RGB-based Simultaneous Localization and Mapping (SLAM): State of the Art

Johannes Decker (johannes.decker@hm.edu)
Report related repository: https://github.com/JohannesDecker95/hm_hauptseminar_vcml

***Abstract*—Recent advances in learning-based monocular Simultaneous Localization and Mapping (SLAM) exploit neural scene representations. This paper presents a systematic head-to-head evaluation of two state-of-the-art RGB-only pipelines—GlORIE-SLAM, which optimizes a deformable implicit point cloud with proxy-depth supervision, and HI-SLAM2, which reconstructs scenes with geometry-aware 3-D Gaussian splatting. Both methods are benchmarked on the *Replica*, *TUM RGB-D* and *HM* datasets using photometric (PSNR, SSIM, LPIPS), geometric (accuracy, completeness) and the training time. Across all datasets HI-SLAM2 delivers consistently higher image quality—up to 8.7dB PSNR gain and 50% lower LPIPS—while preserving centimetre-level pose accuracy and reducing per-sequence processing time from hours to minutes. GlORIE-SLAM excels in dense appearance recovery when its proxy-depth prior is reliable but suffers from longer runtimes and lower geometric fidelity.**

***Index Terms*—Monocular SLAM, RGB-only SLAM, Neural scene representation, 3-D Gaussian splatting, Neural Radiance Fields, Deformable neural point cloud, Photometric reconstruction, Pose estimation, Comparative benchmark.**

I. Introduction

Simultaneous Localization and Mapping (SLAM) is a foundational problem in computer vision and robotics that aims to construct a spatial map of an environment while simultaneously estimating the position of a moving camera. Traditional SLAM systems have relied heavily on depth sensors (e.g., Red Green Blue - Depth (RGB-D) or Light Detection and Ranging (LiDAR)) to extract geometric information, limiting deployment due to hardware costs and environmental constraints. In contrast, SLAM methods based solely on monocular RGB images offer a lightweight, scalable alternative, yet they face intrinsic challenges such as depth ambiguity, scale indeterminacy, and global consistency. In recent years, neural representations and learning-based tracking have enabled a new generation of dense RGB-only SLAM systems. These systems attempt to reconstruct photorealistic and geometrically faithful 3D scenes from color input alone, often relying on auxiliary monocular priors and novel implicit scene models. Despite significant progress, a trade-off remains between realism, accuracy, and computational efficiency in 3D reconstruction from monocular imagery. This paper addresses a central question in developing RGB-based SLAM: How can a model of a 3D scene be created as realistically and efficiently as possible using only monocular color images? This paper investigates this question by analyzing and comparing two recent state-of-the-art systems: Globally Optimized RGB-only Implicit Encoding Point Cloud SLAM (GlORIE-SLAM) [1] and HI-SLAM2: Geometry-Aware Gaussian SLAM for Fast Monocular Scene Reconstruction (HI-SLAM2) [2]. Both methods leverage RGB input exclusively and tackle the limitations of monocular SLAM through fundamentally different scene representations and optimization strategies. GlORIE-SLAM introduces a deformable dense neural point cloud that allows efficient online bundle adjustment (BA) and loop closure (LC) without costly backpropagation, aiming to improve tracking accuracy and rendering quality. In contrast, HI-SLAM2 employs a geometry-aware 3D Gaussian Splatting (3DGS) framework with spatially varying monocular scale alignment, enabling accurate surface modeling and high-fidelity appearance reconstruction. Through an in-depth comparative analysis of these two systems—across architecture, optimization, and experimental performance—we aim to clarify the advantages and limitations of current RGB-only SLAM paradigms and outline key directions for future work in realistic, efficient monocular scene modeling.

II. Related Work

A. Neural Radiance Fields

A *neural radiance field* (NeRF) represents a static scene as a continuous five-dimensional function

$$F_\Theta : (\mathbf{x}, \mathbf{d}) \longmapsto (\sigma, \mathbf{c}),$$

where the input is a 3-D spatial position $\mathbf{x} = (x, y, z) \in \mathbb{R}^3$ and a 2-D viewing direction expressed as a unit vector $\mathbf{d} \in \mathbb{S}^2$, and the output is the *volume density* $\sigma(\mathbf{x}) \in \mathbb{R}_{\geq 0}$ and the *view-dependent emitted radiance* (RGB colour) $\mathbf{c}(\mathbf{x}, \mathbf{d}) \in [0, 1]^3$. The mapping F_Θ is realised by a fully–connected multi-layer perceptron (MLP); its weights and biases $\Theta = \{W_\ell, b_\ell\}_{\ell=1}^{L}$ constitute the *learnable parameters* optimised during training [3].

a) Volume rendering: Given a calibrated image, a camera ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$, $t \in [t_n, t_f]$, is cast through the scene. The expected colour $C(\mathbf{r})$ observed along the ray is obtained by integrating the differential contributions of all points on the ray:

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\, \sigma\big(\mathbf{r}(t)\big)\, \mathbf{c}\big(\mathbf{r}(t), \mathbf{d}\big)\, dt$$
$$T(t) = \exp\left(- \int_{t_n}^{t} \sigma\big(\mathbf{r}(s)\big)\, ds \right) \tag{1}$$

where $T(t)$ is the accumulated *transmittance*, i.e. the probability that the ray reaches depth t without being occluded [3].

In practice, the integral in (1) is estimated with stratified sampling. Drawing N sample points $\{t_i\}_{i=1}^{N}$ along the ray and defining $\delta_i = t_{i+1} - t_i$, the discrete colour estimate is

$$\hat{C}(\mathbf{r}) = \sum_{i=1}^{N} T_i \big(1 - e^{-\sigma_i \delta_i}\big)\, \mathbf{c}_i, \quad T_i = \exp\Big(-\sum_{j=1}^{i-1} \sigma_j \delta_j\Big), \tag{2}$$

with $\sigma_i = \sigma\big(\mathbf{r}(t_i)\big)$ and $\mathbf{c}_i = \mathbf{c}\big(\mathbf{r}(t_i), \mathbf{d}\big)$. Because the compositing rule 2 is differentiable, the parameters Θ can be optimised end-to-end by minimising the squared difference between rendered and ground-truth pixel colours over all training images [3].

b) Learning objective and parameters.: Let $\mathcal{R}$ be the set of rays sampled from the training images and $C^\star(\mathbf{r})$ their ground-truth colours. The loss minimized is

$$\mathcal{L}(\Theta) = \sum_{\mathbf{r}\in\mathcal{R}} \big\| \hat{C}(\mathbf{r}) - C^\star(\mathbf{r}) \big\|_2^2. \tag{3}$$

Optimisation is performed with stochastic gradient descent (Adam), updating *only* Θ; no explicit voxel grids or mesh vertices are stored, which makes NeRF a memory-efficient continuous scene representation [3].

Together, Figure 1 and Figure 2 illustrate the training pipeline: sparse calibrated photographs provide colour targets, camera rays define the sampling domain, and the learnable parameters Θ encode the scene's continuous geometry and appearance. Once trained, F_Θ can render photorealistic novel views by evaluating 3 for any desired camera pose.

