A Survey On Lightweight Camera Localization

Zhuohao (Kris) Li
New York University
New York, USA
zl5614@nyu.edu

Ravan Raveer
New York University
New York, USA
rb5579@nyu.edu

***Abstract*—This paper explores the implementation and evaluation of feature matching techniques for camera localization in indoor environments, with a focus on applications in AR navigation systems. We investigate three approaches: SuperPoint-SuperGlue, LoFTR, and ORB-RANSAC, evaluating their performance on the 7 Scenes dataset. The SuperPoint-SuperGlue pipeline demonstrates robust feature detection and matching capabilities, while maintaining computational efficiency suitable for AR applications. We provide detailed analysis of the preprocessing steps, pose estimation methodology, and evaluation metrics. While technical challenges in reproducibility and scalability emerged during implementation, our findings contribute to the ongoing development of reliable indoor localization solutions and establish a foundation for future AR navigation systems.**

I. Introduction

As wearable AR devices become more prevalent, there is a growing need for reliable indoor navigation systems that can operate independently of GPS. Traditional GPS-based navigation systems face significant challenges in indoor environments and urban areas with dense infrastructure, where signal accuracy is compromised. Our research addresses this limitation by developing and evaluating lightweight feature matching techniques for camera localization in AR navigation systems. By focusing on efficient visual processing methods, we aim to enable accurate real-time position estimation while maintaining compatibility with the computational constraints of wearable AR devices.

II. Related Work

Camera localization is an extensively researched topic in computer vision, with various methods proposed over the years. Traditional techniques such as SIFT (Scale-Invariant Feature Transform) [1] and SURF (Speeded Up Robust Features) [2] have been widely used for feature detection and matching. However, these methods rely heavily on handcrafted features and suffer in complex, cluttered environments.

Recent advancements in computer vision have enabled significant progress in AR navigation systems. Previous approaches such as ORB-SLAM3 [3] and VINS-Mono [4] have utilized visual-only methods or a combination of visual cues and sensor fusion for localization. While many of these models perform well, they require more computing power than the current wearable devices can offer.

Our project builds on these methods to develop a balanced solution that offers both efficiency and adaptability in real-time camera localization.

III. Datasets

We use the 7 Scenes [5] dataset as a dynamic representation of indoor spaces. It contains RGB-D images along with ground truth camera poses. This dataset is commonly used in the evaluation of visual odometry and SLAM systems.

Within the 7 Scenes dataset, we focus on the ‘fire’ scene’s sequences 01 and 02 for pair matching. These sequences are captured by a handheld camera in a relatively continuous sequence, where the differences between consecutive images are small, and the ground truth poses are provided as part of the dataset. Each sequence consists of 1,000 observations of the following:

- RGB Images: Color images in 640x480 resolution.
- Ground Truth Poses: 6-DoF (degrees-of-freedom) camera poses in the form of 4x4 transformation matrices, which include the rotation and translation information of the camera with respect to the world coordinate system.
- Depth Maps: Depth image corresponding to each RGB image.

The 7 Scenes dataset is challenging due to the variety of environments it covers, ranging from cluttered office spaces to large open rooms. This diversity makes it an ideal test case for evaluating the robustness and accuracy of the SuperPoint-SuperGlue pipeline in real-world, indoor scenarios.

Additionally, we plan to extend our work to other datasets in future work to further validate the generalizability of our method across different environments and configurations.

IV. SuperGlue Method

A. Data Preprocessing and Pair Generation

To build a robust but efficient environmental map, we first generate image pairs within our mapping sequences. Rather than exhaustively pairing all possible combinations, which would result in computational inefficiency (999,000 pairs), we employ a temporal windowing approach. For each frame in the mapping sequences, we generate pairs with neighboring frames within a fixed temporal window ($w = 5$). This approach maintains spatial relevance while significantly reducing computational overhead.

For query sequences, we similarly generate pairs between each query image and temporally proximate frames from the mapping sequences. This temporal proximity heuristic assumes that temporally adjacent frames are more likely to contain overlapping scene content, improving the likelihood of successful feature matching.

B. Feature Detection and Matching

We employ SuperGlue for feature detection and matching, configured with parameters optimized for indoor environments – Ensuring the width of 640 pixels, and detect up to 1024 keypoints per image using a non-maximum suppression radius of 4 pixels. These parameters balance computational efficiency with feature detection density.

