

ToF-Splatting: Dense SLAM using Sparse Time-of-Flight Depth and Multi-Frame Integration

Andrea Conti^{1,2*} Matteo Poggi¹ Valerio Cambareri²
 Martin R. Oswald³ Stefano Mattocchia¹

¹University of Bologna, Italy ²Sony DepthSensing Solutions, Belgium ³University of Amsterdam, Netherlands



Figure 1. **Overview of our ToF-Splatting method.** Our method combines sparse ToF depth, multi-view geometry from a buffer of keyframes, and monocular cues (left) to perform into a unique end-to-end dense SLAM framework enabled by a Gaussian Splatting.

Abstract

Time-of-Flight (ToF) sensors provide efficient active depth sensing at relatively low power budgets; among such designs, only very sparse measurements from low-resolution sensors are considered to meet the increasingly limited power constraints of mobile and AR/VR devices. However, such extreme sparsity levels limit the seamless usage of ToF depth in SLAM. In this work, we propose ToF-Splatting, the first 3D Gaussian Splatting-based SLAM pipeline tailored for using effectively very sparse ToF input data. Our approach improves upon the state of the art by introducing a multi-frame integration module, which produces dense depth maps by merging cues from extremely sparse ToF depth, monocular color, and multi-view geometry. Extensive experiments on both real and synthetic sparse ToF datasets demonstrate the advantages of our approach, as it achieves state-of-the-art tracking and mapping performances on reference datasets.

1. Introduction

Simultaneous Localization and Mapping (SLAM) [6, 8, 36, 43] entails the joint estimation of the current camera pose

against a 3D scene representation (*i.e.*, the map) and the update of this representation with data from the current camera view. It is a fundamental task with applications in AR/VR, robotics, and autonomous navigation [14, 18]. In recent years, SLAM witnessed a revolution [33] following the success of Neural Radiance Fields (NeRF) [23] and 3D Gaussian Splatting (3DGS) [13] as scene representations from the adjacent field of novel view synthesis.

In this paper we focus on SLAM pipelines that leverage RGB-D video inputs, *i.e.*, where color images and dense depth maps are collected by synchronized, accurate color and depth sensors. Beyond global scale, depth sensors provide accurate local geometry that grants better tracking and mapping quality. A common choice for active depth sensing are Time-of-Flight (ToF) cameras. Despite the increased accuracy provided by reliable depth measurements, the power consumption and cost of such sensors constrain their adoption to high-end industrial or automotive applications. A recent trend to favor ToF sensor integration in handheld consumer devices is to reduce the specifications of ToF sensors to provide low-resolution but more reliable depth measurements. Among other examples, LiDAR (an instance of ToF) appears in consumer mobile phones [20] and is known to feature up to at most 576 dots (interpolated to 256×192 resolution via depth completion). Other recent ToF sensors provide only up to 64 depth points but consume a re-

* work started while visiting University of Amsterdam.

ported 200 mW [19]. In such settings a SLAM pipeline can expect to receive, *e.g.*, 8×8 very noisy, very sparse depth measurements per frame. These specifications prevent their straightforward integration in SLAM systems, as low-resolution noisy depth is known to harm the accuracy of dense SLAM systems built with NeRF and not originally designed to deal with this scenario [30, 36, 46]. This is even more evident with the more recent 3DGS-based SLAM systems [22], where accurate and dense depth is paramount to achieve satisfactory results.

To overcome these limitations, a straightforward strategy [38] would consist in recovering dense and accurate depth maps using a single-image depth completion framework [25]. However, models trained for this task are known to generalize poorly across domains and across input depth noise levels, potentially affecting the downstream SLAM pipeline accuracy and yielding both drifted camera trajectory and inaccurate mapping. Therefore, we explore 3DGS as an explicit representation that allows for injecting domain-specific knowledge and that is easily extended to, *e.g.*, dynamic scenes [21] and accurate large mesh reconstruction [2, 9]. On the one hand, we argue that depth completion alone is insufficient to allow sparse ToF depth usage in SLAM systems. On the other, we acknowledge that even sparse and noisy depth measurements can bootstrap SLAM if properly assisted with geometry. Accordingly, inspired by frame-to-frame SLAM systems relying on pre-trained trackers for localization, we believe that *multi-frame integration* fusing the noisy depth measurements with multi-view geometry across a small set of local frames could supply the dense SLAM system with the supervision necessary to fill this gap.

In this paper, we propose ToF-Splatting, a novel 3DGS-based SLAM system enabling dense reconstruction from sparse ToF sensors. ToF-Splatting harmoniously alternates tracking and mapping, with each one benefiting from the other. The former is carried out through backpropagation during the optimization of the 3DGS model on color images; the latter exploits a pre-trained network [3] as a multi-frame integration module, combining sparse depth measurements with multi-view geometry according to a buffer of keyframes. For each of the keyframes, we use the camera pose estimated through tracking and providing dense depth maps to improve the supervision given to the 3DGS model itself. Figure 1 shows an overview of the setting in which ToF-Splatting operates, as well as the quality of the 3D reconstructions it yields.

ToF-Splatting is tested on the real-world ZJUL5 dataset [19], where we measure tracking and mapping quality and find the superiority of our framework with respect to existing solutions built over sparse ToF sensors [38]. Moreover, standard benchmarks such as Replica [28] are also provided and confirm the effectiveness of our mapping strat-

egy. Our main **contributions** are as follows:

- We propose ToF-Splatting, the first 3DGS-based SLAM system suited for sparse ToF sensors.
- We introduce a novel multi-frame integration method combining sparse depth, multi-view geometry, and monocular cues for robust noise and outlier handling.
- We establish a new state-of-the-art for dense SLAM systems built on top of sparse ToF depth sensors.

2. Related Work

We briefly review the literature relevant to our work, referring to [33] for a detailed overview of the latest advances.

Traditional SLAM. Traditional visual SLAM pipelines, or the more recent ones introducing deep modules [15, 31, 32], usually perform tracking in frame-to-frame fashion, based on the visual features extracted from past frames either using handcrafted feature extractors [24] or deep networks [31, 32]. These systems are usually defined by four main steps: tracking, mapping, global bundle adjustment, and loop closure. Among the different representations used by traditional SLAM frameworks, depth points [1, 24], surfels [27], and volumetric representations [6] allows for achieving globally consistent 3D reconstruction.