B. 3D Gaussian Splatting

3-D Gaussian Splatting (3DGS) represents a scene as an unstructured set of *volumetric splats*—anisotropic Gaussian ellipsoids that are projected to screen space and blended with classical α-compositing (cf. Fig. 3).

Each splat is defined by a *mean position* $\boldsymbol{\mu} \in \mathbb{R}^3$, a full covariance matrix $\Sigma \in \mathbb{R}^{3\times3}$, an opacity $\alpha \in [0, 1)$, and a set of spherical–harmonic (SH) coefficients $\mathbf{c} = (c_{\ell m})_{\ell \le L}$ that encode view-dependent colour. These quantities constitute the learnable parameters optimised from multi-view imagery:

$$\theta = \big\{\boldsymbol{\mu},\ \mathbf{s},\ \mathbf{q},\ \alpha,\ \mathbf{c}\big\},$$

where Σ is re-parameterised by a scale vector $\mathbf{s} \in \mathbb{R}^3$ and a rotation quaternion $\mathbf{q}$ to guarantee positive-definite covariances during training [5].

a) Gaussian volume: A single 3-D Gaussian evaluates to

$$G(\mathbf{x}) = \exp\Big[-\tfrac{1}{2}\big(\mathbf{x} - \boldsymbol{\mu}\big)^\top \Sigma^{-1} \big(\mathbf{x} - \boldsymbol{\mu}\big)\Big]; \tag{4}$$

the covariance is assembled from scale and rotation as

$$\Sigma = R\, S\, S^\top R^\top, \tag{5}$$

where $S = \mathrm{diag}(\mathbf{s})$ and $R = R(\mathbf{q})$ is the orthonormal matrix induced by $\mathbf{q}$. For rendering, Σ is projected into camera space:

$$\Sigma' = J\, W\, \Sigma\, W^\top J^\top, \tag{6}$$

with the Jacobian J of the local projective transform and the world-to-view matrix W [5].

b) Visibility-aware α-blending: After depth-sorting the splats, colour is accumulated along each pixel ray via

$$\begin{aligned} C &= \sum_{i=1}^{N} T_i\, \alpha_i\, \mathbf{c}_i, \\ \alpha_i &= 1 - \exp(-\sigma_i \delta_i), \\ T_i &= \prod_{j<i} (1 - \alpha_j) \end{aligned} \tag{7}$$

which is equivalent to volumetric ray-marching but requires only one evaluation per Gaussian. Here σ_i denotes the local density implied by the Gaussian, and δ_i is the ray-segment length intersecting that splat [5].

c) Optimisation: Training alternates gradient-based updates of θ with adaptive densification (Fig. 4): Gaussians with large position gradients are *cloned*, while oversized splats in high-variance regions are *split* into finer ones. The differentiable tile-based rasteriser delivers real-time forward and backward passes and removes any cap on the number of contributing splats per pixel, enabling high-fidelity, interactive novel-view synthesis [5].

The compact, explicit nature of 3DGS therefore combines the optimisation flexibility of continuous radiance fields with the rendering efficiency of point-based graphics, achieving state-of-the-art image quality.

C. Dense Neural Point Clouds

Dense Neural Point Clouds (DNPCs) serve as a hybrid 3D–implicit representation in which *continuous* scene appearance and geometry are anchored to a *discrete*, data-adaptive set of points. Formally, the map at time t is a set of N neural points

$$\mathcal{P} = \big\{(\mathbf{p}_i,\, \mathbf{f}_i^g,\, \mathbf{f}_i^c) \;\big|\; i = 1, \ldots, N\big\}, \tag{8}$$

where $\mathbf{p}_i \in \mathbb{R}^3$ is the 3-D location, while $\mathbf{f}_i^g, \mathbf{f}_i^c \in \mathbb{R}^{32}$ are learnable **geometric** and **color** feature descriptors, respectively. Points are inserted on–the-fly with a density that follows image-space gradients, so high-frequency regions (edges, textures) receive many anchors whereas textureless surfaces are represented sparsely [6].

a) Ray sampling and feature interpolation: For each camera ray with origin $\mathbf{O}$ and direction $\mathbf{d}$, M samples

$$\mathbf{x}_i = \mathbf{O} + z_i \mathbf{d}, \qquad i = 1, \ldots, M, \tag{9}$$

are drawn in the depth band surrounding an observed sensor depth D. At every sample $\mathbf{x}_i$ the K nearest neural points inside a radius $2r$ are gathered, and their features are fused by inverse–squared-distance weighting

$$\begin{aligned} \mathbf{f}^g(\mathbf{x}_i) &= \frac{\sum_{k=1}^{K} w_k \mathbf{f}_k^g}{\sum_{k=1}^{K} w_k}, \\ \mathbf{f}^c(\mathbf{x}_i) &= \frac{\sum_{k=1}^{K} w_k \mathbf{f}_k^c, x_i}{\sum_{k=1}^{K} w_k}, \\ w_k &= \frac{1}{\|\mathbf{p}_k - \mathbf{x}_i\|_2^2}. \end{aligned} \tag{10}$$

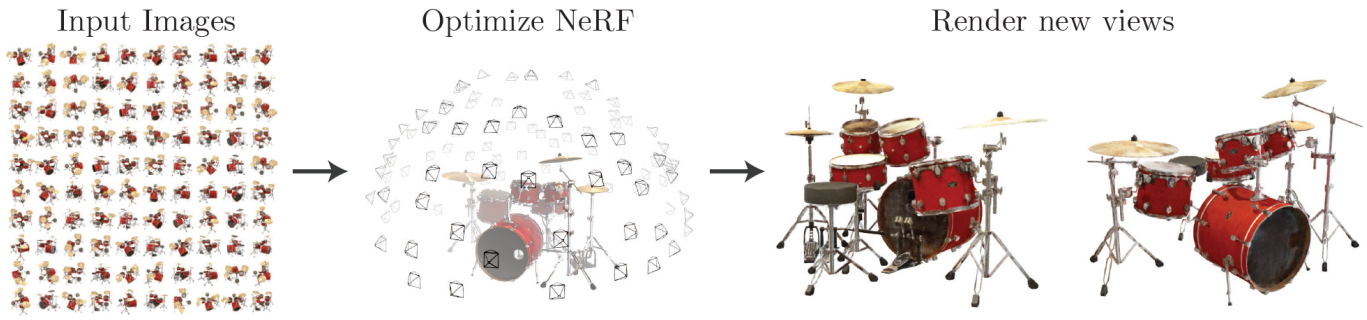

Fig. 1. NeRF learns a continuous 5-D radiance field from sparse input images. The network is queried along camera rays and trained by differentiable volume rendering [3].

