate the texture.

From an observational perspective, this workflow is well suited to real-time applications; however, it may not provide the highest level of precision. On the positive side, the weighted merging of surfels can progressively reduce noise over time. Nevertheless, the quality of the reconstruction remains strongly dependent on resolution: lower resolution results in fewer points, causing surfels to have larger radii and thus to average out fine details. Furthermore, the method is sensitive to noise, which can lead to instability and difficulties in accurately reconstructing sharp edges. The Poisson reconstruction step may also introduce inaccuracies and reduce consistency between the point cloud and the final mesh.

The accuracy of the results can be further affected by potential drift in ICP pose estimation, even though motion validation can help mitigate this issue. While confidence-based culling effectively removes unstable surfels, it may also discard useful data, resulting in holes or loss of detail. In addition, sparse point clouds tend to produce poor-quality Poisson reconstructions. Parameter tuning is critical, yet the optimal configuration may vary considerably between scenes. Although the method is likely sufficient for medium-scale scenes such as horse back scans, scalability to significantly larger environments could be limited. Finally, because normals are estimated locally, inconsistencies may occur, which in turn could negatively affect the mesh quality.

Proposed Workflow (TSDF-Based Volumetric Reconstruction) The proposed workflow relies on a volumetric representation based on a Truncated Signed Distance Function (TSDF). Pose estimation is performed using a Frame-to-Model ICP approach, in which incoming RGB-D frames are first unprojected into 3D space to generate point clouds. To reduce noise and limit drift, a bilateral filter is applied to the data, and the model is updated every five frames rather than continuously. As in the StandardCyborg library, it is also possible to evaluate the quality of the ICP alignment and pose estimation by analysing both linear and angular velocities, although this step is optional in the present implementation.

Once the pose has been estimated, the RGB-D frames are integrated into the TSDF volume, following the process described previously. Before integration, the frames are filtered using one of the methods presented in Sec. 3 to enhance reconstruction accuracy; a key objective here is to determine which filter is most effective for this task. After the volumetric representation has been updated, a point cloud and mesh are extracted using the marching cubes algorithm, adapted to operate directly on volumetric data.

From an observational perspective, the method is suitable for real-time applications, although it is computationally more demanding than a surfel-based approach, requiring a trade-off between processing speed and accuracy. One advantage of the volumetric representation is that reconstruction precision is decoupled to a greater extent from the input image resolution, being instead determined by the voxel size. However, this introduces a need for parameter tuning, as the choice of voxel size must balance efficiency against precision. Similarly, the truncation distance must be carefully set to optimise performance.

The approach offers several benefits: noise is mitigated through voxel weighting and incremental updates, and because the TSDF is voxel-based, the marching cubes extraction process ensures

geometric consistency between the reconstructed point cloud and mesh, resulting in higher mesh fidelity. Potential drift from ICP remains possible but is mitigated in the same way as in the StandardCyborg implementation.

On the other hand, the use of the Open3D framework limits access to deeper optimisations, such as the integration of advanced confidence measures, adaptive weighting strategies, or dynamic voxel pruning, unless modifications are made directly to the source code. Therefore, even after all the process that is done to help reduce the noise, the region that are seen in less frames and less updated can still be noisy and looking like oscillating.

Summary Comparison While both methods are designed for real-time reconstruction, the proposed TSDF-based approach theoretically demonstrates several advantages in terms of mesh consistency, resilience to noise, and independence from image resolution. These benefits come at the cost of potential higher computational demand and reduced flexibility due to current library limitations. Given the goal of improving scan fidelity for specific use cases (e.g., scanning of equine backs), the proposed method represents a compelling alternative, particularly where fine detail and surface accuracy are prioritized.

4.1.2 Evaluation Targets

Three primary targets are used to evaluate the proposed workflow, determine the most suitable filtering method, and compare it to the current approach. These targets are designed to validate the new process progressively, starting from controlled scenarios and moving toward real application cases. Each target is associated with specific evaluation methods to assess reconstruction accuracy. The filters tested are those described in Sec. 3, namely: the bilateral filter, the joint bilateral filter (applied after upsampling the depth map with interpolation, as little difference was observed compared to joint bilateral upsampling), the unified multi-lateral filter, and the guided filter. For reference, a TSDF reconstruction without any filtering is also included.

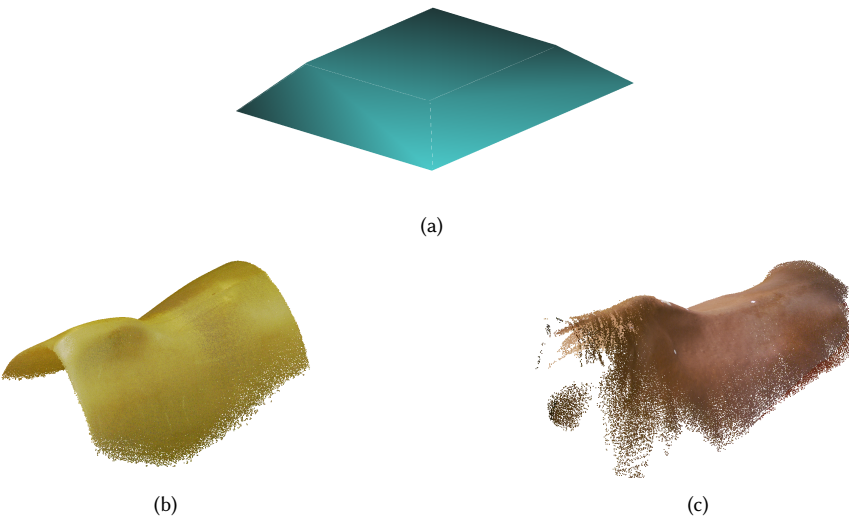


Figure 12. Ground truth of the evaluation targets: (a) Frustum, (b) *Courbette*, and (c) Horse.

Frustum Model

Rationale: The frustum model is selected as the first evaluation target because it provides a simple geometric shape that facilitates the validation of scale and reconstruction quality. It is fabricated with known distances, allowing for straightforward and precise dimensional verification. A slight angular variation is introduced to also assess reconstruction under non-perpendicular viewing conditions. The model’s geometry is available as an accurate CAD file, serving as a precise reference.

Evaluation Method: All filters are applied to this dataset, as it serves as the initial benchmark for precision measurements. The reconstructed scans are compared against the CAD reference, focusing on dimensional accuracy and the ability of each filtering method to preserve the true geometry of the model. The capacities of the filters will also be looked at visually. For the tuning of the parameters of the filters, i.e. the kernel sizes, range and spatial variances will be defined dynamically as explained in 3. In the case of quantization 8 bins are used.

Courbette (Horse Back Model for Prototyping)

Rationale: The *Courbette* model presents a more complex geometry that closely resembles the intended final application target. As a static object, it avoids the motion-related challenges encountered when scanning a live horse. Landmarks in the form of stickers can be placed on the surface to enable distance measurements, and the model’s smooth surface helps assess the performance of filters in low-texture regions. This scenario also allows for testing whether incorporating RGB data improves reconstruction, particularly in the withers area where steeper angles may occur.

Evaluation Method: Reference data is acquired using a FARO laser scanner with sub-millimetre accuracy (under 0.5 mm). All filters are tested to determine whether edge-preserving methods, such as guided depth super-resolution, offer measurable benefits. The focus is on two aspects: smoothing in flat areas and edge accuracy in regions with notable surface curvature. Most specifically for the filters, the aim is also to see if the edges are significant enough to justify the use of RGB data. Before looking at the precision of the scans and influence of the filters, multiple voxel sizes from 1 to 5 mm will be assessed to find the best trade-off between precision and computing time.

Live Horses

Rationale: Scanning live horses provides the most realistic and challenging evaluation scenario. It introduces variability in shape, colour, and environmental conditions, all of which can influence reconstruction quality. Since horses are not perfectly static, this target captures the challenges of motion and deformation, including subtle changes in posture or skin stretch between scans.

Evaluation Method: Multiple scans are performed on different horses, and in some cases repeated on the same horse to assess process repeatability. Landmarks for distance measurement are not used, as skin movement could easily introduce measurement bias. Reference scans are acquired using a FARO laser scanner; however, due to changes in the subject’s pose, direct geometric comparison is not always possible. Instead, the evaluation focuses on detecting anomalies in precision and surface smoothness. Depending on the results, comparisons may be limited to the most promising filters from the previous evaluation rather than all available options.

Acquisition Workflow for Evaluation

In all cases, the reconstructions are performed without modifying the main scanning application. During acquisition, RGB-D frames are saved as binary files along with their associated metadata. The proposed workflow is then executed offline on an external computer, rather than directly on the iPhone, to ensure flexibility in testing and analysis.

4.1.3 Evaluation Metrics

This section details the common metrics used in 3D reconstruction evaluations, which in this case are primarily based on the distance from the model to its ground-truth reference.

Standard Deviation The standard deviation measures the amount of variation or dispersion in the distance values. It quantifies how much the individual distances deviate from the mean distance $\bar{d}$. It is defined as:

$$\text{STD}(\mathbf{d}) = \sqrt{\frac{1}{N} \sum_{i=1}^N (d_i - \bar{d})^2} \quad (33)$$

where d_i is the distance from the reconstructed point to the ground truth, $\bar{d}$ is the mean of all distances, and N is the total number of points.

Root Mean Square Error The root mean square error (RMSE) is a commonly used metric that provides a single measure of the magnitude of the error. It penalizes larger errors more heavily due to the squaring of the distance values. RMSE is defined as:

$$\text{RMSE}(\mathbf{d}) = \sqrt{\frac{1}{N} \sum_{i=1}^N (d_i)^2} \quad (34)$$

where d_i is the distance from each point in the reconstructed scan to the corresponding ground truth, and N is the number of such distances.

Mean Absolute Error The mean absolute error (MAE) provides an average of the absolute distance errors, offering a straightforward measure of the average deviation regardless of direction. It is defined as:

$$\text{MAE}(\mathbf{d}) = \frac{1}{N} \sum_{i=1}^N |d_i| \quad (35)$$

where $|d_i|$ represents the absolute distance between the reconstructed and ground truth point for each i , and N is the total number of points.

Precision Also referred to as *accuracy*, precision is defined as the proportion of distances from the reconstructed 3D scan to the ground truth that fall below a given threshold. It is formally expressed as:

$$P(\mathbf{d}, \tau) = \frac{100}{N} \sum_{i=1}^N [d_i < \tau] \quad (36)$$

where d_i denotes the minimum distance from a point in the reconstructed scan to the nearest point in the ground truth model, and $[\cdot]$ is the Iverson bracket, which evaluates to 1 if the condition is true and 0 otherwise. This metric can be applied to either point clouds or meshes by computing distances from the reconstructed vertices to the ground truth surface.

By reversing the direction of the distance computation—that is, measuring distances from the ground truth to the reconstructed model—a complementary metric known as *completeness* (or *recall*) can be evaluated. Those two metrics are commonly used in benchmarking of 3D reconstruction such as in Tanks and Temples dataset by Knapitsch et al. [75]. However, completeness is not considered in this work, as it is not relevant to the specific evaluation objectives.

Roughness For a given mesh vertex $\mathbf{v}_i \in \mathbb{R}^3$, the local roughness is defined as the absolute perpendicular distance between the vertex and a best-fitting plane estimated from its local neighbourhood.

Let:

- $\mathcal{N}_i$ be the set of neighbouring vertices within a kernel of radius r centred at $\mathbf{v}_i$,
- $\mathbf{p}_i$ be the centroid of $\mathcal{N}_i \cup \{\mathbf{v}_i\}$,
- $\mathbf{n}_i$ be the unit normal vector of the best-fitting plane to the neighbourhood

Then, the local roughness R_i is given by:

$$R_i = |\mathbf{n}_i \cdot (\mathbf{v}_i - \mathbf{p}_i)| \quad (37)$$

This scalar value quantifies the deviation of vertex $\mathbf{v}_i$ from the locally approximated surface and provides a measure of surface irregularity at that point. The average of the roughness of the vertices can also be done to assess the overall smoothness of the reconstructed scans. CloudCompare implementation is used for this with the windows size dynamically defined based on the neighbours distances ensuring comparison between meshes with different resolutions.

Measurements For the *Courbette* dataset, which is static and therefore more reliable for evaluating the workflows, five stickers are placed on the surface prior to scanning to serve as landmarks. Measurements between these landmarks, as illustrated in Fig. 13, are taken manually for comparison with the reconstructed mesh. The measurements correspond to arcs, defined as the intersection between a plane and the mesh connecting two points.

To compute these, the points are first selected manually. For each pair of points corresponding to a measurement, a third point is defined to construct a plane. From this, the plane equation of the intersection is expressed as $Ax + By + Cz + D = 0$, where, given two vectors $\mathbf{u}$ and $\mathbf{v}$ formed from the three points P_1, P_2, P_3 , the plane coefficients can be determined as follows:

$$\mathbf{n} = \mathbf{u} \times \mathbf{v} = (A, B, C), \quad D = -\mathbf{n} \cdot P_1 \quad (38)$$

Once the plane is defined, its intersection with the mesh can be computed. A mesh is represented as a collection of triangles, each defined by three vertices $T(v_1, v_2, v_3)$. Therefore, the intersection of the plane and the mesh can be understood as the intersection between the plane and the edges of these triangles. To determine this, it is necessary to evaluate the position of the triangle vertices relative to the plane by computing their distance to it. For a vertex $v_i = (x_i, y_i, z_i)$, this distance can be computed as follows:

$$d(v_i) = Ax_i + Bx_i + Cz_i + D \quad (39)$$

where

- If $d(v_i) > 0$ the vertex lies on one side of the plane
- If $d(v_i) < 0$ the vertex lies on the other side
- If $d(v_i) = 0$ the vertex lies exactly on the plane

For each edge, defined by two vertices v_i and v_j , the distances d_i and d_j to the plane are computed. If the signs of these distances are the same, the edge does not intersect the plane. If the signs are different, the edge crosses the plane, and the intersection point can be computed as

$$t = \frac{d_i}{d_i - d_j}, \quad 0 \leq t \leq 1 \quad (40)$$

$$v_{int} = v_i + t(v_j - v_i)$$

In this way, the intersection points are collected to form a curve that lies on the surface of the mesh. However, the set of intersection points often contains many points, some of which lie beyond the desired measurement range. To retain only the points within the range defined by the two landmarks, the 3D intersection points are first projected into 2D, with the axes chosen depending on the measurement being considered. A principal component analysis (PCA) is then performed on the points to determine the principal axis for projection, and only the points falling between the two landmark points along this axis are retained.

This procedure is preferred over using the raw Cartesian coordinates because, at this stage, the scan may still be subject to disorientation. For certain measurements, such as 1a and 1b, the scan can be poorly reconstructed due to steep angles, in which cases these measurements are considered outliers and excluded.

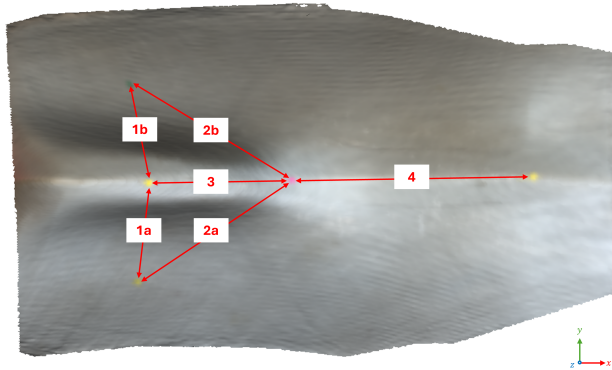


Figure 13. Measurements used for evaluating the quality of *Courbette* meshes.

This provides additional information on the quality of the mesh, its scale relative to reality, and the potential impact of the chosen reconstruction method. Furthermore, it supports subsequent analyses by confirming the feasibility of performing automated measurements on the meshes.