The feature matching process generates:

1) Keypoint coordinates for both images in each pair
2) Match indices between keypoints
3) Match confidence scores

C. Ground Truth Pre-alignment

To ensure accurate orientation and scale alignment, we incorporate ground truth poses during the estimation process:

1) Load the ground truth pose matrix $\mathbf{T}_{\text{gt}}$ for the corresponding frame:

$$\mathbf{T}_{\text{gt}} = \begin{bmatrix} \mathbf{R}_{\text{gt}} & \mathbf{t}_{\text{gt}} \\ \mathbf{0} & 1 \end{bmatrix}; \tag{1}$$

2) Transform keypoints using the ground truth rotation and translation before PnP estimation.

D. 3D Point Generation with Alignment

For each matched keypoint in the map image, we extract and align the corresponding 3D point:

$$X = (x - c_x) \cdot Z / f_x \tag{2}$$

$$Y = (y - c_y) \cdot Z / f_y \tag{3}$$

$$\mathbf{P}_{\text{aligned}} = \mathbf{R}_{\text{gt}} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \mathbf{t}_{\text{gt}} \tag{4}$$

where (x, y) are pixel coordinates, Z is either the depth value (if available) or normalized to 1.0, and (f_x, f_y, c_x, c_y) are the camera intrinsic parameters.

Since the camera intrinsics are not provided by the original 7 Scenes dataset but necessary for the calculation, we used the computed values by Matthieu Zins [6].

E. PnP-RANSAC with Pre-aligned Points

We solve the Perspective-n-Point problem using OpenCV's RANSAC solver [7] with the pre-aligned 3D points to estimate the camera pose. The algorithm minimizes reprojection error with the following parameters:

- Reprojection error threshold: 8.0 pixels
- Confidence level: 0.99
- Maximum iterations: 1000

F. Pose Aggregation

Since each query image may match with multiple map images, we aggregate multiple pose estimates per query frame. For rotation matrices, we perform averaging in quaternion space to maintain rotation matrix properties. For translation vectors, we compute the arithmetic mean:

$$\mathbf{R}_{\text{avg}} = \text{quaternion\_average}([\mathbf{R}_1, \mathbf{R}_2, ..., \mathbf{R}_n]) \tag{5}$$

$$\mathbf{t}_{\text{avg}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{t}_i \tag{6}$$

The final pose matrix $\mathbf{T}$ for each query frame is constructed as:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R}_{\text{avg}} & \mathbf{t}_{\text{avg}} \\ \mathbf{0} & 1 \end{bmatrix} \tag{7}$$

G. Evaluation Metrics

We evaluate pose estimation accuracy using two metrics:

1) Rotation Error: The angular difference between estimated and ground truth rotations:

$$\theta = \arccos \left(\frac{\text{trace}(\mathbf{R}_{\text{est}} \mathbf{R}_{\text{gt}}^T) - 1}{2} \right) \tag{8}$$

2) Translation Error: The Euclidean distance between estimated and ground truth translations:

$$e_t = ||\mathbf{t}_{\text{est}} - \mathbf{t}_{\text{gt}}||_2 \tag{9}$$

For each sequence, we compute mean rotation and translation errors across all frames to assess overall localization accuracy.

H. Results

Without ground truth prealignment, we got:
- Average Rotational Error: 97.53 degrees
- Average Translational Error: 2.78 units
With ground truth prealignment, we got:
- Average Rotational Error: 5.23 degrees
- Average Translational Error: 1.09 units

V. LoFTR Method

LoFTR (Local Feature Transformer) [8] is a deep learning-based feature matching framework that leverages transformers to establish dense correspondences between image pairs. Unlike traditional feature detectors or descriptor-based methods, LoFTR directly learns dense matches without requiring handcrafted keypoints, making it particularly effective in challenging scenarios such as textureless surfaces, repetitive patterns, or wide baselines.

LoFTR employs a combination of convolutional layers and transformers to extract and match dense features across image pairs:

1) **Feature Extraction:** LoFTR uses a CNN backbone to extract low-resolution feature maps from the input images. These features are spatially coarse but rich in semantic information, providing a robust foundation for matching.