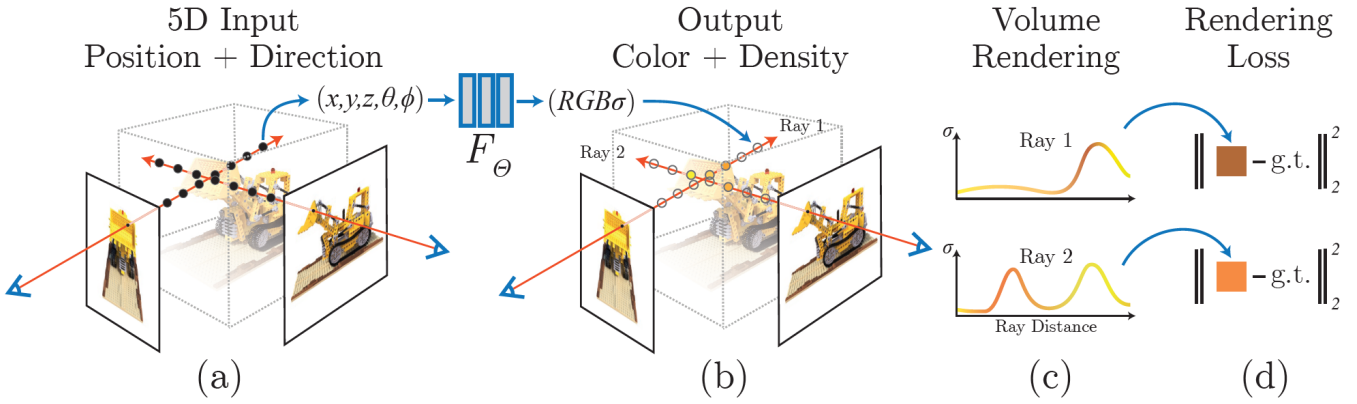

Fig. 2. Differentiable rendering pipeline used to supervise the radiance field. Samples along each ray are evaluated by the MLP and composited according to 2 [3].

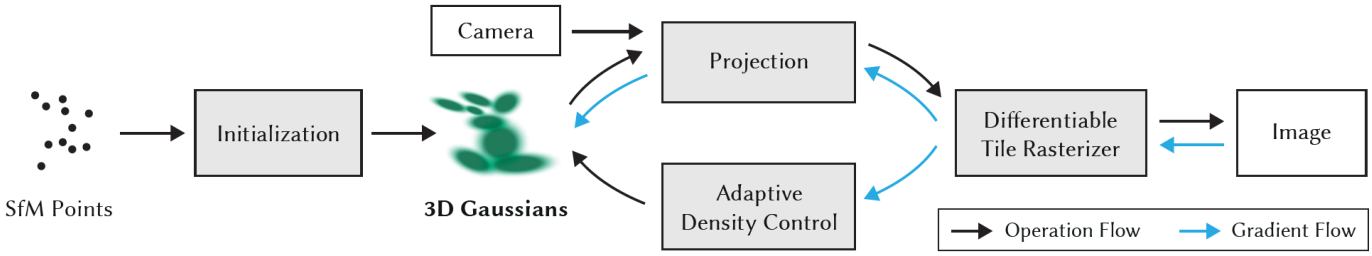

Fig. 3. Pipeline overview of 3-D Gaussian Splatting showing the initialization from sparse Structure-from-Motion (SfM) points [4], adaptive density control, and the differentiable tile-based rasterizer [5].

with an analogous expression for the color feature $\mathbf{f}^c(\mathbf{x}_i)$ after a small non-linear "appearance" transform $F_\theta(\cdot)$ [6].

b) Neural decoding: Two shallow multilayer perceptrons (MLPs) convert these aggregated features into density and colour:

$$o_i = h_\phi\big(\mathbf{x}_i, \mathbf{f}^g(\mathbf{x}_i)\big), \tag{11}$$

$$\mathbf{c}_i = g_\xi\big(\mathbf{x}_i, \mathbf{f}^c(\mathbf{x}_i)\big), \tag{12}$$

where h_ϕ (occupancy decoder) and g_ξ (color decoder) share a sinusoidal positional encoding of $\mathbf{x}_i$ [6].

c) Volume compositing: Per-ray termination probabilities

$$\alpha_i = o_i \prod_{j<i}(1 - o_j) \tag{13}$$

yield rendered depth $\hat{D}$ and colour $\hat{\mathbf{I}}$ via

$$\hat{D} = \sum_{i=1}^{M} \alpha_i z_i, \qquad \hat{\mathbf{I}} = \sum_{i=1}^{M} \alpha_i \mathbf{c}_i. \tag{14}$$

d) Learnable parameters: DNPC training proceeds *solely* at test time by minimizing an RGB-D re-rendering loss [6].

The optimised variables are:

- the point-wise feature vectors $\{\mathbf{f}_i^g, \mathbf{f}_i^c\}$,
- the weights ξ of the color decoder g_ξ,
- the weights θ of the appearance MLP F_θ,

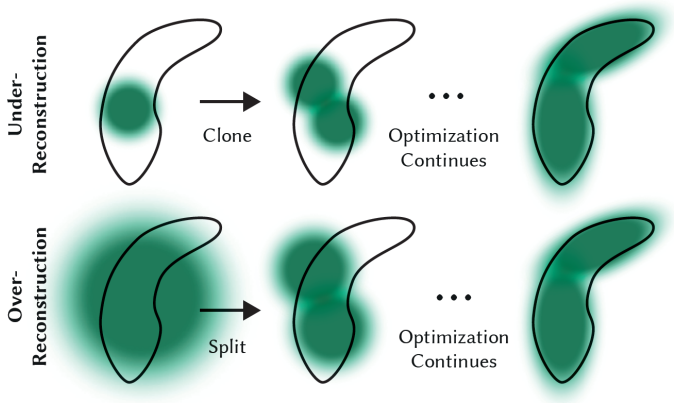

Fig. 4. Adaptive densification. Top: under-reconstruction is addressed by cloning and displacing small Gaussians along their gradient; bottom: over-reconstruction is handled by splitting large Gaussians into two smaller ones [5].

III. Methods

A. GlORIE-SLAM

GlORIE-SLAM (Globally Optimized RGB-only Implicit Encoding SLAM) is a dense SLAM system designed to work with only RGB inputs, achieving globally consistent camera tracking and high-fidelity scene reconstruction. It extends the Dense Neural Point Cloud (DNPC) paradigm introduced in Sec. II-C, but introduces several innovations to enable mapping and tracking without depth sensors [1].

a) Neural Scene Representation: The scene is represented by a deformable neural point cloud

$$\mathcal{P} = \{(\mathbf{p}_i, \mathbf{f}_i^g, \mathbf{f}_i^c, k_i, u_i, v_i, D_i) \,|\, i = 1, \ldots, N\}, \tag{15}$$

where $\mathbf{p}_i \in \mathbb{R}^3$ denotes 3D point location, $\mathbf{f}_i^g, \mathbf{f}_i^c \in \mathbb{R}^{32}$ are geometric and color features, and k_i, (u_i, v_i), D_i are the keyframe index, originating image coordinates, and initial depth, respectively. These points deform during global updates to preserve consistency across LCs and BAs [1].

b) Mapping with Proxy Depth: Since no depth sensor is used, GlORIE-SLAM builds a *proxy depth map* D_c for each keyframe by fusing a noisy tracker-estimated depth $\widetilde{D}_c$ with a monocular depth prior D_c^{mono}. Reliable depth values are extracted via multi-view consistency checks and fused with scaled and shifted monocular predictions:

$$D_c(u,v) = \begin{cases} \hat{D}_c(u,v), & \text{if consistent,} \\ \theta_c D_c^{\text{mono}}(u,v) + \gamma_c, & \text{otherwise.} \end{cases} \tag{16}$$

The parameters (θ_c, γ_c) are optimized using least squares fitting [1].

c) Tracking via Optical Flow, Disparity, Scale and Pose Optimization: Camera pose tracking in GlORIE-SLAM is performed via frame-to-frame optical flow estimation, inspired by DROID-SLAM [7]. A recurrent network predicts dense optical flow between incoming frames and previously selected keyframes. These flow predictions are used to construct a factor graph $G = (V, E)$, where each node $v_i \in V$ corresponds to a keyframe with associated camera pose $\omega_i \in \mathrm{SE}(3)$ and disparity map d_i, and each edge $e_{ij} \in E$ stores the optical flow constraints between keyframes i and j.

