*big data and cognitive computing*

MDPI

*Article*

# A Model Development Approach Based on Point Cloud Reconstruction and Mapping Texture Enhancement

**Boyang You [1] and Barmak Honarvar Shakibaei Asli [2,*]**

[1] College of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China; boyang.you.435@cranfield.ac.uk
[2] Centre for Life-Cycle Engineering and Management, Faculty of Engineering and Applied Sciences, Cranfield University, Bedford MK43 0AL, UK
* Correspondence: barmak@cranfield.ac.uk

**Abstract:** To address the challenge of rapid geometric model development in the digital twin industry, this paper presents a comprehensive pipeline for constructing 3D models from images using monocular vision imaging principles. Firstly, a structure-from-motion (SFM) algorithm generates a 3D point cloud from photographs. The feature detection methods scale-invariant feature transform (SIFT), speeded-up robust features (SURF), and KAZE are compared across six datasets, with SIFT proving the most effective (matching rate higher than 0.12). Using K-nearest-neighbor matching and random sample consensus (RANSAC), refined feature point matching and 3D spatial representation are achieved via antipodal geometry. Then, the Poisson surface reconstruction algorithm converts the point cloud into a mesh model. Additionally, texture images are enhanced by leveraging a visual geometry group (VGG) network-based deep learning approach. Content images from a dataset provide geometric contours via higher-level VGG layers, while textures from style images are extracted using the lower-level layers. These are fused to create texture-transferred images, where the image quality assessment (IQA) metrics SSIM and PSNR are used to evaluate texture-enhanced images. Finally, texture mapping integrates the enhanced textures with the mesh model, improving the scene representation with enhanced texture. The method presented in this paper surpassed a LiDAR-based reconstruction approach by 20% in terms of point cloud density and number of model facets, while the hardware cost was only 1% of that associated with LiDAR.

**Keywords:** 3D reconstruction; Poisson surface reconstruction; style transfer; deep learning; image quality assessment; texture mapping; computer vision; computer graphics



**Citation:** You, B.; Honarvar Shakibaei Asli, B. A Model Development Approach Based on Point Cloud Reconstruction and Mapping Texture Enhancement. *Big Data Cogn. Comput.* **2024**, *8*, 164. https://doi.org/10.3390/bdcc8110164

Academic Editors: Moulay A. Akhloufi and Jun Wang

Received: 30 August 2024
Revised: 11 November 2024
Accepted: 15 November 2024
Published: 20 November 2024



## 1. Introduction

A digital twin is an accurate virtual replica of a physical asset or system, capable of simulating and predicting various scenarios, encompassing the geometric structure, behavior, and performance of the entity [1]. The rapid advancement of the digital twin concept has led to its application in diverse industrial fields, such as manufacturing, construction, and energy. A critical requirement for digital twins is accurate real-time representation of the target entity through high-quality 3D models. However, the current geometrical model development approaches exhibit the following limitations: the modeling of specialized equipment is expensive, the modeling process is complex, the costs associated with modeling are high [2], and the accuracy of the models still falls short of the required standards [3], creating significant barriers to widespread adoption. Developing a cost-effective and fast modeling method is essential for the progress of the digital twin industry, facilitating broader application and yielding substantial industrial benefits.

To achieve a model development method that is low-cost and efficient for rapid development in digital twin scenarios, while accurately preserving the geometric feature information and texture details of the reconstructed objects and improving the display of

textures on the model surface, a pipeline of methods is proposed in this paper that combines SFM-based point cloud data acquisition, surface reconstruction, style-transfer-based texture enhancement, and texture mapping. In Section 3, the methodology is explained. Experimental results and discussions are given in Section 4. Finally, an analysis of the results and future improvements are provided in Section 5.

## 2. Related Work

### 2.1. There-Dimensional Reconstruction

Current methodologies for geometric model development rely on a variety of costly hardware to acquire point cloud data for the purpose of the three-dimensional reconstruction of objects. Point cloud data are a widely used 3D representation, typically containing six dimensions, 3D coordinates $(x, y, z)$, and RGB values or other information for each point, which represent geometric relationships within complex 3D objects. The main point cloud acquisition methods available are laser-scanner-based, time-of-flight (ToF)-camera-based, structured-light-camera-based, stereo-vision-camera-based, and photogrammetry methods, whose equipment is shown in Table 1.

**Table 1.** Comparison of point cloud data acquisition equipment [1].

| Hardware | Cost (GBP) | Accuracy (mm) | Measurement Range (m) | Outdoor Work |
|---|---|---|---|---|
| Lidar scanner | 600+ | 1–3 [4] | 200+ | Unaffected |
| Structured light camera | 200–4000 | 0.01–0.32 [5] | 0.3–10 [5] | Highly affected |
| ToF camera | 400–30,000 | 0.5–2.2 [6] | 0.5–6.0 [6] | Minimally affected |
| Stereo vision camera | 200+ | 0.05–1 [7] | 10–100 [7] | Unaffected |
| Photogrammetry methods | N/A | 1–10 | 10–100 | Unaffected |

[1] The data presented in the table without references are derived from the operational experiences of the authors.

LiDAR systems perceive depth information by emitting laser pulses and measuring their round-trip time, while simultaneously recording the angle of the reflected light, which relies on the speed of light and precise temporal measurements to generate high-precision three-dimensional point clouds. Pose graph clustering [8] and NeRF [9] were introduced to precisely map large-scale environments. The LiDAR-based method is ideal for large-scale reconstruction due to its high efficiency, and independence from ambient light. However, it is costly and unable to reconstruct surface color and texture, and it struggles with geometric interference and environmental occlusion.

Depth cameras are categorized into ToF cameras, structured light cameras, and stereo vision cameras. ToF cameras use an active emitter to project light onto a target and an optical sensor to collect the reflected light. The 3D information is obtained by calculating the time delay between the emission and collection of the signal. Unlike structured light and binocular vision technologies, the ToF-camera-based method does not require triangulation to compute 3D information [10]. The compact hardware system of ToF cameras facilitates their deployment in the execution of scanning tasks. Structured light camera technology employs optical encoding methods, such as projecting laser stripes, Gray codes, or sinusoidal patterns onto the object's surface. Other patterns include point maps [11], line maps [12], and crossed line maps [13]. However, structured light cameras using optical coding methods typically employ statistical rather than precise mathematical models, limiting their accuracy to within a few millimeters. With a stereo vision camera, depth information is obtained by determining the angle between the projected rays and their baselines. To achieve robust matching of the two camera views, various stereo-matching methods have been proposed [14–16]. The measurement accuracy of stereo vision is comparable to structured light techniques, achieving 0.05 mm to 0.1 mm, and even 0.001 mm in microscopic applications [17].

Representative algorithms for photogrammetry methods include structure-from-motion (SfM) and multi-view stereo (MVS) [18]. Current SfM algorithms can be categorized into incremental SfM, global SfM, SfM integrated with deep learning, and combined SfM

with MVS reconstruction methods. Gao et al. [19] developed a multi-view 3D reconstruction system fusing SIFT and SURF features for enhanced point detection, applying scale constraints for robust matching and using RANSAC for error removal. These enhancements significantly outperformed bundle adjustment [20] in robustness and completeness. With continuous algorithmic advancements, global SFM offers enhanced reconstruction accuracy and efficiency compared to incremental SFM, due to the simultaneous consideration of all image factors [21]. The main framework of global SFM includes global similarity estimation, feature extraction and matching, and camera pose estimation [22]. To enhance algorithm efficiency, the visual bag-of-words model is used in global similarity estimation [23]. Recent algorithms such as NAPSAC, PROSAC, and P-NAPSAC have been developed to expedite robust feature point estimation. PROSAC starts by sampling from points with the highest probability based on global prior ranking [24], NAPSAC utilizes spatial coherence by sampling from points with high inner line ratios [25], and P-NAPSAC combines local and global sampling methods for improved accuracy [22].

With advancements in deep learning, key aspects of the SFM algorithm are being enhanced by using neural networks for improved efficiency. CNN-based global descriptors are used for efficient feature matching of unordered image sequences [26]. Scholars [27] from Fudan University integrated deep learning into the bundle adjustment (BA) process, enhancing the 3D reconstruction robustness for untextured or non-Lambertian surfaces.

SFM combined with MVS can create denser point cloud data, computing the depth of each pixel in a 2D image by identifying corresponding points in multiple images and generating dense 3D point clouds based on these depth maps. Some open-source algorithms integrate SfM and MVS, such as COLMAP [28], *OpenMVS* [29], *MVE* [30] and *VisualSFM* [31].

### 2.2. Surface Reconstruction

A mesh model with a continuous surface can be generated using surface reconstruction technology. Due to optical reflections, environmental noise, and other factors, point cloud data often contain outliers and noise [32], necessitating denoising before surface reconstruction. Additionally, to effectively address the issue of point cloud data discreteness and provide normal information for the reconstructed surface, surface reconstruction of the point cloud is required [33–35]. Wu et al. [36] proposed a skip-attention-based correspondence filtering network (SACF-Net) for point cloud filtering and point cloud registration, Lu et al. [37] achieved automatic prediction for filtering by training a classification network model based on patch samples, and Zhang et al. [38] used graph-based denoising to capture geometric details by treating the point cloud as a graph signal. Deep learning effectively filters point clouds through supervised methods like PointNet [39], which uses MLPs for feature learning, and PointNet++ [40], which captures geometric relationships using a hierarchical network. Unsupervised methods like *TotalDenoising* [41] use Monte Carlo convolution-based encoder–decoders to reduce noise by leveraging spatial locality and bilateral appearance, without needing ground truth samples. Liu et al. [42] used partial differential equations and IMQ radial basis functions for 3D surface reconstruction and repair of scattered data. Dai et al. [43] introduced a novel point-based representation termed Gaussian surfels, which integrates the flexible optimization of 3D Gaussian points with the surface alignment properties of surfels, enhancing the optimization stability and surface alignment. Researchers have accelerated surface solving with deep learning, exemplified by Comi et al. [44], who improved DeepSDF, a neural network representing shapes with a continuous signed distance function for high-quality shape representation and interpolation.

### 2.3. Style Transfer

A mesh model exhibits a uniform coloration, lacking a reproduction of the surface textures characteristic of the target object. Style transfer is a technical way of extracting the texture of a style image and combining it with core features from the content image

to change the texture of the content image, which can be applied to enhance the texture of mesh models. With the increasing influence of convolutional neural networks in image processing, it has become the consensus that shallow convolutional layers extract texture and deep convolutional layers extract image features [45]. Zhang et al. [46] proposed an inversion-based style transfer method (InST) that efficiently and accurately extracted the pivotal information from images, thereby capturing and transferring the artistic style of a painting. Lin et al. [47] used a generative adversarial neural network (GAN) and added a convolutional layer to extract the features of the content image for image-to-image style conversion, as shown below. Feedforward networks allow for fast synthesis, but these methods often lack diversity and quality. To improve the diversity of the texture features learned by network models, Gatys et al. [48] introduced control over the spatial location, color information, and cross-spatial scaling, to achieve high-resolution controlled stylization. In recent research [49–51], network models with complex and large model architectures have been replaced by efficient encoder–decoder architectures. Zhang et al. [52] introduced an edge loss function within the transformer model, which enhances the content details and prevents the generation of blurred results due to the excessive rendering of style features, while simultaneously mitigating the issue of content leakage. To improve the architecture, Zhu et al. [53] proposed a novel all-to-key attention mechanism that integrates distributed attention and progressive attention, matching each position in content features to stable key positions in style features, demonstrating exceptional performance in preserving semantic structures and rendering consistent style patterns.

### *2.4. Texture Mapping*

Through texture mapping technology, texture images can be combined with mesh models to create realistic 3D models, which can provide visual details like texture and material. Texture mapping, which uses a 2D image to represent surface appearance, is widely used. Neural scene representations, such as *NeRF* [54] and deep reflectance volumes [55], use volume rendering but fail to separate geometry from appearance, limiting surface editing. Thies et al. [56] addressed this by optimizing neural textures on 3D mesh proxies. Xiang et al. [57] further improved on this with the *neutex* model, which uses volumetric representation for geometry and 2D texture maps for surface appearance, enabling scene reconstruction and traditional texture editing from multi-view images.

## 3. Methodology

### *3.1. Dataset*

A public dataset (hereinafter referred to as Dataset 1) was used to evaluate the application of the reconstruction algorithm in large-scale scenes, such as the external environment of factories, and contains 200 images of building structures in eight categories, with corresponding internal camera parameter files. In addition, a private dataset (hereinafter referred to as Dataset 2) was established, which contains 48 high-definition images of two kinds of robotic arms and two types of CNC machining centers, as representatives of common factory equipment. Examples of the two kinds of datasets are shown in Figure 1.

Figure 2 shows the dataset (hereinafter referred to as Dataset 3) collected for the style transformation task in deep learning, containing six different style images and the corresponding data sources.

Photogrammetry algorithms require a camera's intrinsic parameter matrix for calculations. The calibration process, results, and intrinsic parameters for the images in Dataset 2 are detailed in Appendix A.

(a) Samples of Dataset 1 (b) Samples of Dataset 2

**Figure 1.** Samples from Dataset 1 (Source: https://github.com/Abhishek-Aditya-bs/MultiView-3D-Reconstruction/tree/main/Datasets accessed on 18 November 2024) and samples from Dataset 2.