NeRF-based SLAM. The advent of NeRF in the field of novel view synthesis also reached adjacent research areas, as well as SLAM a few years later. NeRF-based SLAM systems estimate camera poses and reconstruct dense meshes by modeling scenes via MLPs. iMAP [30] and NICE-SLAM [45] are the first SLAM frameworks following this approach, with the latter introducing dense feature grids supporting the MLPs. Point-SLAM [26] and Loopy-SLAM [17] switched feature grids with neural point embeddings, allowing for higher flexibility and correcting or adjusting local maps after global optimization, yet with slow processing prohibiting deployment in robotics. ESLAM [11] and Co-SLAM [34] deploy tri-planes and hash grid embeddings, improving both processing speed and reconstruction accuracy, with PLGSLAM [7] further improving the representation capacity in large indoor scenes. GO-SLAM [44] and KN-SLAM [37] implement frame-to-frame systems thanks to external trackers, yielding a good trade-off between tracking/mapping accuracy and frame rate.

3DGS-based SLAM. More recent advances in novel view synthesis brought to the development of new SLAM systems using 3D Gaussian Splatting [13] to represent the mapped scene. Several concurrent frameworks emerged built over 3DGS, such as MonoGS [22], GaussianSLAM [42], SplaTAM [12], and GS-SLAM [39]. Different from NeRF-based SLAM systems, these frameworks allow for higher interpretability and higher rendering speed.

Sparse ToF-based SLAM. Only lately, the use of more compact and energy-efficient depth sensors has gained attention in the SLAM research community. ToF-SLAM [38]

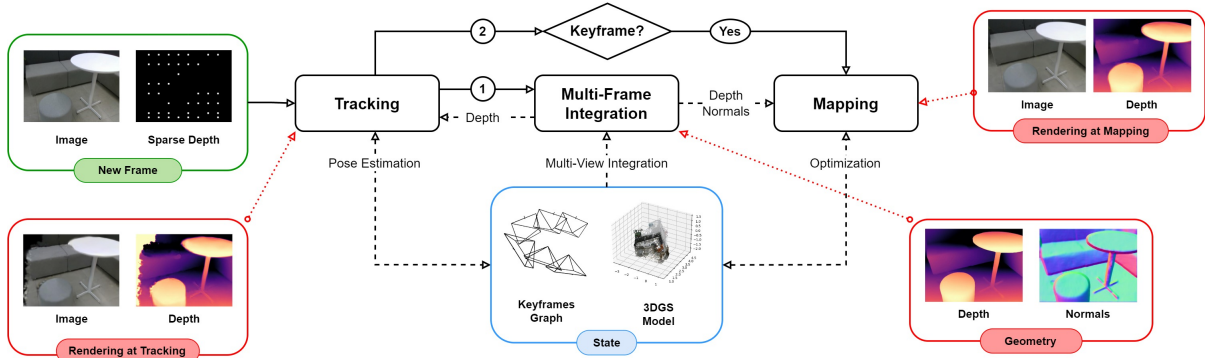


Figure 2. **ToF-Splatting Pipeline.** Our method involves three main modules: a Tracking frontend estimating camera poses, a Multi-Frame Integration module that predicts dense depth maps from sparse ToF measurements and multi-view geometry, and a Mapping backend modeling the 3D scene representation via 3D Gaussian Splatting.

represents the only attempt to exploit “lightweight”, sparse depth measurements within a dense SLAM system, aided by a depth completion model [16]. However, depth completion alone fails to deal with noise in the sparse input depth, a shortcoming we aim to specifically overcome within our ToF-Splatting framework, the first one based on 3DGS and suited for sparse ToF sensors.

3. Pipeline Architecture

In this section, we present the core components of our framework. ToF-Splatting is a frame-to-model SLAM system conceptually divided into three modules: (i) a mapping backend, (ii) a tracking frontend, and (iii) a peculiar multi-frame integration module yielding dense depth maps based on a subset of the keyframe graph, the current RGB frame, and the current sparse ToF depth map.

Figure 2 provides an overview of our pipeline. Our framework is characterized by three heterogeneous modules that are tightly related and influence each other. The tracking module stands as the most influencing element in the pipeline, as it selects the keyframes used by both the mapping and multi-frame integration parts, moreover, it computes the poses used for multi-frame depth estimation. Nonetheless, it relies on the quality of the mapping step to perform correct ego-motion estimation and exploits the depth cues to properly behave when photometric information is lacking or ambiguous.

The depth perception part is mainly sustained by the tracking step and highly influences the mapping part since it is used to seed new Gaussians and as supervision. Finally, the mapping phase not only creates a unique smooth representation but also defines the rendered opacity that is used for keyframe selection. These connections lead to a smoothly integrated framework that not only exploits a multi-frame backbone but more importantly, defines an effective way to use such information in the SLAM scenario.

3.1. Background: 3D Gaussian Splatting

3D Gaussian Splatting [13] (3DGS) represents one of the latest advances in novel view synthesis, fitting a dynamically defined set of multivariate Gaussian distributions over the scene to enable image rendering from arbitrary viewpoints. Each Gaussian $G(\mathbf{x})$ is parametrized by its mean $\boldsymbol{\mu} \in \mathbb{R}^3$ and covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{3 \times 3}$, reading

$$G(\mathbf{x}) = \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right) \quad (1)$$

Moreover, opacity $o \in \mathbb{R}$ and RGB color $\mathbf{c} \in \mathbb{R}^3$ are assigned to each Gaussian.

Given a set of images from different viewpoints all around the scene, the Gaussian parameters are optimized using the splatting technique [35]. With such a method, 3D Gaussians are projected through rasterization over the 2D image plane, containing a 2D Gaussian distribution

$$\boldsymbol{\mu}^{2D} = \pi(\mathbf{T}_{wc}\bar{\boldsymbol{\mu}}) \quad \boldsymbol{\Sigma}^{2D} = \mathbf{J}\mathbf{R}_{wc}\boldsymbol{\Sigma}\mathbf{R}_{wc}^\top\mathbf{J}^\top \quad (2)$$

where $\bar{\boldsymbol{\mu}}$ is $\boldsymbol{\mu}$ in homogeneous coordinates, $\mathbf{T}_{wc} \in \mathbb{R}^{4 \times 4}$, $\mathbf{R}_{wc} \in \mathbb{R}^{3 \times 3}$ are respectively the world-to-camera transformation and its rotational component, π is the projection matrix and $\mathbf{J} \in \mathbb{R}^{2 \times 3}$ is the Jacobian of the projective transformation [49]. The color C of a pixel is then given by the combination of the M weighted Gaussians, *i.e.*,

$$\mathbf{C} = \sum_{i \in M} \mathbf{c}_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j) \quad (3)$$

with $\alpha_i = o_i G(\mathbf{x}_i)$. Since $\boldsymbol{\Sigma}$ is positive semi-definite, it is parametrized with a diagonal scaling matrix \mathbf{S} and a rotation matrix \mathbf{R} , so that $\boldsymbol{\Sigma} = \mathbf{R}\mathbf{S}\mathbf{S}^\top\mathbf{R}^\top$. \mathbf{R} is internally represented with quaternions.