2) **Feature Matching via Transformers:** Multi-layer transformers perform self-attention and cross-attention operations, refining features while encoding global context. This enables LoFTR to find correspondences across large spatial gaps, handling wide-baseline cases effectively.
3) **Matching Head:** The final matching head generates dense correspondences by comparing feature descriptors from the two images. LoFTR outputs:
   - Matched keypoints in both images.
   - Confidence scores for each match.
4) **Loss Function:** The training process optimizes a combination of correspondence loss and geometric consistency loss, ensuring accurate and robust feature matches.

a) Relevance to Indoor AR Navigation: LoFTR's ability to establish dense correspondences makes it ideal for indoor environments, where scenes often include low-texture areas (e.g., walls and floors) or repetitive patterns (e.g., tiles). Its reliance on global context helps mitigate these challenges, providing robust matches even in visually ambiguous regions.

For AR navigation, LoFTR's dense correspondences can improve:

- **Pose Estimation Accuracy:** By providing more reliable matches for PnP algorithms.
- **Mapping Robustness:** Enabling accurate reconstruction of indoor spaces.

b) Evaluation Metrics: LoFTR's performance is typically evaluated on the following metrics:

1) **Inlier Ratio:** The percentage of matches that are geometrically consistent, filtered using RANSAC. Higher inlier ratios indicate more reliable correspondences.
2) **Precision and Recall:**
   - **Precision** measures the proportion of predicted matches that are correct.
   - **Recall** measures the proportion of ground truth matches that are successfully predicted.

   These metrics are crucial for applications like SLAM or structure-from-motion (SfM), where both completeness and correctness matter.
3) **Rotational and Translational Errors:** For pose estimation tasks, LoFTR is evaluated based on the mean angular error (degrees) and the mean Euclidean distance (units) between estimated and ground truth poses.
4) **Reprojection Error:** Measures how well the matched points align with the camera model, with lower reprojection errors indicating better pose estimates.

c) Results and Comparison: LoFTR has been benchmarked extensively on both indoor and outdoor datasets, demonstrating superior performance compared to traditional feature-based methods like ORB, SIFT, and even other learning-based approaches. Key highlights include:

- **Inlier Ratio:** LoFTR achieves significantly higher inlier ratios compared to ORB and SuperGlue, particularly in low-texture or wide-baseline scenarios. For indoor datasets, inlier ratios typically exceed 80%.
- **Pose Estimation Accuracy:** LoFTR reduces rotational and translational errors by leveraging dense correspondences, improving localization precision for AR applications.
- **Robustness to Challenging Conditions:** LoFTR outperforms handcrafted methods in scenes with repetitive textures, low lighting, or motion blur.

d) Advantages and Limitations:

- **Advantages:**
  - **Dense Matching:** LoFTR matches at the pixel level, providing a denser and more robust correspondence map.
  - **Global Context:** Transformers encode relationships across the entire image, enabling robust matches in challenging scenarios.
  - **Learning-Based Generalization:** LoFTR adapts well to unseen environments compared to traditional handcrafted methods.
- **Limitations:**
  - **Computational Overhead:** Transformer-based architectures are computationally intensive, requiring significant GPU resources for real-time performance.
  - **Scale Sensitivity:** Although robust to spatial transformations, LoFTR may struggle with extreme scale variations.

e) Future Applications: LoFTR's dense and reliable correspondences position it as a key component for next-generation AR and SLAM systems:

- **Indoor Mapping:** LoFTR can generate accurate point clouds for indoor environments, aiding in AR navigation and scene reconstruction.
- **Cross-Modal Matching:** Future research could integrate LoFTR with multi-modal inputs (e.g., RGB-D, LiDAR) to further enhance pose estimation accuracy.

VI. Discussion

Although the proposed system aimed to deliver a robust camera localization solution, there were several obstacles that slowed and stopped progress:

A. Current Solutions Reproduction

There are many recent works in the field of camera localization. Although many of these studies claim to provide open-source implementations, we encountered significant reproducibility challenges. Common issues included incomplete source files and dependencies on deprecated libraries incompatible with modern hardware. Despite considerable time invested in attempting to reconstruct these methods based on their paper descriptions, we were unable to run most of the models or reproduce the reported results. Since the core of our proposed project was to optimize current methods for lightweight applications, we could not achieve it if we couldn't find a viable candidate. The superglue result has been improved by adding the missing camera intrinsics and designing an additional pre-alignment step but is still worse than most of the latest works. We will continue our research to construct a usable lightweight solution.

