MocapGS: Robust Online 3D Mapping Using Motion Capture and Gaussian Splatting

Theodor Kapler[a,*], Markus Hillemann[a], Robert Langendörfer[a] and Markus Ulrich[a]

[a]*Institute of Photogrammetry and Remote Sensing, Karlsruhe Institute of Technology, Englerstr. 7, Karlsruhe, 76131, Baden-Württemberg, Germany*

ARTICLE INFO

Keywords:
Online 3D Reconstruction
Motion Capture
Gaussian Splatting
Time Synchronization
Camera Calibration
Hand-eye Calibration

ABSTRACT

Image-based online 3D mapping allows scenes to be reconstructed incrementally, with new images continuously integrated to provide direct feedback on the reconstructed scene. However, many online reconstruction methods, typically SLAM-applications, rely on image-based camera pose estimation, which is prone to drift and loop-closure errors. To address this issue, we present MocapGS, an online 3D reconstruction method that directly integrates accurate camera poses obtained from a Motion Capture system directly into a 3D Gaussian Splatting-based reconstruction pipeline. By decoupling pose estimation from scene reconstruction, MocapGS leverages externally provided, metrically accurate camera poses to improve robustness and global consistency. To enable this integration, a complete workflow is developed, including temporal synchronization between the camera and the Motion Capture system, camera calibration, hand-eye calibration, and online reconstruction from sequential image-pose pairs. The proposed method is evaluated on multiple datasets and compared to methods estimating poses from image data. Experimental results show that MocapGS yields more stable camera trajectories and improved reconstruction quality compared to such methods, particularly with respect to global consistency. The results demonstrate that Motion Capture can serve as a reliable primary source of camera poses for online 3D reconstruction, highlighting the potential of Motion Capture-driven reconstruction pipelines.

1. Introduction

Image-based 3D reconstruction has become a fundamental technology in computer vision, photogrammetry, and robotics, with applications ranging from industrial inspection [21] over autonomous driving [1] to digital heritage preservation [7]. As cameras are inexpensive, flexible, and widely available, image-based approaches remain particularly attractive compared to active sensing modalities.

Classical 3D reconstruction pipelines are predominantly batch-based. Methods such as Structure from Motion (SfM) [18] estimate camera poses, intrinsic parameters, and sparse geometry jointly from a complete set of images, followed by a densification step using Multi-view Stereo (MVS) [19] or learning-based representations, like 3D Gaussian Splatting (3DGS) [11]. While these approaches can achieve good reconstruction quality, they fundamentally require all images to be available in advance. As a consequence, they are unsuitable for scenarios where image data is acquired sequentially and where immediate feedback is desired.

Incremental reconstruction approaches address these issues. Simultaneous Localization and Mapping (SLAM)-methods [4] estimate camera poses and scene structure on-the-fly, enabling online operation and continuous map expansion as new images arrive. More recently, learning-based scene representations such as 3DGS have been adapted to online image acquisition [14]. These methods make it possible to visualize a scene in real-time while it is being reconstructed, providing immediate feedback and enabling adaptive data acquisition.

Despite these advantages, online approaches typically estimate camera poses sequentially from image data alone. As a result, they are prone to drift and loop-closure errors, which can accumulate over time and lead to globally inconsistent reconstructions.

In parallel, Motion Capture (Mocap) systems offer an alternative means of camera pose estimation. By tracking a rigid body with a mounted camera, Mocap systems can provide camera poses with high accuracy and global consistency, independent of visual scene content. However, in the context of 3D reconstruction, such systems are rarely used as the primary source of camera pose retrieval. Instead, they are typically employed for benchmarking, validation, or ground truth generation, leaving their potential for direct integration into online reconstruction pipelines largely unexplored.

This thesis aims to fill this gap by introducing *MocapGS*, a method that integrates accurate camera poses obtained from a Mocap system directly into an online 3D reconstruction pipeline based on 3DGS. To enable this integration, we propose a complete workflow consisting of:

1. Temporal Synchronization,
2. Camera Calibration,
3. Hand-eye Calibration,
4. Online 3D Reconstruction, and
5. 3D Mesh Extraction.

The main contributions of this work are threefold. First, we present the proposed workflow for integrating Mocap-based pose estimation into an online 3DGS reconstruction pipeline. Second, we evaluate the accuracy of the temporal

*Corresponding author
theodor.kapler@student.kit.edu (T. Kapler); markus.hillemann@kit.edu (M. Hillemann); robert.langendoerfer@kit.edu (R. Langendörfer); markus.ulrich@kit.edu (M. Ulrich)
ORCID(s): 0000-0002-8906-0450 (M. Hillemann); 0000-0001-8457-5554 (M. Ulrich)

synchronization and calibration procedures. Third, we assess the reconstruction quality and compare our approach against standard 3DGS as well as an online method that estimates camera poses solely from image data.

We find that by decoupling pose estimation from scene optimization, MocapGS combines the online visualization and incremental processing capabilities of modern 3DGS-based methods with the robustness and global consistency of Mocap-based camera tracking.

2. Related Work

In this section, we summarize the related work regarding 3D reconstruction methods, as well as camera tracking with Mocap.

2.1. 3D Reconstruction Using Structure from Motion

Classical 3D reconstruction pipelines are predominantly batch-based and follow a two-stage processing scheme. In a first stage, SfM is applied to a complete set of images to jointly estimate camera poses, intrinsics, and a sparse 3D point cloud via bundle adjustment.

In a second stage, dense reconstruction is performed using the estimated poses. Traditional photogrammetric methods rely on MVS to densify the sparse reconstruction. Neural Radiance Fields (NeRFs) [15], a more recent learning-based approach, replaces explicit point-based reconstruction with a continuous scene representation, enabling high-quality novel view synthesis. 3DGS adopts a more explicit scene representation based on learned 3D Gaussian ellipsoids, enabling very fast rendering.

While highly accurate, these pipelines share a key limitation: SfM is always required as a preprocessing step and depends on the availability of the full image set. Consequently, such methods are non-incremental and do not support real-time reconstruction or interactive scene exploration.

2.2. Online 3D Reconstruction

Online 3D reconstruction methods aim to process image data sequentially as it becomes available, without prior knowledge of the full dataset. These approaches typically combine pose estimation and mapping in a single, incremental pipeline.

Classical point-based methods such as SLAM estimate camera poses on-the-fly while incrementally building a map of the environment. More recent learning-based methods extend this idea to neural or hybrid scene representations. Approaches such as MonoGS [13], WildGS-SLAM [24], or On-the-fly NVS [14] enable incremental reconstruction and real-time rendering while estimating camera poses directly from incoming images.

A major advantage of online methods is the ability to visualize the scene in real time and to actively densify under-reconstructed regions. However, pose estimation is sequential and therefore depends on previously estimated poses. As a result, these methods are susceptible to drift and loop-closure errors, which can lead to globally inconsistent reconstructions.

