F²SLAM: TURNING FEED-FORWARD GEOMETRY INTO PERSISTENT FACTORS FOR SLAM

Anonymous authors
Paper under double-blind review

ABSTRACT

Feed-forward 3D models provide strong multi-view geometric priors, while online simultaneous localization and mapping (SLAM) relies mainly on local measurements and can accumulate drift over long sequences. Existing attempts to combine the two typically treat feed-forward predictions as an external geometric state that is aligned or fused with the online estimate after the fact, which keeps broader multi-view evidence outside the optimizer that refines the SLAM state. We present **F²SLAM**, which instead converts feed-forward geometry directly into optimization-native target-weight measurements attached to a persistent dense factor graph. A high-frequency stream maintains local tracking constraints and graph connectivity, while a low-frequency stream uses wider multi-view context to selectively refresh existing measurements after a state-consistency check. Both streams constrain the same poses, inverse depths, and optional camera intrinsics through a single dense bundle adjustment. Experiments on multiple benchmarks demonstrate consistently strong trajectory estimation and improved dense reconstruction in both calibrated and uncalibrated settings. Notably, the uncalibrated configuration reduces the average ATE RMSE from 0.030 m for the strongest feed-forward baseline to 0.002 m on the Replica dataset.

1 INTRODUCTION

Online visual simultaneous localization and mapping (SLAM) estimates camera motion and scene geometry by maintaining an optimizable geometric state that dense bundle adjustment (BA) continuously revisits, re-weights, and jointly refines as new observations arrive (Mur-Artal & Tardós, 2017; Teed & Deng, 2021). Because these measurements are drawn mainly from short temporal windows, online SLAM remains sensitive to weak overlap, occlusion, viewpoint changes, and accumulated tracking error over long sequences.

Recent feed-forward 3D foundation models (FMs) (Wang et al., 2025a) offer a complementary form of geometric reasoning. By jointly processing multiple images, they directly infer depth, point maps, camera poses, and multi-view structure over much broader context than a local BA window. This has motivated foundation-model-based SLAM (FM-SLAM) systems such as MASt3R-SLAM (Murai et al., 2025), VGGT-SLAM (Maggio et al., 2025), and VGGT-SLAM++ (Mandal et al., 2026), which exploit learned geometry for tracking, mapping, relocalization, loop closure, or submap alignment.

These systems, however, typically compute feed-forward predictions over local windows or submaps and only afterwards reconcile them with the SLAM state through alignment, fusion, or state-level correction, as shown in Fig. 1. Feed-forward geometry therefore remains an external state rather than evidence that the online optimizer can act on directly. This separation is costly in two ways. Broader multi-view evidence never enters the measurement set that dense BA actually optimizes. Once a submap has been aligned, it also cannot be jointly re-weighted against subsequent local evidence, since the correction is applied only once. The missing link is a way to make feed-forward geometric evidence directly and repeatedly participate in persistent SLAM optimization, rather than serve as a one-time external correction.

We address this with **F²SLAM**, which follows one guiding principle: *feed-forward geometry should enter SLAM as optimization-native measurements, not as a separate geometric state.* F²SLAM converts multi-view feed-forward evidence into dense target-weight measurements attached directly

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

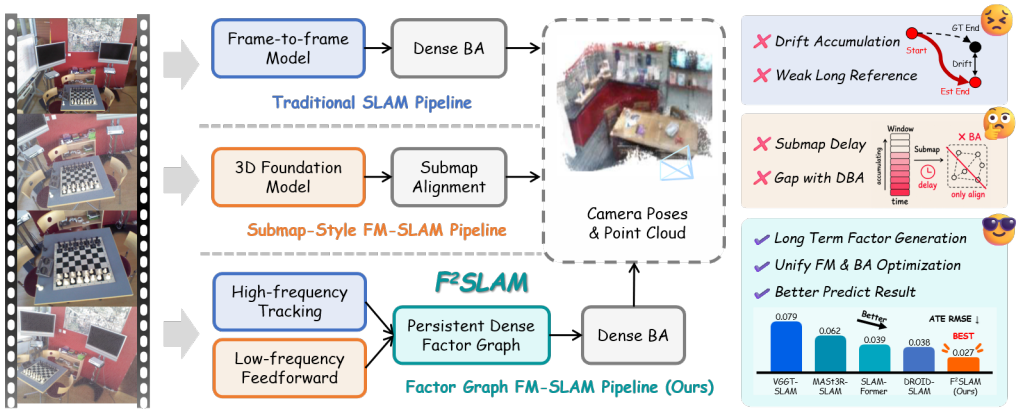


Figure 1: Motivation and overview of F^{2}SLAM. Conventional dense SLAM (top) relies on local frame-to-frame measurements, while foundation-model-based approaches (middle) reconstruct and align local submaps. In contrast, F^{2}SLAM (bottom) integrates high-frequency tracking and low-frequency feed-forward geometry in a persistent dense factor graph, where low-frequency predictions selectively refresh existing measurements for joint refinement through a single dense BA. The inset reports ATE RMSE on the TUM RGB-D dataset Sturm et al. (2012).

to the existing factor graph, so that feed-forward reasoning and recurrent tracking constrain the same keyframe poses, dense inverse depths, and optional camera intrinsics through one dense BA backend. Doing so online requires updating these measurements at two different timescales. A high-frequency stream maintains local constraints and graph connectivity, while a low-frequency stream uses wider multi-view context to selectively refresh existing measurements. Because feed-forward predictions can be unreliable under occlusion or weak overlap, the central difficulty is deciding when a refresh should be trusted. F^{2}SLAM addresses this with a state-consistency check that admits a candidate measurement only when it agrees with the current geometric state, and otherwise leaves the existing measurement unchanged. Feed-forward pose and depth predictions may also provide gauge-aligned initialization, but subsequent geometric refinement remains entirely within the persistent BA state.

In summary, our **contributions** are as follows:

- We formulate feed-forward multi-view geometry as optimization-native target-weight measurements that directly participate in persistent dense BA.
- We develop a dual-timescale measurement-refresh mechanism with state-consistent admission, enabling local tracking and broader multi-view reasoning to update a common factor representation within a single optimized geometric state.
- We instantiate this formulation in a complete online system, F^{2}SLAM, and demonstrate strong trajectory estimation and dense reconstruction across calibrated and uncalibrated indoor and outdoor benchmarks.

2 Related Work

Dense RGB SLAM. Dense RGB SLAM estimates camera motion and scene geometry from video with photometric, geometric, or learned constraints. Classical methods such as ORB-SLAM (Mur-Artal & Tardós, 2017; Campos et al., 2021) and LSD-SLAM (Engel et al., 2014) rely on hand-crafted features, direct alignment, and bundle adjustment, while learning-based methods (Teed & Deng, 2021; Lipson et al., 2024) integrate learned depth, matching, confidence, and motion components for robust estimation. Recent NeRF- and 3DGS-based methods (Zhu et al., 2024; Zhang et al., 2023) improve scene representation but still struggle with long-range consistency.