Structural Similarity Index (SSIM) The Structural Similarity Index (SSIM) is a metric commonly used to assess the similarity between two images. The evaluation is based on a combination of structure, luminance, and contrast. In this work, SSIM is employed to compare the similarity between two depth maps, specifically between the brute-force implementation and the real-time implementation using quantization. SSIM values range from -1 to 1 , where 1 indicates perfect similarity, 0 indicates no similarity, and -1 indicates perfect dissimilarity.

$$SSIM = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (41)$$

where

- μ_x and μ_y are the means of the depths maps compared
- σ_x^2 and σ_y^2 are the variances of the two depth maps
- c_1 and c_2 are two scalars used to stabilize the division

4.2 Geometric Alignment and Surface Refinement in 3D Scans

4.2.1 Proposed Approach

In the context of the reconstructed scans using the proposed method, let x_h represent the principal axis of the scan extending from the front to the back of the horse—potentially aligned with the backline—while y_h denotes the width and z_h the height of the horse. The objective is to align these local axes with the canonical coordinate system x, y, z , with a particular focus on ensuring that the XZ -plane corresponds to the dominant plane of symmetry.

The proposed method (see Fig. 15) is specifically developed for the horse’s back, which presents a challenging scenario in comparison to the objects discussed in Sec. 2. Unlike those objects, which exhibit significant local curvature, the horse’s back is generally smooth and lacks distinctive features that facilitate symmetry detection. As such, the method does not rely on local features, which may be insufficient or unreliable in this context as can be seen in Fig. 14. The application of the work of Sipiran et al. [45] for the extraction of features shows that distinctive features are highlighted on the Armadillo, based on the curvature, while for a horse scan the features are the corners that show no information useful for symmetry plane detection and rather based on the truncation or area scanned.

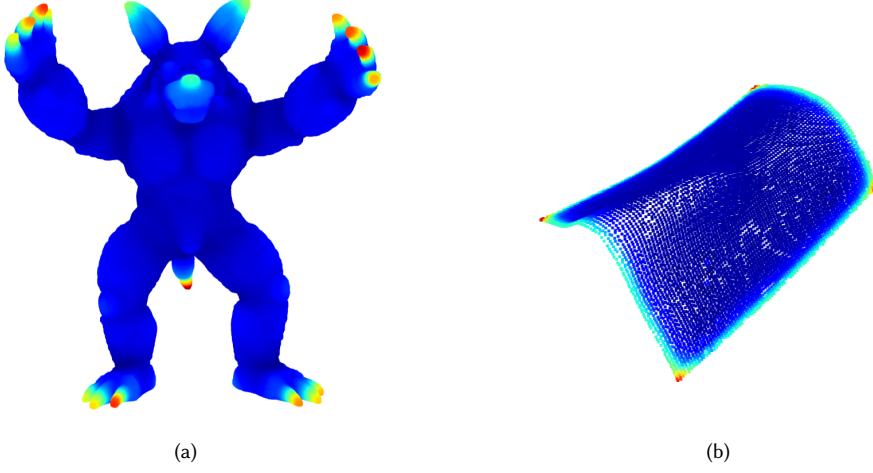


Figure 14. Application of the heat diffusion filter from Sipiran et al. [45] to extract features on **(a)** Armadillo and **(b)** *Courbette* meshes.

Initially, the region of interest is reduced by projecting the point cloud from a top-down perspective and identifying the largest rectangle that fits within the convex hull. This is achieved via a brute-force method, in which the space is discretised into a grid and all candidate rectangles are evaluated accordingly.

Subsequently, Principal Component Analysis (PCA) is employed to determine the main components of the model and guide the orientation process. Either the point cloud derived from the Truncated Signed Distance Function (TSDF) volume or the mesh vertices may be used for this purpose, with the choice potentially affecting computational efficiency. As discussed in Sec. 2, PCA alone is insufficient for achieving reliable alignment due to the partial symmetry of the scanned region and the influence of the scanning device’s orientation—here, an iPhone.

PCA is used in a preliminary step to estimate the orientation of the most appropriate symmetry plane. This involves using the model’s centre point and evaluating each principal component as a potential normal vector for a candidate symmetry plane. For each such plane, a symmetry score is computed by measuring the mean distance between the model and its mirror image. Given a point $p_i \in P$ and a candidate symmetry plane defined by a point $c \in \mathbb{R}^3$ and a normal vector $\mathbf{n}_p$, its mirror $q_i \in Q$ is computed as

$$q_i = p_i - 2 \cdot \frac{(p_i - c) \cdot \mathbf{n}_p}{\|\mathbf{n}_p\|^2} \cdot \mathbf{n}_p \quad (42)$$

where

- c the centre point of the model, $(p_i - c)$ centre the points
- $\mathbf{n}_p$ the normal of the plane used for the symmetry. $(p_i - c) \cdot \mathbf{n}_p$ projects the points onto the normal and $\|\mathbf{n}_p\|^2$ is used to normalize the normal if not already done.

For all the points $p_i \in P$, and corresponding mirrored points $q_i \in Q$, the signed distances to the planes are

$$d_i^P = (p_i - c) \cdot \mathbf{n}_p, \quad d_i^Q = (q_i - c) \cdot \mathbf{n}_p \quad (43)$$

The meshes are partitioned by splitting the point sets based on the sign of their signed distances

$$\begin{aligned} P^+ &= \{p_i \mid d_i^P > 0\}, & P^- &= \{p_i \mid d_i^P \leq 0\} \\ Q^+ &= \{q_i \mid d_i^Q > 0\}, & Q^- &= \{q_i \mid d_i^Q \leq 0\} \end{aligned} \quad (44)$$

Now, for each side of the plane and corresponding meshes halves the mean of the nearest-neighbour distances are computed such as two metrics are obtained

$$d^+ = \frac{1}{|P^+|} \sum_{p \in P^+} \min_{q \in Q^+} \|p - q\|, \quad d^- = \frac{1}{|P^-|} \sum_{p \in P^-} \min_{q \in Q^-} \|p - q\| \quad (45)$$

Then the final symmetry score is computed as the mean of the two side-wise distances.

$$SM = \frac{1}{2}(d^+ + d^-) \quad (46)$$

The plane with the lowest score is selected as the dominant symmetry plane, with its normal vector typically corresponding to the second principal component. In the event that the two lowest symmetry scores are not sufficiently distinctive (i.e. under 5% of relative difference), the second component is used.

The mirrored version of the original model is retained and used to further refine the symmetry estimation. Both the original and mirrored models are then aligned using the Iterative Closest Point (ICP) algorithm, with the point-to-plane variant preferred over point-to-point due to its increased reliability. This alignment is expected to be effective, as the horse's back is relatively smooth and planar, and less affected by pose variations.

Once alignment is complete, two symmetry planes are obtained: one from the original model and one from its mirrored counterpart. These planes are not expected to perfectly align with the horse's anatomical backline and may deviate from both it and the model's centre. However, by averaging the two planes, a refined dominant symmetry plane can be computed. This averaging process reduces the influence of variance in alignment and yields a centre point that is more robustly positioned. Furthermore, the dominant symmetry plane is reoriented to more closely align with the horse's backline, reducing the angular discrepancy.

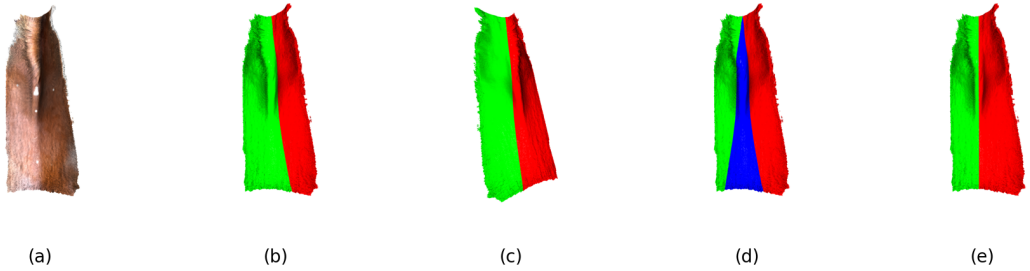


Figure 15. Top-down views illustrating the stages of dominant symmetry plane detection for a 3D horse back model: **(a)** input model prior to orientation, **(b)** model segmented by its principal symmetry plane estimated via PCA, **(c)** mirrored version of the model, also segmented by its PCA-derived symmetry plane, **(d)** original model segmented by both symmetry planes after alignment of the mirrored model to the original, with the region between the planes highlighted in blue, and **(e)** final model segmented by the computed mean symmetry plane, representing the dominant symmetry plane of the horse back scan.

The main limitation of this method is that it cannot easily be transposed to other targets, at least not in an automated way, as the estimation of the best normal works best for a complete model. In the case of missing geometry, the normal must be chosen manually. Still, for missing geometry, a sensitive case would be if one side at the end of the horse’s back is not reconstructed. In such instances, ICP without the invariant part could fail or be imprecise. However, such cases should not pose a problem, as this type of scan is deemed unsatisfactory and should be retaken following a visual check on the mobile app.

Even after the symmetry plane is found and aligned with the XZ plane, some additional orientation is still required, depending on the orientation of the phone during scanning—either not parallel to the ground/horse’s back or due to the position of the camera. These factors will influence the rotation around the y -axis and the side of the neckline of the horse, respectively.

The rotation around the y -axis can be corrected by projecting the principal component of the PCA of the model onto the XZ plane. The angle between the principal component and the x -axis can then be computed, and a rotation applied to the model. This provides an initial adjustment of the scan orientation, but an approach using landmark points must be implemented afterwards for slice extraction and comparison to be accurate.

For the neckline, using only one approach could be unsustainable, as it depends on the orientation of the scan and the extent to which the horse’s back is scanned, which influences the centre point of the scan and its highest points. Therefore, three methods are used, and a flip of the scan direction along the x -axis is performed if at least two methods indicate it is necessary. The methods are as follows:

- Examining the variance along the z -axis as applied after the rest of the orientation has already been done. The scan is split in two based on the position along x of its points or vertices relative to the centre point. Then, the variance is computed for each side. As the withers represent an increase in height compared to the croup, which remains relatively flat, the side with the neckline should show greater variance along the z -axis.
- Two sets of points are selected: one containing points close to the XZ plane, roughly corresponding to the backline, and another set offset along the y -axis on each side of the sym-

metry plane. A polynomial regression is performed for each set of points—only using the x and z coordinates—of degree 4, was deemed suitable for this purpose. The idea is that the withers can be steep compared to the croup, so moving by the same step along y , the height of the corresponding points should be lower near the withers than near the croup. Once the two polynomials are obtained, a set of sample points along x is used to compute the difference in height (by subtracting the offset line from the backline polynomial), resulting in a new function. The maxima and minima of this difference function are taken. The side corresponding to the maximum difference should be the neckline, while the minimum indicates the opposite side. The relative position between these two points is used to determine whether the scan needs to be flipped.

- Slices are taken in the YZ plane to create cross-sections of the horse's back. For each slice, the highest point in z is taken as a reference, and two points—on each side—are selected at a fixed distance in z . The distance between these two points along y is computed for each slice. The logic here is that the horse is narrower around the withers/neckline; therefore, the distance should be smaller at this position compared to the end of the back. As in the previous method, the minimum and maximum values are taken, and their relative positions are used to decide whether the scan needs to be flipped.

Each method has its drawbacks. The first depends on the scanning process: if the phone used is close to the back when scanning the withers and further away at the end, a larger area might be captured, leading to a higher variance in z . Additionally, the extent of the scan influences the centre point and how the scan is split in two. The other two methods are affected by the horse's anatomy. If the back is more rounded and lacks steep angles near the withers, the confidence in the measurement may be reduced. A confidence threshold based on the relative difference between points can be implemented. If the result is deemed unsatisfactory, only the first method is used.

4.2.2 Orientation Evaluation

Compared to the previous section, where the precision of the proposed pipeline can be evaluated using a variety of metrics—each providing different information on the quality of the 3D models—the possibilities for assessing orientation are more limited. While an initial assessment can be made visually, a more precise evaluation must involve quantitative metrics.

To achieve this, the *Courbette* dataset is used. Two stickers are placed along the same line on the x -axis, with their height aligned using a piece of furniture. Although their exact position in real life is not known, these stickers can be used both for the reconstructed model and as a reference for the surface on which it is placed. The aim is to minimise the gap between the x and z coordinates of the centre points of the landmarks. Some tolerance must be allowed, as the stickers are hand-placed and offsets on the order of millimetres may exist. A quick verification can also be performed by cutting the model with a plane parallel to the XY plane to check whether the plane intersects the landmark points or whether there is an offset.

For visual assessment, the model can be coloured differently on each side of the dominant symmetry plane. When viewed in the XZ plane, only one colour should be visible if the symmetry plane has been correctly estimated—though a slight presence of the opposite colour near the neck area may occur, which is not meaningful as it relates to horse movement. Colour can also be used when creating slices to observe how the plane splits the section line.

4.3 Automated Landmark Extraction and Measurement on 3D Surfaces

This section describes the extraction of landmark points from a scan and the measurements that need to be taken. It is important to note that this step is performed after the alignment of the scan, as it relies on a consistent reference frame.

The first step is to extract a reference point from the scan, intended to represent the balance point of the saddle. To achieve this, the back line of the mesh is first determined. Given that the mesh is oriented, two approaches are possible: either directly take the intersection points between the XZ-plane and the mesh, although this may result in the back line not fully following the neck movement; or perform slices of the mesh with planes parallel to the YZ-plane and select the highest z value in each slice. Both methods yield a series of points along the back line.

Next, two specific points are identified along this back line: the lowest point in z after the withers and a local maximum on the neck side. For the lowest point, it is sufficient to focus on the area around the centre of the back line to avoid edge effects from the ends of the scan. Subsequently, 200 evenly spaced points are extracted from left to right along the back line between these two points to serve as input for the trained neural network.

The reference point is estimated using a 1D Convolutional Neural Network (CNN). The model takes the normalized z -coordinates of the 200 points (scaled between 0 and 1) and outputs a single number between 0 and 1, representing the relative position of the reference point along the back line.

The architecture of the network (Fig. 16) is designed to capture both local and global patterns in the z -coordinate sequence. It begins with two 1D convolutional layers, each with 64 filters, a kernel size of 3, and ReLU activation functions, which are effective in detecting local peaks and valleys along the back line. A max pooling layer with pool size 2 is then applied to reduce the dimensionality of the data and summarize local features. The resulting features are flattened into a 1D vector and fed into a fully connected dense network with two hidden layers of 25 neurons each, using ReLU activation. The output layer consists of a single neuron with a linear activation function to produce the relative position.

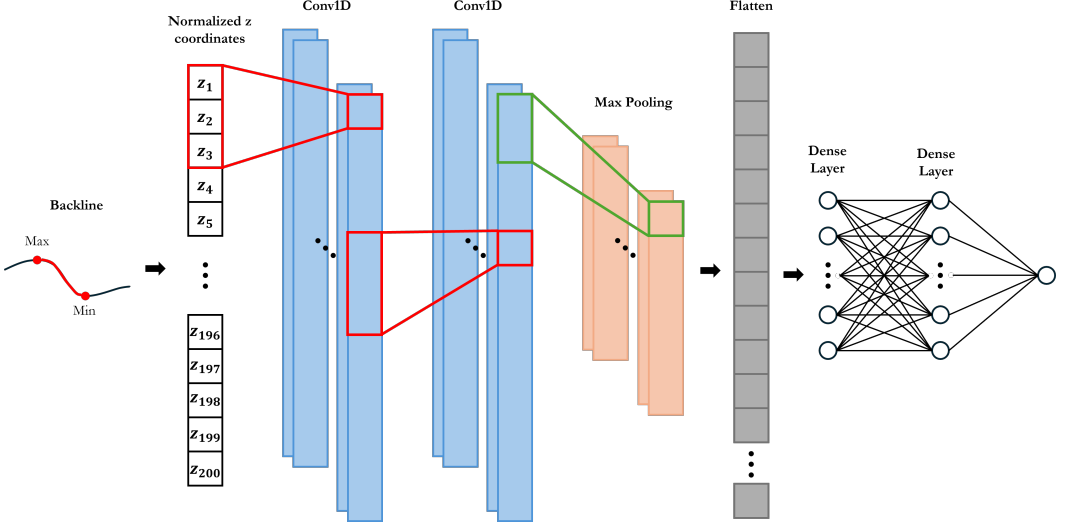


Figure 16. Architecture of the CNN used to extract the relative position of the reference point.

