GSplatLoc : Ultra-Precise Pose Optimization via 3D Gaussian Reprojection

https://github.com/Atticuszz/GsplatLoc

Atticus Zhou, Atticus Zhou, Atticus Zhou, Atticus Zhou

August 7, 2024

Abstract

We present GSplatLoc, an innovative pose estimation method for RGB-D cameras that employs a volumetric representation of 3D Gaussians. This approach facilitates precise pose estimation by minimizing the loss based on the reprojection of 3D Gaussians from real depth maps captured from the estimated pose. Our method attains rotational errors close to zero and translational errors within 0.01mm, representing a substantial advancement in pose accuracy over existing point cloud registration algorithms, as well as explicit volumetric and implicit neural representation-based SLAM methods. Comprehensive evaluations demonstrate that GSplatLoc significantly improves pose estimation accuracy, which contributes to increased robustness and fidelity in real-time 3D scene reconstruction, setting a new standard for localization techniques in dense mapping SLAM.

1 Introduction

We present GSplatLoc, an innovative pose estimation method for RGB-D cameras that employs a volumetric representation of 3D Gaussians. This approach facilitates precise pose estimation by minimizing the loss based on the reprojection of 3D Gaussians from real depth maps captured from the estimated pose. Our method attains rotational errors close to zero and translational errors within 0.01mm, representing a substantial advancement in pose accuracy over existing point cloud registration algorithms, as well as explicit volumetric and implicit neural representation-based SLAM methods. Comprehensive evaluations demonstrate that GSplatLoc significantly improves pose estimation accuracy, which contributes to increased robustness and fidelity in real-time 3D scene reconstruction, setting a new standard for localization techniques in dense mapping SLAM.

2 Related Work

Accurate visual localization commonly relies on estimating correspondences between 2D pixel positions and 3D scene coordinates. Such approaches detect, describe [7,49], and match [32,46,48,73,81,96] local features, maintain an explicit sparse 3D representation of the environment, and sometimes leverage image retrieval [33,86] to scale to large scenes [32,57,70,75,82,87]. Recently, many of these components have been learned with great success [2,23,25,58,60,65,67,71,95], but often independently and not end-to-end due to the complexity of such systems. Here we introduce a simpler alternative to feature matching, finally enabling stable end-to-end training. Our solution can learn more powerful priors than individual blocks, yet remains highly flexible and interpretable. End-to-end learning for localization has recently received much attention. Common approaches encode the scene into a deep network by regressing from an input image to an absolute pose [35,37,59,66,90] or 3D scene coordinates [9,13,16,17,80]. Pose regression lacks geometric constraints and thus does not generalize well to novel viewpoints or appearances [76,78], while coordinate regression is more robust. Both do not scale well due to the limited network capacity [11,82] and require for each new scene either costly retraining or adaptation [16,17]. ESAC [11] improves the scalability by training an ensemble of regressors, each specialized in a scene subset, but is still significantly less accurate than feature-based methods in larger environments. Differently, some approaches regress a camera pose relative to one or more training images [5,24,42,97], often after an explicit retrieval step. They do no memorize the scene geometry and are thus scene-agnostic, but, similar to absolute regressors, are less accurate than feature-based methods [76,97]. Closer to ours, SANet [93] takes the scene representation out of the network by regressing 3D coordinates from an input 3D point cloud. Critically, all top-performing learnable approaches are at least trained per-dataset, if not per-scene, and are limited to small environments [37,80]. In this work we demonstrate the first end-to-end learnable network that generalizes across scenes, including from outdoor to indoor, and that delivers performance competitive with complex pipelines on large real-world datasets, thanks to a differentiable pose solver. Learning

camera pose optimization can be tackled by unrolling the optimizer for a fixed number of steps [21,51,53, 83,91,92], computing implicit derivatives [13,15,18,34,68], or crafting losses to mimic optimization steps [88,89]. Multiple works have proposed to learn components of these optimizers [21,51,83], with added complexity and unclear generalization. Some of these formulations optimize reprojection errors over sparse points, while others use direct objectives for (semi-)dense image alignment. The latter are attractive for their simplicity and accuracy, but usually do not scale well. Like their classical counterparts [26,38], they also suffer from a small basin of convergence, limiting them to frame tracking. In contrast, PixLoc is explicitly trained for wide-baseline cross-condition camera pose estimation from sparse measurements (Figure 2). By focusing on learning good features, it shows good generalization yet learns sensible data priors that shape the optimization objective.