Feed-forward 3D reconstruction priors. Feed-forward reconstruction models have become strong priors for multi-view geometry. Pairwise methods such as DUSt3R (Wang et al., 2024) and MASt3R (Leroy et al., 2024) predict dense structure and correspondences for wide-baseline

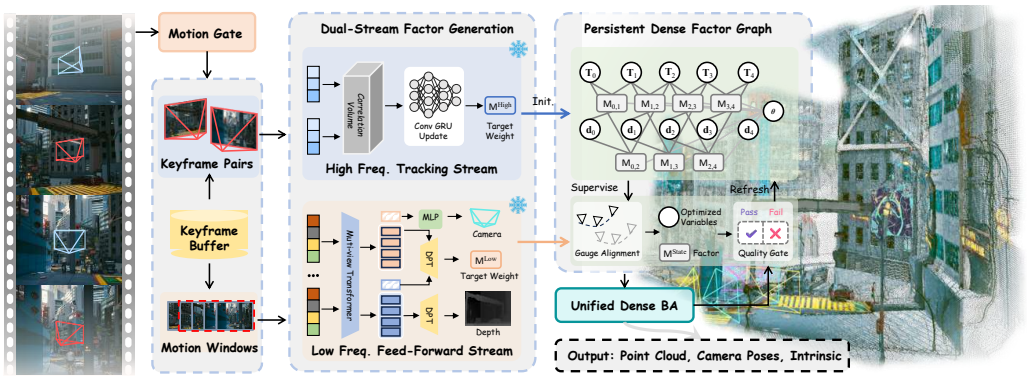


Figure 2: Pipeline of F^{2}SLAM. The high-frequency stream selects keyframes, maintains the graph topology, and frequently updates local target-weight measurements. The low-frequency stream refreshes measurements on selected existing edges using wider multi-view context, without adding new edges. Both streams constrain the same persistent geometric state through a single dense BA.

reconstruction, while multi-view models such as VGGT (Wang et al., 2025a) extend this to joint camera, depth, point-map, and track prediction. Streaming variants (Zhuo et al., 2025; Chen et al., 2026) support online or long-sequence reconstruction with memory or causal processing, but remain window-dependent and lack persistent sequence-level optimization.

SLAM with feed-forward reconstruction priors. Recent foundation-model-based SLAM systems (Liu et al., 2025; Murai et al., 2025; Maggio et al., 2025) have explored using feed-forward reconstruction priors for online tracking and mapping, often by applying them to local windows or submaps and then aligning, fusing, or correcting the resulting poses, depths, and point maps. However, the correction they provide is applied only once, since feed-forward predictions either sit outside the main dense BA or are optimized through a separate alignment stage that is not revisited as new local evidence arrives. F^{2}SLAM instead converts feed-forward outputs into dense target-weight factors and jointly optimizes them with local DROID-style factors in one dense BA backend, directly coupling online tracking with feed-forward measurements predicted from broader multi-view context.

3 Feed-forward Factor SLAM

3.1 Persistent Factor-Graph Formulation

Existing feed-forward SLAM systems typically treat feed-forward geometry as an external state that must be aligned or fused with the online estimate. In contrast, F^{2}SLAM converts feed-forward predictions into dense measurements that directly constrain the same persistent geometric state as online tracking. Fig. 2 gives the pipeline of F^{2}SLAM.

Given an RGB stream $\{I_t\}_{t=1}^{T}$, we maintain an ordered set of keyframe indices:

$$\mathcal{K}_t = \{k_1, \ldots, k_{N_t}\}, \tag{1}$$

where k_{N_t} is the latest keyframe. The persistent geometric state is:

$$\mathcal{X}_t = \{\mathbf{T}_i, \mathbf{d}_i\}_{i \in \mathcal{K}_t} \cup \{\boldsymbol{\theta}\}, \tag{2}$$

where $\mathbf{T}_i \in SE(3)$ is the world-to-camera transformation, $\mathbf{d}_i \in \mathbb{R}_+^{h \times w}$ is the dense inverse-depth map, and $\boldsymbol{\theta}$ denotes the camera intrinsics. When calibration is known, $\boldsymbol{\theta}$ is fixed and omitted from the optimized variables. We denote the current estimate by $\bar{\mathcal{X}}_t$ and the result of a BA solver by $\mathcal{X}_t^*$.

The active portion of the persistent factor graph is:

$$\mathcal{G}_t = \left(\mathcal{K}_t, \mathcal{E}_{\text{act}}^t\right), \tag{3}$$

where $\mathcal{E}_{\text{act}}^t$ contains the active directed edges. Each edge (i, j) stores a dense target-weight measurement:

$$\mathcal{M}_{ij} = \left(\mathbf{p}_{ij}^*, \mathbf{w}_{ij}\right), \tag{4}$$

where $\mathbf{p}_{ij}^{*}(\mathbf{p}) \in \mathbb{R}^2$ is the predicted target coordinate of source pixel $\mathbf{p} \in \Omega_i$, and $\mathbf{w}_{ij}(\mathbf{p}) \in \mathbb{R}_+^2$ contains the confidence weights for the two reprojection-residual components. We define:

$$\mathbf{\Lambda}_{ij}(\mathbf{p}) = \mathrm{diag}(\mathbf{w}_{ij}(\mathbf{p}))\,, \tag{5}$$

which weights the dense reprojection residual in the BA objective.

All measurements in $\mathcal{G}_t$ use this target-weight representation. Feed-forward predictions may initialize the shared geometric state or generate factor measurements, but do not introduce a separate optimized state. Only $\mathcal{X}_t$ is optimized by the common dense BA.

3.2 Dual-stream Pair Construction and Factor Generation

We update the shared factor representation through two streams operating at different timescales. Following standard visual SLAM practice (Mur-Artal & Tardós, 2017), incoming frames are selected as keyframes according to their image-space motion relative to the latest keyframe. Detailed insertion and low-frequency scheduling rules are provided in Appendix A.1.

After inserting a new keyframe k_{N_t}, its temporal and geometrically proximal neighbors define:

$$\mathcal{N}_{\mathrm{high}}(k_{N_t}) = \mathcal{N}_{\mathrm{temp}}(k_{N_t}) \cup \mathcal{N}_{\mathrm{prox}}(k_{N_t}), \tag{6}$$

from which the high-frequency stream creates directed pairs:

$$\mathcal{P}_{\mathrm{high}}^{t} = \bigcup_{j \in \mathcal{N}_{\mathrm{high}}(k_{N_t})} \{(k_{N_t}, j), (j, k_{N_t})\}\,. \tag{7}$$

These pairs are inserted into $\mathcal{E}_{\mathrm{act}}^{t}$, while factors exceeding the update-age or capacity limits are retired with their measurements retained for possible reuse in local BA. Thus, the high-frequency stream constructs and maintains the persistent graph topology.

Each active edge stores:

$$\mathcal{F}_{ij} = (\mathcal{M}_{ij}, \ell_{ij}, \gamma_{ij})\,, \tag{8}$$

where $\ell_{ij} \in \{\mathrm{high}, \mathrm{low}\}$ records the measurement source and $\gamma_{ij} \in \{0, 1\}$ controls whether high-frequency updates may overwrite it. Note that these metadata are not optimizable geometric variables.