2.3. Camera Tracking with Motion Capture

Mocap systems provide an alternative source of camera pose information. By rigidly mounting a camera to a set of tracked markers, the pose of the camera can be recovered in real-time with high accuracy after hand-eye calibration. Such systems yield globally consistent poses and are not affected by drift or loop-closure failures.

In the literature, Mocap is most commonly used for validation or ground truth generation rather than as the primary source of camera poses for reconstruction [6, 22, 20].

This work is situated at the intersection of online 3D reconstruction and Mocap-based camera tracking. While online 3D reconstruction methods provide real-time feedback and incremental processing, Mocap offers accurate and globally consistent poses. The combination of these complementary properties motivates the approach presented in this work.

3. Notation

This work extensively relies on rigid transformations in three-dimensional space. A rigid transformation in $\mathbb{R}^3$ has six degrees of freedom, comprising three rotational and three translational components. Throughout this paper, we refer to such transformations as *poses*.

Following the standard formulation in projective geometry [9], poses are represented by homogeneous transformation matrices of the form

$$\mathbf{E} = \begin{pmatrix} \mathbf{R} & \boldsymbol{t} \\ \mathbf{0}^\top & 1 \end{pmatrix}, \quad \mathbf{E} \in \mathbb{R}^{4\times 4}, \tag{1}$$

where $\mathbf{R} \in \mathbb{R}^{3\times 3}$ denotes a rotation matrix and $\boldsymbol{t} \in \mathbb{R}^3$ a translation vector. Throughout this paper, bold, capital, monospaced symbols (e.g., $\mathbf{E}$, $\mathbf{T}$) are used to denote homogeneous transformation matrices representing poses.

The mapping from an arbitrary pose parameterization $\boldsymbol{e}$ (e.g. Euler angles) to its associated homogeneous transformation matrix is denoted by $\mathbf{T}(\boldsymbol{e})$.

To describe the relative pose between two coordinate systems, the notation ${}^{\mathrm{A}}\mathbf{T}_{\mathrm{B}}$ is used. This denotes the pose of coordinate system B relative to coordinate system A.

4. Methodology

This section provides a comprehensive overview of the methodology employed in this work. It systematically describes the proposed workflow for performing online 3D reconstruction using a synchronized and calibrated camera–Mocap system. Figure 1 illustrates the complete pipeline. Each component of the workflow is explained in detail throughout the subsequent sections.

4.1. Definition of a Rigid Body

To enable camera pose tracking with a Mocap system, we attach a camera rigidly to a fixed set of Mocap markers,

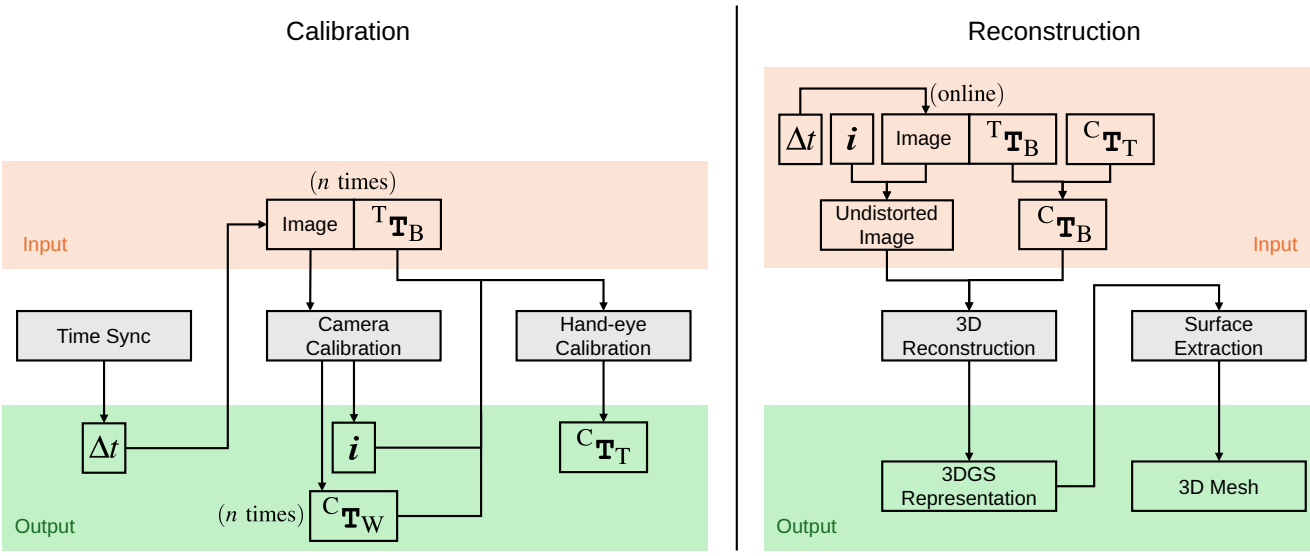


Figure 1: Overview of the proposed workflow. The calibration process yields the latency offset Δt, the intrinsic parameter vector $\boldsymbol{i}$, as well as the hand-eye pose $^{C}\mathbf{T}_{T}$. These quantities then, together with the online acquired images and tool poses $^{T}\mathbf{T}_{B}$, enable the 3DGS-based online 3D reconstruction.

Figure 2: Hand-held rigid body connecting the camera to spatially fixed set of Mocap markers.

which is depicted in Figure 2. These markers define a rigid body that can be tracked by the Mocap system. To maintain the analogy to robotics, in the following, we refer to the rigid body as *tool*. The pose of this tool coordinate system (TCS) can continously be tracked by the Mocap system. It differs from the pose of the camera coordinate system (CCS), that we are interested in, by an unknown pose $^{C}\mathbf{T}_{T}$, which we call *hand-eye pose*.

4.2. Temporal Synchronization

To ensure temporally matching image-pose pairs, it is essential to synchronize the camera and the Mocap system precisely within the employed computer system. In general, such synchronization can be achieved using either hardware-based or software-based triggering mechanisms. To avoid the additional complexity and overhead of dedicated trigger cabling, a software-based synchronization approach is adopted in this work.

Our goal therefore is to determine the latency difference Δt between the camera and the Mocap system that arises on the workstation connected to both systems. To achieve this, we define an event that is temporally well localized and observable by both the camera and the Mocap system. By measuring the arrival times of this event on the workstation, the latency difference Δt can be determined.

Common synchronization events for heterogeneous sensor setups include LED flashes or abrupt motions. However, the temporal resolution of such events is fundamentally limited by the camera frame rate, as they can typically only be detected as present or absent within individual image frames. This limits the achievable temporal precision.

To overcome this limitation, we instead exploit the continuous vertical motion of a Mocap marker as a synchronization signal. The marker's position is directly available from the Mocap data and can also be reliably segmented in the camera stream. From both modalities, we extract time series describing the marker's vertical movement: the vertical coordinate from the Mocap system and the row coordinate of the segmented marker center in the image sequence. We then fit a cubic spline to each time series to obtain a smooth, continuous representation of the motion. By determining the local maxima of the interpolated trajectories, we estimate the precise time points at which the marker reaches its maximum height with sub-frame accuracy. These time instants are subsequently used as high-precision synchronization events.