3 Method

Overview: PixLoc localizes by aligning query and reference images according to the known 3D structure of the scene. The alignment consists of a few steps that minimize an error over deep features predicted from the input images by a CNN (Figure 3). The CNN and the optimization parameters are trained end-to-end from ground truth poses

Motivation: In absolute pose and scene coordinate regression from a single image, a deep neural network learns to i) recognize the approximate location in a scene, ii) recognize robust visual features tailored to this scene, and iii) regress accurate geometric quantities like pose or coordinates. Since CNNs can learn features that generalize well across appearances and geometries, i) and ii) do not need to be tied to a specific scene, and i) is already solved by image retrieval. On the other hand, iii) is tackled by classical geometry using feature matching [19,20,28] or image alignment [4,26,27,50] and a 3D representation. We should thus focus on learning robust and generic features, making the pose estimation scene-agnostic and tightly constrained by geometry. The challenge lies in how to define good features to localize. We solve this by making the geometric estimation differentiable and supervise only the final pose estimate. Differently from pose or coordinate regression, we assume that a 3D scene representation is available. This requirement is easily met in practice since the reference poses are usually obtained by sparse or dense 3D reconstruction.

Problem formulation: Our objective is to estimate the 6-DoF pose $(R, t) \in SE(3)$ of a query depth image D_q, where R is the rotation matrix and t is the translation vector in the camera coordinate system. Given a 3D representation of the environment in the form of 3D Gaussians, let $\mathcal{G} = \{G_i\}_{i=1}^N$ denote a set of N 3D Gaussians, and posed reference depth images $\{D_k\}$, which together constitute the reference data.

3.1 Gaussian Splatting

Each Gaussian G_i is characterized by its 3D mean $\boldsymbol{\mu}_i \in \mathbb{R}^3$, 3D covariance matrix $\boldsymbol{\Sigma}_i \in \mathbb{R}^{3\times3}$, opacity $o_i \in \mathbb{R}$, and scale $\mathbf{s}_i \in \mathbb{R}^3$. To represent the orientation of each Gaussian, we use a rotation quaternion $\mathbf{q}_i \in \mathbb{R}^4$.

The 3D covariance matrix $\boldsymbol{\Sigma}_i$ is then parameterized using $\mathbf{s}_i$ and $\mathbf{q}_i$:

$$\boldsymbol{\Sigma}_i = R(\mathbf{q}_i)S(\mathbf{s}_i)S(\mathbf{s}_i)^T R(\mathbf{q}_i)^T$$

where $R(\mathbf{q}_i)$ is the rotation matrix derived from $\mathbf{q}_i$, and $S(\mathbf{s}_i) = \text{diag}(\mathbf{s}_i)$ is a diagonal matrix of scales.

To project these 3D Gaussians onto a 2D image plane, we follow the approach described by [1]. The projection of the 3D mean $\boldsymbol{\mu}_i$ to the 2D image plane is given by:

$$\boldsymbol{\mu}_{I,i} = \pi(P(T_{wc}\boldsymbol{\mu}_{i,\text{homogeneous}}))$$

where $T_{wc} \in SE(3)$ is the world-to-camera transformation, $P \in \mathbb{R}^{4\times4}$ is the projection matrix [2], and $\pi : \mathbb{R}^4 \to \mathbb{R}^2$ maps to pixel coordinates.

The 2D covariance $\boldsymbol{\Sigma}_{I,i} \in \mathbb{R}^{2\times2}$ of the projected Gaussian is derived as:

$$\boldsymbol{\Sigma}_{I,i} = JR_{wc}\boldsymbol{\Sigma}_i R_{wc}^T J^T$$

where R_{wc} represents the rotation component of T_{wc}, and J is the affine transform as described by [3].