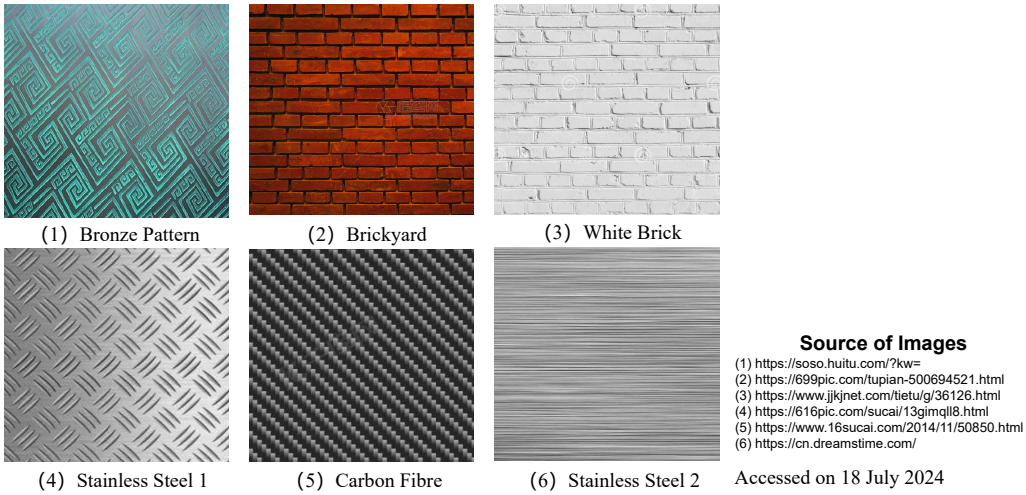



**Figure 2.** Demonstration of Dataset 3.

### *3.2. Point Cloud Data Acquisition*

The SFM algorithm is realized through feature point extraction and matching, camera pose estimation, triangulation, and global optimization, as illustrated in Figure 3.

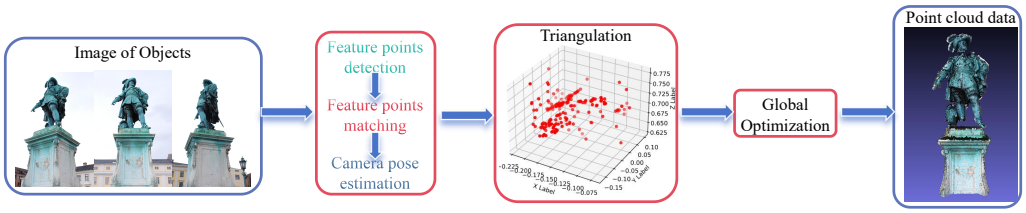



**Figure 3.** Diagram of SFM algorithm.

The feature points, uniquely identifying the scene, are extracted from the image by calculating feature descriptors.

$$D(x,y,\sigma) = [G(x,y,k\sigma) - G(x,y,\sigma)] * I(x,y) = L(x,y,k\sigma) - L(x,y,\sigma). \quad (1)$$

Equation (1) is the computational formula for the image pyramid before SIFT feature detection, where the Gaussian scale space $L$ is first constructed by continuously varying the scale by Gaussian kernel function convolving with the image $I$, $L(x,y,\sigma)$ denotes the Gaussian scale space, and $D(x,y,\sigma)$ is the differential Gaussian image. The differential Gaussian (DoG) response image is obtained by subtracting the images of two neighboring Gaussian spaces, and the location of feature points is detected by the DoG. A gradient

histogram of the image is used to find the stable orientation of the feature points. Then, the SIFT feature point location, scale, and orientation information can be obtained. Speed-up robust features (SURF) [58] and KAZE [59] feature extraction are also used to compare with SIFT.

Feature points are first matched by KNN matching [60], then refined by RANSAC [19] to enhance the accuracy.

The camera pose estimation is realized by estimating rotation and translation matrices, which is based on the camera imaging model and coplanarity condition in photogrammetry.

As shown in Figure 4, the coordinate system transformation is described in the following equation,

$$ZP_{uv} = Z\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \boldsymbol{K}(\boldsymbol{R}P_w + \boldsymbol{t}) = \boldsymbol{K}\boldsymbol{T}P_w, \tag{2}$$

where the matrix consisting of $3 \times 3$ vectors is named the camera intrinsic matrix $K$, with the camera coordinate system rotation matrix $\boldsymbol{R}$ and translation matrix $\boldsymbol{t}$ being called the extrinsic matrix of the camera.

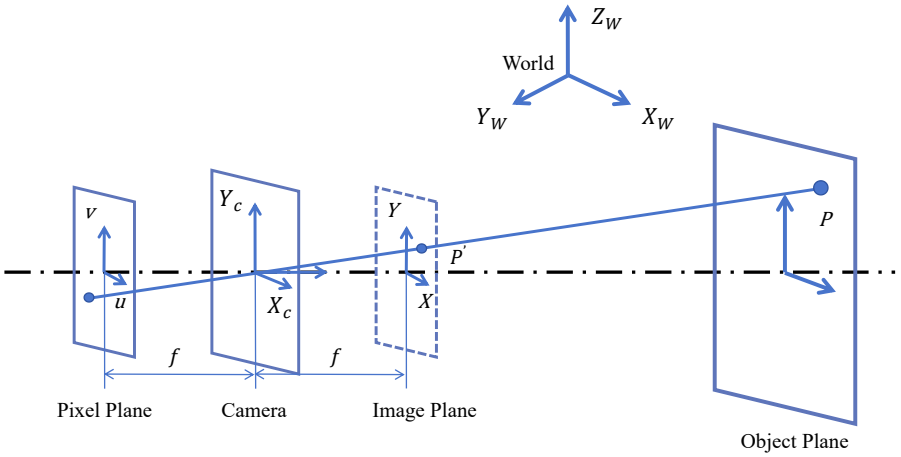



**Figure 4.** Camera imaging model.

Figure 5 describes the geometric correspondence relationship of a point-in-world coordinates when using two cameras. The mathematical expression of the coplanarity condition in photogrammetry is described in Equation (3).

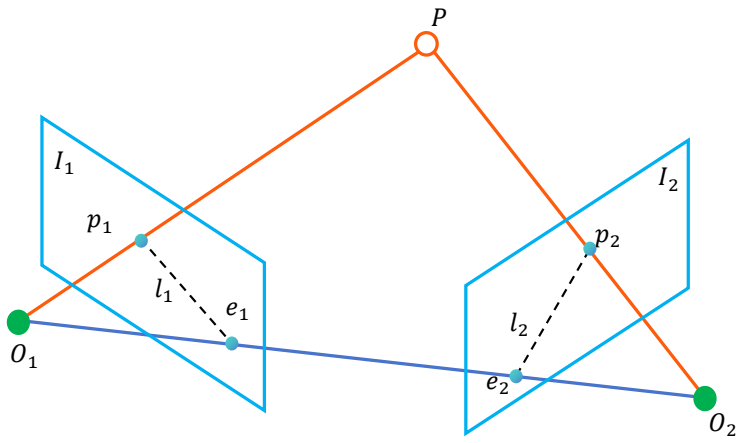



**Figure 5.** Coplanarity condition of photogrammetry.

$$\boldsymbol{p}_2^T \boldsymbol{K}^{-T} \boldsymbol{t}^{\wedge} \boldsymbol{R} \boldsymbol{K}^{-1} \boldsymbol{p}_1 = 0, \tag{3}$$

where $p_1$ and $p_2$ are pixel locations of point $P$ in the pixel planes of two cameras. Given the intrinsic parameters of the camera, the concept of an essential matrix is introduced here [61], simplifying the equation as

$$\boldsymbol{p}_2^T \boldsymbol{E} \boldsymbol{p}_1 = 0. \tag{4}$$

Thus, the problem of solving the camera position involves calculating the camera essential matrix $E$. By expanding the essential matrix $E$ and writing it in the form of a vector and selecting eight pairs of feature points, Equation (5) can be obtained:

$$\begin{pmatrix} u_1^1 u_2^1 & u_1^1 v_2^1 & u_1^1 & v_1^1 u_2^1 & v_1^1 v_2^1 & v_1^1 & u_2^1 & v_2^1 & 1 \\ u_1^2 u_2^2 & u_1^2 v_2^2 & u_1^2 & v_1^2 u_2^2 & v_1^2 v_2^2 & v_1^2 & u_2^2 & v_2^2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ u_1^8 u_2^8 & u_1^8 v_2^8 & u_1^8 & v_1^8 u_2^8 & v_1^8 v_2^8 & v_1^8 & u_2^8 & v_2^8 & 1 \end{pmatrix} \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \\ e_8 \\ e_9 \end{pmatrix} = 0. \tag{5}$$

The $R$, $t$ matrix can be obtained by decomposing the essential matrix $E$ using singular value decomposition (SVD), and finally determined according to the orientation of the camera positive depth.

The point-in-world coordinate is computed based on the coordinate transformation matrix $P$, as follows:

$$P = K[R \mid t]. \tag{6}$$

Decomposing $P$ into three vectors,

$$P_i = \begin{bmatrix} P_{i1} \\ P_{i2} \\ P_{i3} \end{bmatrix} \tag{7}$$

The relationship between the world coordinates and the pixel coordinates of a point can be expressed as follows,

$$\begin{bmatrix} P_{i1} & \tilde{X} \\ P_{i2} & \tilde{X} \\ P_{i3} & \tilde{X} \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}, \tag{8}$$

where $\tilde{X}$ is the location of the point-in-world coordinate, and $x_i$ and $y_i$ are the locations in pixel coordinates. Since $x_1$, $x_2$, $P_1$, and $P_2$ are known, a transformation of the above equation yields Equations (9)–(11).

$$A = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}, \quad A\tilde{X} = 0. \tag{9}$$

$$A_1 = \begin{bmatrix} x_1 P_{13} - P_{11} \\ y_1 P_{13} - P_{12} \end{bmatrix}, \tag{10}$$

$$A_2 = \begin{bmatrix} x_2 P_{23} - P_{21} \\ y_2 P_{23} - P_{22} \end{bmatrix}. \tag{11}$$

Using the least squares method to solve the system of equations, the spatial coordinates of point $\tilde{X}$ can be obtained, while solving the coordinates of the point cloud.

Finally, bundle adjustment [28] is used as a global optimization method to improve the accuracy of the triangulation process by iterative refinement of the camera position parameters.

### *3.3. Surface Reconstruction*

Poisson surface reconstruction is realized by a process involving constructing and solving Poisson's equation and generating an equivalent surface, as shown in Figure 6.

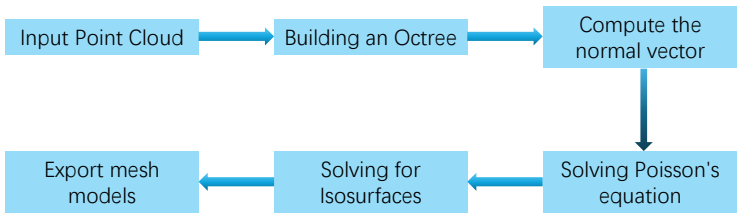



**Figure 6.** Process of surface reconstruction.

First constructing the indicator function $X$, setting the target $M$ inside to 1 and the target $M$ outside to 0, and its exponential function $\chi_M$, where the point cloud normal vector with $\chi_M$ is shown in Figure 7.

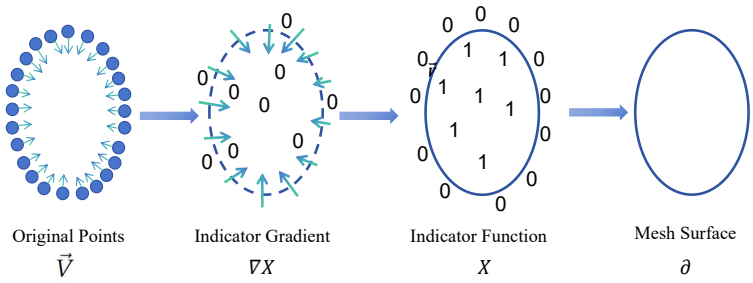



**Figure 7.** Demonstration of isosurface.

The indicator function is smoothed using a smoothing function $\tilde{F}$. For any point $p \in \partial M$, define $N\partial M(p)$ as the inward surface normal vector and $\tilde{F}_p(q_0)$ as a smoothing filter.

$$\nabla(\chi_M * \tilde{F})(q_0) = \int_{\partial M} \tilde{F}_p(q_0)\vec{N}_{\partial M}(p)dp. \tag{12}$$

According to *Gauss's divergence theorem*, the vector space and the indicator function satisfy the constraints, $\nabla\tilde{\chi} = \tilde{V}$. Since vector fields cannot be integrated, the derivation of both sides of the above equation yields the Laplace equation. Morphing the equation yields

$$\Delta\chi \equiv \nabla \cdot \nabla\chi = \nabla \cdot \vec{V}. \tag{13}$$

By solving this Poisson equation, the indicator function can be solved. The function $F_0(q)$ corresponding to a node $o$ in the octree is

$$F_0(q) \equiv F\left(\frac{q - o.c}{o.w}\right)\frac{1}{(o.w)^3}, \tag{14}$$

where $o.c$ is the center of $o$ and $o.w$ is the width of o. The vector space $\vec{V}(q)$ can be approximated as

$$\vec{V}(q) \equiv \sum_{s\in S}\sum_{o\in Ngb_0(s)} \alpha_{o,s}F_o(q)s.\vec{N}. \tag{15}$$

Defining $\tilde{\chi} = \sum_0 x_0 F_0$, then solving for $\chi$ is the same as solving for $x_o$. Let the number of nodes in the octree be N , and the mesh can be built by calculating value of the N × N matrix $L$ at the position $(0, O')$

$$L_{o,0'} \equiv \left\langle \frac{\partial^2 F_o}{\partial x^2}, F_{o'} \right\rangle + \left\langle \frac{\partial^2 F_o}{\partial y^2}, F_{o'} \right\rangle + \left\langle \frac{\partial^2 F_o}{\partial z^2}, F_{o'} \right\rangle. \tag{16}$$

### *3.4. Style-Transfer-Based Texture Enhancement*

Style transfer is achieved by combining the VGG-19 network and the Gram matrix. The VGG19 network (Figure 8) is capable of advanced image feature extraction, synthesis, and manipulation. It can extract content features from images. The network comprises 16 convolutional layers, 5 pooling layers, 3 fully connected layers, and a final softmax layer.