For each unlocked edge $(i, j) \in \mathcal{E}_{\mathrm{act}}^{t}$, the high-frequency recurrent updater predicts:

$$\big(\mathbf{h}_{ij}^{+}, \Delta\mathbf{p}_{ij}^{\mathrm{high}}, \mathbf{w}_{ij}^{\mathrm{high}}\big) = \Psi_{\mathrm{high}}\big(\mathbf{h}_{ij}, \mathbf{x}_i, \mathbf{C}_{ij}, \mathbf{m}_{ij}\big). \tag{9}$$

Here, $\mathbf{h}_{ij}$ and $\mathbf{h}_{ij}^{+}$ are the recurrent hidden states, $\mathbf{x}_i$ is the source context feature, $\mathbf{C}_{ij}$ contains correlation features sampled at the current reprojection, and $\mathbf{m}_{ij}$ encodes geometry-induced flow and the discrepancy from the stored target. The resulting measurement is:

$$\begin{aligned} \mathbf{p}_{ij}^{*,\mathrm{high}}(\mathbf{p}) &= \hat{\mathbf{p}}_{ij}\left(\mathbf{p}; \bar{\mathcal{X}}_t\right) + \Delta\mathbf{p}_{ij}^{\mathrm{high}}(\mathbf{p}), \\ \mathcal{F}_{ij} &\leftarrow \left(\left(\mathbf{p}_{ij}^{*,\mathrm{high}}, \mathbf{w}_{ij}^{\mathrm{high}}\right), \mathrm{high}, 0\right). \end{aligned} \tag{10}$$

The constructed target remains fixed during the ensuing BA solve.

Meanwhile, newly inserted keyframes are accumulated into a scheduled low-frequency window:

$$\mathcal{W}_t = \{k_{a_t}, \ldots, k_{b_t}\}, \tag{11}$$

which provides wider multi-view context rather than defining a separate submap or factor graph. The low-frequency stream operates only on selected existing active edges:

$$\mathcal{P}_{\mathrm{low}}^{t} \subseteq \left\{(i, j) \in \mathcal{E}_{\mathrm{act}}^{t} \;\middle|\; i, j \in \mathcal{W}_t,\ i \neq j\right\}. \tag{12}$$

Multi-view features are extracted as:

$$\{\mathbf{H}_i\}_{i \in \mathcal{W}_t} = \mathrm{MVTrans}\left(\{E_{\mathrm{feat}}(I_i)\}_{i \in \mathcal{W}_t}\right), \tag{13}$$

where $\mathrm{MVTrans}$ aggregates per-image backbone features across the window. When enabled, gauge-aligned pose and depth predictions initialize the shared geometric state before pair selection and validation, but do not directly modify the predicted flow targets.

For each $(i,j) \in \mathcal{P}^t_{\text{low}}$, the multi-view factor decoder H_{factor} converts the multi-view features into a dense displacement-confidence pair:

$$\left(\mathbf{f}^{\text{low}}_{ij}, \mathbf{c}^{\text{low}}_{ij}\right) = H_{\text{factor}}\left(\mathbf{H}_i, \mathbf{H}_j\right), \tag{14}$$

where H_{factor} includes decoding and conversion to the common BA grid. Importantly, $\mathbf{f}^{\text{low}}_{ij}$ is the full source-to-target displacement in BA coordinates, rather than a correction to the current geometric reprojection. The confidence vector $\mathbf{c}^{\text{low}}_{ij}$ uses identical horizontal and vertical components derived from the scalar confidence gate.

The candidate measurement is constructed for each source pixel $\mathbf{p} \in \Omega_i$ as:

$$\begin{aligned} \mathbf{p}^{*,\text{low}}_{ij}(\mathbf{p}) &= \mathbf{p} + \mathbf{f}^{\text{low}}_{ij}(\mathbf{p}), \\ \mathbf{w}^{\text{low}}_{ij} &= \eta_{\text{low}} \mathbf{c}^{\text{low}}_{ij}, \end{aligned} \tag{15}$$

where $\eta_{\text{low}} > 0$ is a fixed scalar controlling the relative weight of low-frequency measurements. The resulting candidate measurement is denoted by:

$$\mathcal{M}^{\text{low}}_{ij} = \left(\mathbf{p}^{*,\text{low}}_{ij}, \mathbf{w}^{\text{low}}_{ij}\right). \tag{16}$$

These candidates are validated against the current SLAM state before being allowed to refresh the corresponding existing factor measurements.

3.3 State-Consistent Low-Frequency Factor Refresh

Low-frequency predictions may be unreliable under occlusion or limited overlap. We therefore validate each candidate against the current SLAM state before refreshing its existing factor measurement. For $(i,j) \in \mathcal{P}^t_{\text{low}}$, the reprojection residual is:

$$\mathbf{e}^{\text{low}}_{ij}(\mathbf{p}) = \hat{\mathbf{p}}_{ij}\left(\mathbf{p}; \bar{\mathcal{X}}_t\right) - \mathbf{p}^{*,\text{low}}_{ij}(\mathbf{p}), \tag{17}$$

where $\hat{\mathbf{p}}_{ij}(\mathbf{p}; \bar{\mathcal{X}}_t)$ denotes the reprojection of source pixel $\mathbf{p}$ into keyframe j under the current pose and inverse-depth estimates $\bar{\mathcal{X}}_t$.

Validating a candidate requires aggregating this residual, together with its predicted confidence, into per-pair statistics. A source pixel belongs to the *support* set when its projection is valid and its confidence exceeds a fixed floor, and is further counted as an *inlier* when its reprojection residual additionally satisfies $\|\mathbf{e}^{\text{low}}_{ij}(\mathbf{p})\|_2 \le \tau_{\text{geo}}$, where τ_{geo} is a fixed pixel-level tolerance. We then compute an acceptance score for each candidate pair:

$$s_{ij} = \frac{\bar{c}_{ij} r^{ij}_{\text{in}} \sqrt{r^{ij}_{\text{sup}}}}{1 + \bar{e}^{\text{low}}_{ij}}, \tag{18}$$

where $\bar{c}_{ij}$ denotes the mean confidence over the support set, r^{ij}_{sup} and r^{ij}_{in} the support and inlier ratios defined above, and $\bar{e}^{\text{low}}_{ij}$ the mean reprojection error over inliers.

A candidate is accepted when:

$$(i,j) \in \mathcal{S}^t_{\text{low}} \iff r^{ij}_{\text{sup}} \ge \tau_{\text{sup}} \ \wedge \ s_{ij} \ge \tau_{\text{score}}. \tag{19}$$

where $\mathcal{S}^t_{\text{low}}$ denotes the set of accepted low-frequency pairs, and τ_{sup} and τ_{score} are fixed thresholds for support and the acceptance score, respectively. Accepted candidates refresh the existing factor-state as:

$$\mathcal{F}_{ij} \leftarrow \left(\mathcal{M}^{\text{low}}_{ij}, \text{low}, 1\right), \qquad (i,j) \in \mathcal{S}^t_{\text{low}}, \tag{20}$$

and remain protected from high-frequency updates for the remaining factor lifetime. Rejected candidates leave the factor unchanged. Thus, low-frequency refresh changes only the stored measurement of an existing edge, without altering the graph topology or introducing a separate geometric state.

The two streams differ substantially in architecture, a local recurrent correlation-volume updater versus a global multi-view transformer, yet both are required to produce a candidate in the same BA coordinate system and the same target-weight format. Gauge alignment removes cross-source scale and reference-frame discrepancies before a candidate is even considered, and the acceptance criterion in Eq. (19) evaluates every candidate against the current geometric state rather than trusting either source's own confidence estimate. Unification is therefore enforced by this admission procedure, not merely asserted by the shared measurement format.

Table 1: Trajectory evaluation (ATE RMSE [m] ↓) on the TUM RGB-D dataset (Sturm et al., 2012). $*$ denotes results reproduced on our hardware.

