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

https://spla-tam.github.io

Atticus Zhou, Atticus Zhou, Atticus Zhou, Atticus Zhou

July 23, 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

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.

3 Method

Depth-only In our methodology, we initiate 3D Gaussians from a dense point cloud acquired via a RGB-D camera.

3.1 Pre-process

To enhance the optimization process and improve the accuracy of our 3D Gaussian representation, we implement a robust normalization procedure for the input data. This preprocessing step ensures that the initial Gaussian distributions are positioned and scaled optimally within a standardized coordinate system, facilitating more effective optimization and reconstruction.

Our normalization process consists of two primary stages: similarity transformation and principal axis alignment. First, we apply a similarity transformation $?\mathbf{T}_s \in SE(3)$ to align and scale the camera positions. This transformation is computed as:

$$\mathbf{T}_s = \begin{bmatrix} s\mathbf{R}_a & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}$$

where $s \in \mathbb{R}$ is a scaling factor, $\mathbf{R}_a \in SO(3)$ is a rotation matrix that aligns the world up vector with the camera space up vector, and $\mathbf{t} \in \mathbb{R}^3$ is a translation vector that centers the scene. The scaling factor s is calculated based on the distribution of camera positions, using either the maximum (for strict scaling) or median distance from the center.

Following the similarity transformation, we perform principal component analysis (PCA)[1] on the transformed point cloud to align its principal axes with the coordinate axes. This results in a second transformation matrix $\mathbf{T}_p \in SE(3)$:

$$\mathbf{T}_p = \begin{bmatrix} \mathbf{R}_p & -\mathbf{R}_p\mathbf{c} \\ \mathbf{0}^T & 1 \end{bmatrix}$$

where $\mathbf{R}_p \in SO(3)$ is the rotation matrix formed by the eigenvectors of the point cloud's covariance matrix, and $\mathbf{c} \in \mathbb{R}^3$ is the centroid of the point cloud.

The final normalization transformation $\mathbf{T} \in SE(3)$ is the composition of these two transformations:

$$\mathbf{T} = \mathbf{T}_p\mathbf{T}_s$$

This combined transformation is applied to both the point cloud and the camera poses. For a point $\mathbf{p} \in \mathbb{R}^3$, the normalized point $\mathbf{p}'$ is computed as:

$$\mathbf{p}' = \mathbf{T}_{3\times3}\mathbf{p} + \mathbf{T}_{3\times1}$$

where $\mathbf{T}_{3\times3}$ and $\mathbf{T}_{3\times1}$ are the rotation and translation components of $\mathbf{T}$, respectively.

For camera poses represented by transformation matrices $\mathbf{C} \in SE(3)$, we apply the normalization as:

$$\mathbf{C}' = \mathbf{T}\mathbf{C}$$

To maintain the scale information of the original scene, we compute a scale factor λ for each camera pose:

$$\lambda = \|\mathbf{C}'_{1:3,1}\|_2$$

where $\mathbf{C}'_{1:3,1}$ is the first column of the rotational component of the transformed camera pose. We then normalize the rotational component of $\mathbf{C}'$ by this scale factor to preserve the original scale information.

This comprehensive normalization procedure ensures that our initial set of 3D Gaussians is optimally positioned and scaled within a standardized coordinate system. By aligning the principal axes of the scene with the coordinate axes and centering the data, we create a more favorable starting point for subsequent optimization steps. This approach not only improves the convergence of our optimization algorithms but also enhances the overall quality and consistency of the 3D reconstruction across various input datasets.

3.2 Gaussian Splatting

Let $\mathcal{G} = \{G_i\}_{i=1}^N$ denote a set of N 3D Gaussians. Each Gaussian G_i is characterized by its 3D mean $\boldsymbol{\mu}_i \in \mathbb{R}^3$, 3D covariance matrix $\boldsymbol{\Sigma}_i \in \mathbb{R}^{3\times3}$, and opacity $o_i \in \mathbb{R}$. We initialize these Gaussians from a point cloud, where each point corresponds to a Gaussian's mean $\boldsymbol{\mu}_i$. Unlike traditional 3D reconstruction methods[2] that often rely on structure-from-motion techniques[3], our approach is tailored for direct point cloud input, offering greater flexibility and efficiency in various 3D data scenarios.

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.

To represent the orientation of each Gaussian, we use a rotation quaternion $\mathbf{q}_i \in \mathbb{R}^4$. 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.

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 [2]. 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 [4], 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:

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

where R_{wc} represents the rotation component of T_{wc}, and J is the affine transform as described by [5].

3.3 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 [2]:

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

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

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

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.

3.4 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.

3.5 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 optimiza-

tion, 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.

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] A. Maćkiewicz and W. Ratajczak, "Principal components analysis (PCA)," *Computers & Geosciences*, vol. 19, no. 3, pp. 303–342, 1993, doi: 10.1016/0098-3004(93)90090-R.

[2] 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.

[3] 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

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

[5] 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.