To jointly optimize poses and geometry, GlORIE-SLAM first applies a dense BA layer that minimizes photometric reprojection error using the Mahalanobis-weighted cost:

$$\min_{\omega,d} \sum_{(i,j)\in E} \left\| \tilde{\mathbf{p}}_{ij} - K\omega_j^{-1} \left(\omega_i \frac{1}{d_i} K^{-1} [\mathbf{p}_i, 1]^T \right) \right\|_{\Sigma_{ij}}^2, \tag{17}$$

where $\tilde{\mathbf{p}}_{ij}$ is the predicted pixel location in keyframe j corresponding to pixel $\mathbf{p}_i$ in keyframe i, K is the camera intrinsic matrix, and Σ_{ij} is a diagonal confidence matrix encoding flow uncertainty.

However, since GlORIE-SLAM operates in an RGB-only setting, it must resolve scale ambiguities. To address this, the system introduces the *Disparity, Scale and Pose Optimization (DSPO)* layer. This module refines camera poses ω_i, disparity maps d_i, and learns a per-keyframe scale θ_i and shift γ_i to align the inverse of the monocular depth prior D_i^{mono} with the optimized disparity. The DSPO loss is defined as:

$$\begin{aligned} \min_{\theta,\gamma,d^h} & \sum_{(i,j)\in E} \left\| \tilde{\mathbf{p}}_{ij} - K\omega_j^{-1} \left(\omega_i \frac{1}{d_i^h} K^{-1} [\mathbf{p}_i, 1]^T \right) \right\|_{\Sigma_{ij}}^2 \\ & + \alpha_1 \sum_i \left\| d_i^h - (\theta_i (1/D_i^{\text{mono}}) + \gamma_i) \right\|^2 \\ & + \alpha_2 \sum_i \left\| d_i^l - (\theta_i (1/D_i^{\text{mono}}) + \gamma_i) \right\|^2 \end{aligned} \tag{18}$$

where d_i^h and d_i^l are the high-error and low-error components of the disparity map (determined via multi-view consistency), and $\alpha_1 < \alpha_2$ are weighting parameters that prioritize reliable disparities during scale-shift estimation.

Optimization alternates between DBA and DSPO steps to iteratively refine the trajectory and geometry. The scale and shift parameters (θ_i, γ_i) are initialized via least squares fitting (see Eq. 16) and are updated only during DSPO phases. This two-stage tracking strategy improves robustness under depth ambiguity and enables globally consistent optimization of camera poses [1].

d) Rendering and Optimization: Color and depth rendering use volume compositing (as detailed in Sec. II-C), but samples are drawn around the proxy depth to reduce computation. Rendered depth $\hat{D}$ and color $\hat{\mathbf{I}}$ are supervised via photometric and geometric re-rendering losses:

$$\mathcal{L} = \lambda_{\text{geo}} \|\hat{D} - D_c\|_1 + \lambda_{\text{color}} \|\hat{\mathbf{I}} - \mathbf{I}_c\|_1 + \lambda_{\text{pix}} \|\mathbf{I}_c - \mathbf{I}_k(u', v')\|_1. \tag{19}$$

Neural features $\mathbf{f}_i^g, \mathbf{f}_i^c$ and decoder weights are optimized during test time [1].

e) Data Requirements and Preparation: GlORIE-SLAM requires only an RGB video stream and intrinsic camera parameters. It leverages an off-the-shelf monocular depth estimator to generate the depth prior used in proxy depth construction and DSPO optimization. Specifically, the method uses the Dense Prediction Transformer (DPT) [8] as the

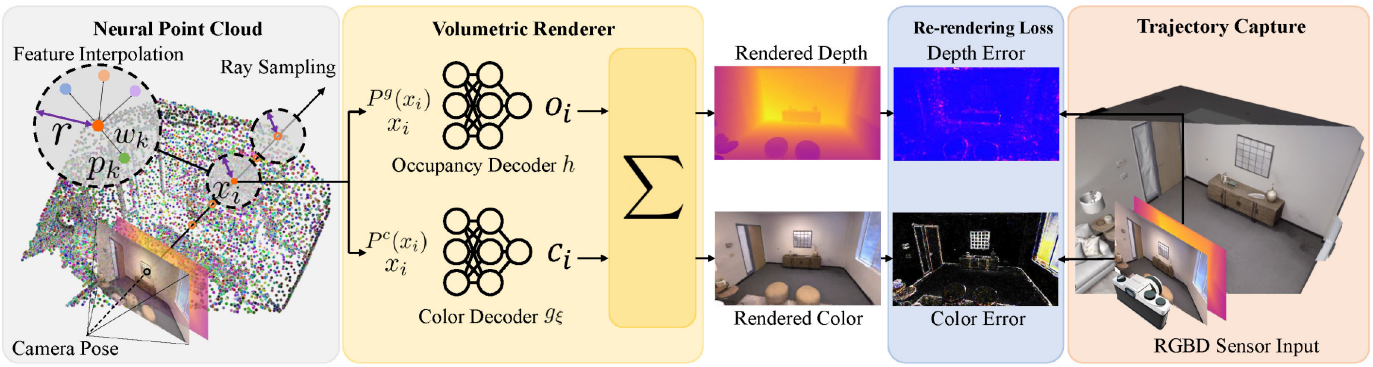

Fig. 5. Dense Neural Point Cloud pipeline (adapted from [6]). For each incoming RGB-D frame new neural points are spawned (orange) and collectively optimised by rendering-based losses that couple mapping and tracking.