Method	Uncalib.	360	desk	desk2	floor	plant	room	rpy	teddy	xyz	Avg.
DROID-SLAM (Teed & Deng, 2021)	✗	0.111	0.018	0.042	**0.021**	0.016	**0.049**	0.026	0.048	0.012	0.038
DPV-SLAM (Lipson et al., 2024)	✗	0.112	0.018	0.029	0.057	0.021	0.330	0.030	0.084	0.010	0.076
DPV-SLAM++ (Lipson et al., 2024)	✗	0.132	0.018	0.029	0.050	0.022	0.096	0.032	0.098	0.010	0.054
GO-SLAM (Zhang et al., 2023)	✗	0.089	**0.016**	0.028	0.025	0.026	0.052	**0.019**	0.048	0.010	0.035
MASt3R-SLAM (Murai et al., 2025)	✗	**0.049**	**0.016**	**0.024**	0.025	0.020	0.061	0.027	0.041	**0.009**	0.030
F²SLAM (ours)	✗	0.070	0.017	0.026	**0.021**	**0.014**	**0.049**	0.023	**0.034**	0.010	**0.029**
DROID-SLAM (Teed & Deng, 2021)	✓	0.202	0.032	0.091	0.064	0.045	0.918	0.056	0.045	0.012	0.163
MASt3R-SLAM (Murai et al., 2025)	✓	0.070	0.035	0.055	0.056	0.035	0.118	0.041	0.114	0.020	0.060
VGGT-SLAM (Maggio et al., 2025)	✓	0.071	0.025	0.040	0.141	0.023	0.102	0.030	0.034	0.014	0.053
VGGT-SLAM 2.0* (Maggio & Carlone, 2026)	✓	**0.050**	0.025	0.029	0.102	0.026	0.063	0.026	0.038	0.014	0.041
SLAM-Former* (Yuan et al., 2026)	✓	0.067	0.018	0.026	0.079	0.021	0.082	**0.017**	0.030	**0.011**	0.039
Lingbot-Map* (Chen et al., 2026)	✓	0.091	0.034	0.038	0.055	0.032	0.064	0.023	0.039	0.016	0.044
F²SLAM (ours)	✓	0.066	**0.016**	**0.023**	**0.015**	**0.013**	**0.043**	0.026	**0.029**	0.012	**0.027**

3.4 Dense Bundle Adjustment

F^{2}SLAM jointly optimizes the persistent geometric state using the measurements selected for the current BA solve. The backend is agnostic to their source. ℓ_{ij} and γ_{ij} affect measurement updates but do not enter the objective.

For a source pixel $\mathbf{p} \in \Omega_i$, its projection into keyframe j is:

$$\hat{\mathbf{p}}_{ij}\left(\mathbf{p}; \mathcal{X}_t\right) = \pi_{\boldsymbol{\theta}}\left(\mathbf{T}_j \mathbf{T}_i^{-1}\, \pi_{\boldsymbol{\theta}}^{-1}\left(\mathbf{p}, \mathbf{d}_i(\mathbf{p})\right)\right), \tag{21}$$

yielding the reprojection residual:

$$\mathbf{r}_{ij}\left(\mathbf{p}; \mathcal{X}_t\right) = \hat{\mathbf{p}}_{ij}\left(\mathbf{p}; \mathcal{X}_t\right) - \mathbf{p}^*_{ij}(\mathbf{p}). \tag{22}$$

Here, $\mathcal{E}^t_{\mathrm{BA}}$ contains the active factors and any recent inactive factors selected for the current solve.

Using $\mathbf{\Lambda}_{ij}$ from Eq. (5), we optimize:

$$\begin{aligned}\mathcal{X}_t^* = \arg\min_{\mathcal{X}_t} & \sum_{(i,j)\in\mathcal{E}^t_{\mathrm{BA}}} \sum_{\mathbf{p}\in\Omega_i} v_{ij}(\mathbf{p})\, \rho\big(\mathbf{r}_{ij}(\mathbf{p}; \mathcal{X}_t)^\top \mathbf{\Lambda}_{ij}(\mathbf{p}) \mathbf{r}_{ij}(\mathbf{p}; \mathcal{X}_t)\big) \\ & + \frac{\lambda_d}{2} \sum_{i\in\mathcal{K}_t} \|\nabla \mathbf{d}_i\|_2^2 + \frac{\lambda_\theta}{2} \|\boldsymbol{\theta} - \boldsymbol{\theta}_0\|^2_{\mathbf{\Sigma}_\theta^{-1}},\end{aligned} \tag{23}$$

where v_{ij} masks invalid correspondences, ρ is a robust penalty, the second term regularizes dense inverse depth, and the third term constrains camera intrinsics around the calibration prior $\boldsymbol{\theta}_0$. The intrinsic term is omitted when calibration is fixed.

Each edge contributes a single residual using its currently stored measurement. Thus, an accepted low-frequency prediction replaces the measurement on an existing edge without adding another residual or separate geometric state. Both streams constrain the same persistent state through a single dense BA. Detailed linearization and state updates are provided in Appendix A.2.

4 Experiments

4.1 Experimental Setup

We evaluate F^{2}SLAM on four standard SLAM benchmarks: TUM RGB-D (Sturm et al., 2012), 7-Scenes (Shotton et al., 2013), Replica (Straub et al., 2019), and KITTI Odometry (Geiger et al., 2012) datasets. Following (Maggio et al., 2025), we use the `evo` toolkit to Sim(3)-align estimated trajectories and report the root mean square error of absolute trajectory error (ATE RMSE). Following (Yuan et al., 2026), dense reconstruction is evaluated using accuracy (Acc.) and completion (Comp.) on Replica, and additionally Chamfer distance (Chamfer) on 7-Scenes. Unless noted otherwise, the **best** and second-best results in each column are highlighted in green (bold) and orange (underlined), respectively. For benchmarks reported separately by calibration setting (Tabs. 1 to 4), rankings are computed within each setting.

Table 2: Trajectory evaluation (ATE RMSE [m] ↓) on the 7-Scenes dataset (Shotton et al., 2013). ∗ denotes results reproduced on our hardware.

Method	Uncalib.	chess	fire	heads	office	pumpkin	kitchen	stairs	Avg.
DROID-SLAM (Teed & Deng, 2021)	✗	0.036	**0.027**	**0.025**	**0.066**	**0.127**	0.040	0.026	**0.050**
NICER-SLAM (Zhu et al., 2024)	✗	**0.033**	0.069	0.042	0.108	0.200	**0.039**	0.108	0.086
F²SLAM (ours)	✗	0.036	0.029	**0.025**	0.088	0.140	0.042	**0.021**	0.053
DROID-SLAM (Teed & Deng, 2021)	✓	0.047	0.038	0.034	0.136	0.166	0.080	0.044	0.078
MASt3R-SLAM (Murai et al., 2025)	✓	0.063	0.046	0.029	0.103	0.114	0.074	0.032	0.066
VGGT-SLAM* (Maggio et al., 2025)	✓	0.037	0.026	0.018	0.104	0.133	0.061	0.093	0.067
VGGT-SLAM 2.0* Maggio & Carlone (2026)	✓	0.039	0.027	0.021	0.106	0.135	0.053	0.092	0.067
SLAM-Former* Yuan et al. (2026)	✓	0.040	0.033	0.020	**0.065**	**0.066**	**0.035**	0.027	**0.041**
SLAM3R Liu et al. (2025)	✓	0.098	0.060	0.039	0.078	0.152	0.136	0.107	0.096
Lingbot-Map* (Chen et al., 2026)	✓	**0.034**	0.041	0.022	0.134	0.151	0.057	0.090	0.076
F²SLAM (Ours)	✓	0.036	**0.024**	**0.014**	0.087	0.140	0.041	**0.020**	0.051

Table 3: Trajectory evaluation (ATE RMSE [m] ↓) on the Replica dataset (Straub et al., 2019). ∗ denotes results reproduced on our hardware.