4.3. Camera Calibration

After temporally synchronizing the two systems, the camera calibration is performed. This involves estimating

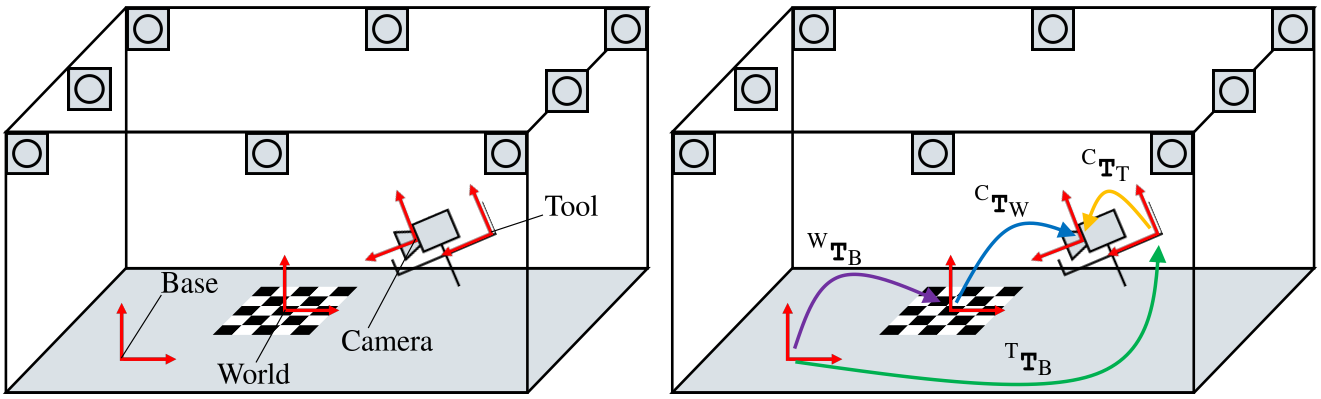


Figure 3: Depiction of our MocapGS setup: The capture volume is depicted by the cuboid with the Mocap tracking cameras at the top. Left shows the relevant coordinate systems and right the relevant transformations between them.

the camera's intrinsic parameter vector $\boldsymbol{i}$, as well as the extrinsic parameters for each calibration image $\boldsymbol{e}_i$. We perform camera calibration using multiple images of a checkerboard, which defines a world coordinate system (WCS). The extrinsic parameters $\boldsymbol{e}_i$ to be estimated for each image describe the pose of the WCS relative to the CCS, so the corresponding homogenous transform is denoted as $\left({}^{C}\mathbf{T}_{W}\right)_i = \mathbf{T}\left(\boldsymbol{e}_i\right)$. Our camera calibration procedure is based on OpenCV [3] and internally consists of an initial parameter estimation followed by a nonlinear refinement.

Initial Parameter Estimation. For estimating initial parameters, the method of Zhang [23] is used. Zhang models the projection of a 3D point in the WCS onto the 2D Image Coordinate System (ICS) through the pinhole camera. This allows for a linear model formulation using homogenous coordinates. The model parameters, namely the intrinsic parameters of the pinhole camera $\boldsymbol{i} = \left(f_x, f_y, c_x, c_y\right)$ and the extrinsic parameters of each image $\boldsymbol{e}_i$, can then linearly be determined.

Nonlinear Refinemant. In the following stage, the linearly estimated parameters are refined via nonlinear optimization. Therefore, the camera model is not constrained to linearity and can thus be extended to incorporate lens distortions. We choose a polynomial camera model using two radial distortion coefficients K_1 and K_2 and two decentering distortion coefficients P_1 and P_2.

Collecting all parameters, we define the vector

$$\boldsymbol{\theta} = \left(f_x, f_y, c_x, c_y, K_1, K_2, P_1, P_2, \boldsymbol{e}_1, \ldots, \boldsymbol{e}_{n_o}\right), \quad (2)$$

which contains all unknowns to be optimized. The distortion parameters K_1, K_2, P_1, and P_2 are typically small, so we assume initial values of zero. We can now define the residual function

$$d_{\mathrm{CC}}(\boldsymbol{\theta}) = \sum_{k=1}^{n_o} \sum_{j=1}^{n_m} \left\| \boldsymbol{m}_{j,k} - \boldsymbol{\pi}\left(\boldsymbol{M}_j, \boldsymbol{\theta}\right) \right\|_2^2 \longrightarrow \min, \quad (3)$$

where $\boldsymbol{m}_{j,k}$ denotes the observed 2D ICS coordinate of the j-th corner in the k-th image. $\boldsymbol{M}_j$ denotes the corresponding 3D WCS coordinate, which is projected into the ICS with the projection $\boldsymbol{\pi}$. The function $d_{\mathrm{CC}}(\boldsymbol{\theta})$ is commonly referred to as the reprojection error, which we minimize using the Levenberg-Marquardt method [12]. After optimization, the reprojection error (3) typically yields a residual value of $d_{\mathrm{CC}} \approx 0.5$ px with our used setup.

4.4. Hand-eye Calibration

During the acquisition of the calibration dataset, we not only capture calibration images but also record the corresponding tool pose from the Mocap system for each image, synchronized via Δt. This allows us to perform a hand-eye calibration.

Hand-eye calibration refers to the determination of the relative pose between a robot's end effector (tool) and a camera that is rigidly mounted to it [5]. Establishing this relationship allows observations made in the CCS to be transformed into the TCS and subsequently into the robot base coordinate system (BCS).

In our case, we want to use hand-eye calbration to estimate the relative pose bewtween the CCS and the TCS, which can be tracked with the Mocap system. This hand-eye pose ${}^{C}\mathbf{T}_{T}$ then allows the tool poses to be corrected, retrieving the camera pose which can then be used for the online 3D reconstruction. Figure 3 illustrates the relevant coordinate systems and transformations in this context. As for the camera calibration, our hand-eye calibration procedure is based on OpenCV and consists of an initial parameter estimation followed by a nonlinear refinement.

Initial Parameter Estimation. The linear approach for estimating initial parameters is based on solving the fundamental equation

$$\mathbf{AX} = \mathbf{XB} \quad (4)$$

for the hand-eye pose $\mathbf{X} = {}^{C}\mathbf{T}_{T}$. $\mathbf{A}$ represents the known motion of the tool between two tool poses $\left({}^{T}\mathbf{T}_{B}\right)_i$ and $\left({}^{T}\mathbf{T}_{B}\right)_j$, which are obtained from the Mocap system. $\mathbf{B}$ represents the corresponding known motion between two camera poses $\left({}^{C}\mathbf{T}_{W}\right)_i$ and $\left({}^{C}\mathbf{T}_{W}\right)_j$, which are obtained by the preceding camera calibration.