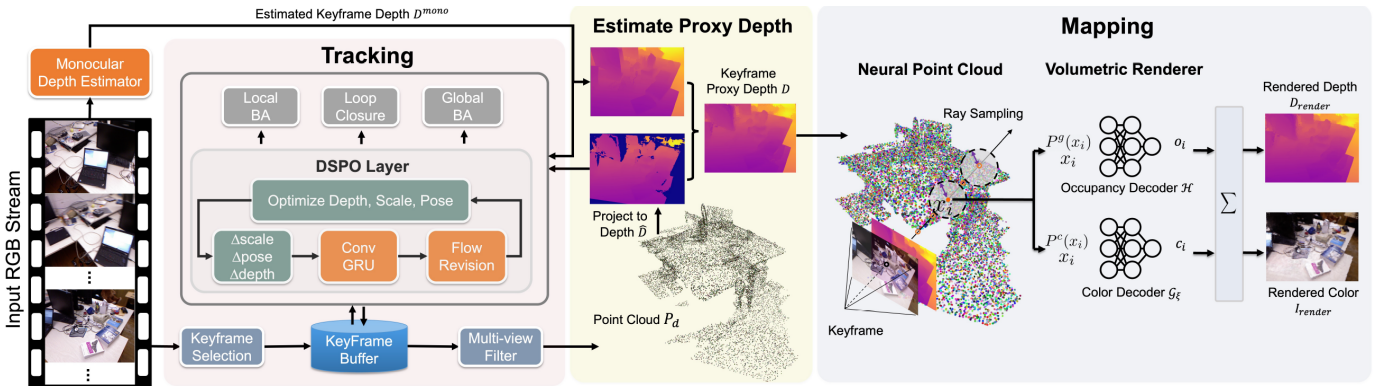

Fig. 6. GlORIE-SLAM architecture (adapted from [1]). The system performs RGB-only tracking and mapping using a deformable neural point cloud and integrates monocular priors via a novel Disparity, Scale and Pose Optimization (DSPO) layer for global optimization.

monocular depth estimator. This model provides dense depth predictions from single RGB images and is not fine-tuned during SLAM operation. The entire GlORIE-SLAM pipeline remains self-supervised at test time, automatically selecting keyframes based on optical flow magnitude and performing mapping and tracking without ground truth depth or scene-specific training [1].

B. HI-SLAM2

HI-SLAM2 is a geometry-aware monocular SLAM system that achieves fast and accurate scene reconstruction using only RGB input. The method combines learning-based dense visual tracking with 3DGS for scene representation. Unlike conventional SLAM pipelines that rely on map-centric representations, HI-SLAM2 follows a hybrid design that decouples tracking and mapping, later performing joint optimization to refine both camera poses and map geometry. The reconstruction pipeline of HI-SLAM2 comprises four main components (see Figure 7): (i) online tracking using a recurrent network for pose and depth estimation, (ii) loop closing via pose graph bundle adjustment (PGBA) in Sim(3) (Similarity Transformation Group in 3D), (iii) continuous mapping using 3DGS, and (iv) offline refinement through joint optimization of camera poses and Gaussian primitives [2].

a) Scene Representation: HI-SLAM2 leverages 3D Gaussian Splatting as described in subsection II-B. Each Gaussian unit is parameterized by a mean $\boldsymbol{\mu}_i \in \mathbb{R}^3$, a covariance matrix Σ_i, an opacity α_i, and a view-dependent color $\mathbf{c}_i$. The covariance is expressed via a scale-rotation decomposition:

$$\Sigma_i = R_i S_i S_i^\top R_i^\top, \tag{20}$$

where S_i is a diagonal scale matrix and R_i is the rotation matrix derived from a quaternion. Projection and differentiable rendering follow the rasterization equations from subsection II-B, using per-ray α-blending [2].

b) Hybrid Tracking and Mapping: Tracking is based on a dense optical flow network [7], generating frame-to-frame correspondences and estimating camera poses $T_i \in \mathrm{SE}(3)$ (Special Euclidean Group in 3D) and depth maps d_i. Keyframes are selected based on optical flow displacement. For improved depth accuracy, monocular priors (depth and normals) are incorporated from a pretrained network [8].

To address scale inconsistencies in monocular priors, HI-SLAM2 introduces a spatially-varying grid-based scale alignment strategy. For a depth prior $\hat{d}_i$ and scale grid coefficients $\mathbf{s}_i$ at a certain pixel $\mathbf{p}$, the aligned depth is:

$$\hat{d}_i^{\text{aligned}}(\mathbf{p}) = \hat{d}_i(\mathbf{p}) \cdot B_i(\mathbf{p}, \mathbf{s}_i), \tag{21}$$

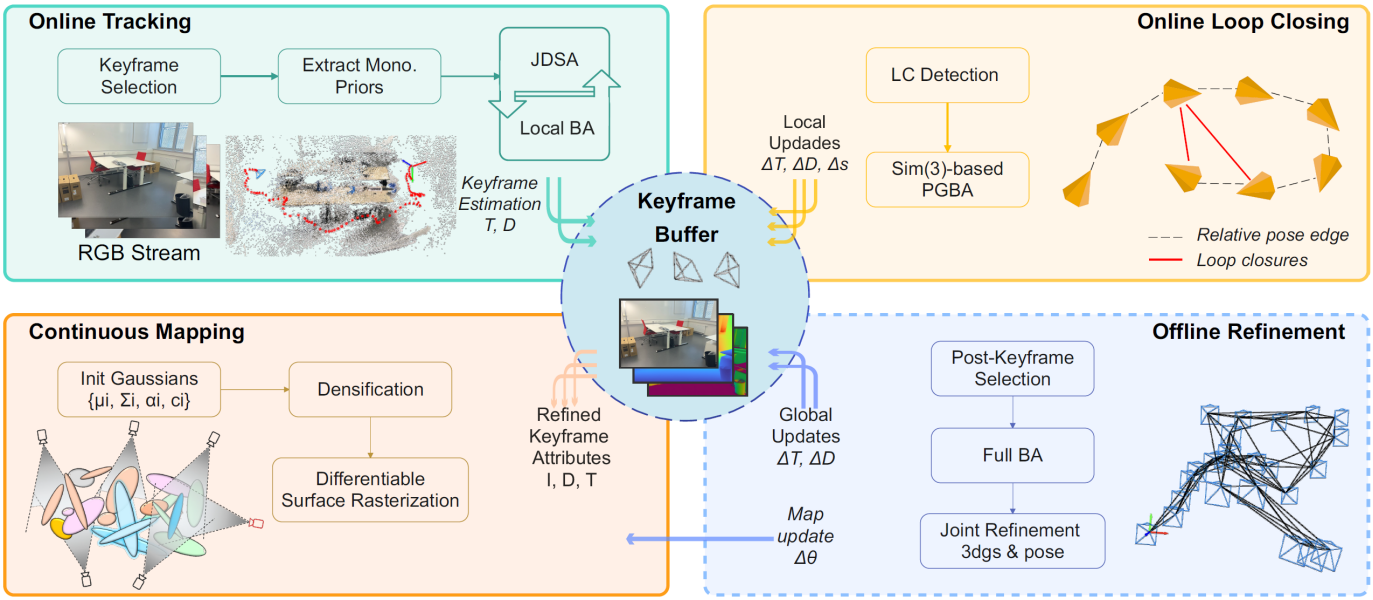

Fig. 7. HI-SLAM2 system overview. The architecture comprises online camera tracking, LC with PGBA, continuous 3D Gaussian mapping, and offline refinement of both geometry and appearance [2].

where B_i is bilinear interpolation over the scale grid. These scale-aligned priors are integrated in a joint depth and scale alignment (JDSA) module, optimizing both $\mathbf{s}_i$ and d_i while keeping the tracking stable [2].

c) Map Updates and Joint Optimization: LC is handled via Sim(3)-based PGBA, correcting scale drift and ensuring global consistency. During pose updates, the 3D Gaussian parameters are transformed accordingly:

$$\begin{aligned} \boldsymbol{\mu}'_j &= (T_i'^{-1} T_i) \cdot \boldsymbol{\mu}_j / s_i, \\ R'_j &= R_i'^{-1} R_i R_j, \quad s'_j = s_i \cdot s_j. \end{aligned} \tag{22}$$