Method	Uncalib.	R0	R1	R2	O0	O1	O2	O3	O4	Avg.
DROID-SLAM (Teed & Deng, 2021)	✗	**0.003**	**0.001**	**0.003**	0.003	0.004	**0.003**	0.005	**0.004**	**0.003**
NICER-SLAM (Zhu et al., 2024)	✗	0.013	0.016	0.011	0.021	0.032	0.021	0.014	0.020	0.019
F²SLAM (Ours)	✗	**0.003**	0.002	**0.003**	**0.002**	**0.003**	**0.003**	**0.003**	**0.004**	**0.003**
VGGT-SLAM (Maggio et al., 2025)	✓	0.030	0.167	0.086	0.042	0.064	0.095	0.039	0.043	0.071
VGGT-SLAM 2.0* (Maggio & Carlone, 2026)	✓	0.030	0.049	0.040	0.026	0.018	0.024	0.024	0.029	0.030
SLAM-Former* (Yuan et al., 2026)	✓	0.031	0.033	0.025	0.030	0.027	0.038	0.033	0.036	0.032
SLAM3R (Liu et al., 2025)	✓	0.046	0.059	0.057	0.112	0.063	0.062	0.050	0.081	0.066
CUT3R (Wang et al., 2025b)	✓	0.145	0.243	0.127	0.159	0.230	0.162	0.088	0.204	0.170
StreamVGGT (Zhuo et al., 2025)	✓	0.113	0.163	0.077	0.076	0.070	0.180	0.153	0.168	0.125
Lingbot-Map* (Chen et al., 2026)	✓	0.202	0.165	0.104	0.156	0.115	0.166	0.101	0.213	0.153
F²SLAM (Ours)	✓	**0.002**	**0.002**	**0.002**	**0.002**	**0.003**	**0.002**	**0.003**	**0.002**	**0.002**

For trajectory evaluation, we compare against a broad range of classical, learning-based, and foundation-model-based SLAM methods (Mur-Artal & Tardós, 2017; Gao et al., 2018; Teed & Deng, 2021; Lipson et al., 2024; Zhu et al., 2024; Zhang et al., 2023; Murai et al., 2025; Maggio et al., 2025; Maggio & Carlone, 2026; Mandal et al., 2026; Yuan et al., 2026; Liu et al., 2025; Wang et al., 2025b; Zhuo et al., 2025; Chen et al., 2026). For reconstruction, we additionally include feed-forward reconstruction baselines (Wang et al., 2024; Leroy et al., 2024; Wang & Agapito, 2025). Baseline results are taken from the corresponding papers or prior benchmark reports, as specified in the table captions.

All experiments are performed on a single NVIDIA A100 GPU (40GB VRAM) with an AMD EPYC 7542 CPU. Runtime and memory usage are reported in Appendix A.4. We use deterministic inference without random sampling or seed-dependent operations, and uncalibrated methods do not use ground-truth camera intrinsics.

Unless stated otherwise, F²SLAM uses pretrained DROID-Net (Teed & Deng, 2021) for high-frequency tracking. For the low-frequency stream, per-image features are extracted with a pretrained DINO (Oquab et al., 2023) backbone and aggregated across the window before Flow3r (Cong et al., 2026) predicts pose, depth, and the displacement-confidence pairs described in Sec. 3.2. Flow3r predictions initialize the SLAM state and, after state-consistency validation, refresh existing measurements in the dense BA backend described in Sec. 3.

4.2 Trajectory Evaluation

As shown in Tabs. 1 to 4, F²SLAM achieves the best average result on three datasets and the second-best on 7-Scenes in the uncalibrated setting.

The strong result in Replica is consistent with favorable conditions for high-frequency DROID-Net tracking, including stable illumination, limited motion blur, and relatively small inter-frame motion. The low-frequency stream supplies complementary multi-view geometry, consistent with the ablations in Tab. 7.

Table 4: Trajectory evaluation (ATE RMSE [m] ↓) on the KITTI Odometry dataset (Geiger et al., 2012). Baseline results are taken from VGGT-SLAM++ (Mandal et al., 2026).

Method	Uncalib.	00	01	02	03	04	05	06	07	08	09	10	Avg.
ORB-SLAM2 (Mur-Artal & Tardós, 2017)	✗	40.65	502.20	47.82	**0.94**	1.30	29.95	40.82	16.04	**43.09**	38.77	**5.42**	69.73
LDSO (Gao et al., 2018)	✗	9.32	11.68	**31.98**	2.85	1.22	**5.10**	13.55	2.96	129.02	**21.64**	17.36	**22.43**
DROID-SLAM (Teed & Deng, 2021)	✗	92.10	5344.60	107.61	2.38	1.00	118.50	62.47	21.78	161.60	72.32	118.70	554.82
DPV-SLAM (Lipson et al., 2024)	✗	112.80	**11.50**	123.53	2.50	0.81	57.80	54.86	18.77	110.49	76.66	13.65	53.03
DPV-SLAM++ (Lipson et al., 2024)	✗	**8.30**	11.86	39.64	2.50	**0.78**	5.74	**11.60**	**1.52**	110.90	76.70	13.70	25.75
F²SLAM (ours)	✗	117.08	64.77	134.50	3.94	1.69	83.39	65.43	25.37	112.10	102.82	27.95	67.19
VGGT-SLAM (Sim(3)) (Maggio et al., 2025)	✓	125.11	120.96	288.82	5.16	0.96	29.92	15.03	14.58	235.80	38.44	18.60	81.22
VGGT-SLAM++ (Mandal et al., 2026)	✓	119.00	109.64	223.21	4.50	0.95	**25.21**	**13.65**	12.17	155.00	**35.26**	15.71	64.94
F²SLAM (Ours)	✓	**39.99**	**58.15**	**28.09**	**1.78**	**0.73**	30.35	14.49	**6.87**	**75.98**	68.20	**11.36**	**30.18**

Table 5: Reconstruction accuracy (Acc. [m] ↓) and completion (Comp. [m] ↓) on the Replica dataset (Straub et al., 2019). + denotes results reported in NICER-SLAM, − denotes results reported in SLAM3R, and ∗ denotes results reproduced on our hardware.