The network is trained using the Mean Squared Error (MSE) loss, which penalizes large deviations more heavily:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (47)$$

where y_i is the true relative position and $\hat{y}_i$ is the predicted value. The Adam optimizer with a learning rate of 0.0001 is used for adaptive learning, and Early Stopping with a patience of 100 epochs prevents over-fitting by stopping training if no improvement is observed.

Once the reference point M_0 is determined, three additional landmark points are obtained by moving along the x -axis from the reference point to positions -16 , $+13$, and $+24$ cm. These points are respectively named M_1 , M_2 , and M_3 . Using these points, the following measurements are extracted:

- **Withers Height:** This is determined by the difference in z -coordinates between M_0 and M_1 . Correction for rotation around the y -axis is important, as an uneven orientation of the mesh can bias the measurement. Since the angle between the principal component and the x -axis has been reduced to 0° , this measurement is equivalent to projecting both points onto the principal axis and taking the distance between the projected points. This ensures consistency across scans.
- **Withers Widths:** Two measurements are taken based on a slice parallel to the YZ plane at M_1 . The first width is measured at $z = z_{M_0}$ (or the closest point), by taking one point on each side of M_1 and computing the distance between them. The second measurement is taken at $z = z_{M_1} - 8$ cm, representing the width at the point of saddle contact. The 8 cm offset accounts for the gap between the top of the withers and the point of saddle contact.
- **Shoulders Width:** Measured similarly to the withers width, but at $z = z_{M_0} - 5$ cm along the slice at M_1 .

- **Withers Angle:** Using the slice at $x = x_{M_1}$, points are selected at ± 5 and 10 cm from the symmetry plane. The angle between the line connecting these points and the z -axis is computed for each side, and the withers angle is defined as the mean of the two angles.
- **Shoulders Angle:** Similar to the withers angle, but with offsets of ± 12 and 17 cm at $x = x_{M_1}$.
- **Back Angle at Straddle Level:** Using a slice at $x = x_{M_2}$ with offsets of ± 3.5 and 7 cm, the angle is computed relative to the y -axis.
- **Back Angle at Teats Level:** Similar to the previous measurement, but using a slice at $x = x_{M_3}$ with offsets of ± 3.5 and 15 cm.

All these measurements and protocols were based on previous work conducted within the company or defined in collaboration with the prototyping team to assess saddle fitting. A database of measurements taken on scans already exists, although it was collected using FARO scans rather than iPhone scans. Nevertheless, these data can still serve as a reference to evaluate some of the measurements and the measurement process.

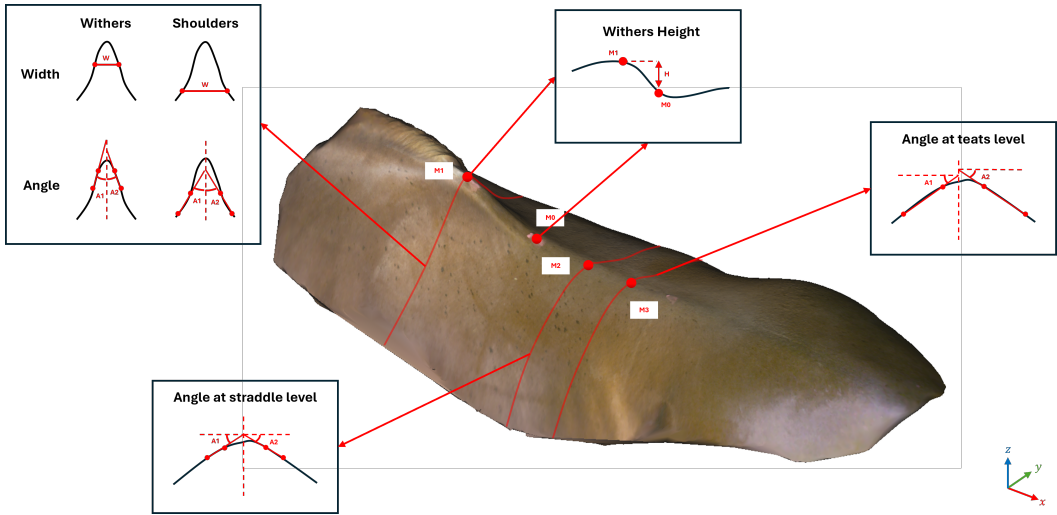


Figure 17. Visualization of extracted landmark points and measurements on a horse scan.

5 RESULTS & DISCUSSION

5.1 3D Scan Enhancement

5.1.1 Brute and Real-Time Implementation Comparative

This section presents an example of the quantization approach used to reduce the computational time of certain filters. Data from the *Courbette* dataset are used for this demonstration. An example of a low-resolution depth map and its corresponding RGB image is shown in Fig. 18.



Figure 18. Example of RGB-D data from the *Courbette* dataset: (a) depth map (320×160) and (b) RGB guidance image (1280×720).

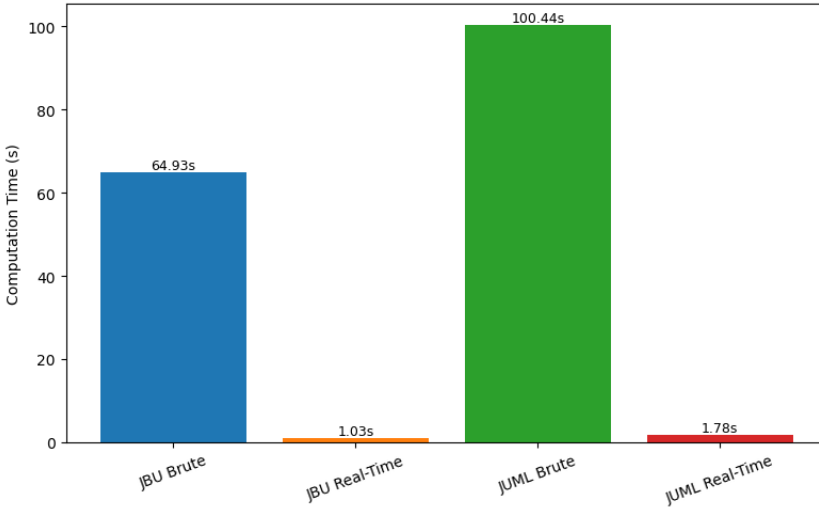


Figure 19. Comparison of computing times between the brute-force implementation and the real-time implementation using quantization for Joint Bilateral Upsampling (JBU) and Joint Unified Multi-Lateral (JUML) filters.

The use of quantization for both the Joint Bilateral Upsampling (JBU) and the Joint Unified Multi-Lateral (JUML) filters results in a significant reduction in computational time. As shown in Fig. 19, the runtime of JBU decreases from approximately 65 s to 1 s ($\sim 65\times$ speed-up), while JUML is reduced from 100 s to 1.78 s ($\sim 56\times$ speed-up). The slightly longer runtime for JUML is

expected, as it extends JBU by incorporating a confidence map derived from the depth gradient, adding computation while providing more information for filtering.

Despite this acceleration, processing still requires slightly over one second due to the relatively high RGB resolution (1280×720), depth resolution (320×160), and the window sizes used in parts of the algorithm. Further improvements could be achieved through additional downsampling, as suggested by Garcia et al. [25]. In the current real-time implementation, a Gaussian blur is used for the spatial filter, as recommended in the original paper. In practice, a recursive Gaussian filter is applied, which has a constant computational complexity ($O(1)$), though optimized implementations in some libraries may be faster depending on the resolution and kernel size. This represents an additional consideration for potential improvements to the real-time filters.

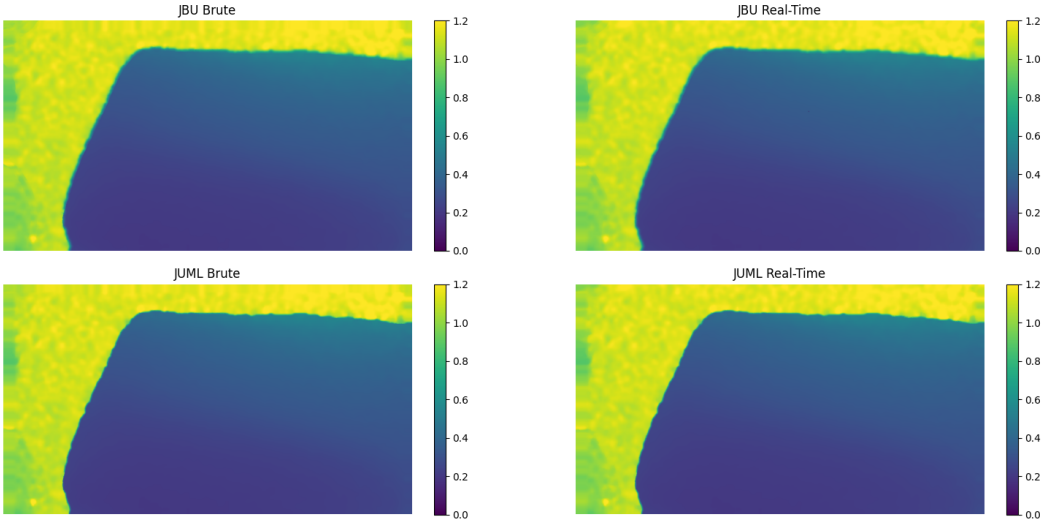


Figure 20. Visual comparison of the brute-force and real-time implementations of Joint Bilateral Upsampling and Joint Unified Multi-Lateral filters on depth maps [units: m].

Fig. 20 shows the depth maps obtained using both the brute-force and real-time implementations. Overall, the real-time quantized filters produce results that are visually similar to the brute-force versions. No major differences are observed, whether at edges where depth transitions are more pronounced and influenced by the number of bins, or on the target surface, where no over-smoothing or stair-step artifacts are visible, indicating that the interpolation in the quantized approach is accurate.

A quantitative analysis can be conducted to confirm the visual similarity. Structural Similarity Index (SSIM) and Mean Absolute Error (MAE) can be used to assess both the structural fidelity and precision differences between the two filtered depth maps. Further analysis can focus on the region of interest, i.e., the depth values on the target, by filtering out background values. This is achieved by retaining only depth values below 0.45 meters. The results of this analysis are summarized in Tab. 2.

Table 2. Comparison of structural similarity and precision between the brute-force and real-time implementations for both filters [units: mm].

Filter	$[D_{min}^f, D_{max}^f]$		$[D_{min}^f, 450]$	
	SSIM $\uparrow$	MAE $\downarrow$	SSIM $\uparrow$	MAE $\downarrow$
JBU	0.998	1.117	1.000	0.123
JUML	0.998	1.317	0.999	0.072

The metrics indicate that the structural similarity is nearly perfect, with SSIM values approaching 1. The minor dissimilarities are likely at the edges, where the difference in depth values between the object and distant background is larger, and some areas may experience slight over-smoothing. The MAE is just above 1 mm for both filters; however, this value can be influenced by the noisy background, making it more meaningful to evaluate the area of interest. Within this region, the error is approximately one-tenth of a millimetre, which is highly satisfactory and represents a negligible difference considering the significant improvement in computation time.

For trimmed regions, SSIM is less informative, as it is an edge-aware metric. In these smooth areas, there are few meaningful edges; the withers are the only potential edges, but their contribution to SSIM is minimal. This suggests that using an RGB guidance image would primarily affect edges, and for smooth horse targets, a classical bilateral filter may be sufficient. This point will be explored further in the following sections.

Conclusion Overall, these results demonstrate that the real-time quantized filters maintain high fidelity while drastically reducing computation time, confirming their suitability for practical applications. While JUML incurs slightly higher computational cost, the additional confidence information may improve robustness in complex regions. Future work could explore further acceleration strategies, including more aggressive downsampling, optimized window sizes, or parallel processing, to enhance real-time performance without compromising depth map quality. For the subsequent experiments, the current real-time implementation will be used as a baseline. In terms of computing time, the bilateral filter implementation is comparable to OpenCV’s optimized version, and the guided filter, being based on a linear formulation, executes rapidly at approximately 0.13 s.

5.1.2 Frustum Model Dataset

In this section, the results from the evaluation of the filters on the frustum are presented to assess their contribution to improving scan accuracy. The model is highly specific and is used to validate both the overall smoothing for enhanced precision and the accuracy of the key measurements.

The first evaluation is conducted by examining the overall precision of the model (Fig. 21) and identifying areas of lower accuracy, using a colour-coded representation of precision on the point cloud.

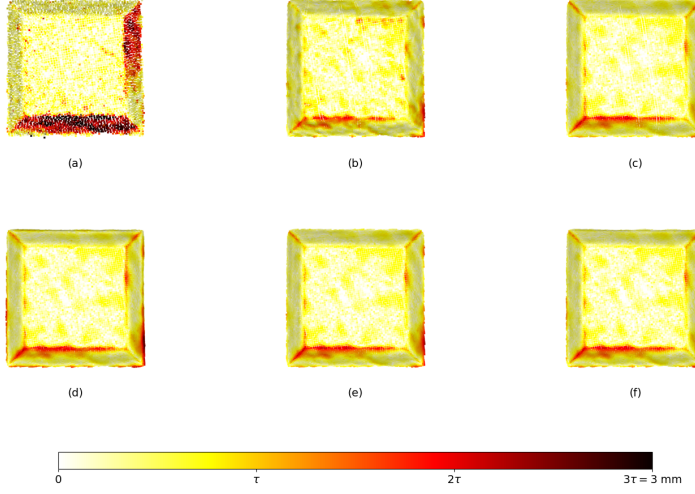


Figure 21. Colour-coded representation of precision on frustum point clouds with threshold $\tau = 1$ mm: **(a)** current surfel-based scan, **(b)** unfiltered TSDF, **(c)** TSDF after bilateral filter, **(d)** TSDF after joint bilateral upsampling, **(e)** TSDF after unified multi-lateral filter and **(f)** TSDF after guided filter.

This visualization provides an initial insight into the influence and quality of the reconstruction techniques and filters used. The current scan (Fig. 21a) appears to be the least precise, with large areas exhibiting errors of up to 3 mm or more. In contrast, the TSDF approach generally achieves higher accuracy, even in its unfiltered form (Fig. 21b). The lower precision of the surfel-based approach used in the current scan can be attributed to the low resolution of the depth data, which is more susceptible to noise—particularly on planar surfaces—and less effectively mitigated by surfels compared to TSDF.

Another notable observation arises from comparing the unfiltered TSDF with its filtered versions (Fig. 21c–f). Filtering introduces slight imprecision along the side edges. While this might initially seem like a drawback, it may instead reflect the fact that the model was handcrafted and compared against a CAD ground truth model with watertight precision. Consequently, the observed discrepancies may indicate the filters’ capacity to preserve edges while enhancing overall precision.

A quantitative analysis further supports these observations, beginning with the precision metrics presented in Fig. 22 and Tab. 3.