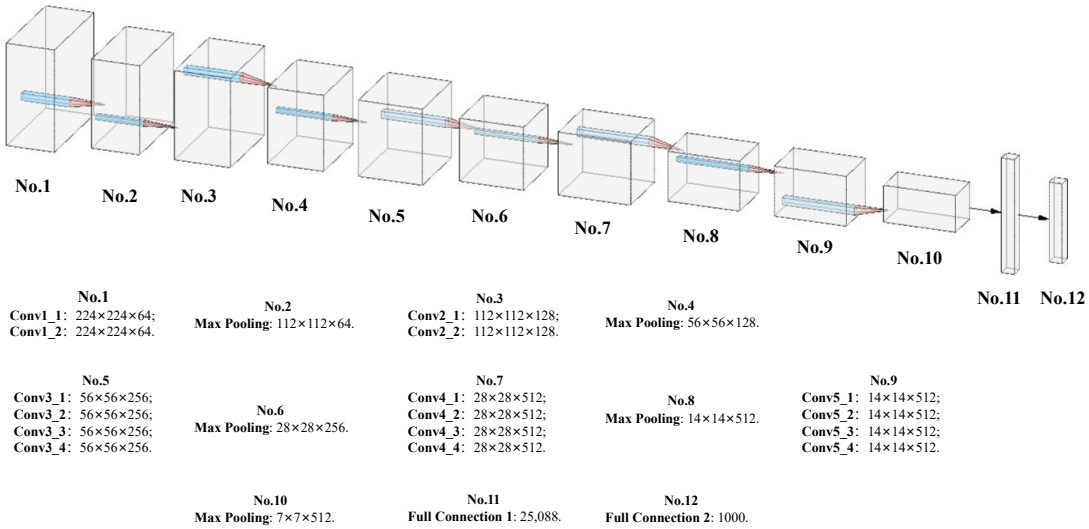



**Figure 8.** Demonstration of VGG network.

The VGG model is divided into two parts: features (containing convolutional and pooling layers) and a classifier (containing fully connected layers). The output of each convolutional layer from the features model is used to compute the content and style loss. To ensure consistency, the *.eval()* function sets the network to evaluation mode. Additionally, before feeding an image into the VGG network, it must be normalized using $mean = [0.485, 0.456, 0.406]$ and $std = [0.229, 0.224, 0.225]$ for each channel.

Define the input content image as $\vec{p}$, the final generated image as $\vec{x}$, and the convolutional layer used in the style migration process as $l$. $F_{ij}^l$ denotes the activation value of the image generated $l$ on the ith convolutional kernel of the first layer located at $j$, and similarly $P_{ij}^l$ denotes the activation value of the content image on the *No. i* convolutional kernel of the first layer located at $j$. Therefore, the loss function of the content of a single layer is

$$\text{Lcontent}(\overrightarrow{\text{p}}, \vec{x}, l) = \frac{1}{2} \sum_{i,j} \left( F_{ij}^l - P_{ij}^l \right)^2. \tag{17}$$

The derivative of the content loss function concerning the activation of layer l is equal to

$$\frac{\partial L_{\text{content}}}{\partial F_{ij}^l} = \begin{cases} \left( F^l - P^l \right)_{ij} & \text{if } F_{ij}^l > 0 \\ 0 & \text{if } F_{ij}^l < 0 \end{cases}. \tag{18}$$

The style of an image is represented by a style matrix. To capture the texture representation of the input image, a Gram matrix [62], as shown in Figure 9, uses a symmetric

matrix formed by the pairwise inner products of $k$ vectors in an $n$-dimensional Euclidean space, representing these vectors' texture features.

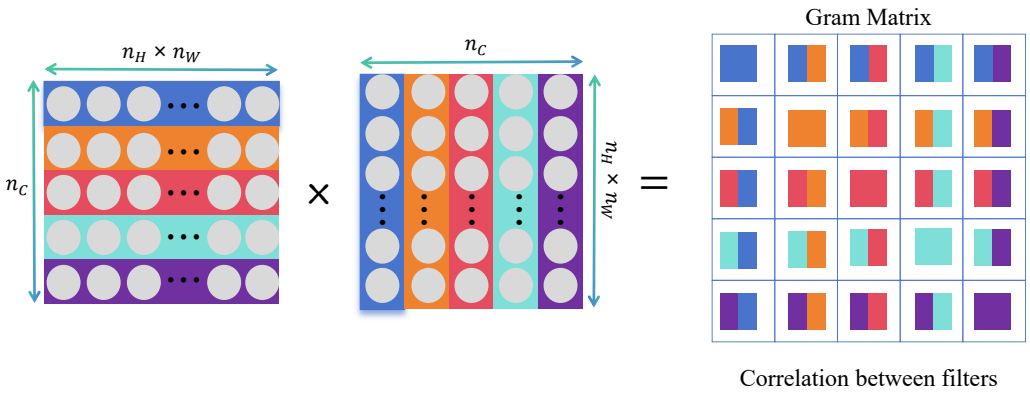



**Figure 9.** Demonstration of Gram matrix.

The feature space of the Gram matrix consists of the correlations between the different filter responses, where the expectation accounts for the spatial extent of the feature mapping. Defined as the activation value of an image $F_{ij}$ located at $j$ on the ith convolution kernel of the first layer, the corresponding Gram matrix is

$$G_{ij}^{l} = \sum_{k} F_{ik}^{l} F_{jk}^{l}. \tag{19}$$

Let the height and width of the convolutional layer $l$ be $N_l$ and $M_l$, respectively, with the Gram matrix $A_{ij}^{l}$ corresponding to the stylized image $\vec{p}$, and the Gram matrix corresponding to the generated image $\vec{a}$, $G_{ij}^{l}$. So the stylized loss function for a single layer is

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} \left( G_{ij}^{l} - A_{ij}^{l} \right)^2. \tag{20}$$

The derivative of the style loss function $E_l$ concerning the layer $l$ activation can be computed analytically, as follows:

$$\frac{\partial E_l}{\partial F_{ij}^{l}} = \begin{cases} \frac{1}{N_l^2 M_l^2} \left( \left( F^l \right)^T \left( G^l - A^l \right) \right)_{ji} & \text{if } F_{ij}^{l} > 0 \\ 0 & \text{if } F_{ij}^{l} < 0 \end{cases}. \tag{21}$$

The total style loss can be calculated as follows:

$$\text{Lstyle}(\vec{a}, \vec{x}) = \sum_{l=0}^{L} w_l E_l. \tag{22}$$

The loss function $L_{\text{total}}$ for style migration consists of the content loss function and the style loss function described above, with the weighting factors $\alpha$ and $\beta$ controlling the balance between content and style reconstruction, as shown in the following equation:

$$L_{\text{total}}(\vec{p}, \vec{a}, x) = \alpha L_{\text{content}}(\vec{p}, \vec{x}) + \beta L_{\text{style}}(\vec{a}, \vec{x}). \tag{23}$$

The Figure 10 illustrates the architecture of style transfer. The style image $\vec{a}$ is processed to compute and store its style representation $A^L$ across all layers. The content image $\vec{p}$ is processed to store its content representation $P^L$ in one layer. A random noise image $\vec{x}$ is then passed through the network, computing its style feature $G^L$ and content feature $F^L$. The style loss $L_{style}$ is calculated as the mean squared difference between $G^L$ and $A^L$ for each layer. The content loss $L_{content}$ is the mean squared difference between $F^L$ and $P^L$.

The total loss $L_{total}$ is a linear combination of these losses. Using error backpropagation, the gradient updates the image $\vec{x}$ iteratively to match both the style features of the style image and the content features of the content image.

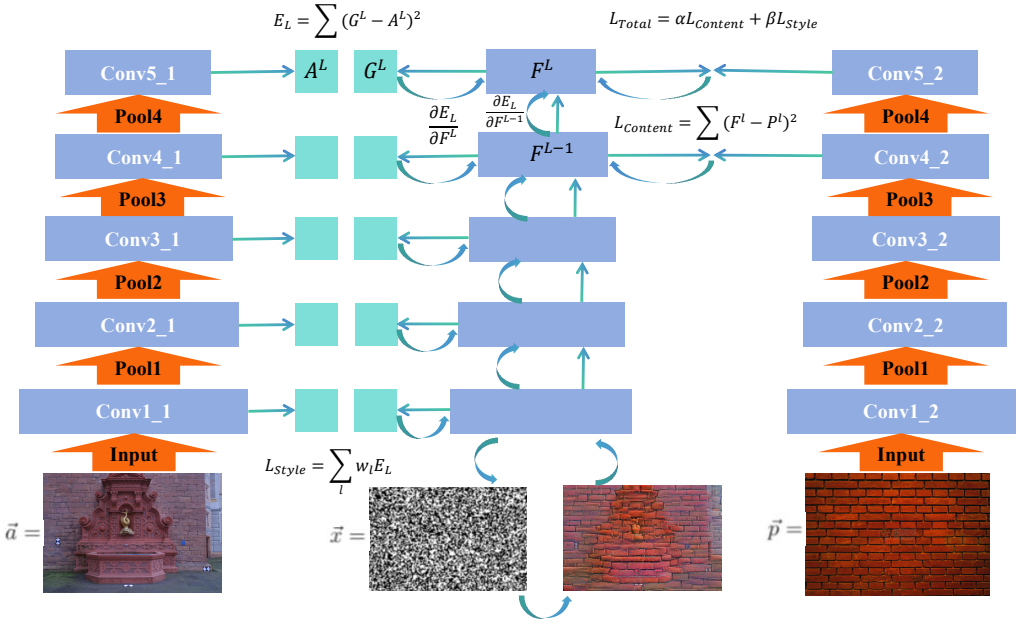



**Figure 10.** Style transformation architecture.

The hyperparameters for style transfer deep learning training based on VGG networks are shown in Table 2.

**Table 2.** Hyperparameters for style transfer training.

| Content Weight | Style Weight | Content Layer | Style Layer | Optimizer | Learning Rate | Ephochs | Iteration |
|---|---|---|---|---|---|---|---|
| 1 | 1000 | $'block4-conv2': 0.5$, $'block5-conv2': 0.5$ | $'block1-conv1': 0.2$, $'block2-conv1': 0.2$, $'block3-conv1': 0.2$, $'block4-conv1': 0.2$, $'block5-conv1': 0.2$ | Adam | 0.03 | 20 | 100 |

### *3.5. Texture Mapping*

Mathematically, a projector function is applied to spatial points to obtain parameter space values, which are then converted to texture space using corresponding functions. The algorithm's flow is illustrated in Figure 11.

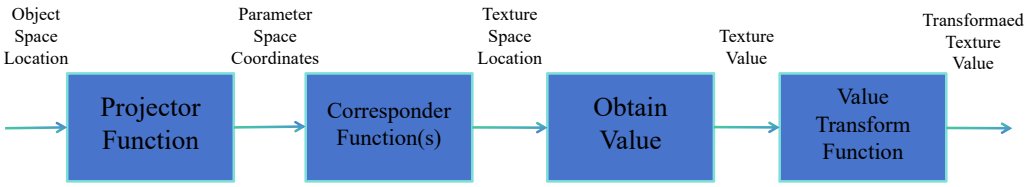



**Figure 11.** Texture mapping process.

- **Step one**. A set of parameter space values are obtained by applying the projector function to points in space, transforming 3D points into texture coordinates. The relationship between the points in the world coordinate system and the points in the pixel coordinate system are shown as follows:

$$\begin{bmatrix} \boldsymbol{P}_{uv} \\ 1 \end{bmatrix} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{z} \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \frac{1}{z} \boldsymbol{K} \boldsymbol{P}_c. \tag{24}$$

- **Step two**. Before accessing the texture with these new values, corresponding functions convert the parameter space values to the texture space. The image appears at position $(u, v)$ on the object's surface with $uv$ values in the normal range of [0,1). Textures outside this range are displayed according to the corresponder function.
- **Step Three**. These texture space locations are used to obtain the corresponding color values from the texture. The built-in functions bilinear and trilinear interpolation sampling are used to map spatial points between the UV space points.
- **Step Four**. The value transform function is used to transform the retrieved results, and finally the new values $L_d$ are used to change the surface properties $k_d$, such as material or coloring normals, and so on.

$$L_d = k_d \times \frac{I}{2} \times \max(0, \vec{l} \cdot \vec{n}), \tag{25}$$

where the vector $n$ represents the normals at the coloring points. Changes in vertex normals alter the coloring results, creating different shades that provide a sense of depth and texture. With the vertices' original positions unchanged, altered vertex normals are used to generate artificial shading effects, enhancing the model's realism.

## 4. Result and Discussion

In this section, Section 4.1 encompasses feature point detection, feature point matching, feature point triangulation, and point cloud generation, using the structure-from-motion algorithm to obtain the point cloud. Section 4.2 presents the surface reconstruction results based on the point cloud data, producing a mesh model composed of triangular facets. Section 4.3 demonstrates the results of the deep-learning-based texture mapping enhancement method. In Section 4.4, the experimental results of the reconstruction model, which integrated texture-enhanced mapping with the mesh model, are exhibited and compared with models generated using the LiDAR-based method. The experiments were conducted in a 64-bit Windows environment , using the Pycharm compilation environment and the Computer Vision toolbox in MATLAB2024a.

### *4.1. SFM-Based Point Cloud Data Acquisition*

#### 4.1.1. Feature Point Detection

The positional distribution of feature points obtained using SIFT, SURF, and KAZE descriptors is shown in Figure 12. It is evident that the KAZE feature descriptor detected significantly more feature points compared to SIFT and SURF. The dense distribution of KAZE feature points across all datasets suggests that KAZE was heavily influenced by environmental geometric features rather than the target object. Conversely, the SIFT feature points effectively detected the geometric edges of the target object across all six dataset types, with fewer points, reducing the likelihood of duality in the geometric information representation.