3.2. Multi-Frame Integration

Dense depth cues are pivotal for proper initialization of the Gaussians position $\boldsymbol{\mu} \in \mathbb{R}^3$, as well as to provide supervision to both tracking and mapping threads – as already

known in the literature [22, 47]. However, sparse ToF sensors alone are insufficient for this purpose, because of the very sparse and noisy depth measurements they provide (e.g., only 64 points for a sensor such as that used in [19]). To overcome this limitation, ToF-SLAM [19] exploits an auxiliary completion model, DELTAR [16], to recover dense depth maps out of ToF measurements. However, depth completion is highly dependent on the quality of the sparse depth in input, rapidly degrading the accuracy of the densified depth maps in the presence of noise.

In this paper, we follow a different path: inspired by frame-to-frame SLAM systems deploying an external tracker, we believe that multi-frame integration can compensate for the noise in the ToF measurements by exploiting both single-view and multi-view geometry cues on a set of local frames, given the camera poses estimated by the tracking frontend. We choose to extend the Depth on Demand (DoD) framework [3] for this purpose.

Multi-Frame Integration. DoD [3] integrates sparse depth and two-view stereo cues for dense depth prediction. Specifically – given a frame \mathbf{F}^k – it processes the associated RGB image I_k (the *target view*), another RGB view I_j from a different frame \mathbf{F}^j (a *source view*), the relative pose P^{jk} between the two, and the sparse depth points H^k to predict a dense metric depth map D^k . Differently from monocular depth prediction, DoD is aware of the scene scale either from H^k or P^{jk} , simplifying the problem of integrating scale-inconsistent geometries in the SLAM frontend. However, DoD is limited to two-view processing: this may lead to suboptimal results in the SLAM scenario, where an abundant amount of keyframes largely overlap and cover the same portion of the scene. Thus, we extend [3] to multi-view processing. This is achieved by observing that the predicted depth map D^k depends on the source view I_j through its relative pose P^{jk} only. DoD iteratively updates D^k for multiple iterations, exploiting this intrinsic property we can integrate multiple views (I_j, P^{jk}) using a different one at each iteration [4]. This approach allows smooth integration of multi-view cues avoiding order dependency. To select the source views I_j , at each prediction we order the keyframes selected by the SLAM pipeline according to their relative pose similarity and select the first N frames with baseline distance close to $b = 15\text{cm}$ to ensure enough parallax. The model is trained from scratch on ScanNetv2 [5] and following the protocol of [3].

Monocular Cues Integration. As shown in [3], DoD heavily relies on both geometry and depth measurements, thus struggling at generalizing in challenging scenarios where the two lack. This may happen in cases where the tight field of view, noisy sparse points, and textureless surfaces hamper multi-view matching. To mitigate such limitations, we integrate explicit monocular cues into DoD by feeding its monocular encoder with I_k and the normalized depth

map $\tilde{D}^k \in [0, 1]^{H \times W}$ obtained through the single-image depth estimator Depth Anything v2 [40]. This approach allows the injection of a robust bias, avoiding issues related to monocular estimation, e.g., slanted surfaces or noisy scale estimates, and delegating to DoD their handling. These cues are included while retraining on ScanNetv2 [5].

Outlier Handling. Furthermore, the noisy depth measurements in input to DoD can severely affect its accuracy. Therefore, we explicitly focus on dealing with outliers in the input sparse depth, that may be perceived by ToF devices over specific surfaces. Such errors are particularly evident in cheap sensors and may largely prejudice the information supplied by the few points they measure. Purposely, we deploy a simple yet effective method to identify such outliers by exploiting multi-view geometry again. Whenever depth needs to be predicted for a new frame, DoD predicts a depth map \hat{D}^k without processing the ToF measurements. Then, the ℓ_1 error between the latter and the depth measured by the ToF sensor H^k is computed and finally sparse measures with error higher than a given quantile q are discarded, and depth is predicted again by DoD by also processing the filtered sparse depth \tilde{H}^k this time. This way, we exploit multi-view cues to extract a clean depth prior to filtering out inconsistent sparse depth measures, and then we integrate the remaining points to improve the original prediction. Even without processing the sparse depth, DoD still predicts metric depth, as the metric scale is enforced by the relative camera poses.

3.3. Tracking Frontend

The tracking part of ToF-Splating estimates the ego-motion of the camera for each new frame \mathbf{F}^k and builds a keyframe graph $\mathbf{G} = \{\mathbf{KF}^k\} \subset \{\mathbf{F}^k\}$ containing a set of meaningful frames for mapping and multi-frame depth estimation.

Pose Estimation. We initialize each new frame \mathbf{F}^k of our frame-to-model tracker with the pose of the previous frame \mathbf{F}^{k-1} and then minimizing the tracking loss with respect to the relative pose between \mathbf{F}^k and \mathbf{F}^{k-1} , following [22]. The tracking loss consists of two terms, L_{rgb} and L_{depth} , respectively accounting for photometric and geometric errors

$$L_{\text{track}} = \lambda_{\text{track}} L_{\text{rgb}} + (1 - \lambda_{\text{track}}) L_{\text{depth}} \quad (4)$$

$$L_{\text{rgb}} = \|I_k - \hat{I}(\mathbf{F}^k)\|_1 \quad L_{\text{depth}} = \|D^k - \hat{D}(\mathbf{F}^k)\|_1 \quad (5)$$

where I_k is the RGB image from \mathbf{F}^k , $\hat{I}(\cdot)$ and $\hat{D}(\cdot)$ denote the rendered RGB and Depth and, D^k is the depth prediction produced by the multi-frame integration module.

Unlike other methods, the tracking process happens in two stages. Initially, for a fixed number of steps η_{rgb} only the photometric loss L_{rgb}^T is optimized. Then, such a pose is used for multi-frame integration, as described in Sec. 3.2, to generate D^k . Finally, both losses in Eq. (5) are used for η_{rgb}^T further steps.