Several methods exist to solve Equation 4. We choose the approach proposed by Daniilidis [5], which uses dual quaternions to represent poses.

For the following nonlinear refinement, we also need an initial estimate of the base-world pose $\mathbf{Y} = {}^{\mathrm{W}}\mathbf{T}_{\mathrm{B}}$. A reasonable initialization for $\mathbf{Y}$ can be derived by a single pair of tool pose and camera pose using

$$\mathbf{Y} = \left({}^{\mathrm{C}}\mathbf{T}_{\mathrm{W}}\right)_i^{-1} \mathbf{X} \left({}^{\mathrm{T}}\mathbf{T}_{\mathrm{B}}\right)_i, \tag{5}$$

which can be seen in Figure 3.

Nonlinear Refinement As for the camera calibration, we adopt a method that minimizes a reprojection error. In contrast to camera calibration, we transform a checkerboard point over a transformation chain containing the base-world pose, the tool pose, and the hand-eye pose before projecting it into the ICS. This allows optimizing for the hand-eye pose and base-world pose. We define the parameter vectors

$$\begin{aligned} \boldsymbol{x} &= \left(\alpha_x, \beta_x, \gamma_x, t_{x,x}, t_{y,x}, t_{z,x}\right), \\ \boldsymbol{y} &= \left(\alpha_y, \beta_y, \gamma_y, t_{x,y}, t_{y,y}, t_{z,y}\right), \end{aligned} \tag{6}$$

which represent a minimal parameterization of the hand-eye pose $\mathbf{X} = {}^{\mathrm{C}}\mathbf{T}_{\mathrm{T}}$ and the base-world pose $\mathbf{Y} = {}^{\mathrm{W}}\mathbf{T}_{\mathrm{B}}$, respectively. These are the parameters to be optimized. The nonlinear refinement is formulated as the minimization of the reprojection error function

$$\begin{aligned} d_{\mathrm{HEC}}(\boldsymbol{x}, \boldsymbol{y}) &= \sum_{k=1}^{n_o} \sum_{j=1}^{n_m} \left\| \boldsymbol{m}_{j,k} - \boldsymbol{\pi}\left(\boldsymbol{M}_{j,k}, \boldsymbol{i}\right) \right\|_2^2 \longrightarrow \min, \\ \boldsymbol{M}_{j,k} &= \mathbf{T}(\boldsymbol{x}) \left({}^{\mathrm{T}}\mathbf{T}_{\mathrm{B}}\right)_k \mathbf{T}(\boldsymbol{y})^{-1} \boldsymbol{M}_j, \end{aligned} \tag{7}$$

where $\boldsymbol{m}_{j,k}$ denotes the observed 2D image coordinate of the j-th calibration point in the k-th image. $\boldsymbol{M}_j$ is the corresponding homogenous 3D WCS coordinate, which is transformed to the CCS of the k-th image through the transformation chain illustrated in Figure 3, yielding $\boldsymbol{M}_{j,k}$. $\boldsymbol{M}_{j,k}$ is then projected into the image via the projection function $\boldsymbol{\pi}$ using the intrinsic camera parameters $\boldsymbol{i}$. The reprojection error (7) is minimized using the Levenberg-Marquardt algorithm [12], yielding optimal estimates for both the hand-eye pose $\mathbf{X}$ and the base-world pose $\mathbf{Y}$. After optimization, the reprojection error (7) typically yields a residual value of $d_{\mathrm{HEC}} \approx 1.5$ px with our used setup.

4.5. Online 3D Reconstruction

At this stage, the following quantities are available from the preceding synchronization and calibration procedure:

1. the latency difference Δt,
2. the intrinsic parameter vector $\boldsymbol{i}$,
3. the hand-eye pose ${}^{\mathrm{C}}\mathbf{T}_{\mathrm{T}}$.

With these quantities, we can now perform the 3DGS-based online 3D reconstruction. During the incremental reconstruction stage, Δt enables us to continuously recieve temporally matched pairs of an image and its corresponding tool pose ${}^{\mathrm{T}}\mathbf{T}_{\mathrm{B}}$. The obtained image is then, using the previously determined intrinsic parameter vector $\boldsymbol{i}$, undistorted and shifted so that the principle point lies in the image center. In the same time, the tool pose ${}^{\mathrm{T}}\mathbf{T}_{\mathrm{B}}$ can, togehter with the previously determined hand-eye pose ${}^{\mathrm{C}}\mathbf{T}_{\mathrm{T}}$, be composed to recieve the desired pose of the BCS with respect to the CCS

$${}^{\mathrm{C}}\mathbf{T}_{\mathrm{B}} = {}^{\mathrm{C}}\mathbf{T}_{\mathrm{T}}\,{}^{\mathrm{T}}\mathbf{T}_{\mathrm{B}}. \tag{8}$$

The shifted and undistorted image can then, together with its pose ${}^{\mathrm{C}}\mathbf{T}_{\mathrm{B}}$, be used as an incremental input to the 3DGS-based online 3D reconstruction to perform reconstruction directly in the BCS.

As a baseline for our reconstruction algorithm, we use the method On-the-fly NVS [14]. This approach already handles incremental 3DGS, as well as initial camera pose and intrinsic parameter estimation and their refinement in the 3DGS optimization.

Our goal therefore is to streamline this baseline approach by eliminating the initial camera pose and intrinsic parameter estimation and replacing those by our well-known intrinsic parameter vector $\boldsymbol{i}$ and on-the-fly acquired camera pose ${}^{\mathrm{C}}\mathbf{T}_{\mathrm{B}}$. This enables us to perform online, incremental 3D reconstruction using 3DGS, already in the metric BCS.

To investigate whether additional pose refinement is beneficial, we implement and evaluate three different variants. We refer to these methods as follows:

1. **MocapGS-Fixed:** The initial camera poses obtained from the Mocap system are used directly and are not further refined.
2. **MocapGS-FullOpt:** The initial camera poses are refined jointly with the Gaussian parameters during the 3DGS optimization.
3. **MocapGS-RotOpt:** Only the rotational components of the initial camera poses are refined during 3DGS optimization, while the translational components are kept fixed in order to preserve the metric scale of the reconstruction.

4.6. 3D Mesh Extraction

After completing the online 3DGS-based 3D reconstruction, we perform an offline surface extraction step to recover the explicit geometry of the observed scene or object. For this purpose, we adopt the efficient mesh extraction pipeline proposed in SuGaR [8], which is straightforward to integrate with a previously trained Gaussian representation.

SuGaR extracts surface points by identifying locations of equal density within the scene. Specifically, multiple depth maps are rendered from different viewpoints, and pixels are sampled from each depth map. For every corresponding camera ray, the method determines whether a point with a predefined density value exists along the ray and, if so, computes its 3D position. The collection of these points yields a sparse sampling of the underlying surface. Finally, Poisson surface reconstruction [10] is applied to the extracted surface points to generate a triangular mesh.