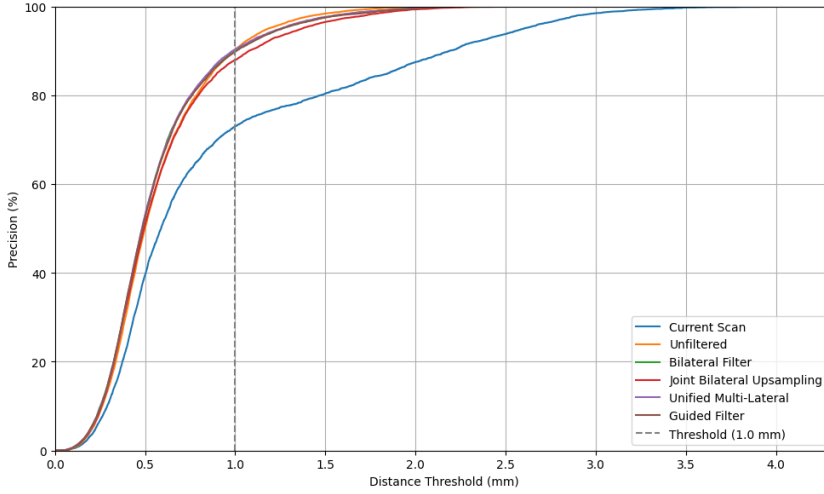


Figure 22. Precision plot for frustum point clouds

Table 3. Precision metrics across different filtering methods for Frustum 3D scans [unit: mm].

Method	Max ↓	STD ↓	RMSE ↓	MAE ↓	P(t) ↑
Current Scan	4.318	0.754	1.177	0.903	73.007
Unfiltered	2.334	0.309	0.652	0.575	90.257
Bilateral Filter	2.349	0.332	0.659	0.569	89.838
Joint Bilateral Upsampling	2.681	0.366	0.698	0.595	87.902
Unified Multi-Lateral	2.326	0.329	0.655	0.567	90.336
Guided Filter	2.372	0.336	0.663	0.571	89.806

This complementary data indicates that the proposed methods can increase the precision of the extracted scans by up to a factor of two compared to the current approach. The mean distance from the reference is significantly reduced—from 0.9 mm to approximately 0.5 mm—corresponding to a precision improvement of up to 13%, with the majority of points in the TSDF approach falling within one millimetre.

For scans reconstructed using TSDF, precision generally remains within a similar range (Fig. 22), with a slight exception observed for the Joint Bilateral Upsampling (JBU) method. The comparatively lower precision for JBU (Tab. 3) can be attributed to the model’s texture, which includes text elements. These intensity variations are not correlated with depth changes and may produce texture-copying artefacts, a known limitation of this filter as discussed in Sec. 2. In contrast, the Bilateral Filter, which relies solely on depth information, and the Unified Multi-Lateral and Guided Filter, which are designed to address this limitation, are not affected and thus produce more accurate results. Fig. 23 highlights some excavated areas on the sides for JBU, while also illustrating the influence of the other filters.

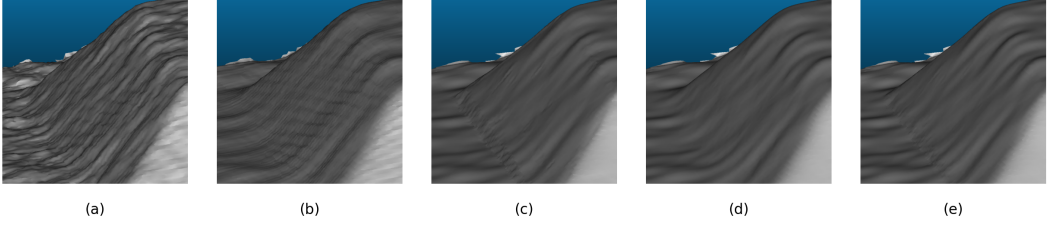


Figure 23. Zoom on TSDF scans: (a) unfiltered, (b) after bilateral filter, (c) after joint bilateral upsampling, (d) after unified multi-lateral filter and (e) after guided filter.

Based on the quantitative data, the unfiltered scan is considered precise, exhibiting lower variance than the other methods, as indicated by the standard deviation (STD) and root mean square error (RMSE) metrics. A more detailed analysis can be conducted by focusing on planar surfaces, slightly eroding the data to minimize the influence of handcrafted geometry and better assess the smoothing capabilities of the filters. Tab. 4 and Tab. 5 summarize the precision metrics for the upper surface and the average of the side surfaces, respectively.

Table 4. Precision of planar surfaces — upper surface [unit: mm].

Method	Max ↓	STD ↓	RMSE ↓	MAE ↓
Current Scan	2.058	0.251	0.587	0.531
Unfiltered	1.682	0.237	0.564	0.512
Bilateral Filter	1.085	0.185	0.494	0.458
Joint Bilateral Upsampling	1.018	0.178	0.480	0.445
Unified Multi-Lateral	1.152	0.189	0.496	0.459
Guided Filter	1.086	0.191	0.499	0.461

This table indicates that the unfiltered scan is less accurate than the filtered versions, which smooth the surface and yield a more precise reconstruction. The effect of smoothing is evident in the deviation from the mean values: the standard deviation (STD) is lower for the filtered scans compared to the unfiltered scan. The current scan remains the least accurate, with distances to the ground truth roughly twice as large as those observed for the filtered and TSDF scans.

Interestingly, the JBU scan exhibits the highest precision on the upper surface, despite being the least accurate among the filtered scans in terms of overall precision. This suggests that texture-copying artefacts may have a greater impact on the side planes, where the depth data is less reliable.

Table 5. Precision of planar surfaces — combined sides (mean of sides 1 to 4) [unit: mm].

Method	Max ↓	STD ↓	RMSE ↓	MAE ↓
Current Scan	2.673	0.432	1.264	1.183
Unfiltered	1.457	0.252	0.599	0.543
Bilateral Filter	1.341	0.242	0.579	0.524
Joint Bilateral Upsampling	1.527	0.262	0.610	0.548
Unified Multi-Lateral	1.300	0.239	0.581	0.528
Guided Filter	1.380	0.248	0.593	0.537

The table confirms the previous observation, showing that the JBU method is less accurate on the side edges and highlighting the potential benefit of mitigating confidence in the depth map, as implemented in UML. Additionally, the currently used approach appears to lose precision as the distance to the phone increases, with metrics roughly twice as large as those of the unfiltered scans, whereas the values for the upper plane were closer.

The final aspect to be evaluated is the accuracy of the measurements, as GDSR employs techniques to smooth surfaces while preserving edges. The results are summarized in Tab. 6.

Table 6. Mean errors for edge groups by filtering method [unit: mm].

Method	Upper Edges	Lower Edges	Side Edges
Current Scan	3.651	1.760	1.559
Unfiltered	2.198	0.670	0.393
Bilateral Filter	2.338	0.403	0.864
Joint Bilateral Upsampling	2.600	0.403	1.305
Unified Multi-Lateral	2.013	0.403	1.205
Guided Filter	2.175	0.403	1.061

The first observation is that the TSDF approach also yields more precise results, with errors up to 1.5 mm greater when using the surfel-based method. The effect of GDSR is particularly pronounced on the lower edges, which are farther from the scanning device. By leveraging either depth changes or intensity variations—especially between the supporting table and the model—precision is improved, reducing the average error to approximately 0.4 mm for a 16 cm edge, compared to 0.67 mm using only TSDF and nearly 2 mm with the current scan.

Conclusion Overall, this evaluation demonstrates that the proposed TSDF-based reconstruction produces more accurate scans than the current method, roughly halving the errors in precision. The impact of the filters is also evident in the smoothing of planar surfaces and the enhancement of edges in the reconstructed model.

5.1.3 Voxels Size Influence & TSDF Computational Time

For the frustum model, the voxel size and computational time of the TSDF reconstruction were not prioritized. The main objective was to evaluate the pipeline process and ensure that the reconstructed model was accurate. The voxel size was chosen empirically to capture the relevant geometric details. However, for the subsequent datasets—the *Courbette* and horse scans—the voxel size must be optimized to balance precision and computational efficiency for real-time applications.

To assess the influence of voxel size on both precision and computational time, the *Courbette* dataset is used, as it is more suitable for evaluation than horse scans for the reasons detailed in Sec. 4. The first evaluation examines the computational time for nine voxel sizes ranging from 1 mm to 5 mm (Fig. 24 and Tab. 7).

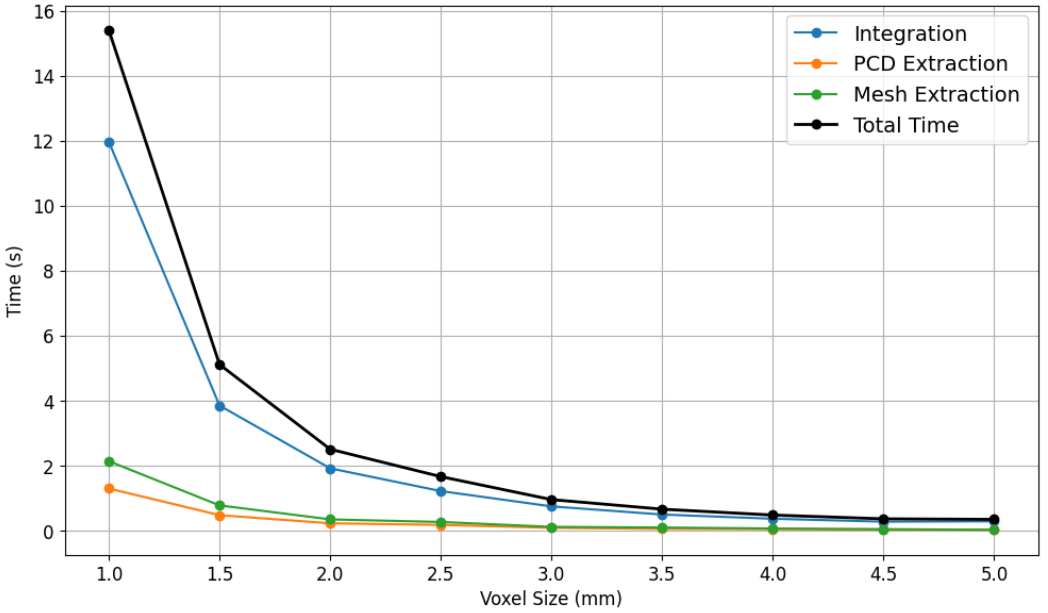


Figure 24. Impact of voxel size on computational time for TSDF RGB-D frame integration, point cloud generation, and mesh extraction.

The computational time can be divided into two main steps: the integration of the TSDF volume and the extraction of the model (although a mesh is used in the pipeline, a point cloud can also be extracted for evaluation purposes). As shown in Fig. 24, the integration step is the most computationally expensive, as it processes all frames and projects the voxel centres onto the plane. This step depends on both the number of frames and the voxel size, making the choice of an appropriate voxel size critical for computational efficiency. In contrast, the extraction step depends only on the voxel size and its interaction with the marching cubes algorithm for meshes. Overall, the computational time decreases significantly as the voxel size increases, exhibiting a converging-like behaviour.

Table 7. Timing breakdown by voxel size

Voxel Size (mm)	Integration (s)	Extract PCD (s)	Extract Mesh (s)	Total (s)
1.0	11.91	1.26	2.20	15.371
1.5	4.07	0.45	0.75	5.276
2.0	1.88	0.25	0.35	2.485
2.5	1.17	0.14	0.20	1.516
3.0	0.77	0.11	0.13	1.005
3.5	0.53	0.08	0.08	0.688
4.0	0.36	0.07	0.06	0.496
4.5	0.30	0.05	0.06	0.415
5.0	0.25	0.04	0.03	0.314

The quantitative data show that computational time decreases significantly as the voxel size increases. While a voxel size of 1 mm is impractical due to the resulting computation time of approximately 15 seconds, voxel sizes of 3 to 3.5 mm appear suitable for real-time applications. The next step is to verify that precision is not compromised when using voxels of this size.

Further analysis of computational time can be conducted for mesh extraction, considering that the complexity of the marching cubes algorithm is $O(n^3)$. Doubling the voxel size reduces the number of voxels by a factor of $2^3 = 8$, implying a theoretical reduction in extraction time by the same factor. This trend is approximately observed in practice; for example, the extraction time for voxel sizes of 1 mm and 2 mm shows a ratio of $\frac{2.20}{0.35} \approx 6.3$. Thus, the computational time for mesh extraction can be considered nearly cubic, and this behaviour extends, to some extent, to the overall reconstruction process.

Having established that the proposed workflow can operate in real time—achieving reconstruction in under a second—it is necessary to ensure that scan quality is not degraded, for example, through excessive smoothing with larger voxel sizes. An initial assessment can be performed by visualizing the precision on the mesh in Fig. 25.

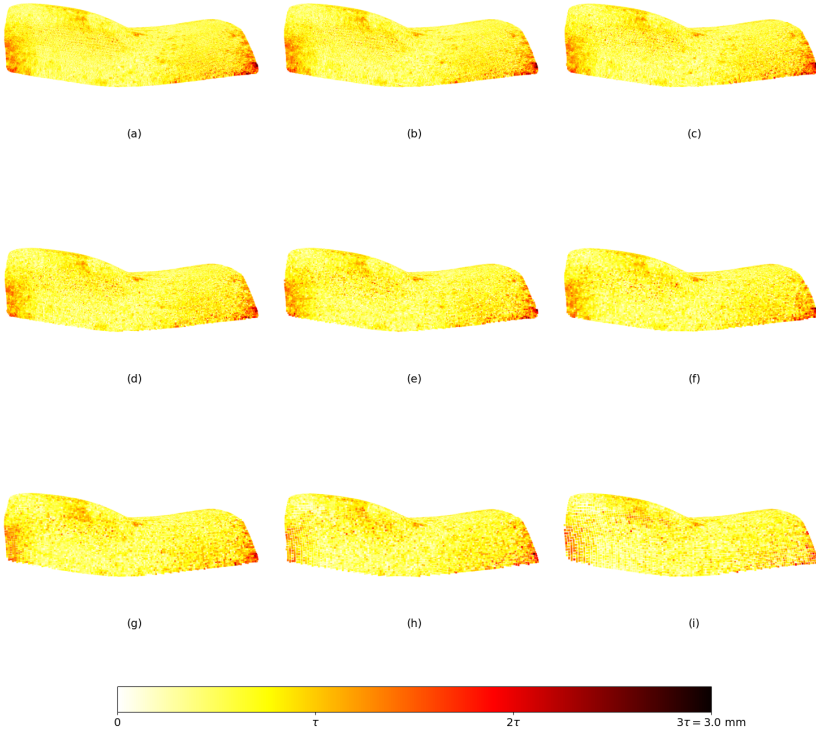


Figure 25. Colour-coded representation of precision on *Courbette* meshes with threshold $\tau = 1$ mm at varying TSDF voxel size: (a) 1 mm, (b) 1.5 mm, (c) 2 mm, (d) 2.5 mm, (e) 3 mm, (f) 3.5 mm, (g) 4 mm, (h) 4.5 mm and (i) 5 mm

The colour-coded representations show little difference across voxel sizes, indicating that precision does not appear to be noticeably degraded at first glance. For the coarser meshes with voxel sizes of 4, 4.5, and 5 mm (Fig. 25g–j), potential over-smoothing could have been expected around the withers, which are less planar, yet the precision remains consistent. The voxel resolution is sufficient to preserve the overall shape. These observations are quantitatively confirmed in Tab. 8.

Table 8. Precision metrics across different voxel sizes for *Courbette* 3D scans [unit: mm].