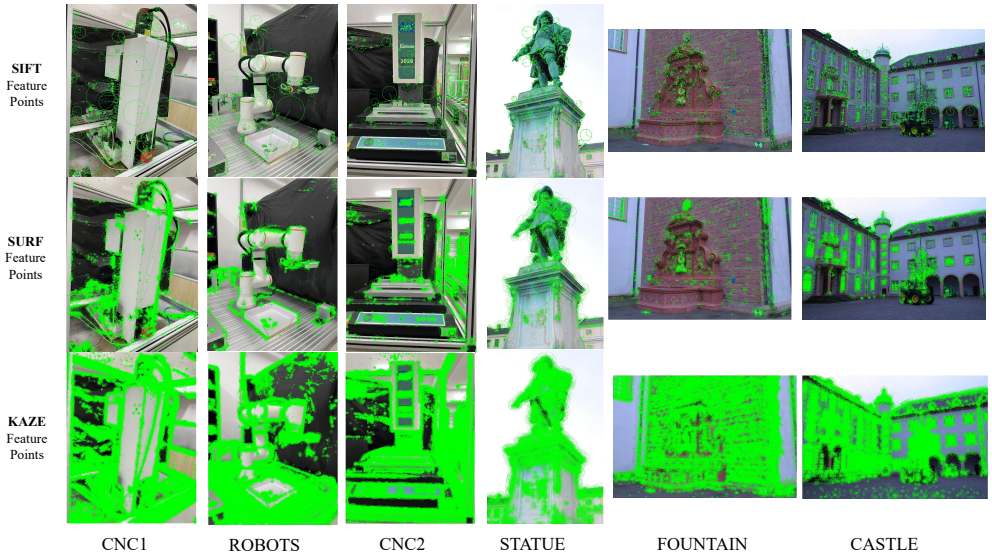



**Figure 12.** Demonstration of the three kinds of feature descriptors used on Dataset 1 and Dataset 2.

#### 4.1.2. Feature Point Matching

The feature points obtained from the three image descriptors were coarsely matched through a K-nearest neighbor search (KNN), and the number of matched pairs and the matching rate were counted according to the spatial distance of the feature points as the filtering condition, as shown in Tables 3–5.

Tables 3–5 illustrate that, as the threshold increased, both the matching rate and the number of matched point pairs rose. However, the above tables show that for the same target object, there was no direct connection between the number of feature points and the point matching rate, which also means that more feature points will not have a direct positive impact on 3D reconstruction. Among the three feature point detection methods, as shown in Figure 13, the SIFT descriptor detected the smallest number of feature points, but its point-matching rate was the highest. Notably, the SIFT-detected feature points were more sensitive to matching threshold variation.

**Table 3.** Statistics of SIFT feature point matching.

| **Threshold** | **0.65** | | **0.70** | | **0.75** | | **0.80** | | **0.85** | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Rate** | **Quantity** | **Rate** | **Quantity** | **Rate** | **Quantity** | **Rate** | **Quantity** | **Rate** | **Quantity** |
| CNC1 | 0.0596 | 95 | 0.0734 | 117 | 0.0891 | 142 | 0.1350 | 215 | 0.1965 | 313 |
| CNC2 | 0.0519 | 93 | 0.0664 | 119 | 0.0859 | 154 | 0.1233 | 221 | 0.2015 | 361 |
| ROBOTS | 0.0109 | 16 | 0.0157 | 23 | 0.0321 | 47 | 0.0560 | 82 | 0.1051 | 154 |
| STATUE | 0.0548 | 91 | 0.0759 | 126 | 0.1048 | 174 | 0.1452 | 241 | 0.2133 | 325 |
| FOUNTAIN | 0.1247 | 328 | 0.1433 | 377 | 0.1699 | 447 | 0.1984 | 522 | 0.2535 | 667 |
| CASTLE | 0.1777 | 513 | 0.2040 | 589 | 0.2442 | 705 | 0.2910 | 841 | 0.3543 | 1023 |

**Table 4.** Statistics of SURF feature point matching.

| **Threshold** | **0.65** | | **0.70** | | **0.75** | | **0.80** | | **0.85** | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Rate** | **Quantity** | **Rate** | **Quantity** | **Rate** | **Quantity** | **Rate** | **Quantity** | **Rate** | **Quantity** |
| CNC1 | 0.0902 | 326 | 0.0999 | 361 | 0.1151 | 416 | 0.1300 | 470 | 0.1494 | 540 |
| CNC2 | 0.0506 | 347 | 0.0573 | 393 | 0.0642 | 440 | 0.0713 | 489 | 0.0791 | 542 |
| ROBOTS | 0.0288 | 93 | 0.0349 | 113 | 0.0470 | 152 | 0.0591 | 191 | 0.0724 | 234 |
| STATUE | 0.0617 | 103 | 0.0707 | 118 | 0.0779 | 130 | 0.0845 | 141 | 0.0911 | 152 |
| FOUNTAIN | 0.1173 | 160 | 0.1254 | 171 | 0.1334 | 182 | 0.1400 | 191 | 0.1481 | 202 |
| CASTLE | 0.1696 | 573 | 0.1835 | 620 | 0.1995 | 674 | 0.2175 | 735 | 0.2362 | 798 |

**Table 5.** Statistics of KAZE feature point matching.

| Threshold | 0.65 | | 0.70 | | 0.75 | | 0.80 | | 0.85 | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Rate | Quantity | Rate | Quantity | Rate | Quantity | Rate | Quantity | Rate | Quantity |
| CNC1 | 0.0552 | 2470 | 0.0652 | 2918 | 0.0764 | 3418 | 0.0890 | 3981 | 0.1030 | 4609 |
| CNC2 | 0.0367 | 2151 | 0.0432 | 2543 | 0.0510 | 2993 | 0.0595 | 3492 | 0.0697 | 4090 |
| ROBOTS | 0.0225 | 1185 | 0.0298 | 1566 | 0.0380 | 1996 | 0.0485 | 2549 | 0.0604 | 3177 |
| STATUE | 0.0533 | 738 | 0.0599 | 800 | 0.0652 | 870 | 0.0710 | 947 | 0.0767 | 1024 |
| FOUNTAIN | 0.1685 | 5716 | 0.1712 | 5806 | 0.1736 | 5889 | 0.1759 | 5967 | 0.1784 | 6051 |
| CASTLE | 0.1866 | 5174 | 0.1947 | 5379 | 0.2020 | 5601 | 0.2100 | 5823 | 0.2170 | 6017 |

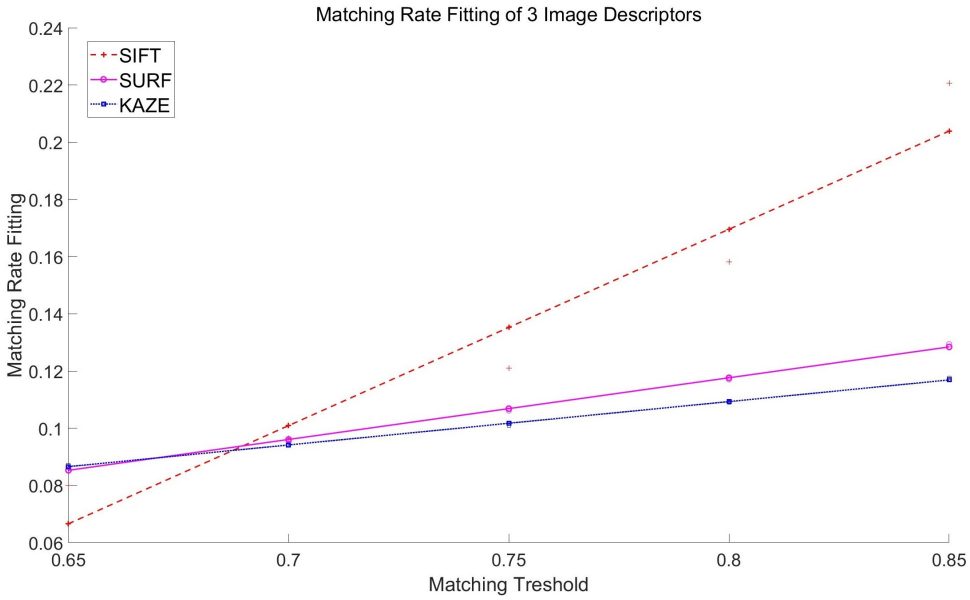



**Figure 13.** Matching rate fitting of three kinds of image descriptors.

As shown in the above figure, the feature points obtained by the SIFT descriptor achieved a higher matching rate compared to the SURF and KAZE descriptors across all datasets. This suggests that matching the feature points obtained by KAZE and SURF was more challenging, and these points did not cluster well around key geometric elements. Thus, SIFT proved to be the best feature detection and extraction method for the datasets in this paper. The results of feature point extraction using SIFT and matching with KNN are shown in Figures 14 and 15 as examples.

In Figures 14 and 15, the shooting angles of the two images used for matching are similar, so, ideally, each pair of feature point lines should be nearly parallel. However, as the matching threshold decreased, the cross confusion of the feature point connecting lines improved, proving that increasing the matching threshold can help reduce the occurrence of false matches. It is evident that Dataset 1 (building class), shown in Figure 15, had significantly fewer point-matching line crossings under the same matching threshold conditions compared to Dataset 2 (industrial equipment class), shown in Figure 14. This difference can be attributed to the fact that the building class dataset has more surface texture features, whereas the industrial equipment class dataset lacks surface texture.

In Figure 16b, the blue box represents the projection of the left image plane onto the right image plane, visualizing the homography matrix between the camera planes. It can be observed that false matches were corrected by RANSAC, and each pair of feature points is correctly positioned in the images.

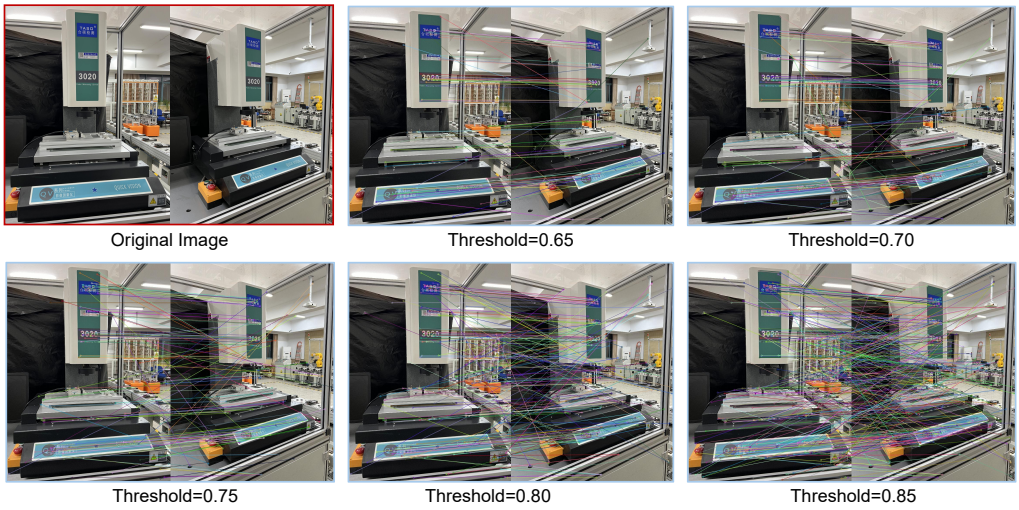



**Figure 14.** SIFT point matching for *CNC1* object under different thresholds.

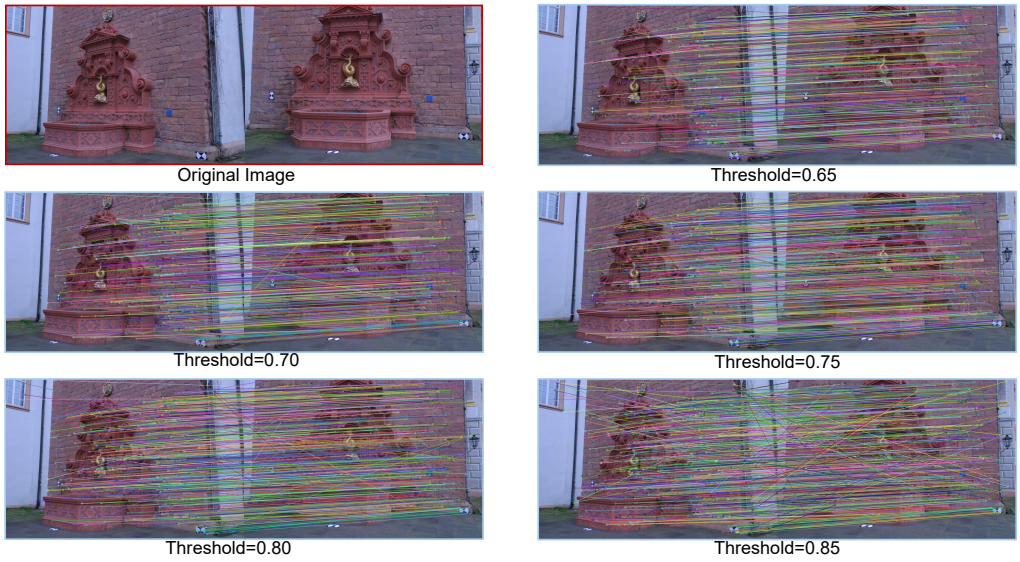
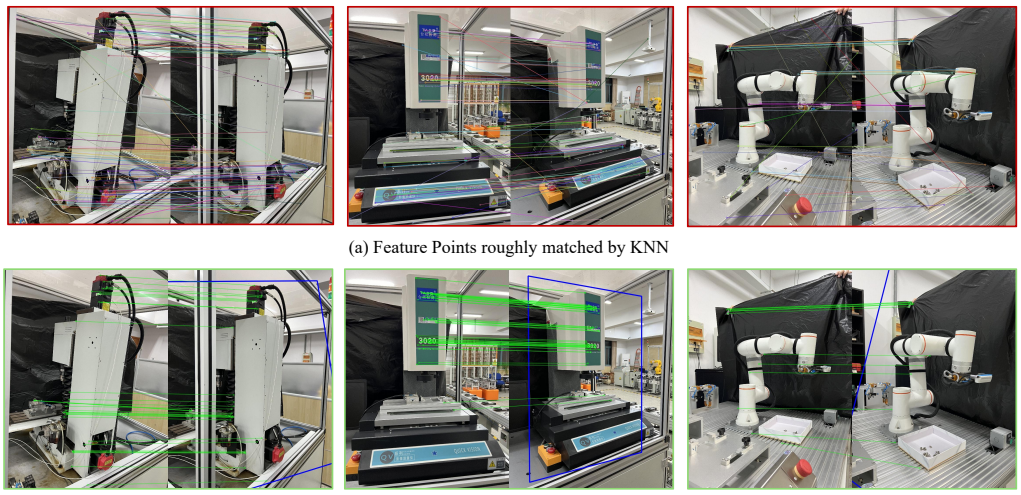



**Figure 15.** SIFT point matching for *Fountain* object under different thresholds.



(a) Feature Points roughly matched by KNN

(b) Feature Points precisely matched by RANSAC

**Figure 16.** Matching result of Dataset 2 using RANSAC method.

#### 4.1.3. Triangulation of Feature Points

The results of the matching points triangulation, from solving the homography matrix to obtain a camera pose matrix, are shown in Figures 17 and 18.