Keyframe Selection Policy. We deploy a simple yet effective keyframe selection policy that considers i) novel content to be mapped and ii) instabilities in the already mapped areas. Unlike other methods using complex approaches – *e.g.*, frustum intersection, point clouds and pose analysis [22, 41] – we observe that the rendered opacity for a novel frame contains low opacity values where either Gaussians are not present or unstable. The first case highlights the presence of novel areas of the environment to be mapped, the second happens due to the pruning procedure involved in the mapping step (see Sec. 3.4). We set a frame \mathbf{F}^k as a keyframe \mathbf{KF}^k if its novelty factor ν^k is higher than a threshold ν^{th} , where we define ν^k as

$$\nu^k = \frac{\sum_{i \in \mathbf{KF}^k} \mathbf{M}(\mathbf{KF}_i^k)}{\sum_{i \in \mathbf{KF}^k} (1 - \mathbf{M}(\mathbf{KF}_i^k))} \quad (6)$$

$$\mathbf{M}(\mathbf{KF}_i^k) = \begin{cases} 1 & \text{if } \hat{O}(\mathbf{KF}_i^k) < \sigma \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

with $\hat{O}(\mathbf{KF}_i^k)$ being the rendered opacity at pixel i , and $\mathbf{M}(\mathbf{KF}_i^k)$ a binary uncertainty map.

This approach allows for skipping several frames when the camera moves slowly or focusing deeply on areas where the ToF-Splatting struggles the most, by mapping it consecutively. Nonetheless, we enforce mapping after skipping a certain number of frames to avoid mapping too few frames.

3.4. Mapping Backend

Whenever a new keyframe \mathbf{KF}^k is identified, mapping is triggered. This phase seeks to embed the new frame in the global 3DGS model, involving the following two steps.

Initialization. In this step, new Gaussians are seeded in the 3DGS model. To limit the size of the model and reduce outliers, only areas where the rendered uncertainty $\mathbf{M}(\mathbf{KF}^k)$ is high are seeded with new points, with the new Gaussians also being randomly downsampled by a constant factor. Each Gaussian is initialized using depth D^k predicted in the multi-frame integration step, with color from the RGB frame I_k , and scale initialized with a constant value. Unlike other 3DGS pipelines [22, 41], we prove this approach effective and particularly efficient, replacing any complex 3D heuristics to a simple use of rendered opacity $O(\cdot)$, already a side-product of the rasterization process.

Optimization. Firstly, ToF-Splatting smoothly integrates the new information in the global model and secondly tunes the existing representation. This is achieved by optimizing the new keyframe \mathbf{KF}^k and a random subset of 5 keyframes among the last N keyframes $\{\mathbf{KF}^j \mid j \geq \max(k - N, 0)\}$. These are optimized for a fixed number of steps η_{rgb}^M . Moreover, we enforce a sampling rate of 60% for \mathbf{KF}^k , with the remaining 40% for the remaining \mathbf{KF}^j to avoid forgetting. During this step, gradients are also back-propagated to the

camera poses of the sampled keyframes, thus acting as a global bundle adjustment. Finally, we prune Gaussians with opacities lower than 0.5 every $\eta_{\text{rgb}}^M/2$ steps. In addition to the terms in Eq. (5), we optimize structural similarity as in Eq. (8) and, following [2], the normals as in Eq. (9), *i.e.*,

$$L_{\text{dssim}} = \frac{1 - \text{SSIM}(I_k, \hat{I}(\mathbf{KF}^k))}{2} \quad (8)$$

$$L_{\text{normals}} = \|N^k - \hat{N}\|_1 + \mathbf{1} - \langle N^k, \hat{N} \rangle \quad (9)$$

with $\hat{N} := \bar{\nabla} \hat{D}(\mathbf{KF}^k)$, $N^k := \bar{\nabla} D^k$ denoting the normals estimated via 3D gradients $\bar{\nabla}$ for the rendered and multi-frame predicted depth respectively, $\langle \cdot, \cdot \rangle$ their inner product, and $\bar{\cdot}$ the average. L_{normals} is masked on edges to preserve sharpness, as identified by running a Sobel filter.

Finally, we introduce a regularization loss L_{iso} to penalize elongated Gaussians (*i.e.*, promote their isotropy) that usually leads to rendering artifacts as already observed in [22]. Eq. (10) defines L_{iso} , where $\text{diag}(\cdot)$ extracts the diagonal values from its argument, and $\text{diag}(\mathbf{S}_j)$ is the average of the resulting vector, *i.e.*, of the elements of the j -th diagonal scale matrix \mathbf{S}_j as defined in Sec. 3.1. Thus,

$$L_{\text{iso}} = \frac{1}{|\mathcal{G}|} \sum_j \|\text{diag}(\mathbf{S}_j) - \overline{\text{diag}(\mathbf{S}_j)} \cdot \mathbf{1}_{3 \times 1}\|_1 \quad (10)$$

The final mapping loss aggregates the aforementioned loss terms with corresponding weights as follows:

$$L_{\text{map}} = \lambda_{\text{map}} (\lambda_{\text{visual}} L_{\text{rgb}} + (1 - \lambda_{\text{visual}}) L_{\text{dssim}}) + (1 - \lambda_{\text{map}}) L_{\text{depth}} + \lambda_{\text{normals}} L_{\text{normals}} + \lambda_{\text{iso}} L_{\text{iso}} \quad (11)$$

4. Experiments

This section assesses both the tracking and mapping performance of ToF-Splatting on the following datasets.

ZJUL5. ZJUL5 [19] is the only existing public-domain real dataset providing sparse depth data from a low-resolution, low-cost ToF sensor. A VL53L5CX ToF sensor is assembled on a calibrated rig with an Intel RealSense 435i to provide dense depth ground truth. This benchmark comprises 7 diverse indoor scene recordings. Ground truth 3D meshes and camera poses are obtained following the ScanNet protocol [5]. This sensor provides 8×8 depth points by counting the number of photons returned in each discretized time range. Of the at most 64 depth points provided, many are extremely noisy and introduce large outliers. This challenging real-world dataset provides real measurement noise that is not accurately represented in mainstream datasets.

TUM RGB-D. This is a real dataset [29] providing indoor sequence captures and widely used as benchmark for RGB-D SLAM pipelines. The depth stream provides dense VGA depth maps from a high-resolution Kinect v1 depth sensor.