5. Evaluation

This section presents a comprehensive evaluation of the proposed MocapGS method. We first describe the comparative setup and introduce the datasets employed in our experiments. Subsequently, we report both quantitative and qualitative results.

5.1. Comparative Setup

We compare our approach against 3DGS, as well as against On-the-fly NVS, which forms the basis of our system. The key difference between the methods lies in how camera poses and intrinsic parameters are obtained and optimized.

3DGS estimates all camera poses and intrinsics in advance using COLMAP [19]. The comparison is therefore not completely fair, since it is not an online method and is significantly slower. Nevertheless, we include it as a baseline because it represents a strong, widely adopted offline approach.

On-the-fly NVS is an online method and estimates intrinsics from a bundle adjustment of the first 8 frames. Initial camera poses for each subsequent frame are then also estimated using bundle adjustment and refined during the following scene optimization.

MocapGS is also an online method and leverages externally provided and calibrated camera parameters. Camera intrinsics are assumed to be known and fixed, while camera poses are obtained from the Mocap system. To study the influence of pose optimization during training, we evaluate the three variants of our approach, MocapGS-Fixed, MocapGS-FullOpt, and MocapGS-RotOpt.

5.2. Datasets

We evaluate all methods on three reconstruction datasets we captured using our Mocap setup:

Buddha. This dataset consists of 411 RGB images capturing a Buddha statue from multiple concentric viewpoints, each with its corresponding Mocap pose. The object exhibits both coarse geometric structure and fine-grained surface details, making it well suited for quantitative 3D evaluation. A highly accurate 3D ground truth of the statue is available in the form of a dense mesh acquired using a structured-light scanning system, for details refer to [17, 16].

Robot. This dataset consists of 409 RGB images, each with an associated Mocap pose. It captures a working table setup that includes a robot arm, controller, monitor, computers, and various cables. The scene is captured from multiple concentric viewpoints. This dataset poses a particularly challenging reconstruction scenario due to its complex and heterogeneous geometry, as well as the presence of thin structures and significant clutter. A 3D ground truth of the working table is available in the form of a dense point cloud acquired using a Zoller+Fröhlich Imager 5016 laser scanner.

PhotoLab. This dataset consists of 176 RGB images capturing the room in which the Mocap system is installed. Each image is accompanied by its corresponding Mocap pose. Unlike the previous datasets, the scene is captured in an eccentric, inside-out manner: instead of observing a single object from the outside, the camera traverses the room along a looped trajectory, capturing the surrounding environment. This dataset is particularly challenging, especially for image-based pose estimation methods, as many visual features are only observed from a limited number of viewpoints. A 3D ground truth of the room is not available.

5.3. General Considerations

In contrast to COLMAP and On-the-fly NVS, MocapGS is not constrained to a minimum image overlap for pose estimation. Despite this advantage of MocapGS over COLMAP and On-the-fly NVS, all datasets provide sufficient image overlap, enabling COLMAP and On-the-fly NVS to estimate camera poses for each image.

MocapGS uses the Mocap poses as input, while COLMAP+3DGS and On-the-fly NVS only use the captured images as input and infer the pose solely from image data. COLMAP+3DGS and On-the-fly NVS therefore operate with an unknown scale factor, while MocapGS is able to operate in metric units due to the calibrated Mocap system.

5.4. Metrics

2D Metrics. For assessing the rendering quality of the output 3DGS representation, we report the Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM). These metrics are computed using ground truth image-pose pairs, that have not been used for training.

For COLMAP+3DGS, we use the camera poses estimated from COLMAP as ground truth pose.

On-the-fly NVS jointly optimizes camera poses and the scene representation. Consequently, a strict separation between scene representation and ground truth poses is less straightforward. In their evaluation protocol, the authors estimate the poses of test images during joint 3DGS optimization while keeping the scene representation fixed, resulting in no scene update for a given test image-pose pair.

For MocapGS-Fixed, we use the camera poses acquired by the Mocap system as ground truth pose. MocapGS-FullOpt and MocapGS-RotOpt further refine the test poses similar to On-the-fly NVS.

3D Metrics. For the 3D evaluation, we convert the optimized 3DGS representation into an explicit triangle mesh following the procedure described in Section 4.6. The reconstructed mesh is then compared to the ground truth 3D data, which consists of a triangle mesh for the Buddha dataset and a point cloud for the Robot dataset.

Before comparison, we align the reconstructed mesh to the ground truth using a two-stage registration procedure: an initial coarse alignment based on manually selected point correspondences, followed by fine registration using the Iterative Closest Point (ICP) algorithm [2]. We then compute Cloud-to-Mesh (C2M) distances.

For the Buddha dataset, C2M distances are measured from the reconstructed mesh vertices to the ground truth

Table 1
Quantitative results on the Buddha, Robot, and PhotoLab datasets. We report runtime (including pose estimation), PSNR, SSIM, and σ_{3D}. The best and second best results are color-coded among the online methods (On-the-fly NVS and MocapGS variants).

	Buddha				Robot				PhotoLab			
	Time↓	PSNR↑	SSIM↑	σ_{3D} (mm)↓	Time↓	PSNR↑	SSIM↑	σ_{3D} (mm)↓	Time↓	PSNR↑	SSIM↑	σ_{3D} (mm)↓
On-the-fly NVS	02:00	29.10	0.83	6.2	01:58	26.17	0.82	4.5	00:45	25.89	0.78	–
MocapGS-Fixed	01:53	28.80	0.82	5.4	01:52	26.82	0.83	4.7	00:42	31.52	0.85	–
MocapGS-FullOpt	01:55	29.56	0.84	5.1	01:53	28.05	0.85	4.2	00:43	31.73	0.85	–
MocapGS-RotOpt	01:55	29.47	0.84	5.1	01:54	27.61	0.84	5.0	00:43	31.68	0.85	–
COLMAP + 3DGS	12:42	31.67	0.92	14.2	12:06	31.53	0.93	7.0	11:13	32.14	0.90	–

Ground Truth | 3DGS | On-the-fly NVS | MocapGS-Fixed | MocapGS-FullOpt | MocapGS-RotOpt

Figure 4: Test image that has not been used during training (left) and rendered scene for all corresponding methods. Buddha dataset on top, Robot dataset in the middle and PhotoLab dataset below.

mesh. For the Robot dataset, due to the sparsity of the ground truth point cloud, distances are computed from the ground truth points to the reconstructed mesh.

From all C2M distances, the mean μ_{3D} and standard deviation σ_{3D} can be calculated. In principle, μ_{3D} should be small if the reconstructed mesh is registered perfectly to the ground truth. μ_{3D} therefore serves as a measure of the quality of the registration. The residual uncertainty and noise are characterized by the standard deviation σ_{3D}, which therefore is the metric we report.