This maintains consistency between map and updated poses. The final refinement stage jointly optimizes all camera poses, Gaussian parameters, and exposure compensation to enhance rendering quality and geometric accuracy [2].

d) Rendering and Optimization: The volume rendering and compositing of HI-SLAM2 follow the same principles as described in subsection II-A and subsection II-B. Specifically, the expected color along a camera ray is computed using transmittance-aware α-blending, and optimization is performed via differentiable rasterization. Loss functions combine photometric error, depth consistency, and normal supervision to guide the learning process [2]:

$$\mathcal{L} = \lambda_c \mathcal{L}_{\text{color}} + \lambda_d \mathcal{L}_{\text{depth}} + \lambda_n \mathcal{L}_{\text{normal}} + \lambda_s \mathcal{L}_{\text{scale}}.$$

e) Data Preparation: During online tracking in HI-SLAM2, the RGB input is processed into a sequence of keyframes using a learning-based dense SLAM approach [7] that estimates both camera poses and depth maps. Each incoming RGB frame is evaluated, and if selected as a keyframe (based on optical flow thresholds), monocular priors such as depth and surface normals are extracted. These priors are used to guide the optimization process. A keyframe graph is constructed to represent co-visibility relations, and local BA is performed to refine camera poses and depths using per-pixel optical flow correspondences. To address the scale inconsistency of monocular depth, a novel grid-based scale alignment strategy (JDSA) is introduced, which adjusts depth priors with spatially-varying scale coefficients for accurate alignment. During LC, the system continuously checks for LC candidates by comparing optical flow and orientation metrics between new and earlier keyframes. If a match is found, a Sim(3)-based PGBA is performed to correct both pose and scale drift. This involves optimizing relative pose constraints derived from refined dense correspondences while using Sim(3) transformations to account for scale inconsistencies. The LC updates are efficiently propagated to the 3D Gaussian map by deforming its elements according to the refined keyframe poses, ensuring that the scene representation remains globally consistent in real-time [2] [9] [10].

IV. Experiments

A. Datasets

1) Replica: The Replica dataset is a collection of 18 highly photorealistic 3D reconstructions of indoor environments designed to support research in machine learning, computer vision, and embodied AI. Each scene features dense geometry, high-dynamic-range (HDR) textures, semantic class and instance annotations, and accurately modeled mirror and glass reflectors. Created using a custom RGB-D capture rig and advanced reconstruction techniques, Replica achieves an unprecedented level of realism, enabling active perception and realistic rendering from arbitrary viewpoints [11].

TABLE I
RESULTS OF THE GLORIE-SLAM METHOD (MEASURED VALUES HIGHLIGHTED WITH BEST (GREEN), WORST (RED) AND CLOSEST-TO-AVERAGE (ORANGE)).

Dataset	Subset	PSNR ↑	SSI ↑	LPIPS ↓	Computation time [h:min:sec] ↓	Nr. frames
Replica	office0	33.939	0.972	0.113	3:51:17	2000
	office1	36.084	0.986	0.089	3:13:38	2000
	office2	26.446	0.957	0.193	4:24:32	2000
	office3	26.402	0.951	0.170	3:47:48	2000
	office4	28.751	0.955	0.214	4:07:54	2000
	room0	27.015	0.945	0.174	3:36:58	2000
	room1	28.201	0.944	0.190	4:31:33	2000
	room2	28.484	0.945	0.192	4:08:31	2000
	Avg.	29.415	0.957	0.167	03:57:46	–
TUM RGB–D	desk	16.845	0.715	0.652	1:12:27	595
	desk2	15.578	0.614	0.764	3:33:32	639
	room	15.798	0.607	0.746	8:28:37	1360
	xyz	21.014	0.855	0.357	1:20:02	3666
	household	16.923	0.622	0.745	2:39:32	2509
	Avg.	17.232	0.683	0.653	03:26:50	–

TABLE II
RESULTS OF THE HI-SLAM2 METHOD (MEASURED VALUES HIGHLIGHTED WITH BEST (GREEN), WORST (RED) AND CLOSEST-TO-AVERAGE (ORANGE)).

Dataset	Subset	PSNR ↑	SSI ↑	LPIPS ↓	Acc.[cm] ↓	Comp.[cm] ↓	Comp. Rat[%] ↑	Computation time [min:sec] ↓	Nr. of frames
Replica	office0	42.641	0.984	0.021	0.014	0.021	0.894	2:42	2000
	office1	43.621	0.984	0.021	0.011	0.032	0.856	2:27	2000
	office2	36.812	0.969	0.042	0.020	0.044	0.864	2:52	2000
	office3	36.867	0.969	0.036	0.018	0.040	0.878	2:32	2000
	office4	38.524	0.972	0.043	0.019	0.044	0.853	2:55	2000
	room0	35.231	0.959	0.041	0.015	0.034	0.902	2:29	2000
	room1	36.655	0.965	0.047	0.013	0.031	0.888	2:56	2000
	room2	38.204	0.973	0.037	0.018	0.038	0.886	2:50	2000
	Avg.	38.569	0.972	0.036	0.016	0.035	0.878	2:43	-
TUM RGB-D	desk	21.210	0.797	0.197	1.566	10.363	0.002	20:36	595
	desk2	21.132	0.794	0.237	1.214	13.210	0.019	21:10	639
	room	20.923	0.773	0.266	1.248	16.867	0.014	26:47	1360
	xyz	24.207	0.819	0.133	6.559	171.178	-	18:10	3666
	household	21.921	0.824	0.186	1.248	13.268	0.046	11:56	2509
	Avg.	21.878	0.801	0.204	2.367	44.977	0.017	19:44	-
HM	first	15.181	0.640	0.610	2.393	788.925	0.015	10:44	1857
	second	15.846	0.667	0.559	7.010	1647.933	0.016	10:53	2616
	Avg.	15.514	0.654	0.584	4.702	1218.429	0.016	10:49	-

2) TUM RGB-D: The TUM RGB-D dataset, curated by the Computer Vision Group at the Technical University of Munich, is a comprehensive benchmark specifically designed for evaluating RGB-D visual odometry and SLAM systems and contains a variety of indoor scenes. It comprises synchronized color (RGB) and depth streams captured at 640×480 resolution and 30Hz using a Microsoft Kinect sensor, alongside highly accurate ground-truth trajectories acquired via an external motion-capture system at 100Hz. Complementing the image data, the dataset includes inertial measurements from the Kinect, intrinsic and distortion calibration parameters [12].