	Room0		Room1		Room2		Office0		Office1		Office2		Office3		Office4		Avg.	
Method	Acc.	Comp.	Acc.	Comp.	Acc.	Comp.	Acc.	Comp.	Acc.	Comp.	Acc.	Comp.	Acc.	Comp.	Acc.	Comp.	Acc.	Comp.
DROID-SLAM⁺ (Teed & Deng, 2021)	0.1218	0.0896	0.0835	0.0607	0.0326	0.1601	0.0301	0.1619	0.0239	0.1619	0.0566	0.1556	0.0449	0.0973	0.0465	0.0963	0.0550	0.1229
NICER-SLAM⁺ (Zhu et al., 2024)	**0.0253**	0.0304	0.0393	0.0410	0.0340	0.0342	0.0549	0.0609	0.0345	0.0442	0.0402	0.0429	0.0334	0.0403	**0.0303**	0.0387	0.0365	0.0416
MASt3R-SLAM* (Murai et al., 2025)	0.2468	0.2675	0.0393	0.0266	0.0258	0.0187	0.0336	0.0211	0.0359	0.0303	0.0609	0.0484	0.0881	0.0824	0.0742	0.0577	0.0756	0.0691
SLAM-Former* (Yuan et al., 2026)	0.0339	0.0247	**0.0248**	0.0171	0.0244	**0.0166**	**0.0208**	0.0153	**0.0178**	0.0138	0.0345	0.0222	**0.0296**	0.0213	0.0305	0.0214	0.0270	0.0187
VGGT-SLAM 2.0* (Maggio & Carlone, 2026)	0.0305	0.0622	0.0564	0.0765	0.0277	0.0629	0.0475	0.0457	0.0340	0.0595	**0.0249**	0.0792	0.0322	0.0739	0.0408	0.0743	0.0367	0.0657
DUSt3R⁻ (Wang et al., 2024)	0.0347	0.0250	0.0253	0.0186	0.0295	0.0176	0.0492	0.0351	0.0309	0.0221	0.0401	0.0310	0.0327	0.0225	0.0366	0.0261	0.0349	0.0248
MASt3R⁻ (Leroy et al., 2024)	0.0401	0.0410	0.0361	0.0325	0.0313	0.0215	0.0257	0.0163	0.1285	0.0813	0.0313	0.0199	0.0467	0.0315	0.0369	0.0247	0.0471	0.0336
Spann3R⁻ (Wang & Agapito, 2025)	0.0975	0.1294	0.1551	0.1294	0.0728	0.0850	0.0546	0.1875	0.0524	0.1664	0.0933	0.1180	0.1600	0.0903	0.1397	0.1602	0.1032	0.1333
SLAM3R⁻ (Liu et al., 2025)	0.0319	0.0240	0.0312	0.0234	0.0272	0.0200	0.0428	0.0260	0.0317	0.0234	0.0384	0.0278	0.0390	0.0316	0.0432	0.0336	0.0357	0.0262
F²SLAM (Ours)	0.0313	**0.0226**	0.0259	**0.0166**	**0.0242**	**0.0166**	0.0230	**0.0148**	**0.0178**	**0.0131**	0.0258	**0.0169**	0.0327	**0.0210**	0.0321	**0.0210**	**0.0266**	**0.0174**

On KITTI Odometry, it substantially outperforms the evaluated uncalibrated foundation-model-based baselines, including VGGT-SLAM (Sim(3)) and VGGT-SLAM++, although some calibrated systems remain stronger. In the calibrated setting, F²SLAM improves substantially over plain DROID-SLAM (554.82 m to 67.19 m average ATE), while it trails methods with explicit loop closure, such as LDSO and DPV-SLAM++, consistent with the absence of an explicit loop-closure mechanism in the current system (Sec. 5).

Additional qualitative trajectory visualizations are provided in Appendix A.6.

4.3 Reconstruction Evaluation

We evaluate dense reconstruction on the Replica and 7-Scenes datasets, with quantitative results in Tabs. 5 and 6 and qualitative comparisons in Fig. 3. For a fair comparison, we do not apply method-specific post-processing such as confidence-based point filtering (Yuan et al., 2026).

On Replica, F²SLAM achieves the best average reconstruction accuracy and completion among the compared methods. The lower completion error suggests that complementary low-frequency geometry helps recover regions weakly constrained by local tracking.

On 7-Scenes, F²SLAM matches the best reconstruction accuracy at 0.024 m, while reducing completion error from 0.030 m to 0.015 m and Chamfer distance from 0.027 m to 0.019 m relative to SLAM-Former. These correspond to approximately 50% and 30% improvements, respectively. Qualitative results show the same trend, with more continuous surfaces and fewer missing regions.

Additional qualitative reconstruction results are provided in Appendix A.7.

4.4 Ablation

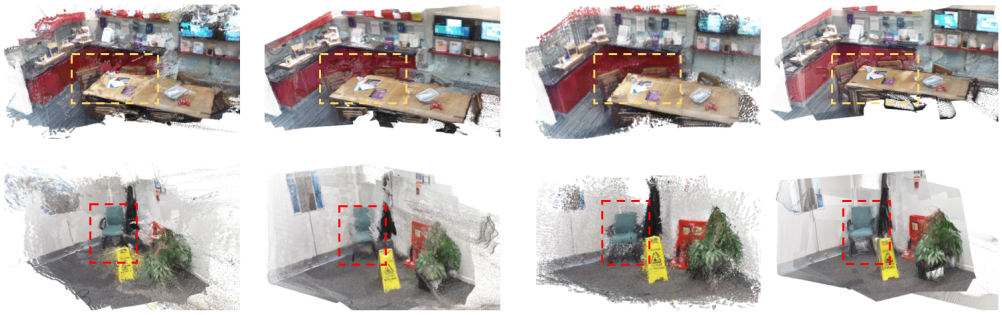
Intrinsic refinement. In the uncalibrated setting, F²SLAM jointly optimizes camera intrinsics, poses, and inverse depths rather than fixing ground-truth calibration. The paired results in Tabs. 1 to 4 show consistently lower average ATE, suggesting that joint refinement better adapts the geometric state to each sequence without requiring accurate calibration in advance. These gains may also reflect compensation for systematic reprojection biases from resizing, cropping, rectification, or camera-model conventions, without implying inaccurate dataset calibration.

Table 6: Reconstruction evaluation [m] on 7-Scenes dataset (Shotton et al., 2013). Results are averaged over all sequences.

Method	Acc. ↓	Comp. ↓	Chamfer ↓
MASt3R-SLAM (Murai et al., 2025)	0.089	0.085	0.087
VGGT-SLAM++ (Mandal et al., 2026)	0.054	0.060	0.057
SLAM-Former (Yuan et al., 2026)	**0.024**	0.030	0.027
SLAM3R (Liu et al., 2025)	0.026	0.028	0.027
F²SLAM (Ours)	**0.024**	**0.015**	**0.019**

Table 7: Ablation studies [m] on Replica dataset (Straub et al., 2019). Results are averaged over all sequences.

	Method	ATE ↓	Acc. ↓	Comp. ↓
(a)	Basic Tracking	0.0027	0.0303	0.0188
(b)	(a) + FF Init.	0.0026	0.0281	0.0177
(c)	(a) + Meas. Refresh	0.0026	0.0291	0.0179
(d)	Full w/o Gauge Align.	0.0024	0.0290	0.0194
(e)	Full w/o Gate	0.5848	0.1889	0.2186
(f)	**Full (Ours)**	**0.0023**	**0.0266**	**0.0174**



DROID-SLAM VGGT-SLAM **F²SLAM (Ours)** GT

Figure 3: Qualitative reconstruction comparison on representative sequences. F²SLAM recovers more continuous geometry with fewer missing regions and artifacts.

Ablation of feed-forward integration. Tab. 7 evaluates feed-forward initialization, measurement refresh, gauge alignment, and state-consistent gating on Replica dataset. Both initialization and measurement refresh independently improve performance over basic tracking. The full model achieves the strongest overall performance, reducing ATE from $0.0027\,\mathrm{m}$ to $0.0023\,\mathrm{m}$ and improving reconstruction accuracy and completion from $0.0303\,\mathrm{m}$ and $0.0188\,\mathrm{m}$ to $0.0266\,\mathrm{m}$ and $0.0174\,\mathrm{m}$, respectively. Removing gauge alignment degrades both trajectory and reconstruction quality, highlighting the importance of expressing predictions in the current SLAM gauge. Without state-consistent gating, performance degrades severely, demonstrating the need to reject inconsistent low-frequency predictions before measurement refresh. Further ablations on the low-frequency target-motion threshold, context window size, and gating threshold are provided in Appendix A.5.