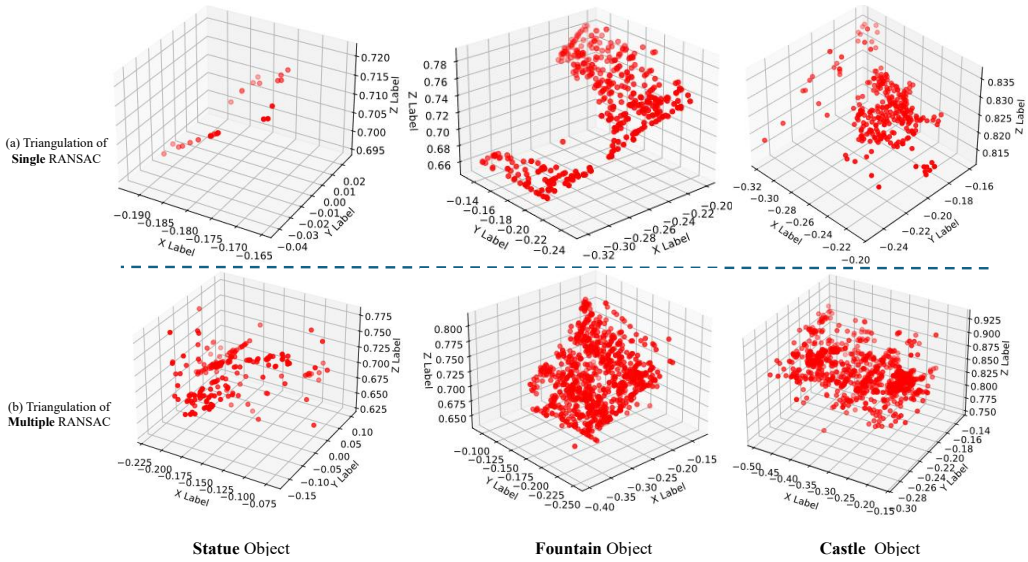



**Figure 17.** Triangulation presentation of feature points obtained from objects in Dataset 1.

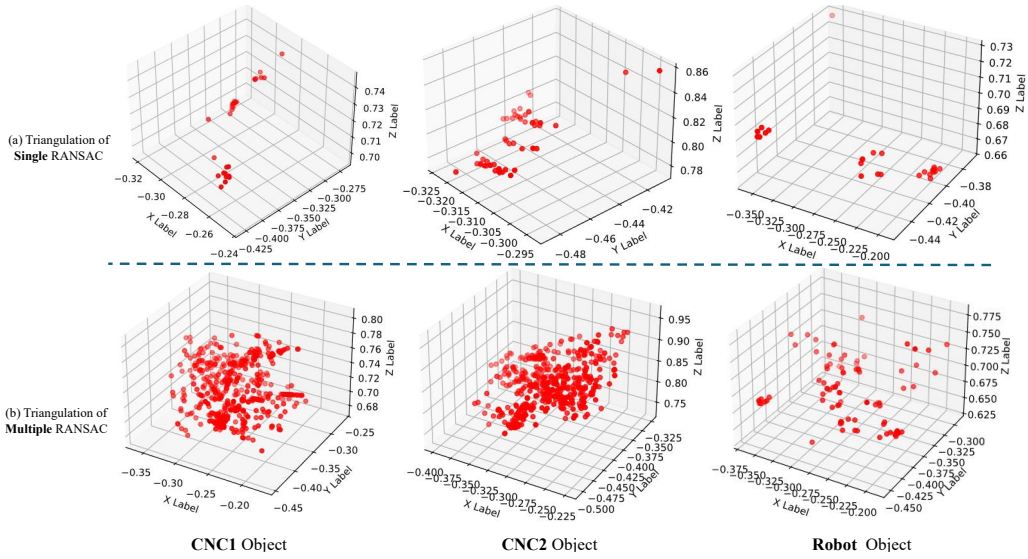
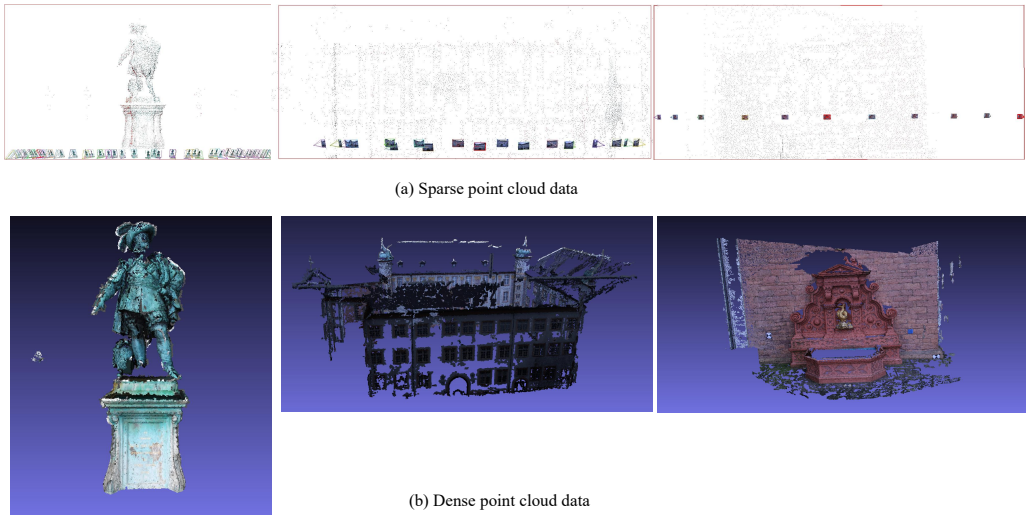



**Figure 18.** Triangulation presentation of feature points obtained from objects in Dataset 2.

The points in Figures 17b and 18b are denser than those in Figures 17a and 18a suggesting that the spatial points obtained through multiple RANSAC matches were significantly more than those from a single RANSAC match. The inherent randomness of the RANSAC algorithm means that multiple iterations increased the matching range of feature points.

#### 4.1.4. Point Cloud Data

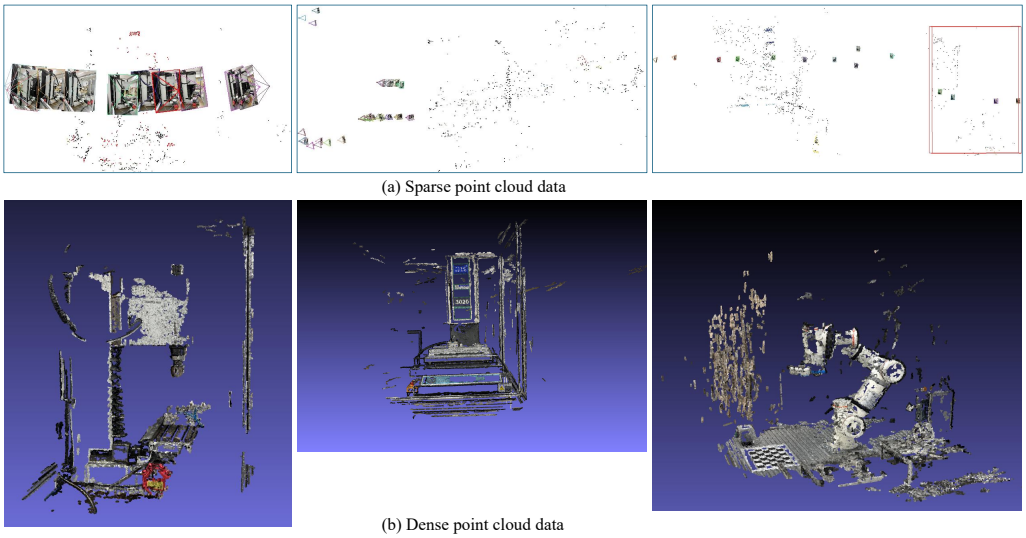
A comparison of Figures 19 and 20 shows that the reconstruction result of Dataset 1 was superior to that of Dataset 2. This difference is attributed to Dataset 1's composition of large buildings, which offer richer surface textures, more detectable feature points, and more extractable geometric information. In contrast, the objects in Dataset 2 have simple geometric contours and larger untextured planes, making it difficult to detect feature points at the center of these planes.



(a) Sparse point cloud data

(b) Dense point cloud data

**Figure 19.** Point cloud data of objects in Dataset 1.



(a) Sparse point cloud data

(b) Dense point cloud data

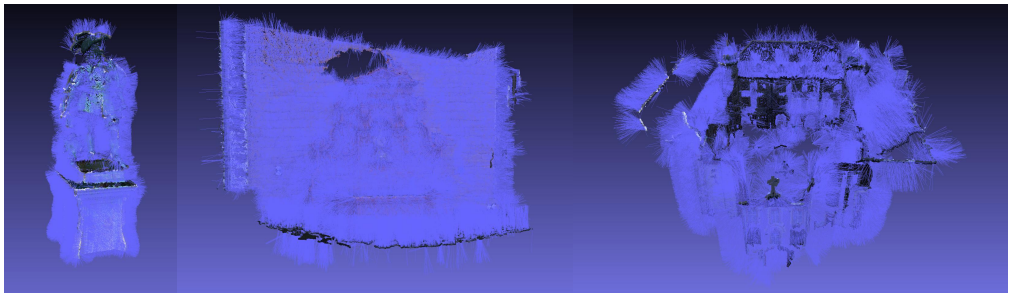
**Figure 20.** Point cloud data of objects in Dataset 2.

### *4.2. Surface Reconstruction*

The results of the normal vector solution for Poisson surface reconstruction are shown below.

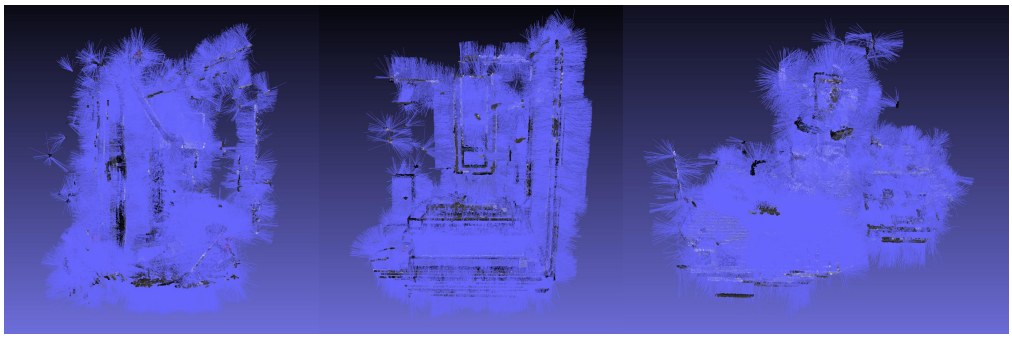
The results in Figures 21 and 22 indicate that the normal vectors were uniformly distributed across the entire point cloud, with no significant outliers or anomalies. In Dataset 2, which primarily consists of flat surfaces, the normal vectors were consistent in the smooth regions. In Dataset 1, composed of complex geometric elements, the normal vectors exhibited smooth transitions in areas with more edges and curvature. The lengths of all normal vectors were consistent across all distribution plots for each point cloud set. The computed normal vectors were accurate in terms of directional consistency, distribution pattern, and length uniformity.

As shown in Figures 23 and 24, the visual reconstruction effect of Dataset 1 (architecture) was superior to Dataset 2 (workshop equipment). In Figure 23a, the statue's head has an extraneous tubular extension due to an upward acquisition angle, resulting in a lack of top data and an incomplete reconstructed surface. In Figure 24b, the surface transition of the small pool in the fountain's center is uneven, failing to restore the real water surface. This is due to the acquisition angle causing the water surface to almost overlap from the proximal to distal ends, making it difficult to extract geometric profile information, leading to significant reconstruction errors.



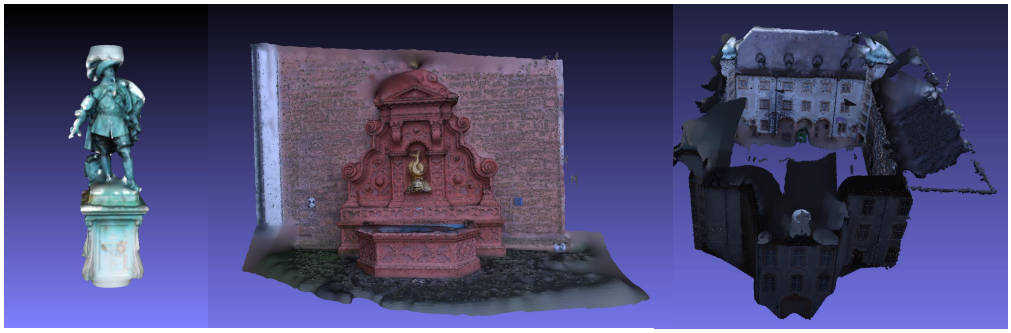
(a) Statue (b) Fountain (c) Castle

**Figure 21.** Normal vector presentation of the points set obtained from objects in Dataset 1.



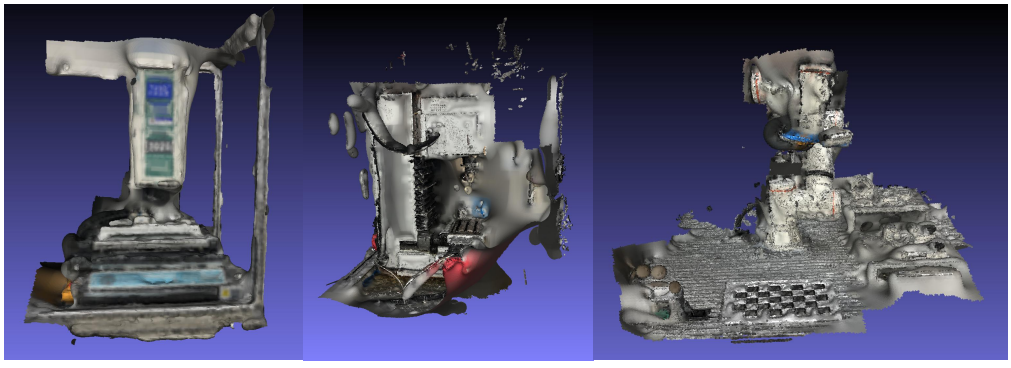
(a) CNC1 (b) CNC2 (c) Robot

**Figure 22.** Normal vector of the points set obtained from objects in Dataset 2.



(a) Statue (b) Fountain (c) Castle

**Figure 23.** Poisson surface reconstruction results of objects in Dataset 1.



(a) CNC1 (b) CNC2 (c) Robot

**Figure 24.** Poisson surface reconstruction results of objects in Dataset 2.

### *4.3. Texture Enhancement*

The style transfer results are shown in Figures 25–30.