5.5. Quantitative Evaluation

Table 1 summarizes the numerical reconstruction results for all compared methods. We report all previously introduced evaluation metrics as well as the training duration for each approach. For 3DGS, the reported training duration corresponds to 30,000 optimization iterations. Since 3DGS is not incremental, we additionally account for the preprocessing time required for pose estimation using COLMAP.

For On-the-fly NVS and our MocapGS variants, the training duration is measured as the cumulative runtime after performing 30 iterations per sequential input image. As all datasets are pre-captured and include recorded Mocap poses, these timings represent a best-case runtime scenario for the respective methods.

Considering the 2D evaluation metrics PSNR and SSIM, 3DGS consistently achieves the highest scores across all datasets. This performance, however, comes at the cost of substantially longer training times due to the additional COLMAP overhead and the fixed number of optimization iterations. Among the four online methods, we observe that MocapGS-FullOpt and MocapGS-RotOpt consistently outperform On-the-fly NVS by a noticeable margin. This behaviour is especially distinctive on the PhotoLab dataset,

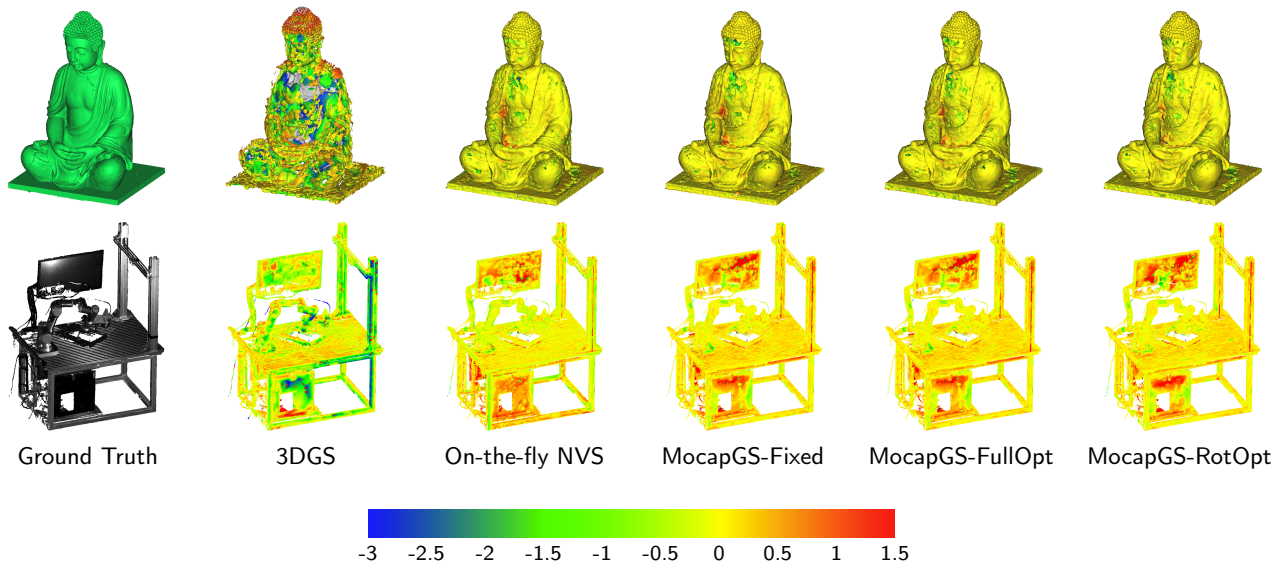


Figure 5: Extracted 3D geometry of the Buddha dataset (top) and the Robot dataset (bottom) colored according to the C2M distances to the ground truth representation. Color bar in cm.

which is particularily challenging for image-based pose estimation.

Turning to the 3D evaluation metric σ_{3D} for the Buddha and Robot datasets, it becomes evident that 3DGS is not competitive with respect to extracted 3D geometry. This is likely due to a mismatch between the scene representation learned by 3DGS and our employed surface extraction strategy. Among the four online methods, the lowest σ_{3D} values, and thus the most accurate and stable 3D reconstructions, are consistently achieved by MocapGS variants, highlighting their advantage for high-quality 3D surface extraction.

5.6. Qualitative Evaluation

Figure 4 shows one test view for each dataset. The ground truth image is depicted along with the rendered views generated from all compared methods from the repsective ground truth pose of the image. Overall, all methods produce visually compelling novel views, which is also reflected in reasonably high PSNR and SSIM values in Table 1. Noticable blur can only be observed on the PhotoLab dataset reconstructed by On-the-fly NVS.

Figure 5 shows the generated 3D geometry for the Buddha and the Robot dataset. The 3D representations are colored according to their C2M distances to the ground truth. It becomes evident that 3DGS in combined with our employed mesh extraction strategy does not produce a compelling 3D mesh. All other methods procuce visually compelling meshes with an accuracy in the range of few millimeters.

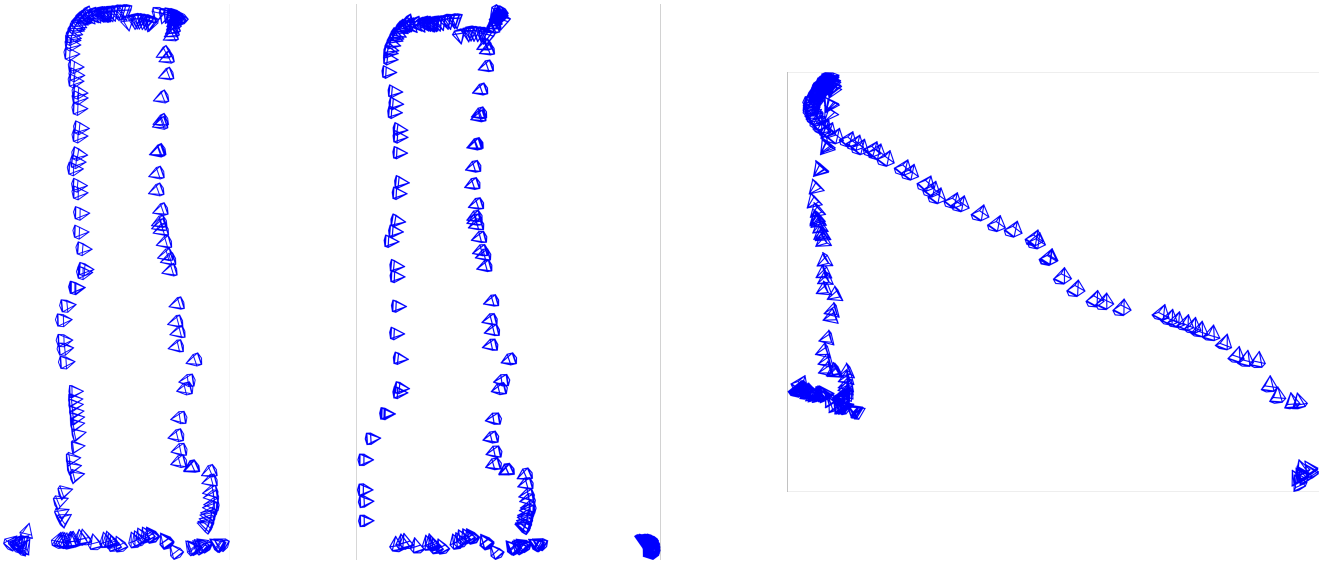
Figure 6 presents a top-view comparison of the estimated camera poses for the PhotoLab scene. The MocapGS poses are directly provided by the Mocap system, the 3DGS poses are obtained via COLMAP, and the On-the-fly NVS poses are estimated online. MocapGS and 3DGS show close overall agreement, although 3DGS exhibits several outliers. In contrast, the On-the-fly NVS approach suffers from significant pose drift, resulting in a noticeable loop-closure error.