Voxel Size (mm)	Max ↓	STD ↓	RMSE ↓	MAE ↓	P(t) ↑
1.0	2.622	0.264	0.653	0.597	92.642
1.5	2.597	0.263	0.653	0.597	92.593
2.0	2.640	0.264	0.653	0.597	92.792
2.5	2.612	0.267	0.651	0.593	92.566
3.0	2.323	0.260	0.643	0.589	93.004
3.5	2.163	0.264	0.651	0.595	92.716
4.0	1.849	0.262	0.644	0.588	93.215
4.5	2.505	0.265	0.652	0.596	92.588
5.0	2.501	0.268	0.654	0.596	92.270

Overall, the precision metrics remain within a similar range across all voxel sizes. However, a parabolic trend is observed, with the highest precision achieved for voxel sizes of 3 mm or 4 mm. When the voxel size is too small, the TSDF volume is not always the most precise, as it may be more sensitive to noise in the low-resolution depth data. This highlights the importance of enhancing the depth data through filtering. Conversely, coarser voxel sizes can over-smooth certain areas, leading to a slight loss of precision. Therefore, voxel sizes of 3 mm or 4 mm provide the best balance between noise smoothing and preservation of geometric detail. Additionally, at these sizes, model computation takes less than one second, making them suitable for real-time applications.

Conclusion The choice of voxel size depends on the intended purpose. For evaluation in the subsequent sections, preserving texture details is important, and larger voxels could cause coloured landmarks to be averaged over a wider area. Thus, a voxel size of 3 mm is adequate for maintaining texture precision while ensuring fast computation. For overall implementation, a voxel size of 4 mm can be used, as precision is retained while computation time is significantly reduced. This has several advantages: it supports real-time implementation, reduces storage requirements—the output models are approximately 20 times smaller compared to 1 mm voxels and even more compact than those from the currently used process—and allows faster post-processing due to a reduced number of mesh vertices (nearly 30 times fewer between 1 mm and 4 mm voxel sizes). Moreover, the mesh appears smoother, being less affected by small noise artifacts present in finer meshes.

5.1.4 *Courbette* Dataset

In this section, all filters are considered, as it remains necessary to assess the usefulness of the RGB image. This is particularly relevant for GDSR, which is effective in edge regions where depth changes correspond to RGB intensity variations. For the horse’s back, which is generally smooth and scanned from above, the smoothing aspect of the filters is more critical than edge preservation. However, the withers can exhibit steep angles, where the RGB image may help improve precision in these regions.

As with the previous dataset, precision is first evaluated using a colour-coded representation, shown in Fig. 26.

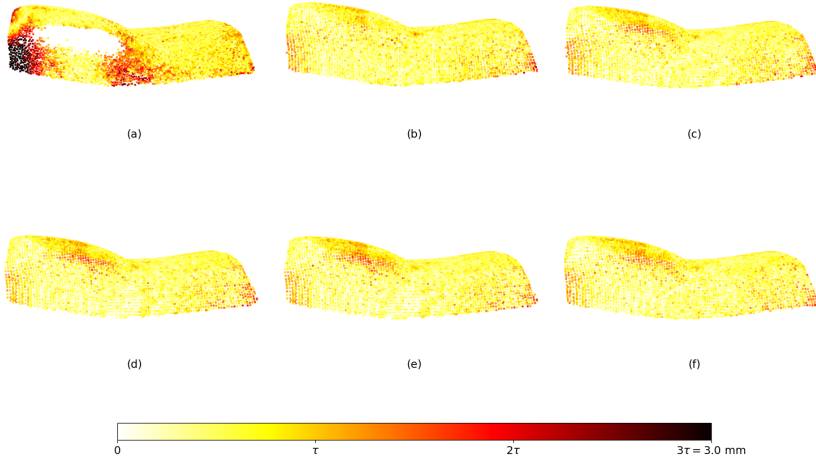


Figure 26. Colour-coded representation of precision on *Courbette* point clouds with threshold $\tau = 1$ mm: **(a)** current surfel-based scan, **(b)** unfiltered TSDF, **(c)** TSDF after bilateral filter, **(d)** TSDF after joint bilateral upsampling, **(e)** TSDF after unified multi-lateral filter and **(f)** TSDF after guided filter.

For this dataset, even though the target is larger, the precision threshold has been kept the same, ensuring that scan accuracy remains high. The main areas of imprecision are observed at both ends of the scans. In the case of the TSDF approach, this is explained by the voxel values being updated based on the number of observations, which effectively provides a confidence metric. Regions at the extremities are observed in fewer frames, resulting in higher noise levels.

In the proposed pipeline, no additional features are used, representing a limitation of the workflow that should be addressed in future improvements. For the surfel-based approach, multiple steps are taken to select which depth values to integrate, including weighting based on the angle of incidence. Similar considerations explain the observed imprecisions in this method.

Additionally, some regions of the model are not reconstructed. These correspond to areas with steep angles, where the pipeline determined that the depth values were insufficiently reliable. While these missing regions may not be critical for the analysis, gaps in regions of interest could degrade the mesh quality, which is central to this study. Therefore, a precision evaluation is also performed using the mesh vertices (Fig. 27).

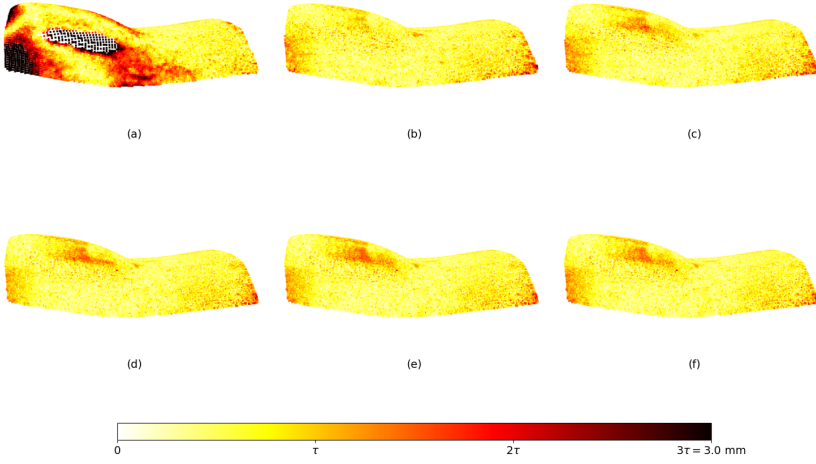


Figure 27. Colour-coded representation of precision on *Courbette* mesh vertices with threshold $\tau = 1$ mm: (a) current surfel-based scan, (b) unfiltered TSDF, (c) TSDF after bilateral filter, (d) TSDF after joint bilateral upsampling, (e) TSDF after unified multi-lateral filter and (f) TSDF after guided filter.

Examining the figure, the TSDF approach demonstrates similar precision and limitations between the extracted point cloud and the mesh generated using the marching cubes algorithm. This confirms both the consistency of TSDF and the suitability of the marching cubes method. For the current scan (Fig. 27a), the missing areas from the point cloud are reconstructed, but with significant imprecision—up to 4.5 mm or more. This aligns with the known limitations of the current approach, where certain regions may be absent, and Poisson reconstruction struggles with coarse or incomplete point clouds.

Visually, the inaccuracies in this scan appear more pronounced in the mesh, likely due to the higher point density. Regardless, the comparison between the point cloud and mesh for both the current and proposed approaches highlights a loss of precision during reconstruction. This observation is further supported by additional qualitative data (Fig. 28) and quantitative metrics (Tab. 9 and Tab. 10).

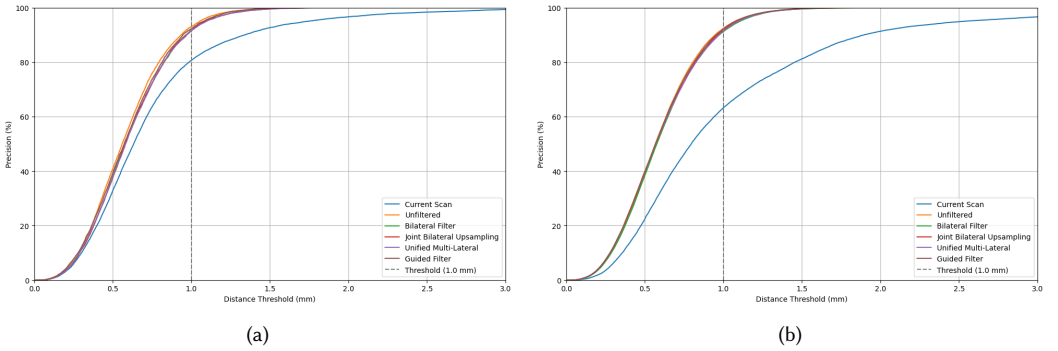


Figure 28. Precision plot for *Courbette* (a) point clouds, (b) mesh vertices

Fig. 28 illustrates this decline in precision, showing that the TSDF approach maintains consistent accuracy, while the precision of the current scan decreases by nearly 20% at $\tau = 1$ mm. This further highlights the limitations of the current workflow and the advantages of the proposed method. In the current surfel-based approach, the mesh is reconstructed from a point cloud that is already imprecise and inconsistent. Poisson reconstruction applied to such a cloud relies on an indicator function and mesh extraction algorithm that are not fully reliable, resulting in further loss of precision.

In contrast, the TSDF approach generates the mesh directly from a detailed volumetric representation, avoiding the intermediate point cloud that may contain holes and inconsistent normals. The observed consistency between the point cloud and mesh in TSDF arises because both are derived from the same volumetric source, serving as two complementary representations rather than successive processing steps.

Table 9. Precision metrics across different filtering methods for *Courbette* 3D point clouds [unit: mm].

Method	Max ↓	STD ↓	RMSE ↓	MAE ↓	P(t) ↑
Current Scan	4.527	0.501	0.904	0.752	80.689
Unfiltered	2.323	0.260	0.643	0.589	93.004
Bilateral Filter	1.986	0.269	0.666	0.609	91.658
Joint Bilateral Upsampling	2.274	0.270	0.668	0.611	91.643
Unified Multi-Lateral	2.155	0.271	0.669	0.611	91.449
Guided Filter	1.958	0.265	0.654	0.598	92.269

The current scan using surfel-based reconstruction performed the worst across all metrics. It exhibited the highest maximum deviation (4.527 mm) and largest standard deviation (0.501 mm), indicating more irregular surface estimates. Additionally, it had the highest RMSE (0.904 mm) and the lowest $P(t)$ (80.689%), reflecting relatively poor reconstruction quality compared to the volumetric methods. This clearly demonstrates that TSDF-based methods significantly outperform the surfel-based approach in high-precision 3D reconstruction for this dataset. The volumetric integration inherent in TSDF-based methods helps to smooth noise and recover more accurate surfaces, as evidenced by the observed error reductions.

Focusing on the TSDF results, both unfiltered and filtered outputs achieve similar precision, although the unfiltered scan appears slightly more accurate. Examination of the colour-coded point clouds reveals minor over-smoothing around the horse’s withers, attributable to the chosen filter parameters. While these parameters could be adjusted to reduce over-smoothing, maintaining them as dynamically defined ensures consistency across scans. Furthermore, the precision threshold is set at 1 mm, making the evaluation more sensitive; in real-world scanning of horses, these differences may be less pronounced. Although the bilateral and guided filters appear most effective, further analysis using the mesh is necessary. Additionally, precision is not the sole metric to assess the influence of filtering techniques—surface roughness can also provide important insight into the impact of these steps.

Table 10. Precision metrics across different filtering methods for *Courbette* 3D mesh vertices [unit: mm].

Method	Max ↓	STD ↓	RMSE ↓	MAE ↓	P(t) ↑
Current Scan	8.195	0.805	1.307	1.030	63.289
Unfiltered	2.867	0.266	0.655	0.598	92.376
Bilateral Filter	2.420	0.273	0.668	0.610	91.131
Joint Bilateral Upsampling	2.609	0.272	0.660	0.601	91.696
Unified Multi-Lateral	2.472	0.274	0.663	0.604	91.357
Guided Filter	2.325	0.267	0.655	0.598	92.218

The mesh from the current reconstruction process still appears to be the least precise. Quantitative data confirm the loss of precision due to the Poisson reconstruction step: the standard deviation increased from 0.5 mm to 0.8 mm, indicating a more irregular surface, while the maximum error nearly doubled, particularly around the withers and in regions missing from the surfel point cloud. More importantly, overall precision dropped to 63%.

For the TSDF results, the unfiltered mesh is the most precise. Among the filtered meshes, the guided filter appears to be the most effective, whereas the bilateral filter shows slightly lower precision. Inspection of the colour-coded mesh suggests that the main errors occur near the borders, likely due to the combination of filtering on low-resolution depth data—which reduces edge fidelity compared to guidance-based filters—and fewer observations in these regions. Overall, the filtering results are similar and do not show a clear preference for any specific filter on the *Courbette* dataset. The effect of filtering on surface smoothness can now be evaluated through roughness analysis (Tab. 11) to assess their influence in greater detail.

Table 11. Roughness of each *Courbette* scan [unit: mm].

Method	Roughness ↓
Current Scan	0.14
Unfiltered	0.12
Bilateral Filter	0.07
Joint Bilateral Upsampling	0.05
Unified Multi-Lateral	0.05
Guided Filter	0.05

Roughness is defined as the average distance of a vertex to the local plane fitted from its neighbouring vertices. Since the meshes are inherently smooth, with few small-scale details or rapid variations between neighbouring vertices, the roughness values are expected to be small. While increasing the neighbourhood window for roughness computation could provide a less localized metric, this approach may be overly influenced by curvature rather than the noise or imprecision that we aim to assess. Additionally, oscillatory behaviours that might be captured with larger windows are more relevant to dynamic scenes than the static models evaluated here. For consistency, all meshes are truncated to the same size when computing roughness.

The results show that the current scan and the unfiltered TSDF mesh have roughly similar

roughness values, with the TSDF mesh demonstrating a slight improvement. This improvement likely arises from the volumetric representation and voxel-based updates, which provide a more regular vertex positioning. For the filtered meshes, roughness decreases to 0.07 mm for the bilateral filter and 0.05 mm for the other filters. This indicates that filtering effectively smooths the surface, as also observed visually for the frustum, while maintaining high precision, highlighting the importance of these filtering steps.

The difference in roughness between the bilateral filter and the other filters can be attributed to the applied parameters and resolution. The bilateral filter is applied at a lower resolution, providing fewer depth values relative to the voxel size and fewer updates to the signed distance field. In contrast, the other filters use upsampled depth maps with dynamically adjusted parameters, ensuring consistent smoothing. The equal roughness values of 0.05 mm among the upsampled filters indicate that the dynamic parameter adjustment achieves comparable smoothing across these methods.

For this scan, a final evaluation is performed by examining the errors in the measurements extracted from the mesh. The results are summarized in Tab. 12.

Table 12. Errors in key measurements for each filtering method [unit: cm].

Method	1a	1b	2a	2b	3	4	Mean	Max
Current Scan	–	0.86	0.01	0.42	1.2	0.08	–	–
Unfiltered	0.02	0.45	0.72	0.72	0.17	0.73	0.47	0.72
Bilateral Filter	0.15	0.75	0.6	0.59	0.3	0.37	0.46	0.75
Joint Bilateral Upsampling	0.01	0.68	0.52	0.5	0.33	0.45	0.42	0.68
Unified Multi-Lateral	0.02	0.66	0.56	0.44	0.34	0.33	0.39	0.66
Guided Filter	0.01	0.81	0.59	0.49	0.25	0.26	0.4	0.81

The first notable observation is that holes and the resulting imprecision in the mesh affect the measurements. For instance, measurement 1a is not considered usable for evaluation and is treated as an outlier. Variations in the errors of the current scan provide insights into the reconstruction method itself. The surfel-based approach appears to find a trade-off between smoothness, geometric integrity, and precision, resulting in irregularities in the measurements with errors ranging from 0.01 to 1.2 cm. In comparison, the errors from TSDF reconstruction are relatively consistent and closely clustered around the mean.