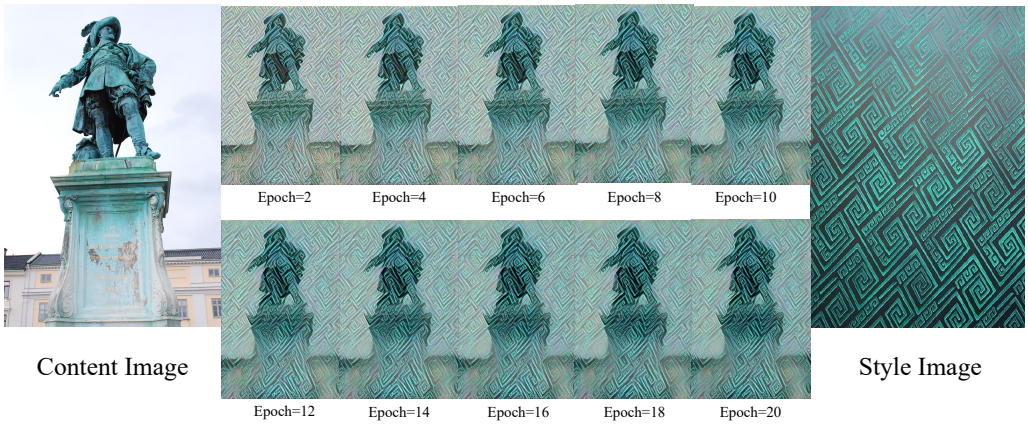



**Figure 25.** Style transfer result of *Statue* object.

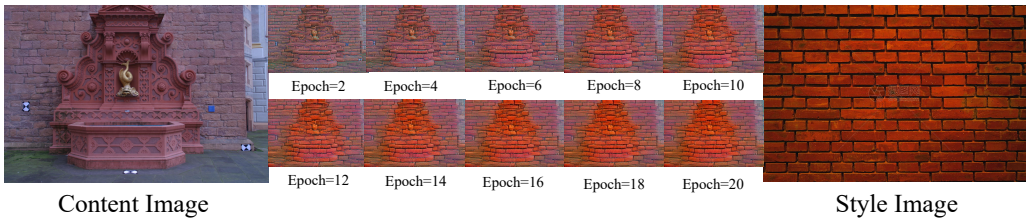



**Figure 26.** Style transfer result of *Fountain* object.

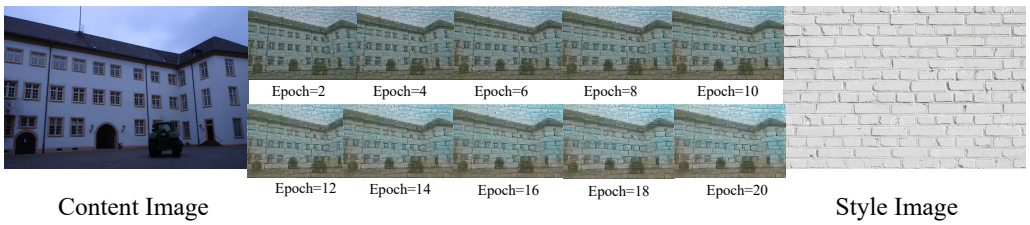



**Figure 27.** Style transfer result of *Castle* object.

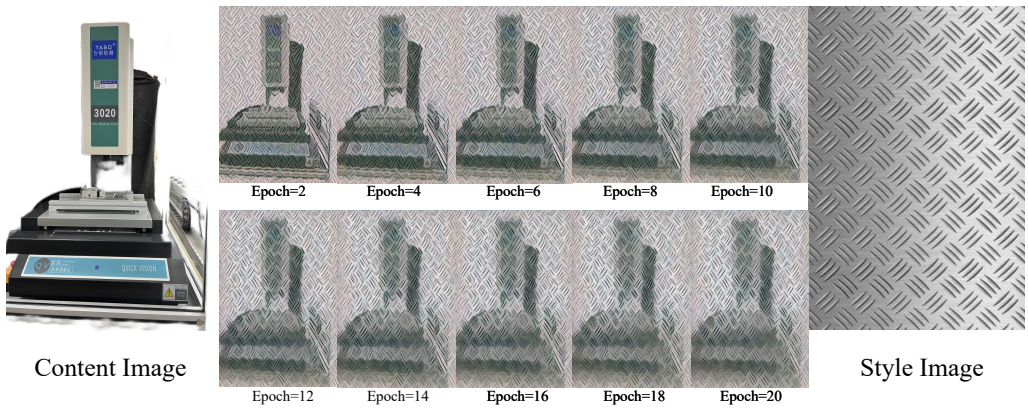



**Figure 28.** Style transfer result of *CNC1* object.

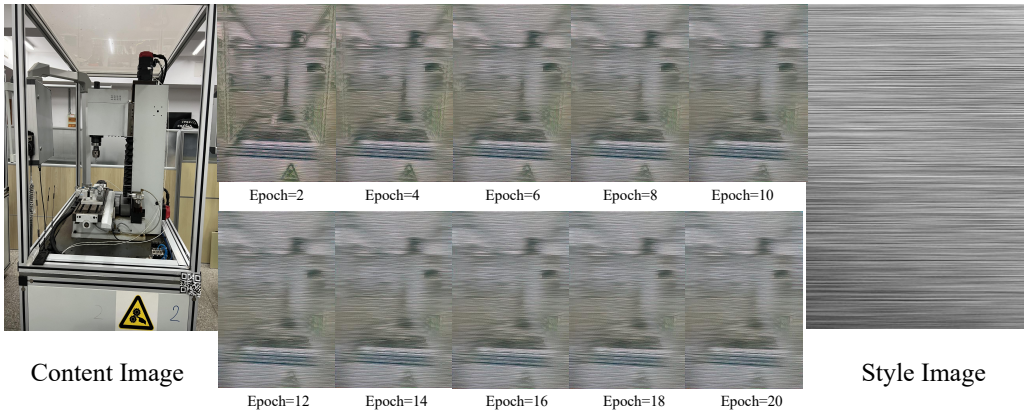



**Figure 29.** Style transfer result of *CNC2* object.

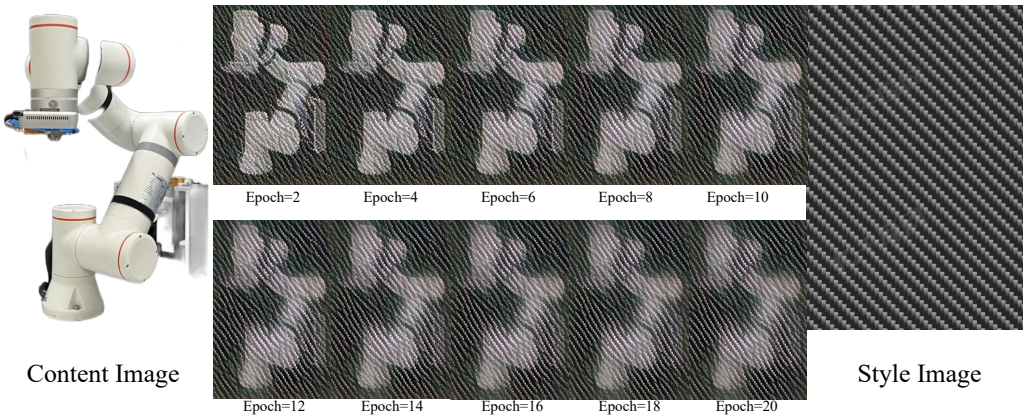



**Figure 30.** Style transfer result of *Robot* object.

In the figures above, the number of training epochs increased from 0 to 20 (iterations from 0 to 2000), causing the edges in the generated image to gradually disintegrate and form local contours consistent with the style image. This indicates a deepening fusion of content and style with more training epochs. Comparing Figures 25–30, when the texture features in stylized images exhibit recurring complex structures and these structures are relatively large in scale, the stylization process is more perceptible to the human eye. In contrast, the scale ratios of the stainless steel texture and carbon fiber texture in Figures 29 and 30 are relatively small, resulting in minimal visual differences in the stylization results from epochs 6 to 20. Additionally, the stainless steel texture in Figure 29 mainly varies in the lateral direction, leading to blurred lateral features and an unclear preservation of contours in the content images.

The loss function during the training process exhibited a consistent pattern of variation across the six categories of objects. The training loss of the experiment carried on the CNC1 object, as an example, is shown in Figure 31.

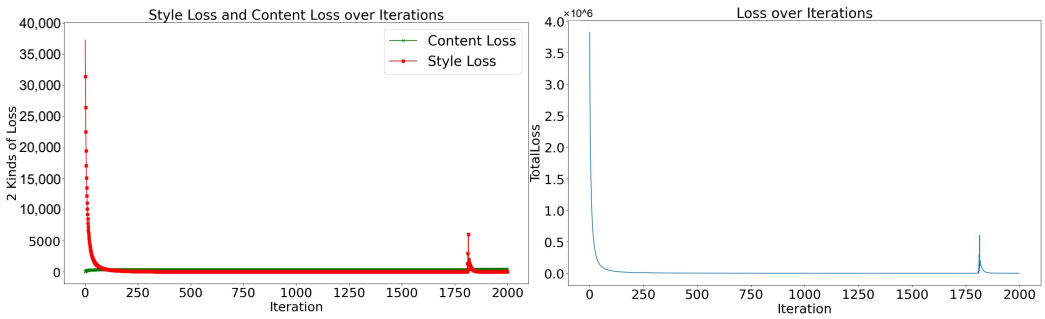



**Figure 31.** Training loss in style transfer for **CNC1** object.

The subfigure on the left in Figure 31 depicts the style and content loss over the iterations, with the red line representing style loss and the green line content loss. The content loss did not increase with more training rounds, indicating that the geometric contours of objects in the generated images and content images remained similar as perceived by the convolutional neural network, preserving the details of the content image. The style loss decreased rapidly within the first five training rounds, indicating a significant reduction in texture disparity between the generated and style images, suggesting the model quickly learned and integrated the style image's texture.

The subfigure on the right in Figure 31 demonstrates the total loss over the iterative process, where total loss is a weighted sum of style and content loss. As detailed in Table 2, the style loss weight was substantially higher than that of the content loss. Consequently, the total loss primarily followed the trend of style loss, gradually decreasing with additional training epochs.

The image quality assessments for these six datasets exhibit a uniform pattern, with CNC1 serving as an example, as shown in Figure 32 The result for each epoch was evaluated using both PSNR and SSIM methods, with the content image as the reference image.

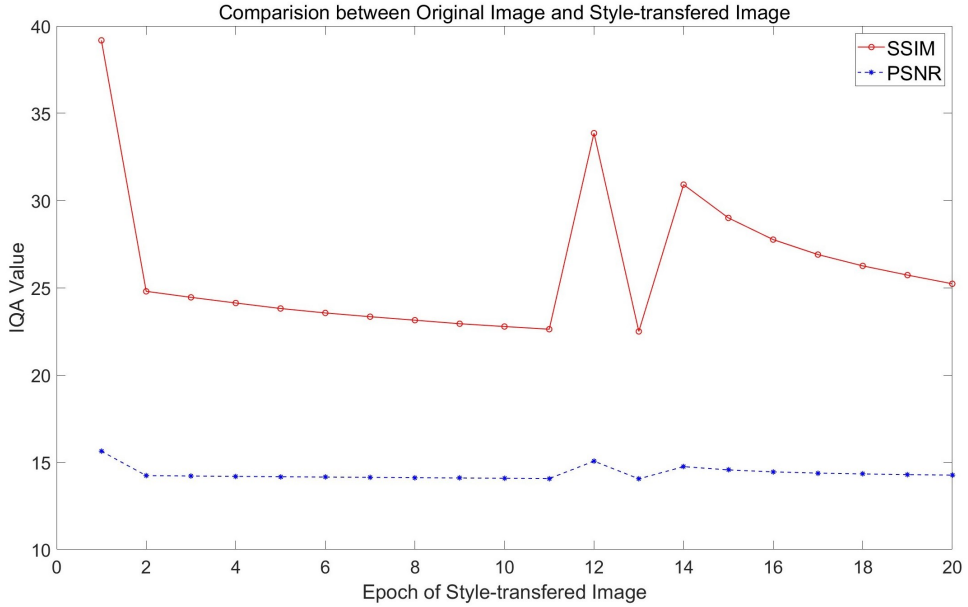
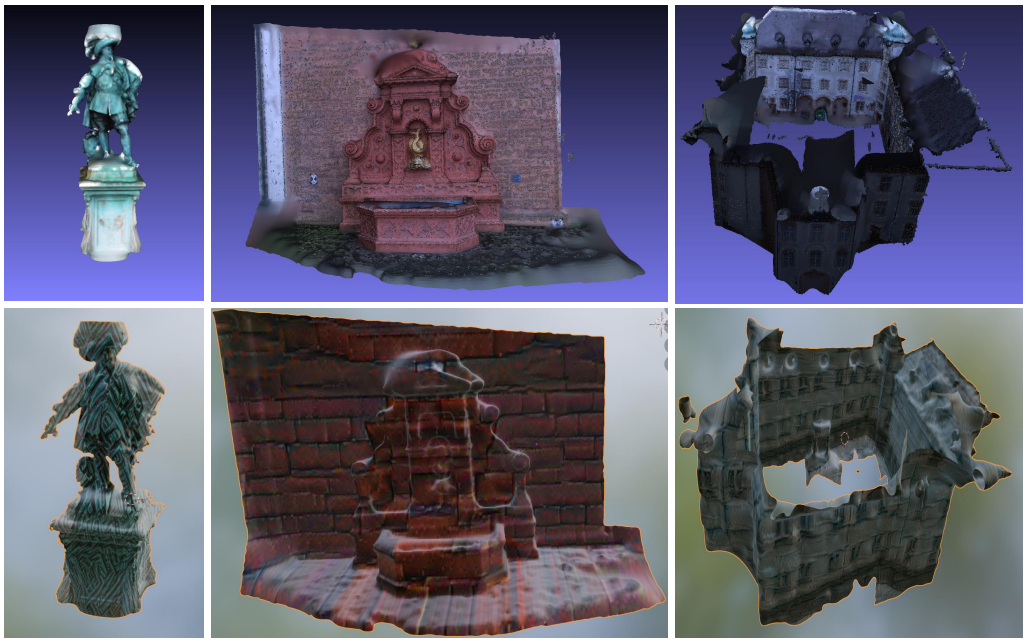