3) HM Hochschule München University of Applied Sciences: Besides the Replica and the TUM RGB-D datasets two captures from the onboard camera of autonomes model racing car from HM were used to examine the better performing method. The data was captured as part of the "Intelligente autonome Systeme (Projektstudium)" lecture from the monocular camera capturing the route ahead of the car. The capturing **060525_SLAM_1_2** was selected as the first dataset and the capturing **260525_SLAM_LONG_VARIOUS_SPEED_2** was selected as the second dataset. To be able to use the captures the monocular videos had to pe preprocessed before which was done with COLMAP software tool that provides a SfM [9] and Multi-View Stereo (MVS) pipeline [10]. A monocular video is first decomposed into an ordered set of overlapping frames. In COLMAP the frames pass through three principal stages. At first **incremental SfM** [9] detects SIFT (Scale-Invariant Feature Transform) features, matches them sequentially/exhaustively, and incrementally registers the cameras, yielding a sparse point cloud and per-frame poses [13]. SIFT features are distinctive points in an image that remain consistent even when the image is scaled, rotated, or subjected to changes in illumination. Second **MVS** [10] takes this calibrated model, undistorts the images, estimates per-view depth/normal maps with PatchMatch stereo, and fuses them into a dense coloured point cloud stored as fused.ply [13]. Next the **Surface reconstruction** converts the fused cloud to a mesh with either the Poisson or a Delaunay mesher, producing

files such as meshed-poisson.ply or meshed-delaunay.ply that conform to the PLY mesh format [13]. Exporting any of these .ply files delivers a 3-D representation of the scene that subsequent learning-based methods can use as ground-truth geometry or camera-pose initialization. The [10]

B. Evaluation metrics

The Structural Similarity Index (SSIM), Peak Signal-to-Noise Ratio (PSNR), and Learned Perceptual Image Patch Similarity (LPIPS) are complementary full-reference image-quality metrics that evaluate similarity between a reference image and a test image, yet each approaches this task differently. SSIM assesses visual fidelity by comparing luminance, contrast, and structural correlations across local regions—capturing perceptually important changes in image structure rather than relying solely on pixel-level differences—with values ranging from –1 to 1, where 1 indicates near-perfect structural similarity. PSNR, by contrast, quantifies signal fidelity by computing the mean squared error (MSE) between images and converting it into a logarithmic decibel scale—the higher the PSNR, the less distortion—though it often correlates poorly with human perception, particularly when comparing different codecs. LPIPS adopts a learned, perceptual approach: it passes images through convolutional neural networks trained on human judgments, measures the distance between deep feature activations of corresponding patches, and outputs a scalar similarity score where lower values mean greater perceptual similarity, offering improved alignment with human subjective assessments. Together, these metrics provide a nuanced toolkit for scientific image-quality assessment: SSIM for structural fidelity, PSNR for signal-level accuracy, and LPIPS for perceptual alignment with human vision. In evaluating reconstruction quality, the three standard geometric metrics are: Accuracy (Acc.), Completeness (Comp.), and Completeness Ratio (Comp. Rat.). Accuracy measures the average Euclidean distance from each point in the reconstructed model to its nearest neighbor in the ground truth surface, reflecting how precisely the reconstruction aligns with the actual scene geometry. Completeness, conversely, computes the average distance from each ground truth point to its nearest point in the reconstruction, indicating how much of the scene was successfully captured. Finally, the Completeness Ratio reports the percentage of ground truth points that are reconstructed within a predefined threshold distance, quantifying the proportion of the scene that is accurately covered. Lower values for Accuracy and Completeness indicate better geometric precision and coverage, respectively, while a higher Completeness Ratio reflects a more complete and faithful reconstruction.

C. GlORIE-SLAM

For testing the GlORIE-SLAM approach with several datasets a machine with a specific GPU (graphics processing unit) was required for two reasons: At first the GlORIE-SLAM paper mentions that the experiments carried out were run on a NVIDIA GeForce RTX 3090 with 24 GiB GPU memory what is a GPU with a Compute Capability of 8.6 [14]. This circumstance means that a machine with a GPU of the same Compute Capability and with 24 GiB or more of GPU memory is required to be able to repeat the experiments from the GlORIE-SLAM and get the same results. Secondly the corresponding repository of the GlORIE-SLAM paper uses a complex structure of software from third parties and package managers as conda and pip, what reasons in an software environment which is finely tuned on specific versions of software packages as CUDA (What is the parallel computing platform NVIDIA provides for its GPUs) or PyTorch and hardware of a certain generation. To fulfill the requirements of GlORIE-SLAM a remote machine with a NVIDIA GeForce RTX A6000 with 48 GiB GPU memory was rented from runpod.io. Further the rented machine came with 50 GiB of RAM and a 9 core CPU (central processing unit). Unfortunately, due to issues with the CUDA toolkit version 11.8, the accuracy, completeness, and completeness rate for the GlORIE-SLAM approach could not be measured, which partially weakens the comparability with the HI-SLAM2 approach. It also limits the validity of the modeling accuracy of GlORIE-SLAM. Due to that issue it was not considered useful to evaluate the GlORIE-SLAM approach with the HM dataset (subsubsection IV-A3) too.

D. HI-SLAM2

For testing the HI-SLAM2 approach with several datasets a machine with a specific GPU (graphics processing unit) was required for the same reasons as for GlORIE-SLAM. The experiments of the HI-SLAM2 paper were run on a Nvidia RTX 4090 with 24 GiB GPU memory what is a GPU of Compute Capability 8.9 [14]. Further the corresponding repository of the HI-SLAM2 paper is build on software from third parties and package managers too similar to GlORIE-SLAM. To fulfill the requirements of HI-SLAM2 a remote machine with a NVIDIA GeForce RTX 4090 with 24 GiB GPU memory was rented from runpod.io. Further the rented machine came with 60 GiB of RAM and a 9 core CPU also.

V. Conclusion

This study conducted a comparative evaluation of two RGB SLAM methods—**GlORIE-SLAM** and **HI-SLAM2**—using three datasets of varying complexity and origin: the synthetic **Replica** dataset, the real-world **TUM RGB-D** dataset, and the monocular **HM dataset** captured at Hochschule München. The evaluation employed a combination of perceptual (PSNR, SSIM, LPIPS) and geometric (Accuracy, Completeness, Completeness Ratio) metrics as detailed in the *Evaluation Metrics* subsection IV-B, offering a holistic view of visual and structural performance.

Performance Summary

As presented in Table I and Table II, HI-SLAM2 consistently outperformed GlORIE-SLAM across all evaluated datasets and metrics:

- On the **Replica dataset**, HI-SLAM2 achieved an average PSNR of **38.569** and LPIPS of **0.036**, outperforming GlORIE-SLAM's **29.415 PSNR** and **0.167 LPIPS** (Tables I and II). Additionally, HI-SLAM2 showed good geometric reconstruction with an average **accuracy of 0.016 cm**, **completeness of 0.035 cm**, and **completeness ratio of 87.8%**, while GlORIE-SLAM did not report geometric metrics. The computation time further favored HI-SLAM2, with average processing times under 3 minutes per sequence versus nearly 4 hours for GlORIE-SLAM.
- For the **TUM RGB-D dataset**, where real-world depth noise and motion irregularities are more prominent, HI-SLAM2 again demonstrated superiority with perceptual metrics (PSNR: **21.878**, LPIPS: **0.204**) over GlORIE-SLAM (PSNR: **17.232**, LPIPS: **0.653**). Geometric accuracy, while less precise than in Replica due to the dataset's challenges, remained within acceptable ranges for HI-SLAM2 (accuracy: **2.367 cm**).
- On the monocular **HM dataset**, only HI-SLAM2 was evaluated. Despite relatively low image quality scores (PSNR: **15.514**, LPIPS: **0.584**), HI-SLAM2 demonstrated the flexibility to operate on monocular data after COLMAP-based preprocessing (subsubsection IV-A3), with efficient computation times ($\sim$11 minutes per capture). The ability to generalize to such input further emphasizes HI-SLAM2's robustness, especially when considering the difficulties inherent in the HM dataset. Initially, the capturing camera does not focus on a specific scene or object; instead, it is positioned on top of an autonomous model race car and captures the road ahead of it. Secondly the HM dataset is not entirely static, what is caused by the positioning of the camera as well because at the bottom of the camera image the camera captures a part of the model race car so every captured image contains that part of the car what can cause a negative inmpact on the preprocessing of the data and the training of the model afterwards.