This suggests that either a tolerance can be applied based on the error introduced by TSDF reconstruction, or that some errors originate from the measurement extraction process itself. Overall, the maximum errors remain below 1 cm, and the mean errors range from 0.39 to 0.47 cm, indicating that TSDF reconstruction is precise and properly scaled. Consequently, the reconstructed meshes are suitable for extracting measurements and can support applications such as assisting sales representatives. The proposed reconstruction approach yields more consistent results.

Examining the effect of filtering, it is observed that filters generally reduce measurement errors, though they can occasionally increase them. For example, the bilateral filter slightly increases the errors for measurements 1a and 1b, which correspond to the withers region. This minor increase may be attributed to slight over-smoothing. However, the errors remain small and close to the average, suggesting that they may also stem from the measurement extraction process and do not invalidate the use of the filter; rather, this effect should be noted as a point of consideration.

To further develop the comparison, different scanning trajectories are also examined (Fig. 29) in order to evaluate whether they can mitigate the limitations of the current approach and to assess if the proposed method consistently outperforms it.

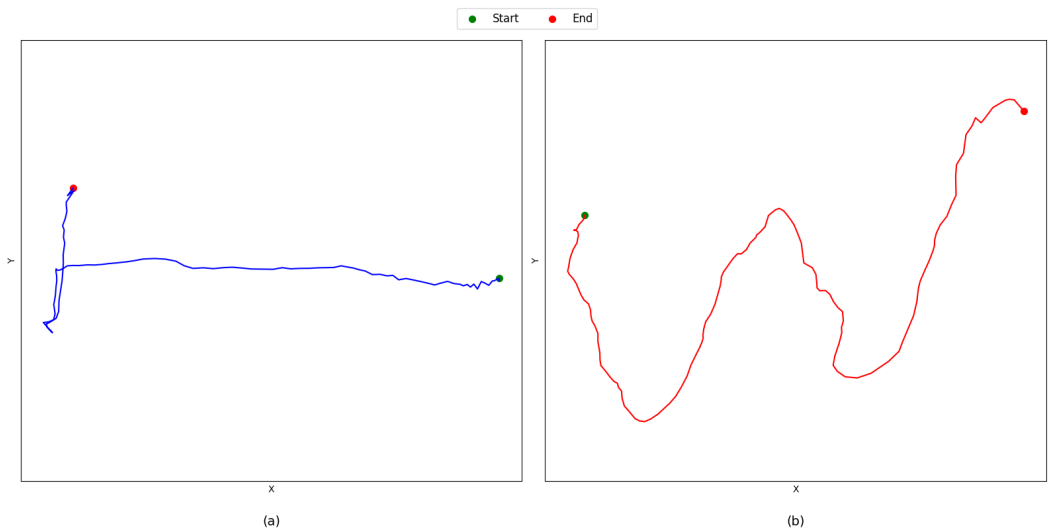


Figure 29. Scanning trajectories used for *Courbette* dataset

Different scanning paths are considered. The first path, used in all previous analyses, is primarily linear with additional focus on the withers at the end, and consists of 102 frames. This path is relatively simple and conventional, designed to ensure a reliable scan. The second path follows a sinusoidal-like trajectory, aiming to cover the object more comprehensively and increase the number of observations per area. This path uses 139 frames.

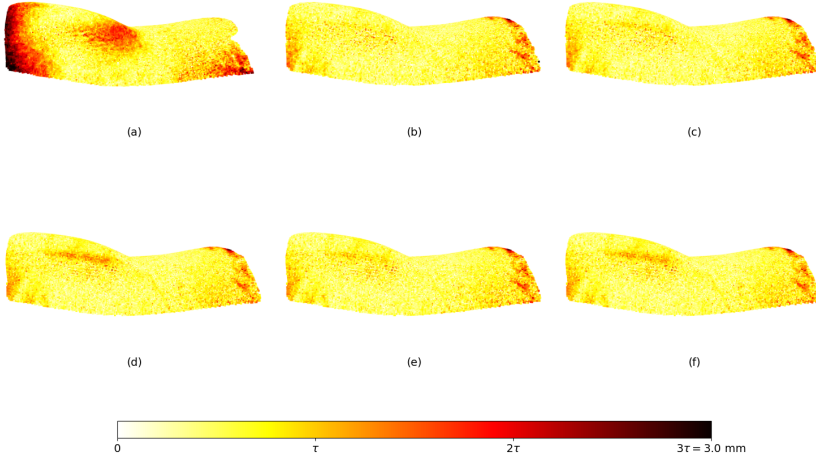


Figure 30. Colour-coded representation of precision using a new trajectory on *Courbette* mesh vertices with threshold $\tau = 1$ mm: (a) current surfel-based scan, (b) unfiltered TSDF, (c) TSDF after bilateral filter, (d) TSDF after joint bilateral upsampling, (e) TSDF after unified multi-lateral filter and (f) TSDF after guided filter.

The colour-coded precision maps on the meshes reveal similar overall trends: the current approach remains less precise than the proposed method, which primarily exhibits minor imprecision at the extremities of the scan, even with the new trajectory. However, a notable difference for the current scan is observed: the side of the withers is now well reconstructed, with a dense vertex distribution, whereas it was previously sparse and imprecise. This indicates that the scanning trajectory also influences reconstruction quality, although the proposed approach still outperforms the current method. These observations can be further supported through plots.

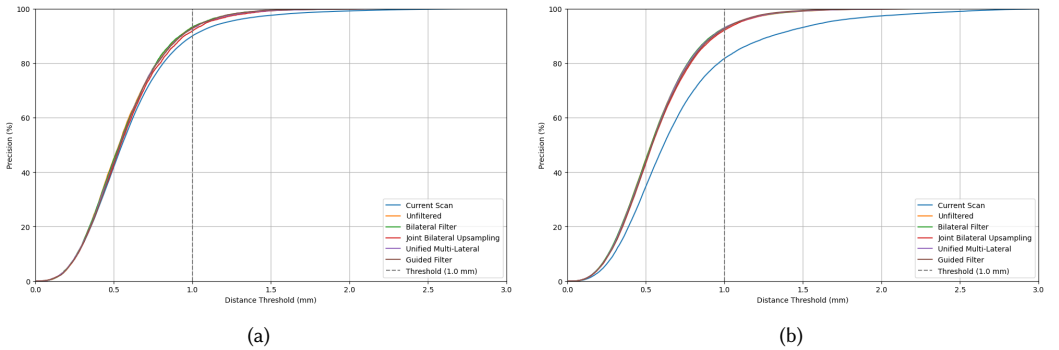


Figure 31. Precision plot with new trajectory for *Courbette* (a) point cloud, (b) mesh vertices

The plot for the point clouds shows that using an improved trajectory that covers a broader area reduces the performance gap between the proposed method and the current approach. A similar trend is observed for the meshes: the precision of the current method increases from 63% with the initial trajectory to approximately 80% with the improved trajectory, whereas the TSDF-based

reconstruction consistently achieves over 90% precision. This reinforces the earlier observation that the main source of error arises from the Poisson reconstruction, which is not suitable for generating a mesh from a surfel cloud.

In contrast, the TSDF approach remains consistent and delivers precise meshes regardless of the scanning trajectory. This is particularly important because the trajectory depends on the operator, such as sales representatives, over the scanning conditions or personal preferences for movement can not be controlled. Even if a recommended base trajectory can be provided, variations and outliers can occur, making the TSDF approach more robust and less dependent on the scanning path. Additionally, the linear trajectory may be less prone to freezing issues compared to the sinusoidal trajectory, based on experimental observations.

Conclusion Based on the qualitative and quantitative analyses of the *Courbette* dataset, several observations and recommendations can be made:

- **Reconstruction:** The main outcome is that the proposed TSDF-based approach provides more precise and consistent results due to the direct extraction of the mesh from the volumetric representation. The primary limitation of the current surfel-based approach is that, among other factors, the low resolution of the depth data and the integration method result in a surfel cloud that is neither complete nor precise enough to serve as input for Poisson reconstruction. Consequently, the output mesh is highly imprecise. The TSDF approach is also more robust, being consistent and independent of the scanning trajectory and the number of frames used. While the differences in precision may only reach millimetres, they can be significant for certain regions. More importantly, the proposed approach ensures consistency in the output mesh, which is crucial for further analysis and downstream applications.
- **Filters:** As expected, the influence of the RGB guidance image is limited for a smooth target such as this. Nevertheless, filtering is important to smooth the surface, which is beneficial for visualization and measurements, while maintaining the precision of the TSDF-based mesh. Among the filters evaluated, two stand out: the Bilateral Filter (BF) and the Guided Filter (GF). The BF is sufficient for this application, performing well for all critical areas and already being used during ICP to reduce noise in point clouds, avoiding redundant filtering. Additionally, the BF is computationally efficient due to its low-resolution implementation, which is advantageous for real-time applications. The GF showed the best performance across all evaluations and does not suffer from artifacts such as text-copying observed in JBU. Its computation is also fast (around 0.13 s) and suitable for real-time use, with potential for further optimization via downsampling. Therefore, if RGB guidance and edge preservation (e.g., through GDSR) are required for other objects with noticeable edges, the GF would be the preferred choice.

5.1.5 Horses Dataset

In this section, the evaluation of a mesh scanned from a horse is presented. Only two filters are considered here—the Bilateral Filter and the Guided Filter—as they were deemed the most suitable for potential implementation. The focus of this evaluation is not on absolute precision values, but rather on verifying that the proposed workflow can be applied in a real-use scenario without issues arising from motion or varying scanning conditions. First, the impact of precision on the scan is

examined visually (Fig. 32), keeping in mind that some motion may have occurred between the acquisition of the ground truth and the iPhone scan, which could affect the observed errors.

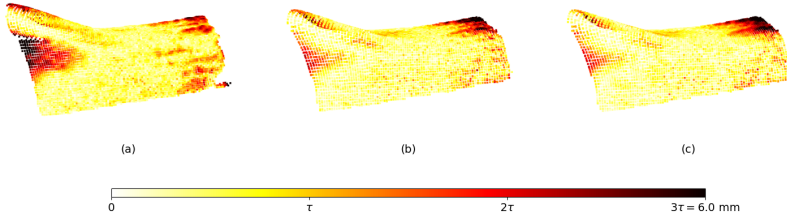


Figure 32. Colour-coded representation of precision on horse mesh vertices with threshold $\tau = 2 \text{ mm}$: **(a)** current surfel-based scan, **(b)** TSDF after bilateral filter, **(c)** TSDF after guided filter.

Since the absolute precision values are less critical in this evaluation compared to previous ones, the precision threshold is increased to 2 mm. Additionally, to minimize the impact of horse motion, only the regions expected to vary least during movement are considered. Specifically, the analysis focuses on the withers and a shortened section of the back, as illustrated in Fig. 32, while the neck is excluded due to the mane, which could introduce variations that are not relevant for this study.

Across all three scans, the areas showing the most imprecision are consistently the top and sides of the withers and the rear portion of the evaluated region. The fact that the same regions are affected suggests that some errors may arise from the time difference between the FARO scanning and the iPhone scan. Notably, the withers exhibit higher imprecision in the current scan, particularly in the steep sections. At the rear of the studied region, the lower precision appears as a bumpy surface, likely caused by horse movement introducing noisy frames or inconsistent depth values, which can affect the TSDF signed distance integration. For the proposed workflow, this region also shows reduced precision, highlighting a limitation of the implementation, as the frames are integrated without additional confidence-based filtering of the depth values, unlike the current method.

These observations will now be further evaluated using quantitative data.

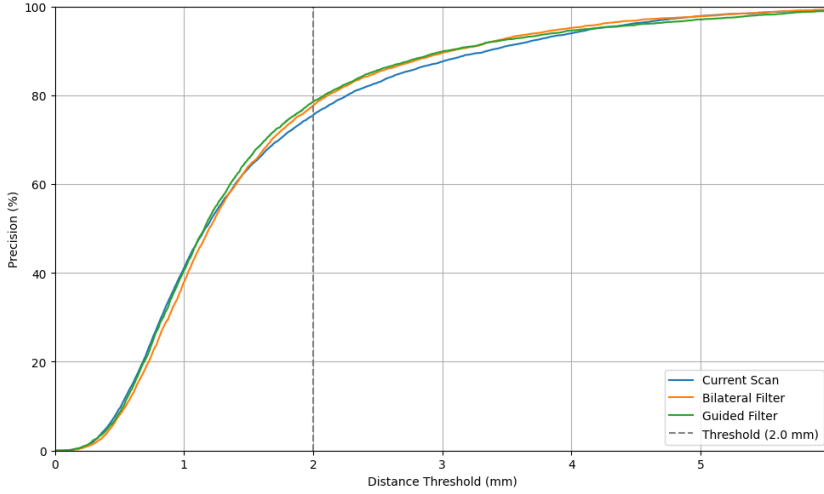


Figure 33. Precision plot for horse meshes

The plots indicate that the difference in precision between the current surfel-based process and the proposed TSDF-based approach is reduced when scanning a horse. For the surfel-based method, the observed precision corresponds roughly to the 1 mm threshold used in the evaluation. The change in target from previous datasets to a live horse appears to have a greater impact on the TSDF-based approach, which results in a smaller gap between the two methods. Nevertheless, the proposed TSDF-based workflow still demonstrates an advantage over the current process.

It is important to note that the relative difference between the methods may be more informative than the absolute difference, as some variations could arise from temporal changes or small movements of the horse between the ground truth scan and the iPhone scan. Despite these factors, the precision percentages remain acceptable, with values around 75%.

Table 13. Precision metrics across different filtering methods for horse 3D mesh vertices [unit: mm].

Method	Max ↓	STD ↓	RMSE ↓	MAE ↓	P(t) ↑
Current Scan	9.447	1.226	1.994	1.573	75.543
Bilateral Filter	7.574	1.123	1.906	1.539	77.660
Guided Filter	7.311	1.181	1.927	1.523	78.519

The table confirms a slight superiority of the proposed TSDF-based approach over the surfel-based method. The lower maximum error and standard deviation indicate that noise is better mitigated and the resulting meshes are smoother, particularly when the mean errors are similar. Additionally, the precision is slightly higher, as observed in the plots. While the guided filter still shows marginally better results, the bilateral filter performs almost equally well, demonstrating its utility.

Conclusion Overall, the evaluations indicate that the proposed workflow can be applied to real use cases. The slight reduction in precision is primarily due to the target itself and the absence of a perfectly reliable ground truth. Nevertheless, the TSDF-based approach consistently outperforms the current surfel-based method. Combined with its demonstrated robustness across multiple scanning trajectories, this confirms the viability and practical applicability of the proposed workflow.

To conclude this section, several reconstructed scans are presented to illustrate the efficiency of the proposed method. These scans were acquired under different conditions and for various purposes, highlighting the robustness and versatility of the workflow.