Figure 7 illustrates the impact of this drift on the reconstructed scene. The figure provides an external view of the 3DGS-based representation of the PhotoLab scene. MocapGS produces a geometrically consistent and coherent reconstruction. While 3DGS encounters difficulties at loop closure, leading to local inconsistencies, the On-the-fly NVS reconstruction is globally distorted due to accumulated pose errors.

6. Discussion

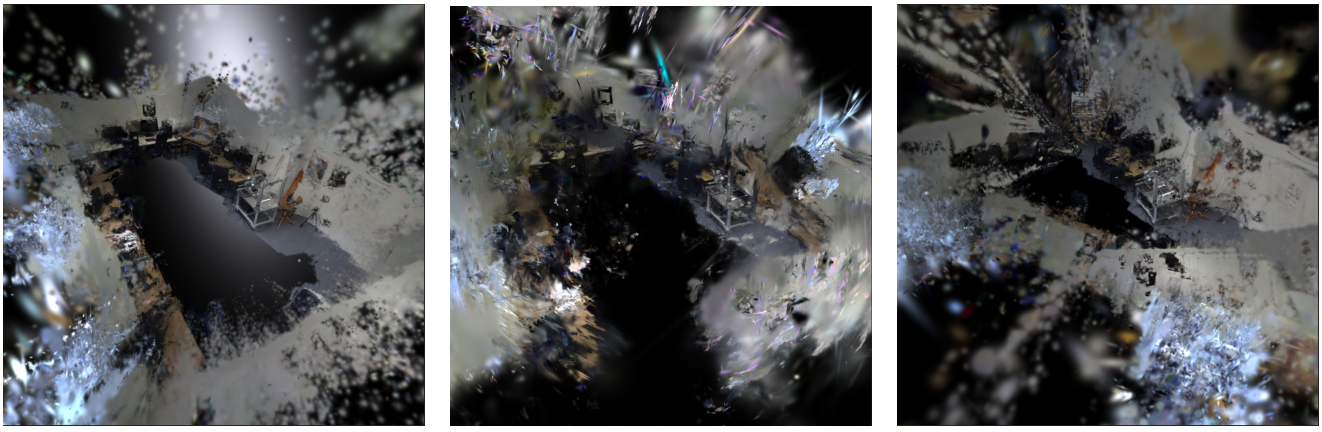
MocapGS Variant Selection. Among the MocapGS variants, pose optimization clearly improves performance. Both MocapGS-FullOpt and MocapGS-RotOpt outperform MocapGS-Fixed in 2D and 3D metrics, while FullOpt and RotOpt show negligible differences in 2D and 3D metrics. We prefer RotOpt because it optimizes only the rotation while keeping translations fixed, preserving metric scale.

Capture Style. On-the-fly NVS, as an example of purely image-based methods, requires an initial pose estimation via bundle adjustment. This restricts newly captured images to small displacements relative to the previous frame. If the user loses track of initial poses, they must reorient the camera toward previously observed regions to re-establish sufficient feature correspondences. This process leads to frequent interruptions during data acquisition. MocapGS addresses this limitation by eliminating the need for initial pose estimation. Initial poses are directly provided by the Mocap system. This enables fully unconstrained camera motion, allowing the user to focus on one part of the scene in one frame and a completely different region in the next without interruption. As a result, our approach enables a seamless, intuitive, and user-friendly data acquisition process.



MocapGS-Fixed 3DGS On-the-fly NVS

Figure 6: Top-view comparison of the camera poses in the PhotoLab scene. MocapGS-Fixed poses are obtained directly with the Mocap system, 3DGS poses are estimated with COLMAP, and On-the-fly NVS poses are estimated online.



MocapGS-Fixed 3DGS On-the-fly NVS

Figure 7: Overview of the Gaussian representation of the PhotoLab scene obtained by MocapGS-Fixed, 3DGS, and On-the-fly NVS.

Robustness. Even when disregarding the facilitated capture setup of MocapGS, which allows for sudden and rapid camera movements, and restricting the comparison to datasets with sufficient overlap for On-the-fly NVS to operate, MocapGS proves to be substantially more robust. As shown in Figures 6 and 7, the camera poses estimated by On-the-fly NVS often exhibit significant drift or are entirely incorrect, particularly for challenging trajectories such as those in the PhotoLab dataset. This behavior can be attributed to the absence of loop closure in On-the-fly NVS. In contrast, MocapGS effectively exhibits no drift, as its pose estimates do not depend on previous observations but are instead provided continuously with high accuracy by the Mocap system. This design enables robust and stable scene reconstruction even under challenging and complex camera trajectories.

Limitations. MocapGS introduces additional overhead compared to the baseline on-the-fly NVS approach. The required synchronization and calibration procedures take approximately five minutes, representing only a modest time investment. However, the financial cost of the motion capture system is significant in comparison to the baseline setup, which requires only a standard camera.

Also, a limitation of the presented approach lies in the separation between reconstruction and surface extraction. While high-quality meshes can be obtained in a post-processing step, the 3DGS representation itself is not explicitly optimized for surface consistency.

Future Work. A promising direction for future work is to incorporate surface-aware constraints or loss terms directly into the MocapGS training process, encouraging Gaussians

to align with surface structures during reconstruction. Such constraints could improve the quality of the extracted geometry and reduce the reliance on offline surface extraction.

7. Conclusion

In this paper, we presented MocapGS, a method for online 3D reconstruction that directly integrates accurate camera poses from a Mocap system into a 3DGS-based pipeline. By leveraging globally consistent and metrically accurate Mocap poses, our approach eliminates the need for image-based pose estimation and avoids the requirement for strong image overlap during capture. In comparison to methods that estimate poses solely from visual data, MocapGS achieves a more globally consistent and robust scene reconstruction, avoiding drift and loop-closure issues. Furthermore, the use of metric Mocap poses enables reconstruction in a fully metric scale without additional alignment steps. We hope that this work demonstrates the potential of Mocap systems not only as validation tools, but as primary pose providers for reliable and efficient 3D reconstruction.