3.2 Depth Compositing

For depth map generation, we employ a front-to-back compositing scheme, which allows for accurate depth estimation and edge alignment. Let d_n represent the depth value associated with the n-th Gaussian, which is the z-coordinate of the Gaussian's mean in the camera coordinate system. The depth $D(p)$ at pixel p is computed as [1]:

$$D(p) = \sum_{n\le N} d_n \cdot \alpha_n \cdot T_n, \quad \text{where } T_n = \prod_{m<n}(1-\alpha_m)$$

Here, α_n represents the opacity of the n-th Gaussian at pixel p, computed as:

$$\alpha_n = o_n \cdot \exp(-\sigma_n), \quad \sigma_n = \frac{1}{2}\Delta_n^T \boldsymbol{\Sigma}_I^{-1}\Delta_n$$

where Δ_n is the offset between the pixel center and the 2D Gaussian center $\boldsymbol{\mu}_I$, and o_n is the opacity parameter of the Gaussian. T_n denotes the cumulative transparency product of

all Gaussians preceding n, accounting for the occlusion effects of previous Gaussians.

To ensure consistent representation across the image, we normalize the depth values. First, we calculate the total accumulated opacity $\alpha(p)$ for each pixel:

$$\alpha(p) = \sum_{n \leq N} \alpha_n \cdot T_n$$

The normalized depth $\text{Norm}_D(p)$ is then defined as:

$$\text{Norm}_D(p) = \frac{D(p)}{\alpha(p)}$$

This normalization process ensures that the depth values are properly scaled and comparable across different regions of the image, regardless of the varying densities of Gaussians in the scene. By projecting 3D Gaussians onto the 2D image plane and computing normalized depth values, we can effectively generate depth maps that accurately represent the 3D structure of the scene while maintaining consistency across different viewing conditions.

3.3 Camera Pose

We define the camera pose as

$$\mathbf{T}_{cw} = \begin{pmatrix} \mathbf{R}_{cw} & \mathbf{t}_{cw} \\ \mathbf{0} & 1 \end{pmatrix} \in SE(3)$$

where $\mathbf{T}_{cw}$ represents the camera-to-world transformation matrix. Notably, we parameterize the rotation $\mathbf{R}_{cw} \in SO(3)$ using a quaternion $\mathbf{q}_{cw}$. This choice of parameterization is motivated by several key advantages that quaternions offer in the context of camera pose estimation and optimization. Quaternions provide a compact and efficient representation, requiring only four parameters, while maintaining numerical stability and avoiding singularities such as gimbal lock. Their continuous and non-redundant nature is particularly advantageous for gradient-based optimization algorithms, allowing for unconstrained optimization and simplifying the optimization landscape.

3.4 Optimization

Based on these considerations, we design our optimization variables to separately optimize the normalized quaternion and the translation. The loss function is designed to ensure accurate depth estimations and edge alignment, incorporating both depth magnitude and contour accuracy. It can be defined as:

$$L = \lambda_1 \cdot L_{\text{depth}} + \lambda_2 \cdot L_{\text{contour}}$$

where L_{depth} represents the L1 loss for depth accuracy, and L_{contour} focuses on the alignment of depth contours or edges. Specifically:

$$L_{\text{depth}} = \sum_{i \in M} |D_i^{\text{predicted}} - D_i^{\text{observed}}|$$

$$L_{\text{contour}} = \sum_{j \in M} |\nabla D_j^{\text{predicted}} - \nabla D_j^{\text{observed}}|$$

Here, M denotes the reprojection mask, indicating which pixels are valid for reprojection. Both L_{depth} and L_{contour} are computed only over the masked regions. λ_1 and λ_2 are weights that balance the two parts of the loss function, tailored to the specific requirements of the application.

The optimization objective can be formulated as:

$$\min_{\mathbf{q}_{cw}, \mathbf{t}_{cw}} L + \lambda_q \|\mathbf{q}_{cw}\|_2^2 + \lambda_t \|\mathbf{t}_{cw}\|_2^2$$

where λ_q and λ_t are regularization terms for the quaternion and translation parameters, respectively.

We employ the Adam optimizer for both quaternion and translation optimization, with different learning rates and weight decay values for each. The learning rates are set to $5{\times}10^-4$ for quaternion optimization and 10^-3 for translation optimization, based on experimental results. The weight decay values are set to 10^-3 for both quaternion and translation parameters, serving as regularization to prevent overfitting.