**Figure 32.** IQA assessment for **CNC1** images after style transfer.

The SSIM and PSNR indices generally decreased with increasing training epochs, as these indices evaluate the similarity between the images generated by style migration and the content images. The image quality evaluation of the six objects exhibited an interesting phenomenon at the 12th training epoch, where both indices first increased and then rapidly decreased. However, this change is not perceptible to the human eye, as the results from the 12th epoch are almost indistinguishable from those of the 10th and 14th epochs in Figures 25–30. Because of the poor interpretability of CNN models, this interesting change is hard to convincingly explain and avoid. Nonetheless, as style transfer experiments prioritize the visualization of results, this phenomenon did not affect the application of the style transfer.

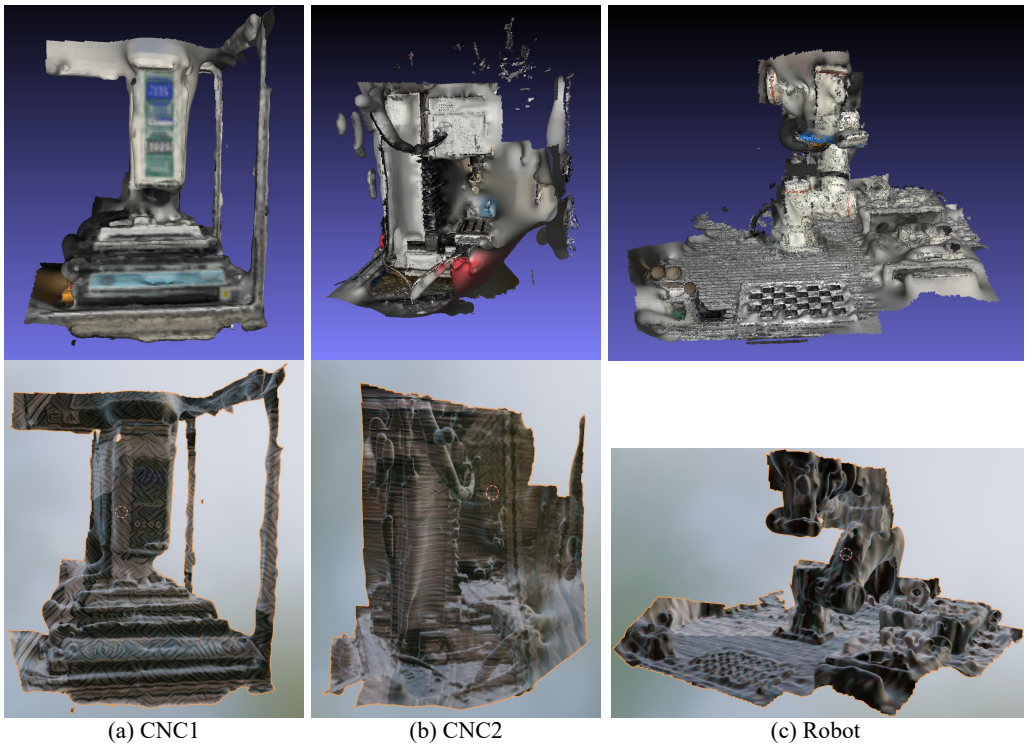
### *4.4. Texture Mapping*

The results of texture mapping are shown in Figures 33 and 34, whose upper parts show the surface reconstruction results, with the lower part displaying the effect after texture mapping.



(a) Statue (b) Fountain (c) Castle

**Figure 33.** Results of texture mapping for Dataset 1.



(a) CNC1 (b) CNC2 (c) Robot

**Figure 34.** Results of texture mapping for Dataset 2.

As shown in Figure 33, the enhanced texture mapping exhibited a precise alignment with the reconstructed surface, markedly improving the outcomes, such as covering the highlights on the *Statue* model and enhancing the metal texture. The *Fountain* object's water surface holes are obscured by the red brick texture, while the integration of the white brick texture with the *Castle* model offers a novel visual impact. The application of texture mapping to Dataset 1 was deemed successful, with an accurate spatial alignment and an improvement in model defects, thus enhancing the overall visual effect.

Figure 34 shows that the *CNC1* in Dataset 2 has optimal mapping. Its geometric edges align closely with the model's contours, respecting *CNC1*'s geometric constraints, without offset or contour loss. For the *Robot* objects, the complex spatial contours, with the main lines on various planes, resulted in texture maps that struggle to perfectly match the device's edges to the model's contours.

The comparisons in the Table 6 and 7 show that the method proposed in this paper outperformed the LiDAR solution by 20% in terms of both point cloud quantity and the number of generated mesh models, while the hardware cost was only 1% of that of the LiDAR solution. However, the modeling time consumption of the proposed method was slightly higher than that of the LiDAR solution.

**Table 6.** Statistics of surface reconstruction on Dataset 2 using LiDAR-based method [1].

| | Vertices | Faces | Time Consumption | Equipment | Equipment Cost |
|---|---|---|---|---|---|
| Robot | 95,556 | 115,736 | 29 min | Leica RTC 360 | 92,800 GBP |
| CNC1 | 175,175 | 204,458 | 40 min | Leica RTC 360 | 92,800 GBP |
| CNC2 | 195,788 | 210,396 | 34 min | Leica RTC 360 | 92,800 GBP |

[1] The solution based on LiDAR involved the use of LiDAR for point cloud acquisition, point cloud filtering, and Poisson reconstruction.

**Table 7.** Statistics of surface reconstruction on Dataset 2 using proposed method.

| | Vertices | Faces | Time Consumption | Equipment | Equipment Cost |
|---|---|---|---|---|---|
| Robot | 172,875 | 145,702 | 33 min | Iphone 13 pro | 949 GBP |
| CNC1 | 216,893 | 234,135 | 37 min | Iphone 13 pro | 949 GBP |
| CNC2 | 221,040 | 200,205 | 38 min | Iphone 13 pro | 949 GBP |

## 5. Conclusions and Future Work

### *5.1. Conclusions*

The paper has presented a cost-effective and rapid 3D modeling technique that leverages point cloud acquisition, surface reconstruction, and texture processing to create visually striking models from mobile or DSLR images, with a demonstrated high visualization performance on architectural datasets, accurately capturing geometric details and constraints without requiring specialized hardware.

This study advances 3D reconstruction by indicating SIFT as a superior feature descriptor for enhancing feature point matching and SfM efficiency. Furthermore, the application of deep learning for texture enhancement in model mapping is identified as a promising area for future research, with the provided data offering guidance on selecting optimal textures for style image enhancement.

### *5.2. Future Work*

Although this study yielded promising outcomes, it was not without limitations. Notably, the initial phase of the systematic modeling approach, particularly the point cloud 3D reconstruction, could benefit from optimization to reduce environmental clutter and enhance geometric details, as observed in Dataset 2. Additionally, the final texture mapping stage, which involves direct application of images to the model's surface, may lead to alignment issues that are challenging to rectify manually.

Future research should aim to refine the point cloud reconstruction and the visual fidelity of the proposed modeling system. This could be accomplished by leveraging deep learning to confine feature points to an object's geometric contours and to optimize their distribution for uniformity, thereby improving the reconstruction quality. Moreover, integrating depth information from the dataset and aligning specific geometric features to precise spatial locations on the model could effectively address mapping discrepancies.

**Author Contributions:** Conceptualization, B.Y. and B.H.S.A.; methodology, B.Y. and B.H.S.A.; resources, B.Y.; writing—original draft preparation, B.H.S.A. and B.Y.; writing—review and editing, B.H.S.A. and B.Y.; supervision, B.H.S.A.; visualization, B.H.S.A. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Data Availability Statement:** The data presented in this study are available on request from the corresponding author.

**Acknowledgments:** We extend our sincere gratitude to the group led by Haihua Zhu at Nanjing University of Aeronautics and Astronautics for their support in constructing the dataset. We also wish to thank Yongjie Xu at Cranfield University, for his assistance in acquiring images for the private dataset.

**Conflicts of Interest:** The authors declare no conflicts of interest.

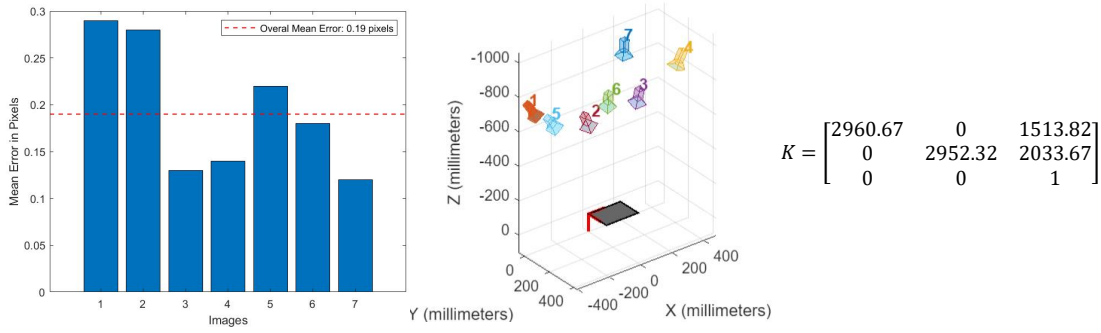
## Abbreviations

The following abbreviations are used in this paper:

| | |
|---|---|
| LiDAR | Laser Scanning/Light Detection and Ranging |
| TOF | Time of Flight |
| ALS | Airborne Laser Scanning |
| MLS | Mobile Laser Scanning |
| TLS | Terrestrial Laser Scanning |
| SFM | Structure from Motion |
| MVS | Multi-View Stereo Vision |
| BA | Bundle Adjustment |
| CNN | Convolutional Neural Network |
| D-CV | Depth-based Cost Volume |
| P-CV | Pose-based Cost Volume |
| CVP-DC | Cost Voxel Pyramid Depth Completion |
| SDF | Signed Distance Function |
| MLP | Multi-Layer Perceptrons |
| EC-Net | Edge-aware Network |
| ICP | Iterative Closest Point |
| IMQ | Inverse Multiquadric |
| SDF | Signed Distance Function |
| GAN | Generative Adversarial Neural Network |
| NeRF | Neural Radiance Fields |
| SIFT | Scale-Invariant Feature Transform |
| DoG | Difference of Gaussians |
| KAZE | KAZE Features |
| SURF | Speeded-Up Robust Features |
| KNN | K-Nearest Neighbor search |
| NPSAC | NAdjacent Points Sample Consensus |
| PROSAC | Progressive Sample Consensus |
| RANSAC | Random Sample Consensus |
| VGG | Visual Geometry Group |
| PSNR | Peak Signal-to-Noise Ratio |
| SSIM | Structural Similarity Index Measure |

## Appendix A. Camera Calibration

The SFM algorithm relies on the intrinsic parameters of the camera, and Dataset 2 is obtained from the iPhone 13 Pro. The camera's intrinsic parameter matrix is calibrated by *ZhangZhengyou* calibration method [63]. From the calibration results in the following Figure A1, it can be seen that the average reprojection error is less than 0.5, indicating that the camera intrinsic matrix obtained from the solution has a high confidence level.




$$K = \begin{bmatrix} 2960.67 & 0 & 1513.82 \\ 0 & 2952.32 & 2033.67 \\ 0 & 0 & 1 \end{bmatrix}$$

(a) Mean Reprojected Error (b) Resolved Position of Cameras (c) Camera Intrinsics Matrix