Method	Kitchen	Sofa	Office	Reception	Living Room	Office2	Sofa2	Avg.
KinectFusion [10]	✗	0.146	0.209	0.157	✗	0.321	0.125	0.192
iMAP [30]	✗	1.658	0.338	0.648	0.679	0.344	0.214	0.647
NICE-SLAM [46]	0.745	0.144	0.155	0.251	0.289	0.228	0.421	0.319
BundleFusion [6]	0.176	0.102	0.135	0.101	✗	0.163	0.120	0.132
ElasticFusion [36]	0.253	0.110	0.070	0.193	0.530	0.121	0.146	0.203
MonoGS [22]	0.231	0.032	0.041	0.044	0.153	0.035	0.073	0.087
ToF-SLAM [19]	0.113	0.081	0.056	0.114	0.200	0.101	0.085	0.107
ToF-Splatting (ours)	0.088	0.024	0.022	0.041	0.122	0.029	0.030	0.051

Table 1. **ZJUL5 Tracking Results.** We show here the tracking performance on the ZJUL5 dataset [19] dataset with the ATE RMSE [m] (\downarrow) metric on the 8 sequences available where ✗ indicates lost tracking. ToF-Splatting provides the better performance by a good margin on each sequence, demonstrating the optimal capability of our proposal to track the camera ego-motion accurately.

Replica. Replica [28] is a synthetic dataset providing extremely realistic indoor sequences. It provides dense depth maps, which we use at different densities to perform experiments that prove the robustness and generalization capabilities of ToF-Splatting in challenging scenarios.

Baselines. We compare our method with both learning-based [19, 30, 46] and traditional pipelines [6, 10, 36]. Following ToF-SLAM [19], baselines requiring RGB-D frames use DELTAR [16] to densify the sparse ToF measurements. Moreover, we adapt MonoGS [22] densifying depth with the state-of-the-art depth completion framework OGNI-DC [48] for a Gaussian Splatting SLAM baseline.

Implementation Details. In all our experiments, we use $N = 4$ source views I_j (*i.e.*, keyframe buffer elements) for the multi-view integration module and a baseline $b = 15\text{cm}$ for selecting keyframes. We use a quantile $q = 0.75$ to filter outliers that occur frequently in the sparse ToF depth of the ZJUL5 dataset [19]. Concerning tracking, we use a novelty factor threshold of $\nu_{th} = 0.1$, $\sigma = 0.98$. The total tracking loss uses $\lambda_{\text{track}} = 0.9$, a total number of $\eta_{\text{rgb}}^T = 30$ and $\eta_{\text{rgbd}}^T = 70$ steps are performed at each iteration. For mapping, we set the loss weights $\lambda_{\text{map}} = 60$, $\lambda_{\text{visual}} = 0.20$, $\lambda_{\text{normals}} = 0.01$, $\lambda_{\text{iso}} = 1.0$ and the number of steps per iteration $\eta_{\text{rgbd}}^M = 60$. We test on a single RTX3090.

4.1. Comparison with the State of the Art

In this section, we compare ToF-Splatting with existing methods in terms of tracking and mapping performances.

Tracking. In Table 1, we report the tracking performance by ToF-Splatting and various baselines on each sequence of the ZJUL5 dataset [19]. Following common practice we use the Absolute Trajectory Error metric [29], which directly compares the global consistency of a trajectory. While various frameworks [6, 10, 30] fail on the most challenging scenes, ToF-Splatting delivers accurate and consistent tracking. This is evident from the good margin it gains against the baselines. In Figure 3, we show the 3D trajectories on a subset of the scenes, with error below 3 cm.

Mapping. In Table 2, we report the mapping performance by ToF-Splatting compared with the other baselines. For each scene, we collect the predicted pose of each frame

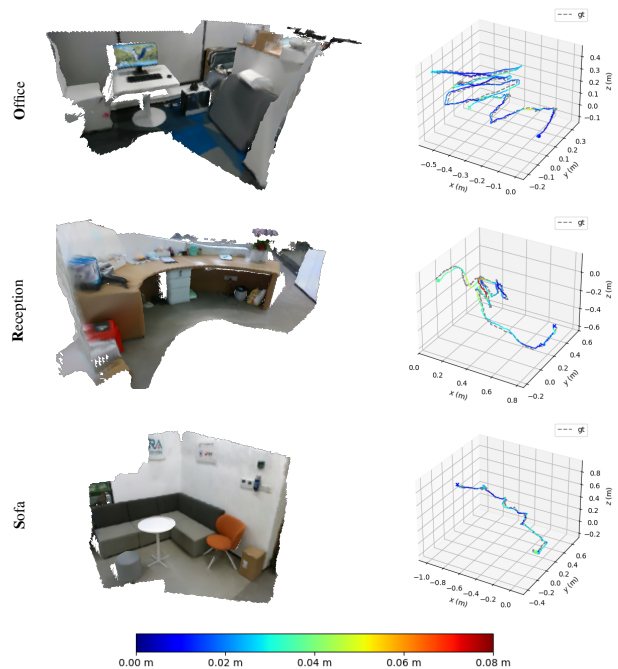


Figure 3. **Qualitative results on the ZJUL5 dataset [19].** We show meshes obtained by fusing rendered depth maps with TSDF and marching cubes (left), and 3D trajectories (right) on 3 scenes selected from the ZJUL5 dataset [19].

and render depth and color from 3DGS. Then, we perform TSDF integration and extract the final mesh through marching cubes. Following [19], the meshes are then evaluated by computing Accuracy, Completion, and F-score versus the ground truth meshes. Accuracy and Completion respectively evaluate the mean distance between each predicted vertex and the nearest ground truth one and vice versa. The F-score takes into account both Accuracy and Completion with an aggregate metric. ToF-Splatting delivers consistent results for mapping, always achieving the highest accuracy and significantly exceeding the baselines on average and specifically on the Sofa, Office, and Reception scenes which account for five over seven sequences. [19] is the second-best method and the main competitor in mapping.