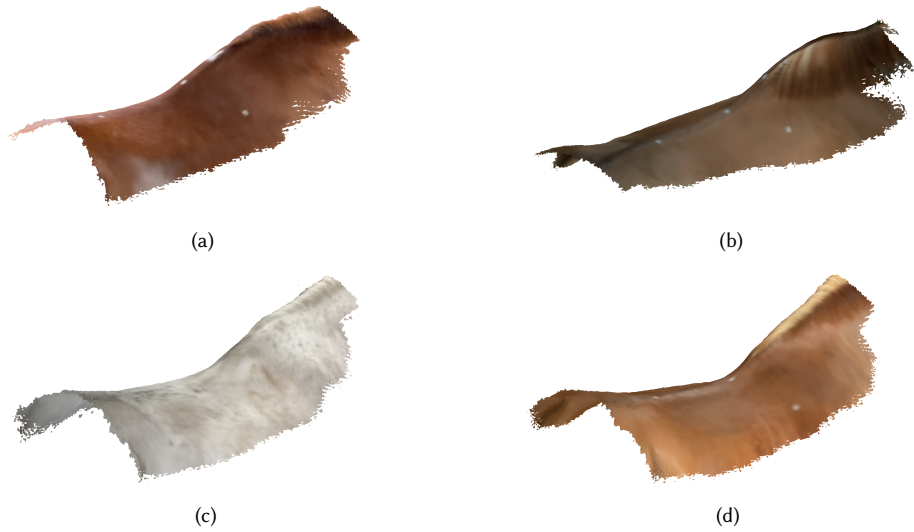


Figure 34. (a) Uniform colour with prominent withers, (b) uniform colour with prominent withers, also used to assess lighting conditions, (c) spotted horse to evaluate the influence on filters, (d) smooth withers.

5.2 Geometric Alignment and Surface Refinement in 3D Scans

In this section, the focus is on evaluating the proposed approach for mesh orientation. The assessment begins with quantitative data (Tab. 14), examining the differences between the two manually placed landmark points that should be aligned in the x and z directions. This comparison helps confirm the effectiveness of the proposed method.

Table 14. Difference in x and z coordinates between the placed landmark points before and after orientation for each scan [unit: cm]

Scan	x		z	
	Before	After	Before	After
1	0.8	0	3.0	0.5
2	3.7	0.6	2.1	0.4
3	3.0	0.9	5.4	0.0
4	3.0	0.5	1.4	1.5
5	0.4	0.7	2.4	1.8
6	5.9	0.7	3.5	0.0
7	1.6	0.7	2.8	0.5
8	1.4	0.4	2.6	0.1

The table shows the offset in centimetres along the axes of interest before and after the orientation process for the eight meshes where it was tested. Prior to orientation, the differences between the coordinates of the two manually placed landmark points exhibit noticeable variance. In some cases, such as Scan 5, the difference along the x -axis is only 0.4 cm, which is within an acceptable range. However, in other cases, the offset can exceed 5 cm, highlighting the visible misalignment and the necessity to normalize the orientation. These differences are largely influenced by the initial orientation of the phone during scanning. Since the first frame is used as the reference for the reconstruction process, even small deviations from a parallel position to the floor can introduce disorientation, as reflected in the table.

After applying the orientation process, the offsets are significantly reduced: the average difference along the x -axis drops from 2.5 cm to 0.6 cm, and along the z -axis from 2.9 cm to 0.6 cm. These results are particularly encouraging considering the tolerance required due to manual placement of the landmarks and hand-picked points on the meshes based on texture. Additionally, the standard deviation for the x -coordinate differences decreases from 1.81 to 0.27, indicating that the orientation process is consistent and repeatable. For the z -axis, improvements are also observed, though two meshes show less effective results, suggesting potential areas for further refinement.

A qualitative evaluation can be performed on one of these meshes (Scan 1 in Tab. 14) by examining the mesh coloured in green and red based on the side of the symmetry plane (Fig. 35), with the XZ plane shown as a reference.

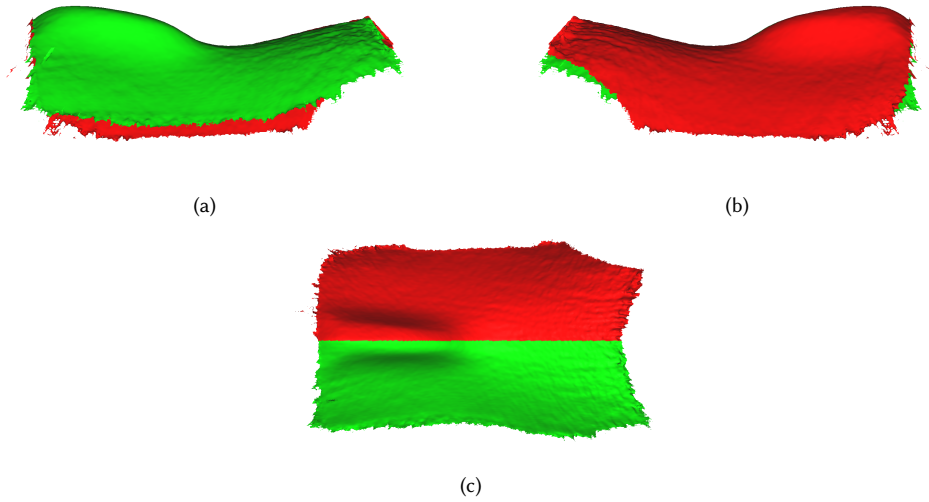


Figure 35. **(a)** Left view, **(b)** Right view and **(c)** Top-down view of a *Courbette* mesh coloured based on symmetry plane

This figure helps to verify that the symmetry plane is coherent by ensuring it correctly passes through the backline. Since the model represents a horse's back without mane, it does not introduce significant complications, and the backline should follow the highest points, as would be observed if slices were taken. Examining the left and right views (Fig. 35a and Fig. 35b), the correctness of the plane is confirmed: in each view, only one side's colour is visible, indicating that the plane passes through the intended highest points. Fig. 35c further supports this, as the separation lines align coherently with the expected plane, such as at the centre of the withers.

This qualitative evaluation can be extended to a horse scan (Fig. 36), as a quantitative evaluation is less suitable due to potential motion of the horse and the difficulty in placing reliable landmarks.

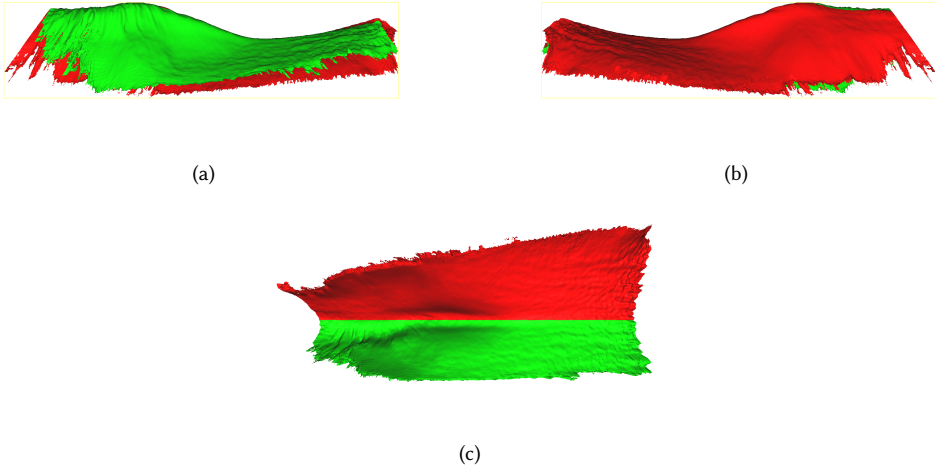


Figure 36. (a) Left view, (b) Right view and (c) Top-down view of a horse mesh coloured based on symmetry plane

Several challenges can potentially arise when working with a horse mesh that were not present in a simplified, static horse model: the mane is primarily present on one side, which can influence the symmetry; movement of the horse's neck can introduce greater asymmetry in the scans; and the horse's muscles are naturally asymmetric, presenting an additional challenge. Despite these factors, the results show that the proposed process performs well and is robust to these challenges, even if their impact is minimal in this case. Similar to the previous evaluation, the symmetry plane cuts the mesh accurately through the centre, particularly in the invariant regions, resulting in a precise orientation of the mesh.

Conclusion The proposed orientation method demonstrates both precision and consistency, with quantitative errors reduced by a factor of four and qualitative evaluations confirming a highly satisfactory alignment.

5.3 Automated Landmark Extraction and Measurement on 3D Surfaces

This section focuses on the automated extraction of measurements from meshes. An example is provided for a mesh obtained from a FARO laser scanner, as shown in Fig. 37.

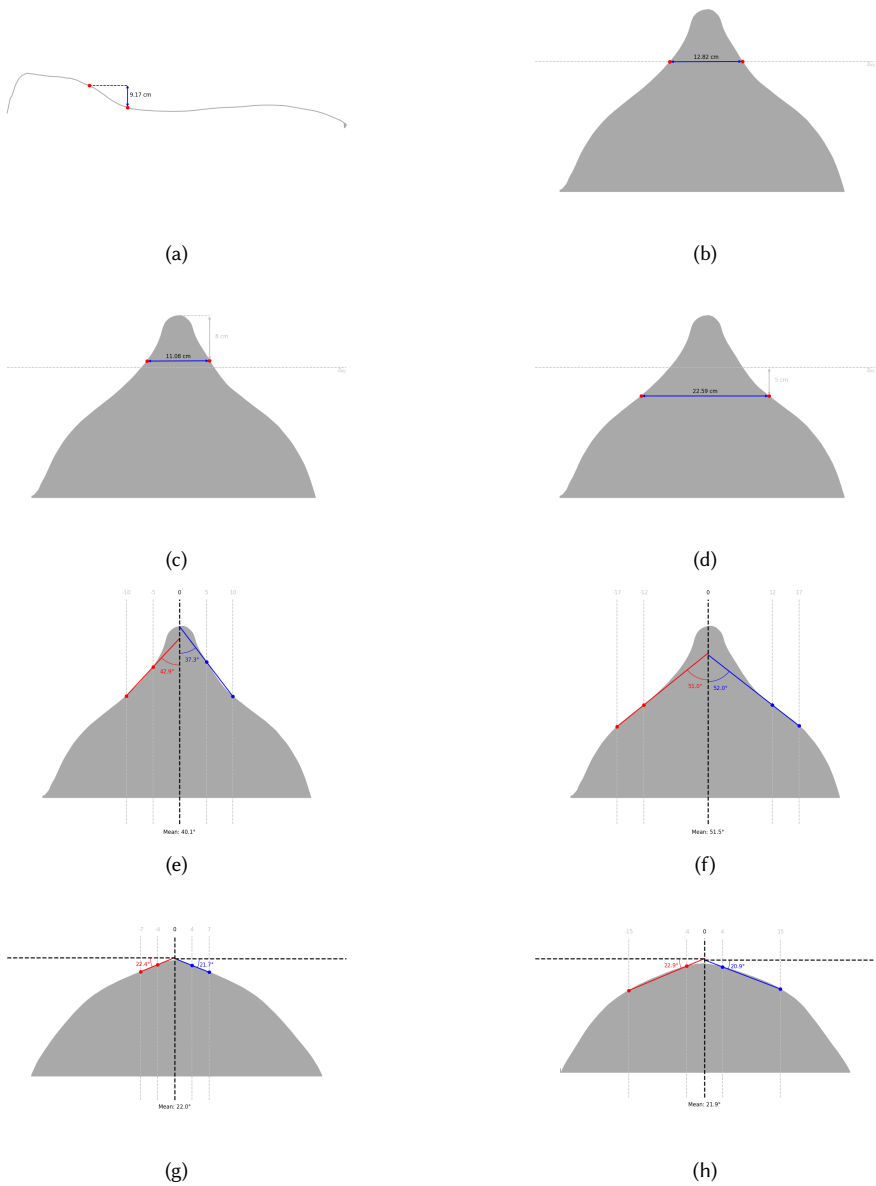


Figure 37. Visualisation of key anatomical measurements extracted from the horse's back mesh: (a) withers height, (b) withers width, (c) withers width measured 8 cm below the highest point, (d) shoulder width, (e) withers angles, (f) shoulder angles, (g) angles at the straddle level, and (h) angles at the teats level.

This scan is used for two main reasons. First, the reference point is clearly visible, which is useful both for manually selecting it and for evaluating the automated extraction process using the CNN. Second, while these scans are highly precise, they are accompanied by measurements obtained in SolidWorks. These measurements serve as a reference to assess the effectiveness of the automated process.

From the construction of the database, some aspects of the measurement extraction process were modified, particularly regarding the plane distances from the reference point, notably for shoulder and withers measurements. Therefore, to evaluate the precision of the obtained measurements, only two measures are considered here. The automated measurements align closely with the manual reference: the angles at straddle level and teats level were 22.35° and 21.5° in SolidWorks, compared to 22° and 21.5° in the automated process (Fig. 37g and Fig. 37h), yielding a difference of less than 0.5° , which can be considered satisfactory.

To further validate the approach, the extracted points can be visualized to qualitatively confirm the reliability of the automated process on new scans generated using the proposed reconstruction workflow.

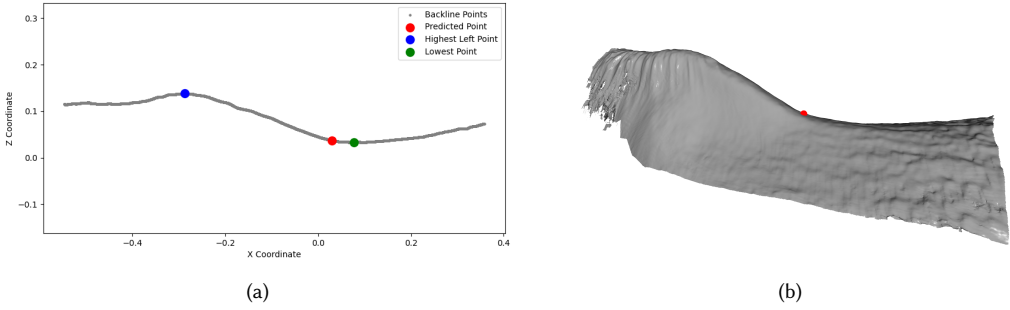


Figure 38. Visualisation of the predicted reference point on (a) the backline and (b) the mesh

The extracted point appears to be consistent with expectations, as the balance point is located near the lower end of the withers. Additionally, the visualization on the 3D mesh shows a version without any additional surface refinement, highlighting the noisy data at the back of the scan. This area could be improved in future work through further post-processing or filtering.

Conclusion To resume, the measures are extracted in an automated way while keeping a great precision and are visualized in a comprehensive way. The process is applicable for scans coming from difference source and most importantly the scans obtained from the proposed reconstruction workflow.

6 FURTHER WORK

Objectives can already be identified both in the short term—given the remaining time in the internship—and in the long term, either to refine the current work or to further develop the implemented features and their applications.

A first priority is to correct the implementation of the spectral processing used for mesh denoising, that was not presented here while the theoretical background was. In the current implementation, the brute-force version provided partial success: trimming eigenvalues produced visible results, but using a larger number of eigenvalues became impractical due to high computational cost. To address this, the MeTiS partitioning approach was tested. While the partitioning itself worked correctly, issues arose during eigenvector computation: using all eigenvalues did not correctly reconstruct the mesh and introduced additional noise. Improved parameter tuning—particularly of the orthogonal iterations—may resolve this. Verifying whether a given solution is suitable would also be an important step. Although this smoothing has limited impact on measurements and precision, it remains valuable for producing visually appealing scans, especially when normals are estimated, since they influence shading and visualization. Without normals, the noise is less apparent in the scans.

In parallel, the next step would be to integrate the completed work into the existing pipeline, first to improve mesh quality and orientation, and then to extract relevant measurements. In the short term, this would be implemented server-side by retrieving the frames and executing the workflow externally. In the long term, the goal is to integrate the entire process directly into the application.

A model will need to be developed to visually present the extracted measurements and their positions on the scan. This visualization will be shared with sales representatives to gather feedback on the measurements used, determine if additional measurements are needed, and assess their potential usefulness. It will also serve as an opportunity to showcase the progress made to the primary users of the application and identify possible long-term objectives.

Another future objective is to implement a fitting prediction system. This system would leverage the automated measurement extraction process to build a dataset, which could then be used to train a model to assist sales personnel in making informed decisions. While the specific model choice remains to be determined, a random forest or tree-based algorithm appears promising, both for its technical suitability and for its interpretability. Such models can visualize the predicted saddle configuration and its defining parameters in a way that is intuitive for sales representatives.