References

[1] Behley, J., Stachniss, C., 2018. Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments, in: Robotics: Science and Systems XIV, Robotics: Science and Systems Foundation. URL: http://www.roboticsproceedings.org/rss14/p16.pdf, doi:10.15607/RSS.2018.XIV.016.

[2] Besl, P., McKay, N.D., 1992. A method for registration of 3-D shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence 14, 239–256. URL: https://ieeexplore.ieee.org/document/121791, doi:10.1109/34.121791.

[3] Bradski, G., 2000. The OpenCV Library. Dr. Dobb's Journal of Software Tools .

[4] Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., Leonard, J.J., 2016. Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age. IEEE Transactions on Robotics 32, 1309–1332. URL: https://ieeexplore.ieee.org/abstract/document/7747236, doi:10.1109/TRO.2016.2624754.

[5] Daniilidis, K., 1999. Hand-Eye Calibration Using Dual Quaternions. The International Journal of Robotics Research 18, 286–298. URL: https://doi.org/10.1177/02783649922066213, doi:10.1177/02783649922066213.

[6] Endres, F., Hess, J., Engelhard, N., Sturm, J., Cremers, D., Burgard, W., 2012. An evaluation of the RGB-D SLAM system, in: 2012 IEEE International Conference on Robotics and Automation, IEEE, St Paul, MN, USA. pp. 1691–1696. URL: http://ieeexplore.ieee.org/document/6225199/, doi:10.1109/ICRA.2012.6225199.

[7] Gomes, L., Regina Pereira Bellon, O., Silva, L., 2014. 3D reconstruction methods for digital preservation of cultural heritage: A survey. Pattern Recognition Letters 50, 3–14. URL: https://www.sciencedirect.com/science/article/pii/S0167865514001032, doi:10.1016/j.patrec.2014.03.023.

[8] Guédon, A., Lepetit, V., 2024. SuGaR: Surface-Aligned Gaussian Splatting for Efficient 3D Mesh Reconstruction and High-Quality Mesh Rendering, in: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5354–5363. URL: https://ieeexplore.ieee.org/document/10655755, doi:10.1109/CVPR52733.2024.00512.

[9] Hartley, R., Zisserman, A., 2004. Multiple View Geometry in Computer Vision. 2 ed., Cambridge University Press, Cambridge. URL: https://www.cambridge.org/core/books/multiple-view-geometry-in-computer-vision/0B6F289C78B2B23F596CAA76D3D43F7A, doi:10.1017/CBO9780511811685.

[10] Kazhdan, M., Bolitho, M., Hoppe, H., 2006. Poisson surface reconstruction, in: Proceedings of the fourth Eurographics symposium on Geometry processing, Eurographics Association, Goslar, DEU. pp. 61–70. doi:10.5555/1281957.1281965.

[11] Kerbl, B., Kopanas, G., Leimkuehler, T., Drettakis, G., 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. ACM Transactions on Graphics 42, 1–14. URL: https://dl.acm.org/doi/10.1145/3592433, doi:10.1145/3592433.

[12] Marquardt, D.W., 1963. An Algorithm for Least-Squares Estimation of Nonlinear Parameters. Journal of the Society for Industrial and Applied Mathematics 11, 431–441. URL: https://www.jstor.org/stable/2098941.

[13] Matsuki, H., Murai, R., Kelly, P.H.J., Davison, A.J., 2024. Gaussian Splatting SLAM, in: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Seattle, WA, USA. pp. 18039–18048. URL: https://ieeexplore.ieee.org/document/10657715/, doi:10.1109/CVPR52733.2024.01708.

[14] Meuleman, A., Shah, I., Lanvin, A., Kerbl, B., Drettakis, G., 2025. On-the-fly Reconstruction for Large-Scale Novel View Synthesis from Unposed Images. ACM Transactions on Graphics 44, 1–14. URL: https://dl.acm.org/doi/10.1145/3730913, doi:10.1145/3730913.

[15] Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R., 2021. NeRF: representing scenes as neural radiance fields for view synthesis. Commun. ACM 65, 99–106. URL: https://dl.acm.org/doi/10.1145/3503250, doi:10.1145/3503250.

[16] Petrovska, I., Jutzi, B., 2024. Vision through Obstacles—3D Geometric Reconstruction and Evaluation of Neural Radiance Fields (NeRFs). Remote Sensing 16, 1188. URL: https://www.mdpi.com/2072-4292/16/7/1188, doi:10.3390/rs16071188.

[17] Petrovska, I., Jäger, M., Haitz, D., Jutzi, B., 2023. Geometric Accuracy Analysis between Neural Radiance Fields (NeRFs) and Terrestrial Laser Scanning (TLS). The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XLVIII-1-W3-2023, 153–159. URL: https://isprs-archives.copernicus.org/articles/XLVIII-1-W3-2023/153/2023/, doi:10.5194/isprs-archives-XLVIII-1-W3-2023-153-2023. conference Name: 2nd GEOBENCH Workshop on Evaluation and BENCHmarking of Sensors, Systems and GEOspatial Data in Photogrammetry and Remote Sensing - 23–24 October 2023, Krakow, Poland.

[18] Schönberger, J.L., Frahm, J.M., 2016. Structure-from-Motion Revisited, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4104–4113. URL: https://ieeexplore.ieee.org/document/7780814, doi:10.1109/CVPR.2016.445.

[19] Schönberger, J.L., Zheng, E., Frahm, J.M., Pollefeys, M., 2016. Pixelwise View Selection for Unstructured Multi-View Stereo, in: Leibe, B., Matas, J., Sebe, N., Welling, M. (Eds.), Computer Vision – ECCV 2016, Springer International Publishing, Cham. pp. 501–518. doi:10.1007/978-3-319-46487-9_31.

[20] Shu, Z., Bei, S., Dai, J., Li, L., Chen, Z., Wang, J., 2026. MoCap2GT: A High-Precision Ground Truth Estimator for SLAM Benchmarking Based on Motion Capture and IMU Fusion. IEEE Robotics and Automation Letters 11, 1538–1545. URL: https://ieeexplore.ieee.org/document/11297812, doi:10.1109/LRA.2025.3643287.

[21] Steger, C., Ulrich, M., Wiedemann, C. (Eds.), 2018. Machine vision algorithms and applications. 2nd, completely revised and enlarged edition ed., Wiley-VCH, Weinheim.

[22] Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D., 2012. A benchmark for the evaluation of RGB-D SLAM systems, in: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, Vilamoura-Algarve, Portugal. pp. 573–580. URL: http://ieeexplore.ieee.org/document/6385773/, doi:10.1109/IROS.2012.6385773.

[23] Zhang, Z., 2000. A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence 22, 1330–1334. URL: https://ieeexplore.ieee.org/document/888718, doi:10.1109/34.888718.

[24] Zheng, J., Zhu, Z., Bieri, V., Pollefeys, M., Peng, S., Armeni, I., 2025. WildGS-SLAM: Monocular Gaussian Splatting SLAM in Dynamic Environments, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11461–11471.