Method	Metric	Kitchen	Sofa	Office	Reception	Living Room	Office2	Sofa2	Avg.
KinectFusion [10]	Accuracy ↓	✗	0.190	0.211	0.261	✗	0.267	0.135	-
	Completion ↓	✗	0.048	0.046	0.064	✗	0.078	0.064	-
	F-score ↑	✗	0.278	0.288	0.285	✗	0.274	0.381	-
ElasticFusion [36]	Accuracy ↓	0.092	0.135	0.084	0.297	0.151	0.096	0.122	0.140
	Completion ↓	0.065	0.048	0.082	0.305	0.216	0.147	0.047	0.130
	F-score ↑	0.553	0.420	0.529	0.274	0.382	0.416	0.481	0.436
BundleFusion [6]	Accuracy ↓	0.170	0.100	0.103	0.122	✗	0.121	0.123	-
	Completion ↓	0.088	0.030	0.038	0.057	✗	0.214	0.034	-
	F-score ↑	0.373	0.571	0.474	0.470	✗	0.442	0.527	-
iMAP [30]	Accuracy ↓	✗	0.135	0.229	0.365	0.225	0.233	0.139	-
	Completion ↓	✗	0.054	0.103	0.245	0.291	0.139	0.069	-
	F-score ↑	✗	0.445	0.315	0.238	0.170	0.255	0.416	-
NICE-SLAM [46]	Accuracy ↓	0.303	0.119	0.116	0.216	0.103	0.156	0.464	0.211
	Completion ↓	0.456	0.042	0.070	0.199	0.089	0.163	0.045	0.152
	F-score ↑	0.221	0.554	0.411	0.402	0.400	0.273	0.401	0.380
MonoGS [22]	Accuracy ↓	0.090	0.086	0.083	0.071	0.069	0.081	0.089	0.081
	Completion ↓	0.246	0.122	0.104	0.238	0.193	0.157	0.170	0.175
	F-score ↑	0.290	0.299	0.346	0.367	0.259	0.374	0.289	0.318
ToF-SLAM [19]	Accuracy ↓	0.081	0.068	0.067	0.079	0.078	0.113	0.121	0.087
	Completion ↓	0.071	0.041	0.045	0.062	0.122	0.085	0.033	0.066
	F-score ↑	0.559	0.661	0.646	0.643	0.496	0.557	0.656	0.604
ToF-Splatting (ours)	Accuracy ↓	0.064	0.031	0.032	0.041	0.059	0.034	0.043	0.043
	Completion ↓	0.095	0.038	0.029	0.056	0.131	0.054	0.046	0.064
	F-score ↑	0.527	0.791	0.840	0.710	0.359	0.779	0.642	0.664

Table 2. **ZJUL5 Mapping Results.** We perform the mapping evaluation on the ZJUL5 dataset (7 indoor sequences) and report results of three metrics including accuracy (Acc.), completion (Comp.), and F-score. The failure cases are marked as ✗.

Monocular Cues	Multi-View Cues	Metric	Kitchen	Sofa	Office	Reception	Living Room	Office2	Sofa2	Avg.
✓	✗	ATE ↓	0.103	0.054	0.046	0.050	0.110	0.041	0.055	0.066
		F-score ↑	0.216	0.675	0.564	0.645	0.241	0.611	0.557	0.501
✗	✓	ATE ↓	0.234	0.024	0.025	0.037	0.109	0.034	0.028	0.072
		F-score ↑	0.284	0.781	0.826	0.703	0.253	0.766	0.683	0.614
✓	✓	ATE ↓	0.088	0.024	0.022	0.041	0.122	0.029	0.030	0.051
		F-score ↑	0.527	0.791	0.840	0.710	0.359	0.779	0.642	0.664

Table 3. **Multi-Frame Integration Ablation Study.** Analysis of the impact of monocular and multi-view cues in the multi-frame integration step. Monocular cues are less impactful than multi-view ones, boosting performance in the challenging scenes where multi-view cues struggle most. Nonetheless, their combination yields the best results.

Method	LC	Input	Density	fr1/desk	fr2/xyz	fr3/office	Avg.
DROID-VO	✗	RGB	0.00%	0.052	0.107	0.073	0.077
MonoGS	✗	RGB	0.00%	0.038	0.046	0.035	0.040
ToF-Splatting	✗	RGB-D	0.02%	0.030	0.022	0.030	0.027
	0.04%		0.027	0.019	0.021	0.022	
ESLAM	✗	RGB-D	100%	0.025	0.011	0.024	0.020
Co-SLAM	✗	RGB-D	100%	0.024	0.017	0.024	0.022
CG-SLAM	✗	RGB-D	100%	0.024	0.012	0.024	0.020
RTG-SLAM	✗	RGB-D	100%	0.017	0.004	0.011	0.011
MonoGS	✗	RGB-D	100%	0.015	0.014	0.015	0.015
ORB-SLAM2	✓	RGB	0.00%	0.019	0.006	0.024	0.016
ORB-SLAM2	✓	RGB-D	100%	0.016	0.040	0.010	0.010

Table 4. **Results on TUM RGB-D dataset.** Top: competitors w/o loop-closure (LC); middle: most representative competitor with 100% density; bottom: competitors w/ loop-closure.

Notably, the other methods – like [6, 36] – may provide good performance on specific scenes but demonstrate unreliability on average usually failing in the particularly challenging Kitchen and Living Room scenes. In Figure 3, we show Office, Reception, and Sofa meshes on the left.

4.2. Ablation Studies

In this section, we perform additional experiments aimed at studying the impact of different factors on ToF-Splatting.

Multi-Frame Integration. In Table 3, we ablate the main components of the multi-frame integration and assess their contribution to the whole SLAM framework. By retaining monocular cues alone at the expense of multi-view geometry, we observe severe drops in both tracking and mapping accuracy, confirming the paramount importance of the latter; monocular cues alone are effective only where multi-view cues are ineffective – *e.g.*, in the absence of texture, as occurs mostly in the Kitchen scene.

TUM RGB-D. In Table 4, we simulate sparse ToF data by sampling 0.02% and 0.04% depth points, and compare ToF-Splatting with existing RGB and RGB-D methods. With as few as 0.02% of points, ToF-Splatting largely improves over RGB methods [22, 32], approaching the 100% density RGB-D approach of the most representative competitor [22] at only 0.04% density.

Replica: Input Depth Sparsity. We conduct additional studies on the Replica dataset, for which Figure 4 provides a qualitative overview of our 3D reconstructions. ToF-Splatting enables performing SLAM with very few sparse points – the 64 sparse points provided by a 8×8 sparse account only for 0.02% of a 640×480 image. We here

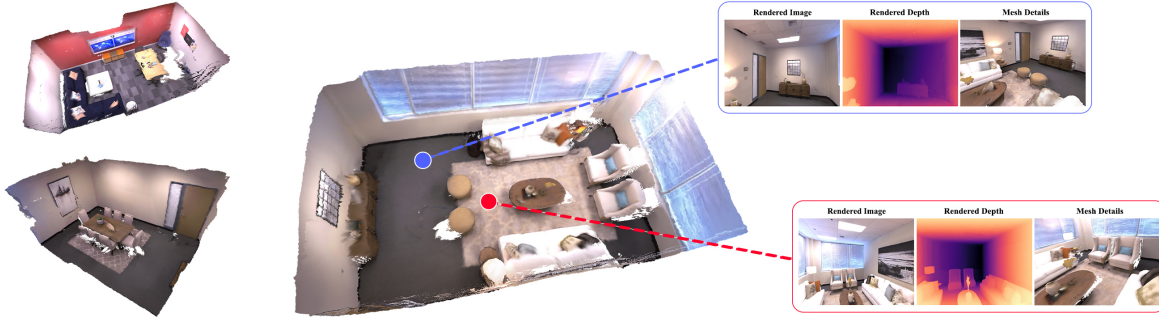


Figure 4. **Replica Qualitatives.** We provide qualitative results on Replica [28] to demonstrate the generalization capabilities of our method. On the left from top to bottom, meshes obtained respectively on scenes Office2 and Room2. On the right, details from the scene Room0. ToF-Splatting delivers accurate details and allows for nice photometric and depth rendering.