In practice, the workflow would involve selecting a saddle along with its possible configurations. Based solely on the extracted measurements, the appropriate saddle tree would be predicted. In a second step, using both the tree characteristics and the measurements, the most suitable panels would be chosen. For model training, automated extraction can be applied to previous scans and linked to the corresponding saddle configuration orders.

Finally, regarding the proposed workflow, further work is needed on weighting the depth data. Incorporating confidence values or masking out low-confidence depths could potentially improve precision in future TSDF-based implementations.

7 CONCLUSION

In resume, through this work, the overall pipeline from scanning to data analysis was improved. A new workflow for reconstruction was proposed, replacing the surfel-based reconstruction with a volumetric representation through the Truncated Signed Distance Function (TSDF). The assessment of the two methods showed the clear superiority of the proposed workflow for static datasets with meaningful metrics. Precision was improved by up to 30% in some cases, as the new workflow ensures that the mesh is extracted in fewer steps, avoiding the accumulation of errors as seen in the current process. Regarding use cases, the real-time integration of the TSDF approach was confirmed with integration of RGB-D data and extraction of a mesh in under a second, while precision in the case of live horse scanning was also validated. Among the filters evaluated, the bilateral filter leveraging only depth data was deemed sufficient, as TSDF already accounts for a significant part of the precision improvement, with filters mainly used for smoothing. Overall, a new precise and consistent scanning workflow was proposed and shown to be implementable.

In addition, a novel approach was proposed to re-orient the scans in space. The method leverages the invariant part of the horse back to align two mirrored meshes and define the symmetry plane as a median plane. This method was quantitatively assessed and showed that precision was improved by a factor of four.

Finally, analysis of the obtained meshes was conducted by adding new features. From the meshes, a batch of measurements was extracted and visualized to provide additional information on the reconstructed models.

For the sales representatives, the new reconstruction approach ensures a more reliable scanning process and greater confidence in the obtained models. The meshes are more precise, more complete, and allow for clearer visualization of details, which can justify choices. Combined with the post-processing, this provides reliable information that is meaningful not only to sales representatives but also to clients, who can better understand the necessary saddle adjustments and track the evolution of their horse over time.

In summary, the proposed pipeline strengthens trust and efficiency for sales representatives, provides clearer insights and transparency for clients, and creates added value for the tools of the company.

REFERENCES

- [1] A Bondi, S Norton, L Pearman, and S Dyson. Evaluating the suitability of an english saddle for a horse and rider combination. *Equine Veterinary Education*, 32:162–172, 2020.
- [2] P De Cocq, PR Van Weeren, and W Back. Saddle pressure measuring: Validity, reliability and power to discriminate between different saddle-fits. *The Veterinary Journal*, 172(2):265–273, 2006.
- [3] Line Greve and Sue Dyson. A longitudinal study of back dimension changes over 1 year in sports horses. *The Veterinary Journal*, 203(1):65–73, 2015.
- [4] K Von Peinen, T Wiestner, Brigitte von Rechenberg, and Michael A Weishaupt. Relationship between saddle pressure measurements and clinical signs of saddle soreness at the withers. *Equine Veterinary Journal*, 42:650–653, 2010.
- [5] Line Greve, Rachel Murray, and Sue Dyson. Subjective analysis of exercise-induced changes in back dimensions of the horse: The influence of saddle-fit, rider skill and work quality. *The Veterinary Journal*, 206(1):39–46, 2015.
- [6] Robert Lange and Peter Seitz. Solid-state time-of-flight range camera. *IEEE Journal of quantum electronics*, 37(3):390–397, 2001.
- [7] Sergi Foix Salmerón, Guillem Alenyà Ribas, and Carme Torras. Exploitation of time-of-flight (tof) cameras. 2010.
- [8] Ivan Eichhardt, Dmitry Chetverikov, and Zsolt Janko. Image-guided tof depth upsampling: a survey. *Machine Vision and Applications*, 28:267–282, 2017.
- [9] Jingyu Yang, Zhongyu Jiang, Xinchun Ye, and Kun Li. Depth super-resolution with color guidance: A review. *RGB-D Image Analysis and Processing*, pages 51–65, 2019.
- [10] Zhiwei Zhong, Xianming Liu, Junjun Jiang, Debin Zhao, and Xiangyang Ji. Guided depth map super-resolution: A survey. *ACM Computing Surveys*, 55(14s):1–36, 2023.
- [11] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *Sixth international conference on computer vision (IEEE Cat. No. 98CH36271)*, pages 839–846. IEEE, 1998.
- [12] Elmar Eisemann and Frédo Durand. Flash photography enhancement via intrinsic relighting. *ACM transactions on graphics (TOG)*, 23(3):673–678, 2004.
- [13] Johannes Kopf, Michael F Cohen, Dani Lischinski, and Matt Uyttendaele. Joint bilateral up-sampling. *ACM Transactions on Graphics (ToG)*, 26(3):96–es, 2007.
- [14] Yangguang Li, Lei Zhang, Yongbing Zhang, Huiming Xuan, and Qionghai Dai. Depth map super-resolution via iterative joint-trilateral-upsampling. In *2014 IEEE Visual Communications and Image Processing Conference*, pages 386–389. IEEE, 2014.
- [15] Frederic Garcia, Bruno Mirbach, Bjorn Ottersten, Frédéric Grandidier, and Angel Cuesta. Pixel weighted average strategy for depth sensor data fusion. In *2010 IEEE International Conference on Image Processing*, pages 2805–2808. IEEE, 2010.

- [16] Kai-Han Lo, Yu-Chiang Frank Wang, and Kai-Lung Hua. Edge-preserving depth map upsampling by joint trilateral filter. *IEEE transactions on cybernetics*, 48(1):371–384, 2017.
- [17] Frédo Durand and Julie Dorsey. Fast bilateral filtering for the display of high-dynamic-range images. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 257–266, 2002.
- [18] Kaiming He, Jian Sun, and Xiaoou Tang. Guided image filtering. *IEEE transactions on pattern analysis and machine intelligence*, 35(6):1397–1409, 2012.
- [19] Zhengguo Li, Jinghong Zheng, Zijian Zhu, Wei Yao, and Shiqian Wu. Weighted guided image filtering. *IEEE Transactions on Image processing*, 24(1):120–129, 2014.
- [20] Ben Weiss. Fast median and bilateral filtering. In *ACM SIGGRAPH 2006 Papers*, pages 519–526. 2006.
- [21] Sylvain Paris and Frédo Durand. A fast approximation of the bilateral filter using a signal processing approach. *International journal of computer vision*, 81:24–52, 2009.
- [22] Fatih Porikli. Constant time $\mathcal{O}(1)$ bilateral filtering. In *2008 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2008.
- [23] Qingxiong Yang, Kar-Han Tan, and Narendra Ahuja. Real-time $\mathcal{O}(1)$ bilateral filtering. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 557–564. IEEE, 2009.
- [24] Frederic Garcia, Djamila Aouada, Bruno Mirbach, Thomas Solignac, and Björn Ottersten. A new multi-lateral filter for real-time depth enhancement. In *2011 8th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 42–47. IEEE, 2011.
- [25] Frederic Garcia, Djamila Aouada, Bruno Mirbach, Thomas Solignac, and Björn Ottersten. Unified multi-lateral filter for real-time depth map enhancement. *Image and Vision Computing*, 41:26–41, 2015.
- [26] James Diebel and Sebastian Thrun. An application of markov random fields to range sensing. *Advances in neural information processing systems*, 18, 2005.
- [27] Yifan Zuo, Qiang Wu, Jian Zhang, and Ping An. Explicit edge inconsistency evaluation model for color-guided depth map enhancement. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(2):439–453, 2016.
- [28] Zhongyu Jiang, Yonghong Hou, Huanjing Yue, Jingyu Yang, and Chunping Hou. Depth super-resolution from rgb-d pairs with transform and spatial domain regularization. *IEEE Transactions on Image Processing*, 27(5):2587–2602, 2018.
- [29] David Ferstl, Christian Reinbacher, Rene Ranftl, Matthias R  ther, and Horst Bischof. Image guided depth upsampling using anisotropic total generalized variation. In *Proceedings of the IEEE international conference on computer vision*, pages 993–1000, 2013.
- [30] Beomjun Kim, Jean Ponce, and Bumsu Ham. Deformable kernel networks for joint image filtering. *International Journal of Computer Vision*, 129(2):579–600, 2021.

- [31] Jiaxiang Tang, Xiaokang Chen, and Gang Zeng. Joint implicit image function for guided depth super-resolution. In *Proceedings of the 29th acm international conference on multimedia*, pages 4390–4399, 2021.
- [32] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. Spie, 1992.
- [33] Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Colored point cloud registration revisited. In *Proceedings of the IEEE international conference on computer vision*, pages 143–152, 2017.
- [34] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *2009 IEEE international conference on robotics and automation*, pages 3212–3217. IEEE, 2009.
- [35] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006.
- [36] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Seminal graphics: pioneering efforts that shaped the field*, pages 347–353. 1998.
- [37] Erica Nocerino, Elisavet Konstantina Stathopoulou, Simone Rigon, and Fabio Remondino. Surface reconstruction assessment in photogrammetric applications. *Sensors*, 20(20):5863, 2020.
- [38] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312, 1996.
- [39] Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (ToG)*, 32(6):1–11, 2013.
- [40] Kaixuan Wang, Fei Gao, and Shaojie Shen. Real-time scalable dense surfel mapping. In *2019 International conference on robotics and automation (ICRA)*, pages 6919–6925. IEEE, 2019.
- [41] Thomas Schöps, Torsten Sattler, and Marc Pollefeys. Surfelmeshing: Online surfel-based mesh reconstruction. *IEEE transactions on pattern analysis and machine intelligence*, 42(10):2494–2507, 2019.
- [42] Thomas Whelan, Stefan Leutenegger, Renato F Salas-Moreno, Ben Glocker, and Andrew J Davison. Elasticfusion: Dense slam without a pose graph. In *Robotics: science and systems*, volume 11, page 3. Rome, 2015.
- [43] Chen He, Lei Wang, Yonghui Zhang, and Chunmeng Wang. Dominant symmetry plane detection for point-based 3d models. *Advances in Multimedia*, 2020(1):8861367, 2020.
- [44] Daisuke Hosoki, Tohru Kamiya, Namiko Kimura-Nomoto, Takako Okawachi, Etsuro Nozoe, and Norifumi Nakamura. Symmetric plane detection and symmetry analysis from a 3d point cloud data of face. In *2020 20th International Conference on Control, Automation and Systems (ICCAS)*, pages 402–406. IEEE, 2020.

- [45] Ivan Sipiran, Robert Gregor, and Tobias Schreck. Approximate symmetry detection in partial 3d meshes. In *Computer graphics forum*, volume 33, pages 131–140. Wiley Online Library, 2014.
- [46] Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Computer graphics forum*, volume 28, pages 1383–1392. Wiley Online Library, 2009.
- [47] Niloy J Mitra, Leonidas J Guibas, and Mark Pauly. Partial and approximate symmetry detection for 3d geometry. *ACM Transactions on Graphics (ToG)*, 25(3):560–568, 2006.
- [48] Kai Wang, Guillaume Lavoué, Florence Denis, and Atilla Baskurt. A comprehensive survey on three-dimensional mesh watermarking. *IEEE Transactions on Multimedia*, 10(8):1513–1527, 2008.
- [49] Jörg Vollmer, Robert Mencl, and Heinrich Mueller. Improved laplacian smoothing of noisy surface meshes. In *Computer graphics forum*, volume 18, pages 131–138. Wiley Online Library, 1999.
- [50] Gabriel Taubin. Curve and surface smoothing without shrinkage. In *Proceedings of IEEE international conference on computer vision*, pages 852–857. IEEE, 1995.
- [51] Hicham Badri, Mohammed El Hassouni, and Driss Aboutajdine. Kernel-based laplacian smoothing method for 3d mesh denoising. In *International Conference on Image and Signal Processing*, pages 77–84. Springer, 2012.
- [52] Hirokazu Yagou, Yutaka Ohtake, and Alexander Belyaev. Mesh smoothing via mean and median filtering applied to face normals. In *Geometric modeling and processing. Theory and applications. Gmp 2002. Proceedings*, pages 124–131. IEEE, 2002.
- [53] Thouis R Jones, Frédo Durand, and Mathieu Desbrun. Non-iterative, feature-preserving mesh smoothing. In *ACM SIGGRAPH 2003 Papers*, pages 943–949. 2003.
- [54] Shachar Fleishman, Iddo Drori, and Daniel Cohen-Or. Bilateral mesh denoising. In *ACM SIGGRAPH 2003 Papers*, pages 950–953. 2003.
- [55] Wangyu Zhang, Bailin Deng, Juyong Zhang, Sofien Bouaziz, and Ligang Liu. Guided mesh normal filtering. In *Computer Graphics Forum*, volume 34, pages 23–34. Wiley Online Library, 2015.
- [56] Shin Yoshizawa, Alexander Belyaev, and H-P Seidel. Smoothing by example: Mesh denoising by averaging with similarity-based weights. In *IEEE International Conference on Shape Modeling and Applications 2006 (SMI’06)*, pages 9–9. IEEE, 2006.
- [57] Hao Zhang, Oliver Van Kaick, and Ramsay Dyer. Spectral mesh processing. In *Computer graphics forum*, volume 29, pages 1865–1894. Wiley Online Library, 2010.
- [58] Zach Karni and Craig Gotsman. Spectral compression of mesh geometry. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 279–286, 2000.

- [59] Daizong Liu, Wei Hu, and Xin Li. Point cloud attacks in graph spectral domain: When 3d geometry meets graph signal processing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5):3079–3095, 2023.
- [60] Gerasimos Arvanitis, Aris S Lalos, and Konstantinos Moustakas. Spectral processing for denoising and compression of 3d meshes using dynamic orthogonal iterations. *Journal of Imaging*, 6(6):55, 2020.
- [61] Bruno Lévy and Hao Zhang. Spectral mesh processing. In *ACM SIGGRAPH 2010 Courses*, pages 1–312. 2010.
- [62] Danish Masood and Jiang Qi. 3d localization of hand acupoints using hand geometry and landmark points based on rgb-d cnn fusion. *Annals of Biomedical Engineering*, 50(9):1103–1115, 2022.
- [63] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.
- [64] Si Zhang, Hanghang Tong, Jiejun Xu, and Ross Maciejewski. Graph convolutional networks: a comprehensive review. *Computational Social Networks*, 6(1):1–23, 2019.
- [65] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.
- [66] Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. Meshcnn: a network with an edge. *ACM Transactions on Graphics (ToG)*, 38(4):1–12, 2019.
- [67] Shunwang Gong, Lei Chen, Michael Bronstein, and Stefanos Zafeiriou. Spiralnet++: A fast and highly efficient mesh convolution operator. In *Proceedings of the IEEE/CVF international conference on computer vision workshops*, pages 0–0, 2019.
- [68] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- [69] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *Proceedings third international conference on 3-D digital imaging and modeling*, pages 145–152. IEEE, 2001.
- [70] Kok-Lim Low. Linear least-squares optimization for point-to-plane icp surface registration. *Chapel Hill, University of North Carolina*, 4(10):1–3, 2004.
- [71] Peng Li, Ruisheng Wang, Yanxia Wang, and Wuyong Tao. Evaluation of the icp algorithm in 3d point cloud registration. *IEEE access*, 8:68030–68048, 2020.
- [72] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deep sdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019.
- [73] Ulrich Elsner. Graph partitioning-a survey. 1997.

- [74] George Karypis. Metis and parmetis. In *Encyclopedia of parallel computing*, pages 1117–1124. Springer, 2011.
- [75] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017.