Strengths and Weaknesses

GlORIE-SLAM's primary strength lies in its potential visual fidelity in synthetic environments under controlled conditions. For example, on the `office1` subset of Replica, it achieved top-tier metrics (PSNR: **36.084**, SSIM: **0.986**, LPIPS: **0.089**). However, its major drawbacks include:

- **Prohibitively high computation time** ($\sim$4 hours per sequence),
- **Lack of geometric reconstruction outputs**,
- **Poor scalability** to real-world or monocular datasets (e.g., no evaluation on HM),
- **Inconsistent performance** across different scenes, with some subsets (e.g., `office3`, `room1`) performing significantly worse than average.

It is also important to note that, due to technical incompatibilities with CUDA toolkit version 11.8, the **accuracy, completeness, and completeness ratio for GlORIE-SLAM could not be measured**. This omission partially weakens the comparability with HI-SLAM2 and limits the validity of any conclusions about GlORIE-SLAM's modeling accuracy.

HI-SLAM2, in contrast, exhibits clear strengths:

- **Good perceptual and geometric accuracy**, particularly on the Replica dataset,
- **Efficient runtimes**
- **Robustness to real-world conditions**
- **Consistent performance** across different scenes and environments.

Its weaknesses are relatively minor but include:

- A drop in reconstruction quality when applied to noisy or unconstrained real-world datasets (e.g., geometric completeness on `xyz` in TUM),
- Slightly higher geometric errors in monocular applications (e.g., HM dataset's accuracy: **4.7 cm**).

Overall Assessment

In conclusion, while GlORIE-SLAM demonstrates some promise in idealized, synthetic conditions, its heavy computational requirements and limited generalization severely constrain its practical applicability. In contrast, HI-SLAM2 provides a **much more balanced and scalable solution**, offering high-fidelity visual reconstructions, accurate geometry, and broad compatibility with different input modalities and hardware setups. Because 3DGS models use an explicit representation of the scene with Gaussians,they typically have fewer trainable parameters than NeRFs which usually use large MLPs. Each Gaussian has a small, fixed number of parameters (e.g., position, scale, color), while NeRFs implicitly model the entire scene with MLPs. This makes Gaussian Splatting more parameter-efficient and often faster to train and render as the assessment has shown. Future work may explore hybrid approaches that combine GlORIE's structural preservation strengths with HI-SLAM2's efficiency and geometric precision, especially in settings requiring both perceptual and spatial accuracy.

VI. Acknowledgement

I want to say thank you to my supervisor Prof. Dr. Alfred Nischwitz and Andrea Maurer for their commitment to finance this work.

VII. Tools

- **runpod.io** was used to execute the experiments on remote machines with specific hardware on both methods
- **gofile.io** was used to transfer data from and to the remote machines
- **overleaf.com** was used to write this report
- **grammarly.com** was used to improve writing style and readability
- **Chat GPT-4o** was used for support in literature analysis, remote machine management, working with LaTeX, coding and debugging
- **GitHub Copilot** was used for support in coding and debugging

- **Visual Studio Code** was the used development environment
- **zotero.org** was used for the source management

REFERENCES

[1] G. Zhang, E. Sandström, Y. Zhang, M. Patel, L. V. Gool, and M. R. Oswald, "GlORIE-SLAM: Globally Optimized RGB-only Implicit Encoding Point Cloud SLAM," May 2024, arXiv:2403.19549 [cs]. [Online]. Available: http://arxiv.org/abs/2403.19549

[2] W. Zhang, Q. Cheng, D. Skuddis, N. Zeller, D. Cremers, and N. Haala, "HI-SLAM2: Geometry-Aware Gaussian SLAM for Fast Monocular Scene Reconstruction," Apr. 2025, arXiv:2411.17982 [cs]. [Online]. Available: http://arxiv.org/abs/2411.17982

[3] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis," Aug. 2020, arXiv:2003.08934 [cs]. [Online]. Available: http://arxiv.org/abs/2003.08934

[4] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: exploring photo collections in 3D," *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 835–846, Jul. 2006. [Online]. Available: https://dl.acm.org/doi/10.1145/1141911.1141964

[5] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3D Gaussian Splatting for Real-Time Radiance Field Rendering," Aug. 2023, arXiv:2308.04079 [cs]. [Online]. Available: http://arxiv.org/abs/2308.04079

[6] E. Sandström, Y. Li, L. V. Gool, and M. R. Oswald, "Point-SLAM: Dense Neural Point Cloud-based SLAM," Sep. 2023, arXiv:2304.04278 [cs]. [Online]. Available: http://arxiv.org/abs/2304.04278

[7] Z. Teed and J. Deng, "DROID-SLAM: Deep Visual SLAM for Monocular, Stereo, and RGB-D Cameras," Feb. 2022, arXiv:2108.10869 [cs]. [Online]. Available: http://arxiv.org/abs/2108.10869

[8] A. Eftekhar, A. Sax, R. Bachmann, J. Malik, and A. Zamir, "Omnidata: A Scalable Pipeline for Making Multi-Task Mid-Level Vision Datasets from 3D Scans," Oct. 2021, arXiv:2110.04994 [cs]. [Online]. Available: http://arxiv.org/abs/2110.04994

[9] J. L. Schönberger and J.-M. Frahm, "Structure-from-Motion Revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[10] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm, "Pixelwise View Selection for Unstructured Multi-View Stereo," in *European Conference on Computer Vision (ECCV)*, 2016.

[11] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, A. Clarkson, M. Yan, B. Budge, Y. Yan, X. Pan, J. Yon, Y. Zou, K. Leon, N. Carter, J. Briales, T. Gillingham, E. Mueggler, L. Pesqueira, M. Savva, D. Batra, H. M. Strasdat, R. D. Nardi, M. Goesele, S. Lovegrove, and R. Newcombe, "The Replica Dataset: A Digital Replica of Indoor Spaces," Jun. 2019, arXiv:1906.05797 [cs]. [Online]. Available: http://arxiv.org/abs/1906.05797

[12] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A Benchmark for the Evaluation of RGB-D SLAM Systems," in *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[13] "COLMAP — COLMAP 3.12.0.dev | 1571fe00 (2025-06-25) documentation." [Online]. Available: https://colmap.github.io/index.html

[14] "NVIDIA CUDA GPU Compute Capability." [Online]. Available: https://developer.nvidia.com/cuda-gpus