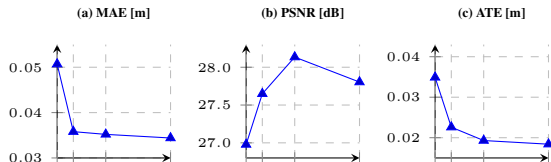


Figure 5. **Impact of depth sparsity.** We test on Replica [28] with different simulated depth sparsity levels to assess the capability to exploit higher input densities. MAE and ATE smoothly decrease, whereas rendering metrics appear to be less affected.

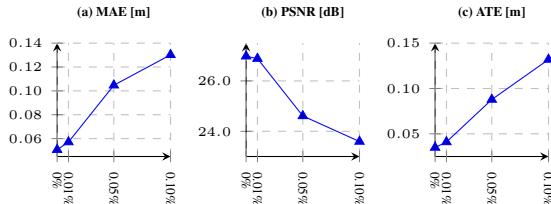


Figure 6. **Impact of depth noise.** We test on Replica [28] injecting different amounts of noise $\xi \in [0.00, 0.01, 0.05, 0.10]$. ToF-Splatting demonstrates to be effective at dealing with noise, with error increasing almost linearly with the injected amount of noise.

explore the impact of higher sparse points density on its overall performance. We simulate on Replica densities of $\{0.02\%, 0.04\%, 0.08\%, 0.16\%\}$ corresponding to about $\{64, 128, 256, 512\}$ points. Figure 5 shows the performance variation on both tracking and mapping metrics using the aforementioned sparsities.

Replica: Input Depth Noise. We study various noise levels impact in the input depth to assess our pipeline robustness. We model noise as additive but dependent on depth, injecting heteroscedastic Gaussian noise $\mathcal{N}(d, \epsilon d)$ where ϵ modulates the variance of the noise source on Replica. Figure 6 shows that MAE, PSNR, and ATE all expose linear degradation trends as ϵ increases.

Runtime. In Table 5 we measure the runtime of ToF-Splatting in comparison with the main baselines. The tracking time of ToF-Splatting already includes depth estimates with DoD, with average runtime $\bar{t}_{\text{DoD}} \approx 90 \text{ ms}$ (compris-

Method Name	Tracking Time	Mapping Time
iMAP [30]	101 ms	448 ms
NICE-SLAM [46]	470 ms	1300 ms
MonoGS [22]	1169 ms	650 ms
ToF-SLAM [19]	116 ms	380 ms
ToF-Splatting (ours)	614 ms	865 ms

Table 5. **Runtime Comparison.** We compare the runtime of ToF-Splatting and other learned baselines measuring their runtime for tracking and mapping. MonoGS [22] tracking time contains also the depth completion inference time [48].

ing DepthAnything v2 for monocular cues inference [40]). To date, 3DGS-based mapping methods are not capable of real-time performances. Nevertheless, exploiting the fact that the mapping step accounts only for just a few frames (typically 4% of the total in sequences of the ZJUL5 dataset [19]) and can be parallelized, the pipeline achieves 1.5 fps without further optimization.

Limitations. Even with state-of-the-art quality, ToF-Splatting manifests some issues that will be addressed in future developments. The most concerning is the runtime, which is too slow for real-time applications and requires memory and compute-intensive backpropagation at deployment. Even though fast 3DGS frameworks exist, such methodologies have not been transferred yet into the SLAM community, that being outside the scope of this work.

5. Conclusion

We presented the first 3DGS-based SLAM pipeline relying on sparse ToF depth sensing, as it provides accurate tracking and mapping from cheap and low-power cameras. Moreover, we showed how this result can be achieved through the integration of multi-view geometry, sparse depth data, and monocular cues, yielding an end-to-end 3DGS-based SLAM system. Finally, we assessed the effectiveness of our approach on the ZJUL5 and Replica datasets.

References

- [1] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. Orb-slam3: An accurate open-source library for visual, visual-inertial, and multi-map slam. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021. 2
- [2] Hanlin Chen, Fangyin Wei, Chen Li, Tianxin Huang, Yunsong Wang, and Gim Hee Lee. Vcr-gaus: View consistent depth-normal regularizer for gaussian surface reconstruction. *arXiv preprint arXiv:2406.05774*, 2024. 2, 5
- [3] Andrea Conti, Matteo Poggi, Valerio Cambareri, and Stefano Mattoccia. Depth on demand: Streaming dense depth from a low frame-rate active sensor. In *European Conference on Computer Vision (ECCV)*, 2024. 2, 4
- [4] Andrea Conti, Matteo Poggi, Valerio Cambareri, and S. Mattoccia. Range-agnostic multi-view depth estimation with keyframe selection. *2024 International Conference on 3D Vision (3DV)*, pages 1350–1359, 2024. 4
- [5] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5828–5839, 2017. 4, 5
- [6] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. BundleFusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics (ToG)*, 36(4): 1, 2017. 1, 2, 6, 7
- [7] Tianchen Deng, Guole Shen, Tong Qin, Jianyu Wang, Wentao Zhao, Jingchuan Wang, Danwei Wang, and Weidong Chen. Plgslam: Progressive neural scene representation with local to global bundle adjustment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19657–19666, 2024. 2
- [8] Zilong Dong, Guofeng Zhang, Jiaya Jia, and Hujun Bao. Keyframe-based real-time camera tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1538–1545. IEEE, 2009. 1
- [9] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. *CVPR*, 2024. 2
- [10] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. KinectFusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User Interface Software and Technology*, pages 559–568, 2011. 6, 7
- [11] Mohammad Mahdi Johari, Camilla Carta, and François Fleuret. Eslam: Efficient dense slam system based on hybrid representation of signed distance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17408–17419, 2023. 2
- [12] Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. Splatam: Splat track & map 3d gaussians for dense rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21357–21366, 2024. 2
- [13] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (TOG)*, 2023. 1, 2, 3
- [14] Georg Klein and David Murray. Parallel tracking and mapping for small AR workspaces. In *Proceedings of the IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 225–234. IEEE, 2007. 1
- [15] Yanyan Li, Nikolas Brasch, Yida Wang, Nassir Navab, and Federico Tombari. Structure-slam: Low-drift monocular slam in indoor environments. *IEEE Robotics and Automation Letters*, 5(4):6583–6590, 2020. 2
- [16] Yijin Li, Xinyang Liu, Wenqian Dong, Han Zhou, Hujun Bao, Guofeng Zhang, Yinda Zhang, and Zhaopeng Cui. Deltar: Depth estimation from a light-weight tof sensor and rgb image. In *European Conference on Computer Vision*, 2022. 3, 4, 6
- [17] Lorenzo Liso, Erik Sandström, Vladimir Yugay, Luc Van Gool, and Martin R Oswald. Loopy-slam: Dense neural slam with loop closures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20363–20373, 2024. 2
- [18] Haomin Liu, Guofeng Zhang, and Hujun Bao. Robust keyframe-based monocular SLAM for augmented reality. In *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality*, pages 1–10. IEEE, 2016. 1
- [19] Xinyang Liu, Yijin Li, Yanbin Teng, Hujun Bao, Guofeng Zhang, Yinda Zhang, and Zhaopeng Cui. Multi-modal neural radiance field for monocular dense slam with a light-weight tof sensor. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1–11, 2023. 2, 4, 5, 6, 7, 8
- [20] Gregor Luetzenburg, Aart Kroon, and Anders A. Bjørk. Evaluation of the Apple iPhone 12 Pro LiDAR for an Application in Geosciences. *Scientific Reports*, 11(1), 2021. 1
- [21] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. *2024 International Conference on 3D Vision (3DV)*, pages 800–809, 2023. 2
- [22] Hidenobu Matsuki, Riku Murai, Paul H.J. Kelly, and Andrew J. Davison. Gaussian splatting slam. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18039–18048, 2023. 2, 4, 5, 6, 7, 8
- [23] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European Conference on Computer Vision*, pages 405–421. Springer, 2020. 1
- [24] Raúl Mur-Artal and Juan D. Tardós. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Transactions on Robotics*, 33(5): 1255–1262, 2017. 2
- [25] Xin Qiao, Matteo Poggi, Pengchao Deng, Hao Wei, Chenyang Ge, and Stefano Mattoccia. Rgb guided tof imag-