5 Conclusion and Limitations

F²SLAM follows a single principle: feed-forward geometry should enter SLAM as optimization-native measurements rather than as a separate geometric state. A high-frequency tracking stream and a low-frequency feed-forward stream convert their predictions into dense target-weight measurements on a persistent factor graph, allowing the same poses, inverse depths, and optional camera intrinsics to be jointly refined through a single dense BA backend with a state-consistency check. The resulting system achieves strong trajectory estimation and dense reconstruction across the evaluated benchmarks.

F²SLAM does not currently include explicit loop closure, which limits performance on long-horizon outdoor sequences such as KITTI Odometry. Performance also depends on feed-forward prediction reliability, which can degrade under weak texture, dynamic objects, large viewpoint changes, and scale ambiguity. Incorporating loop closure and improving the robustness of low-frequency measurements are natural directions for future work.

AI USE STATEMENT

Large language models (LLMs) were not used in the core methodology, experimental pipeline, scientific analysis, or claimed contributions of this work. Specifically, LLMs were not involved in designing the proposed method, generating experimental results, selecting baselines, conducting evaluations, or drawing scientific conclusions.

LLM-based assistance was used only for language editing, such as improving grammar, clarity, and readability of the manuscript. This assistance did not introduce new technical ideas, modify the method, change the experimental results, or affect the originality of the work. All technical decisions, analyses, and claims were made by the authors, who take full responsibility for the content of the paper.

REFERENCES

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 multimap slam. *IEEE transactions on robotics*, 37(6):1874–1890, 2021.

Lin-Zhuo Chen, Jian Gao, Yihang Chen, Ka Leong Cheng, Yipengjing Sun, Liangxiao Hu, Nan Xue, Xing Zhu, Yujun Shen, Yao Yao, et al. Geometric context transformer for streaming 3d reconstruction. *arXiv preprint arXiv:2604.14141*, 2026.

Zhongxiao Cong, Qitao Zhao, Minsik Jeon, and Shubham Tulsiani. Flow3r: Factored flow prediction for scalable visual geometry learning. *arXiv preprint arXiv:2602.20157*, 2026.

Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pp. 834–849. Springer, 2014.

Xiang Gao, Rui Wang, Nikolaus Demmel, and Daniel Cremers. Ldso: Direct sparse odometry with loop closure. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2198–2204, 2018.

Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 3354–3361, 2012.

Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *European conference on computer vision*, pp. 71–91, 2024.

Lahav Lipson, Zachary Teed, and Jia Deng. Deep patch visual slam. In *European Conference on Computer Vision*, pp. 424–440, 2024.

Yuzheng Liu, Siyan Dong, Shuzhe Wang, Yingda Yin, Yanchao Yang, Qingnan Fan, and Baoquan Chen. Slam3r: Real-time dense scene reconstruction from monocular rgb videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 16651–16662, 2025.

Dominic Maggio and Luca Carlone. Vggt-slam 2.0: Real time dense feed-forward scene reconstruction. *arXiv preprint arXiv:2601.19887*, 2026.

Dominic Maggio, Hyungtae Lim, and Luca Carlone. Vggt-slam: Dense rgb slam optimized on the sl (4) manifold. *Advances in Neural Information Processing Systems*, 38:129839–129867, 2025.

Avilasha Mandal, Rajesh Kumar, Sudarshan Sunil Harithas, and Chetan Arora. Vggt-slam++. *arXiv preprint arXiv:2604.06830*, 2026.

Raul 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.

Riku Murai, Eric Dexheimer, and Andrew J Davison. Mast3r-slam: Real-time dense slam with 3d reconstruction priors. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 16695–16705, 2025.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2930–2937, 2013.

Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019.

Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 573–580, 2012.

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.

Hengyi Wang and Lourdes Agapito. 3d reconstruction with spatial memory. In *2025 International Conference on 3D Vision (3DV)*, pp. 78–89. IEEE, 2025.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5294–5306, 2025a.

Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 10510–10522, 2025b.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 20697–20709, 2024.

Yijun Yuan, Zhuoguang Chen, Kenan Li, Weibang Wang, Minghui Qin, Zhijian Fang, Weicheng Zheng, and Hang Zhao. Slam-former: Putting slam into one transformer. In *European Conference on Computer Vision*, pp. 56–74, 2026.

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*, pp. 3727–3737, 2023.

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. In *2024 International Conference on 3D Vision (3DV)*, pp. 42–52, 2024.

Dong Zhuo, Wenzhao Zheng, Jiahe Guo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Streaming 4d visual geometry transformer. *arXiv preprint arXiv:2507.11539*, 2025.

A APPENDIX

A.1 MOTION-BASED KEYFRAME AND WINDOW SCHEDULING

F^{2}SLAM uses image-space motion to schedule keyframe insertion and low-frequency inference. This avoids adding nearly duplicated frames to the persistent graph and adapts the temporal window to the observed camera motion.

Let I_{k_n} denote the latest keyframe and I_t the current frame. The high-frequency stream estimates a dense displacement field $\mathbf{u}_{k_n t}$ from I_{k_n} to I_t. We define the average image-space motion as:

$$m(k_n, t) = \frac{1}{|\Omega|} \sum_{\mathbf{p} \in \Omega} \left\| \mathbf{u}_{k_n t}(\mathbf{p}) \right\|_2 , \tag{24}$$

where Ω denotes the valid pixel set. The current frame is retained as a new keyframe when:

$$m(k_n, t) > \tau_{\mathrm{kf}}. \tag{25}$$

Otherwise, it is used only for transient tracking and is not added as a persistent graph node.

Newly inserted keyframes are accumulated in a low-frequency buffer:

$$\mathcal{B}_t = \{k_s, \ldots, k_e\}. \tag{26}$$

The accumulated image-space motion of the buffer is:

$$M(\mathcal{B}_t) = \sum_{i=s}^{e-1} m(k_i, k_{i+1}). \tag{27}$$

This quantity is used only as a scheduling signal and does not represent metric camera displacement.

Low-frequency inference is triggered when the buffer contains sufficient temporal context and its accumulated motion exceeds a predefined threshold. A maximum buffer length is additionally imposed to bound latency in slow-motion sequences. The corresponding thresholds are specified in the implementation details.

Once triggered, the current buffer forms the low-frequency window:

$$\mathcal{W}_t = \mathcal{B}_t. \tag{28}$$

The window does not define an independent submap or an additional optimized state. Instead, it determines which existing active edges are eligible for low-frequency factor generation and subsequent gating:

$$\mathcal{P}^t_{\mathrm{low}} = \left\{ (i, j) \in \mathcal{E}^t_{\mathrm{act}} \;\middle|\; i, j \in \mathcal{W}_t \right\}. \tag{29}$$

Accepted measurements refresh the corresponding target-weight factor-states, as described in Sec. 3.3. Thus, motion determines when the low-frequency stream is invoked, while both streams share the same graph topology and optimized geometric state.