3.5 Localization pipeline

The GSplatLoc method simplifies the localization process by requiring only posed reference depth images $\{D_k\}$ and a query depth image D_q. Its differentiability in projections of 3D Gaussian facilitates an efficient and smooth convergence during optimization.

Initialization: We initialize these Gaussians from a point cloud projected by $\{D_k\}$, where each point corresponds to a Gaussian's mean $\boldsymbol{\mu}_i$. For the initial parameterization, we set $o_i = 1$ for all Gaussians to ensure full opacity. The scale $\mathbf{s}_i \in \mathbb{R}^3$ of each Gaussian is initialized based on the local point density, allowing our model to adaptively adjust to varying point cloud densities:

$$\mathbf{s}_i = (\sigma_i, \sigma_i, \sigma_i), \text{ where } \sigma_i = \sqrt{\frac{1}{3} \sum_{j=1}^{3} d_{ij}^2}$$

Here, d_{ij} is the distance to the j-th nearest neighbour of point i. In practice, we calculate this using the k-nearest neighbours algorithm with $k = 4$, excluding the point itself. This isotropic initialization ensures a balanced initial representation of the

local geometry. Initially, we set $\mathbf{q}_i = (1, 0, 0, 0)$ for all Gaussians, corresponding to no rotation.

This initialization strategy provides a neutral starting point, allowing subsequent optimization processes to refine the orientations as needed. Unlike traditional 3D reconstruction methods [1] that often rely on structure-from-motion techniques [4], our approach is tailored for direct point cloud input, offering greater flexibility and efficiency in various 3D data scenarios.

Convergence: To determine the convergence of the optimization process, we employ an early stopping mechanism based on the stabilization of the total loss. Extensive experimental results indicate that the total loss stabilizes after approximately 100 iterations. We implement a patience mechanism, set to activate after 100 iterations. If the total loss does not decrease for a consecutive number of patience iterations, the optimization loop is terminated. The pose estimate corresponding to the minimum total loss is then selected as the optimal pose. This approach ensures the reliability and efficiency of the optimization process.

4 Experiments

We present GSplatLoc, an innovative pose estimation method for RGB-D cameras that employs a volumetric representation of 3D Gaussians. This approach facilitates precise pose estimation by minimizing the loss based on the reprojection of 3D Gaussians from real depth maps captured from the estimated pose. Our method attains rotational errors close to zero and translational errors within 0.01mm, representing a substantial advancement in pose accuracy over existing point cloud registration algorithms, as well as explicit volumetric and implicit neural representation-based SLAM methods. Comprehensive evaluations demonstrate that GSplatLoc significantly improves pose estimation accuracy, which contributes to increased robustness and fidelity in real-time 3D scene reconstruction, setting a new standard for localization techniques in dense mapping SLAM.

5 Conclusion

We present GSplatLoc, an innovative pose estimation method for RGB-D cameras that employs a volumetric representation of 3D Gaussians. This approach facilitates precise pose estimation by minimizing the loss based on the reprojection of 3D Gaussians from real depth maps captured from the estimated pose. Our method attains rotational errors close to zero and translational errors within 0.01mm, representing a substantial advancement in pose accuracy over existing point cloud registration algorithms, as well as explicit volumetric and implicit neural representation-based SLAM methods. Comprehensive evaluations demonstrate that GSplatLoc significantly improves pose estimation accuracy, which contributes to increased robustness and fidelity in real-time 3D scene reconstruction, setting a new standard for localization techniques in dense mapping SLAM.

[1] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, pp. 1–14, 2023, doi: 10.1145/3592433.

[2] V. Ye and A. Kanazawa, "Mathematical Supplement for the $\texttt{gsplat}$ Library." Accessed: Jun. 29, 2024. [Online]. Available: http://arxiv.org/abs/2312.02121

[3] M. Zwicker, H. Pfister, J. Van Baar, and M. Gross, "EWA splatting," *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 3, pp. 223–238, 2002, doi: 10.1109/TVCG.2002.1021576.

[4] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4104–4113. Accessed: Jun. 15, 2024. [Online]. Available: https://www.cv-foundation.org/openaccess/content_cvpr_2016/html/Schonberger_Structure-From-Motion_Revisited_CVPR_2016_paper.html