- ing system: A survey of deep learning-based methods. *International Journal of Computer Vision*, pages 1–38, 2024. 2
- [26] Erik Sandström, Yue Li, Luc Van Gool, and Martin R Oswald. Point-slam: Dense neural point cloud-based slam. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18433–18444, 2023. 2
- [27] Thomas Schops, Torsten Sattler, and Marc Pollefeys. Bad slam: Bundle adjusted direct rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 134–144, 2019. 2
- [28] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 2, 6, 8
- [29] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 573–580. IEEE, 2012. 5, 6
- [30] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. iMAP: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6229–6238, 2021. 2, 6, 7, 8
- [31] Keisuke Tateno, Federico Tombari, Iro Laina, and Nassir Navab. Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6243–6252, 2017. 2
- [32] Zachary Teed and Jia Deng. Droid-SLAM: Deep visual SLAM for monocular, stereo, and RGB-D cameras. *Advances in Neural Information Processing Systems*, 34: 16558–16569, 2021. 2, 7
- [33] Fabio Tosi, Youmin Zhang, Ziren Gong, Erik Sandström, Stefano Mattoccia, Martin R. Oswald, and Matteo Poggi. How nerfs and 3d gaussian splatting are reshaping slam: a survey, 2024. 1, 2
- [34] Hengyi Wang, Jingwen Wang, and Lourdes Agapito. Co-slam: Joint coordinate and sparse parametric encodings for neural real-time slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13293–13302, 2023. 2
- [35] Yifan Wang, Felice Serena, Shihao Wu, Cengiz Öztireli, and Olga Sorkine-Hornung. Differentiable surface splatting for point-based geometry processing. *ACM Transactions on Graphics (TOG)*, 38:1 – 14, 2019. 3
- [36] Thomas Whelan, Renato F Salas-Moreno, Ben Glocker, Andrew J Davison, and Stefan Leutenegger. ElasticFusion: Real-time dense SLAM and light source estimation. *The International Journal of Robotics Research*, 35(14):1697–1716, 2016. 1, 2, 6, 7
- [37] Xingming Wu, Zimeng Liu, Yuxin Tian, Zhong Liu, and Weihai Chen. Kn-slam: Keypoints and neural implicit encoding slam. *IEEE Transactions on Instrumentation and Measurement*, 73:1–12, 2024. 2
- [38] Liu Xinyang, Li Yijin, Teng Yanbin, Bao Hujun, Zhang Guofeng, Zhang Yinda, and Cui Zhaopeng. Multi-modal neural radiance field for monocular dense slam with a lightweight tof sensor. In *International Conference on Computer Vision (ICCV)*, 2023. 2
- [39] Chi Yan, Delin Qu, Dan Xu, Bin Zhao, Zhigang Wang, Dong Wang, and Xuelong Li. Gs-slam: Dense visual slam with 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19595–19604, 2024. 2
- [40] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *ArXiv*, abs/2406.09414, 2024. 4, 8
- [41] Vladimir Yugay, Yue Li, Theo Gevers, and Martin R. Oswald. Gaussian-slam: Photo-realistic dense slam with gaussian splatting. *ArXiv*, abs/2312.10070, 2023. 5
- [42] Vladimir Yugay, Yue Li, Theo Gevers, and Martin R. Oswald. Gaussian-slam: Photo-realistic dense slam with gaussian splatting, 2023. 2
- [43] Guofeng Zhang, Jiaya Jia, Tien-Tsin Wong, and Hujun Bao. Recovering consistent video depth maps via bundle optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008. 1
- [44] Youmin Zhang, Fabio Tosi, Stefano Mattoccia, and Matteo Poggi. Go-slam: Global optimization for consistent 3d instant reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3727–3737, 2023. 2
- [45] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12786–12796, 2022. 2
- [46] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. NICE-SLAM: Neural implicit scalable encoding for SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12786–12796, 2022. 2, 6, 7, 8
- [47] Zihan Zhu, Songyou Peng, Viktor Larsson, Zhaopeng Cui, Martin R. Oswald, Andreas Geiger, and Marc Pollefeys. Nicer-slam: Neural implicit scene encoding for rgb slam. *2024 International Conference on 3D Vision (3DV)*, pages 42–52, 2023. 4
- [48] Yiming Zuo and Jia Deng. Ogni-dc: Robust depth completion with optimization-guided neural iterations. *arXiv preprint arXiv:2406.11711*, 2024. 6, 8
- [49] Matthias Zwicker, Hanspeter Pfister, Jeroen van Baar, and Markus H. Gross. Surface splatting. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001. 3