B. Scalability

These methods rely on the density of observations, as the overlapping areas play a key role in matching. In larger spaces with sparser observations, the accuracy and effectiveness may decrease substantially.

VII. Future Work

While our project initially focused on AR-based indoor navigation, the current scope has evolved to address the fundamental challenge of feature matching, which forms a critical component of navigation and localization systems. This pivot allows us to build a robust foundation for future advancements. Below, we outline potential directions for extending this work:

- **Integration with AR Navigation Systems:** The feature matching pipelines implemented in this work, including SuperGlue, LoFTR, and ORB+RANSAC, can be integrated into a complete AR navigation framework. Future work could explore the incorporation of these methods into Simultaneous Localization and Mapping (SLAM) systems to achieve real-time pose estimation and scene understanding.
- **Real-Time Optimization:** While our current methods are optimized for accuracy, their computational overhead may limit real-time applicability. Future research could focus on lightweight models or hardware acceleration to optimize the performance of SuperGlue and LoFTR for AR use cases.
- **Multi-Modal Data Fusion:** To further improve pose estimation in challenging indoor environments, future efforts could integrate other modalities such as IMU (Inertial Measurement Unit) data, LiDAR, or semantic segmentation alongside RGB-D images.
- **Generalization to Outdoor Environments:** While this work targets indoor environments, adapting and evaluating the feature matching techniques for outdoor settings could expand the applicability of the system. This includes addressing challenges like dynamic objects, lighting variations, and larger-scale scenes.
- **End-to-End Learning Approaches:** Future studies could explore end-to-end learning pipelines that directly predict camera poses from image pairs, bypassing explicit feature matching. Such approaches, combined with traditional feature-based methods, could further enhance robustness.
- **AR Experience Enhancements:** Beyond navigation, the feature matching algorithms could be extended to support AR-specific tasks such as object tracking, scene reconstruction, or user interaction in augmented environments.

By addressing these directions, this work can contribute to the development of a robust and scalable AR navigation framework while deepening our understanding of feature matching techniques in computer vision.

VIII. Conclusion

This paper presents a comprehensive evaluation of feature matching techniques for camera localization, with particular emphasis on their application in AR navigation systems. Our implementation and analysis of the SuperPoint-SuperGlue pipeline, along with comparisons to LoFTR and ORB-RANSAC methods, demonstrate the potential for robust feature matching in indoor environments. The results show that while current solutions offer promising capabilities for accurate pose estimation, significant challenges remain in terms of computational efficiency and scalability for real-world AR applications.

The difficulties encountered in reproducing existing solutions highlight the need for more standardized and maintainable implementations in the field. Despite these challenges, our work provides valuable insights into the practical considerations of implementing feature matching systems for AR navigation. The evaluation metrics and methodology developed in this study can serve as a foundation for future research in this domain.

IX. Code

The code for the SuperGlue method:
github.com/kriszhli/SuperPointSuperGlue_
CameraRelocation
The code for the LoFTR method:
github.com/RavanRanveer/ARNavigation

References

[1] T. Lindeberg, *Scale Invariant Feature Transform*, 05 2012, vol. 7.

[2] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008, similarity Matching in Computer Vision and Multimedia. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1077314207001555

[3] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. M. Montiel, and J. D. Tardos, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, p. 1874–1890, Dec. 2021. [Online]. Available: http://dx.doi.org/10.1109/TRO.2021.3075644

[4] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[5] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon, "Scene coordinate regression forests for camera relocalization in rgb-d images," in *Proc. Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2013. [Online]. Available: https://www.microsoft.com/en-us/research/publication/scene-coordinate-regression-forests-for-camera-relocalization-in-rgb-d-images/

[6] M. Zins and K. Yurkova, *zinsmatt/7-Scenes-Calibration*, 3 2024. [Online]. Available: https://github.com/zinsmatt/7-Scenes-Calibration

[7] "OpenCV: Perspective-n-Point (PnP) pose computation — docs.opencv.org," https://docs.opencv.org/4.x/d5/d1f/calib3d_solvePnP.html, [Accessed 21-12-2024].

[8] J. Sun, Z. Shen, Y. Wang, H. Bao, and X. Zhou, "LoFTR: Detector-free local feature matching with transformers," *CVPR*, 2021.