**Figure A1.** Results of Camera calibration.

## References


1. Tao, F.; Xiao, B.; Qi, Q.; Cheng, J.; Ji, P. Digital twin modeling. *J. Manuf. Syst.* **2022**, *64*, 372–389. [CrossRef]
2. Gong, H.; Su, D.; Zeng, S.; Chen, X. Advancements in digital twin modeling for underground spaces and lightweight geometric modeling technologies. *Autom. Constr.* **2024**, *165*, 105578. [CrossRef]
3. Wu, H.; Ji, P.; Ma, H.; Xing, L. A comprehensive review of digital twin from the perspective of total process: Data, models, networks and applications. *Sensors* **2023**, *23*, 8306. [CrossRef] [PubMed]
4. Elaksher, A.; Ali, T.; Alharthy, A. A quantitative assessment of LiDAR data accuracy. *Remote Sens.* **2023**, *15*, 442. [CrossRef]
5. Piedra-Cascón, W.; Meyer, M.J.; Methani, M.M.; Revilla-León, M. Accuracy (trueness and precision) of a dual-structured light facial scanner and interexaminer reliability. *J. Prosthet. Dent.* **2020**, *124*, 567–574. [CrossRef] [PubMed]
6. Frangez, V.; Salido-Monzú, D.; Wieser, A. Assessment and improvement of distance measurement accuracy for time-of-flight cameras. *IEEE Trans. Instrum. Meas.* **2022**, *71*, 1003511. [CrossRef]
7. Bi, S.; Gu, Y.; Zou, J.; Wang, L.; Zhai, C.; Gong, M. High precision optical tracking system based on near infrared trinocular stereo vision. *Sensors* **2021**, *21*, 2528. [CrossRef]
8. Wang, Y.; Funk, N.; Ramezani, M.; Papatheodorou, S.; Popović, M.; Camurri, M.; Leutenegger, S.; Fallon, M. Elastic and efficient LiDAR reconstruction for large-scale exploration tasks. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; IEEE: New York, NY, USA, 2021; pp. 5035–5041.
9. Zhang, J.; Zhang, F.; Kuang, S.; Zhang, L. Nerf-lidar: Generating realistic lidar point clouds with neural radiance fields. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 20–27 February 2024; Volume 38, pp. 7178–7186.
10. Wang, Z. Review of real-time three-dimensional shape measurement techniques. *Measurement* **2020**, *156*, 107624. [CrossRef]
11. Wang, Z.; Zhou, Q.; Shuang, Y. Three-dimensional reconstruction with single-shot structured light dot pattern and analytic solutions. *Measurement* **2020**, *151*, 107114. [CrossRef]
12. Liu, H.; Cao, C.; Ye, H.; Cui, H.; Gao, W.; Wang, X.; Shen, S. Lightweight Structured Line Map Based Visual Localization. *IEEE Robot. Autom. Lett.* **2024**, *9*, 5182–5189. [CrossRef]
13. Cao, D.; Liu, W.; Liu, S.; Chen, J.; Liu, W.; Ge, J.; Deng, Z. Simultaneous calibration of hand-eye and kinematics for industrial robot using line-structured light sensor. *Measurement* **2023**, *221*, 113508. [CrossRef]
14. Liang, Z.; Chang, H.; Wang, Q.; Wang, D.; Zhang, Y. 3D reconstruction of weld pool surface in pulsed GMAW by passive biprism stereo vision. *IEEE Robot. Autom. Lett.* **2019**, *4*, 3091–3097. [CrossRef]
15. Li, Y.; Wang, Z. RGB line pattern-based stereo vision matching for single-shot 3-D measurement. *IEEE Trans. Instrum. Meas.* **2020**, *70*, 5004413. [CrossRef]
16. Jing, J.; Li, J.; Xiong, P.; Liu, J.; Liu, S.; Guo, Y.; Deng, X.; Xu, M.; Jiang, L.; Sigal, L. Uncertainty guided adaptive warping for robust and efficient stereo matching. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 2–6 October 2023; pp. 3318–3327.
17. Hu, Y.; Chen, Q.; Feng, S.; Tao, T.; Asundi, A.; Zuo, C. A new microscopic telecentric stereo vision system-calibration, rectification, and three-dimensional reconstruction. *Opt. Lasers Eng.* **2019**, *113*, 14–22. [CrossRef]
18. Berra, E.; Peppa, M. Advances and challenges of UAV SFM MVS photogrammetry and remote sensing: Short review. In Proceedings of the 2020 IEEE Latin American Grss & ISPRS Remote Sensing Conference (Lagirs), Santiago, Chile, 21–26 March 2020; IEEE: New York, NY, USA, 2020; pp. 533–538.
19. Gao, L.; Zhao, Y.; Han, J.; Liu, H. Research on multi-view 3D reconstruction technology based on SFM. *Sensors* **2022**, *22*, 4366. [CrossRef]
20. Wang, J.; Rupprecht, C.; Novotny, D. Posediffusion: Solving pose estimation via diffusion-aided bundle adjustment. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 2–6 October 2023; pp. 9773–9783.
21. Pan, L.; Baráth, D.; Pollefeys, M.; Schönberger, J.L. Global Structure-from-Motion Revisited. In Proceedings of the European Conference on Computer Vision (ECCV), Milan, Italy, 29 September–4 October 2024.

22. Barath, D.; Mishkin, D.; Eichhardt, I.; Shipachev, I.; Matas, J. Efficient initial pose-graph generation for global sfm. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 14546–14555.
23. Liu, S.; Jiang, S.; Liu, Y.; Xue, W.; Guo, B. Efficient SfM for Large-Scale UAV Images Based on Graph-Indexed BoW and Parallel-Constructed BA Optimization. *Remote Sens.* **2022**, *14*, 5619. [CrossRef]
24. Bond, Y.L.; Ledwell, S.; Osornio, E.; Cruz, A.C. Efficient Scene Reconstruction for Unmanned Aerial Vehicles. In Proceedings of the 2023 Fifth International Conference on Transdisciplinary AI (TransAI), Laguna Hills, CA, USA, 25–27 September 2023; IEEE: New York, NY, USA, 2023; pp. 266–269.
25. Barath, D.; Noskova, J.; Eichhardt, I.; Matas, J. Pose-graph via Adaptive Image Re-ordering. In Proceedings of the BMVC, London, UK, 21–24 November 2022; p. 127.
26. Radenović, F.; Tolias, G.; Chum, O. Fine-tuning CNN image retrieval with no human annotation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *41*, 1655–1668. [CrossRef]
27. Wei, X.; Zhang, Y.; Li, Z.; Fu, Y.; Xue, X. Deepsfm: Structure from motion via deep bundle adjustment. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 230–247.
28. Schonberger, J.L.; Frahm, J.M. Structure-from-motion revisited. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 4104–4113.
29. Li, Z.; Luo, S.; Zeng, W.; Guo, S.; Zhuo, J.; Zhou, L.; Ma, Z.; Zhang, Z. 3d reconstruction system for foot arch detecting based on openmvg and openmvs. In Proceedings of the 2022 5th International Conference on Pattern Recognition and Artificial Intelligence (PRAI), Chengdu, China, 19–21 August 2022; IEEE: New York, NY, USA, 2022; pp. 1017–1022.
30. Lyra, V.G.d.M.; Pinto, A.H.; Lima, G.C.; Lima, J.P.; Teichrieb, V.; Quintino, J.P.; da Silva, F.Q.; Santos, A.L.; Pinho, H. Development of an efficient 3D reconstruction solution from permissive open-source code. In Proceedings of the 2020 22nd Symposium on Virtual and Augmented Reality (SVR), Virtual, 7–10 November 2020; IEEE: New York, NY, USA, 2020; pp. 232–241.
31. Wu, C. Towards linear-time incremental structure from motion. In Proceedings of the 2013 International Conference on 3D Vision-3DV, Seattle, WA, USA, 29 June–1 July 2013; IEEE: New York, NY, USA, 2013; pp. 127–134.
32. Zhou, L.; Sun, G.; Li, Y.; Li, W.; Su, Z. Point cloud denoising review: From classical to deep learning-based approaches. *Graph. Model.* **2022**, *121*, 101140. [CrossRef]
33. Huang, Z.; Wen, Y.; Wang, Z.; Ren, J.; Jia, K. Surface reconstruction from point clouds: A survey and a benchmark. *IEEE Trans. Pattern Anal. Mach. Intell.* **2024**, *46*, 9727–9748. [CrossRef]
34. Azinović, D.; Martin-Brualla, R.; Goldman, D.B.; Nießner, M.; Thies, J. Neural rgb-d surface reconstruction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 6290–6301.
35. You, C.C.; Lim, S.P.; Lim, S.C.; San Tan, J.; Lee, C.K.; Khaw, Y.M.J. A survey on surface reconstruction techniques for structured and unstructured data. In Proceedings of the 2020 IEEE Conference on Open Systems (ICOS), Penang, Malaysia, 17–19 November 2020; IEEE: New York, NY, USA, 2020; pp. 37–42.
36. Wu, Y.; Hu, X.; Zhang, Y.; Gong, M.; Ma, W.; Miao, Q. SACF-Net: Skip-attention based correspondence filtering network for point cloud registration. *IEEE Trans. Circuits Syst. Video Technol.* **2023**, *33*, 3585–3595. [CrossRef]
37. Lu, D.; Lu, X.; Sun, Y.; Wang, J. Deep feature-preserving normal estimation for point cloud filtering. *Comput.-Aided Des.* **2020**, *125*, 102860. [CrossRef]
38. Zhang, S.; Cui, S.; Ding, Z. Hypergraph spectral analysis and processing in 3D point cloud. *IEEE Trans. Image Process.* **2020**, *30*, 1193–1206. [CrossRef] [PubMed]
39. Ren, D.; Ma, Z.; Chen, Y.; Peng, W.; Liu, X.; Zhang, Y.; Guo, Y. Spiking pointnet: Spiking neural networks for point clouds. *arXiv* **2024**, arXiv:2310.06232.
40. Hao, H.; Jincheng, Y.; Ling, Y.; Gengyuan, C.; Sumin, Z.; Huan, Z. An improved PointNet++ point cloud segmentation model applied to automatic measurement method of pig body size. *Comput. Electron. Agric.* **2023**, *205*, 107560. [CrossRef]
41. Hermosilla, P.; Ritschel, T.; Ropinski, T. Total denoising: Unsupervised learning of 3D point cloud cleaning. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 52–60.
42. Liu, X.Y.; Wang, H.; Chen, C.; Wang, Q.; Zhou, X.; Wang, Y. Implicit surface reconstruction with radial basis functions via PDEs. *Eng. Anal. Bound. Elem.* **2020**, *110*, 95–103. [CrossRef]
43. Dai, P.; Xu, J.; Xie, W.; Liu, X.; Wang, H.; Xu, W. High-quality surface reconstruction using gaussian surfels. In Proceedings of the ACM SIGGRAPH 2024 Conference Papers, Denver, CO, USA, 27 July–1 August 2024; pp. 1–11.
44. Comi, M.; Lin, Y.; Church, A.; Tonioni, A.; Aitchison, L.; Lepora, N.F. Touchsdf: A deepsdf approach for 3d shape reconstruction using vision-based tactile sensing. *IEEE Robot. Autom. Lett.* **2024**, *9*, 5719–5726. [CrossRef]
45. Gatys, L.; Ecker, A.; Bethge, M. Texture synthesis and the controlled generation of natural stimuli using convolutional neural networks. *arXiv* **2015**, arXiv:1505.07376.
46. Zhang, Y.; Huang, N.; Tang, F.; Huang, H.; Ma, C.; Dong, W.; Xu, C. Inversion-based style transfer with diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 10146–10156.
47. Lin, C.T.; Huang, S.W.; Wu, Y.Y.; Lai, S.H. GAN-based day-to-night image style transfer for nighttime vehicle detection. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 951–963. [CrossRef]

48. Gatys, L.A.; Ecker, A.S.; Bethge, M.; Hertzmann, A.; Shechtman, E. Controlling perceptual factors in neural style transfer. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 3985–3993.
49. Tang, H.; Liu, S.; Lin, T.; Huang, S.; Li, F.; He, D.; Wang, X. Master: Meta style transformer for controllable zero-shot and few-shot artistic style transfer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 18329–18338.
50. Zhang, C.; Xu, X.; Wang, L.; Dai, Z.; Yang, J. S2wat: Image style transfer via hierarchical vision transformer using strips window attention. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 20–27 February 2024; Volume 38, pp. 7024–7032.
51. Zhang, Z.; Sun, J.; Li, G.; Zhao, L.; Zhang, Q.; Lan, Z.; Yin, H.; Xing, W.; Lin, H.; Zuo, Z. Rethink arbitrary style transfer with transformer and contrastive learning. *Comput. Vis. Image Underst.* **2024**, *241*, 103951. [CrossRef]
52. Zhang, C.; Dai, Z.; Cao, P.; Yang, J. Edge enhanced image style transfer via transformers. In Proceedings of the 2023 ACM International Conference on Multimedia Retrieval, Thessaloniki, Greece, 12–15 June 2023; pp. 105–114.
53. Zhu, M.; He, X.; Wang, N.; Wang, X.; Gao, X. All-to-key attention for arbitrary style transfer. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 1–6 October 2023; pp. 23109–23119.
54. Mildenhall, B.; Srinivasan, P.P.; Tancik, M.; Barron, J.T.; Ramamoorthi, R.; Ng, R. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* **2021**, *65*, 99–106. [CrossRef]
55. Bi, S.; Xu, Z.; Sunkavalli, K.; Hašan, M.; Hold-Geoffroy, Y.; Kriegman, D.; Ramamoorthi, R. Deep reflectance volumes: Relightable reconstructions from multi-view photometric images. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 294–311.
56. Thies, J.; Zollhöfer, M.; Nießner, M. Deferred neural rendering: Image synthesis using neural textures. *ACM Trans. Graph.* **2019**, *38*, 1–12. [CrossRef]
57. Xiang, F.; Xu, Z.; Hasan, M.; Hold-Geoffroy, Y.; Sunkavalli, K.; Su, H. Neutex: Neural texture mapping for volumetric neural rendering. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 7119–7128.
58. Gupta, S.; Thakur, K.; Kumar, M. 2D-human face recognition using SIFT and SURF descriptors of face's feature regions. *Vis. Comput.* **2021**, *37*, 447–456. [CrossRef]
59. Abaspur Kazerouni, I.; Dooly, G.; Toal, D. Underwater image enhancement and mosaicking system based on A-KAZE feature matching. *J. Mar. Sci. Eng.* **2020**, *8*, 449. [CrossRef]
60. Yong, A.; Hong, Z. SIFT matching method based on K nearest neighbor support feature points. In Proceedings of the 2016 IEEE International Conference on Signal and Image Processing (ICSIP), Beijing, China, 13–15 August 2016; IEEE: New York, NY, USA, 2016; pp. 64–68.
61. Özyeşil, O.; Voroninski, V.; Basri, R.; Singer, A. A survey of structure from motion. *Acta Numer.* **2017**, *26*, 305–364. [CrossRef]
62. Sreeram, V.; Agathoklis, P. On the properties of Gram matrix. *IEEE Trans. Circuits Syst. I Fundam. Theory Appl.* **1994**, *41*, 234–237. [CrossRef]
63. Zhang, Z. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 1330–1334. [CrossRef]