A.2 STATE OPTIMIZATION WITH TARGET-WEIGHT FACTOR-STATES

F^{2}SLAM maintains and optimizes a single persistent geometric state using confidence-weighted dense reprojection constraints. For the current keyframe set $\mathcal{K}_t$, the optimized state is:

$$\mathcal{X}_t = \{\mathbf{T}_i, \mathbf{d}_i\}_{i \in \mathcal{K}_t} \cup \{\boldsymbol{\theta}\}, \tag{30}$$

where $\mathbf{T}_i \in SE(3)$ denotes the camera pose of keyframe i, and $\mathbf{d}_i \in \mathbb{R}_+^{h \times w}$ denotes its dense inverse-depth map. The camera intrinsics are parameterized as:

$$\boldsymbol{\theta} = (f_x, f_y, c_x, c_y)^\top . \tag{31}$$

When the camera calibration is known, $\boldsymbol{\theta}$ is fixed. Otherwise, it is refined jointly with the camera poses and inverse-depth maps.

Each active directed edge $(i, j) \in \mathcal{E}_{\text{act}}^t$ stores a mutable factor-state:

$$\mathcal{F}_{ij} = (\mathcal{M}_{ij}, \ell_{ij}, \gamma_{ij}), \tag{32}$$

where

$$\mathcal{M}_{ij} = \left(\mathbf{p}_{ij}^*, \mathbf{w}_{ij}\right) \tag{33}$$

is the current target-weight measurement. Here, $\mathbf{p}_{ij}^*$ is the dense target-coordinate field, $\mathbf{w}_{ij}$ is its per-pixel precision weight, $\ell_{ij} \in \{\text{high}, \text{low}\}$ records the latest measurement source, and $\gamma_{ij} \in \{0, 1\}$ is a temporary update lock.

During each bundle-adjustment step, $\mathcal{M}_{ij}$ is treated as a fixed observation. The source label ℓ_{ij} and lock flag γ_{ij} control factor-state updates but do not enter the optimization objective.

For a source pixel $\mathbf{p} \in \Omega_i$, the current state predicts its target coordinate in keyframe j as:

$$\hat{\mathbf{p}}_{ij}(\mathbf{p}; \mathcal{X}_t) = \pi_{\boldsymbol{\theta}}\left(\mathbf{T}_j \mathbf{T}_i^{-1} \pi_{\boldsymbol{\theta}}^{-1}(\mathbf{p}, \mathbf{d}_i(\mathbf{p}))\right), \tag{34}$$

where $\pi_{\boldsymbol{\theta}}$ and $\pi_{\boldsymbol{\theta}}^{-1}$ denote projection and inverse projection under the camera model specified by $\boldsymbol{\theta}$.

The corresponding dense target-coordinate residual is:

$$\mathbf{r}_{ij}(\mathbf{p}; \mathcal{X}_t) = \hat{\mathbf{p}}_{ij}(\mathbf{p}; \mathcal{X}_t) - \mathbf{p}_{ij}^*(\mathbf{p}). \tag{35}$$

The per-pixel precision defines the diagonal information matrix:

$$\boldsymbol{\Lambda}_{ij}(\mathbf{p}) = \operatorname{diag}(\mathbf{w}_{ij}(\mathbf{p})). \tag{36}$$

The persistent state is optimized by minimizing:

$$\begin{aligned} \mathcal{X}_t^* = \arg\min_{\mathcal{X}_t} & \sum_{(i,j) \in \mathcal{E}_{\text{BA}}^t} \sum_{\mathbf{p} \in \Omega_i} v_{ij}(\mathbf{p})\, \rho\left(\mathbf{r}_{ij}(\mathbf{p}; \mathcal{X}_t)^\top \boldsymbol{\Lambda}_{ij}(\mathbf{p}) \mathbf{r}_{ij}(\mathbf{p}; \mathcal{X}_t)\right) \\ & + \frac{\lambda_d}{2} \sum_{i \in \mathcal{K}_t} \|\nabla \mathbf{d}_i\|_2^2 \\ & + \frac{\lambda_\theta}{2} \|\boldsymbol{\theta} - \boldsymbol{\theta}_0\|_{\boldsymbol{\Sigma}_\theta^{-1}}^2 . \end{aligned} \tag{37}$$

where $v_{ij}(\mathbf{p}) \in \{0, 1\}$ masks invalid projections, $\rho(\cdot)$ is a robust penalty, and λ_d and λ_θ control the strengths of the inverse-depth and intrinsic regularization terms, respectively. The intrinsic prior is omitted when the camera calibration is fixed.

The first term in Eq. equation 37 enforces agreement between the current geometric state and the target-coordinate measurements stored on all active edges. The second term regularizes spatial variation in the dense inverse-depth maps. The third term constrains the optimized intrinsics around the calibration prior $\boldsymbol{\theta}_0$, with uncertainty described by $\boldsymbol{\Sigma}_\theta$.

For compact notation, we stack all inverse-depth maps as:

$$\mathbf{d} = \left[\mathbf{d}_1^\top, \ldots, \mathbf{d}_{N_t}^\top\right]^\top, \tag{38}$$

and denote the corresponding block-diagonal spatial-gradient operator by $\mathbf{D}$. The inverse-depth regularizer can then be written as:

$$\mathcal{R}_d(\mathbf{d}) = \frac{\lambda_d}{2} \|\mathbf{D}\mathbf{d}\|_2^2 . \tag{39}$$

Similarly, the intrinsic prior is:

$$\mathcal{R}_\theta(\boldsymbol{\theta}) = \frac{\lambda_\theta}{2} \|\boldsymbol{\theta} - \boldsymbol{\theta}_0\|_{\boldsymbol{\Sigma}_\theta^{-1}}^2 . \tag{40}$$

At the current estimate, the residual for each correspondence is linearized as:

$$\begin{aligned} \mathbf{r}_{ij}(\mathbf{p}; \mathcal{X}_t \boxplus \Delta) \approx \mathbf{r}_{ij}(\mathbf{p}; \mathcal{X}_t) & + \mathbf{J}_{ij}^{\xi_i}(\mathbf{p}) \delta\boldsymbol{\xi}_i \\ & + \mathbf{J}_{ij}^{\xi_j}(\mathbf{p}) \delta\boldsymbol{\xi}_j + \mathbf{J}_{ij}^{d}(\mathbf{p}) \delta\mathbf{d}_i(\mathbf{p}) \\ & + \mathbf{J}_{ij}^{\theta}(\mathbf{p}) \delta\boldsymbol{\theta}, \end{aligned} \tag{41}$$

where $\delta\boldsymbol{\xi}_i \in \mathbb{R}^6$ is the pose increment in the Lie algebra of $SE(3)$, $\delta\mathbf{d}_i$ is the inverse-depth increment, and $\delta\boldsymbol{\theta}$ is the intrinsic increment. When the intrinsics are fixed, the final term is omitted.

Using iteratively reweighted least squares for the robust penalty, all active residuals are stacked into:

$$\mathbf{r} + \mathbf{J}\Delta, \tag{42}$$

where $\widetilde{\mathbf{W}}$ combines the target precision, projection-validity masks, and robust weights.

The regularization terms contribute:

$$\mathbf{H}_d = \lambda_d \mathbf{D}^\top \mathbf{D}, \qquad \mathbf{g}_d = \lambda_d \mathbf{D}^\top \mathbf{D}\mathbf{d}, \tag{43}$$

and

$$\mathbf{H}_\theta = \lambda_\theta \boldsymbol{\Sigma}_\theta^{-1}, \qquad \mathbf{g}_\theta = \lambda_\theta \boldsymbol{\Sigma}_\theta^{-1} (\boldsymbol{\theta} - \boldsymbol{\theta}_0). \tag{44}$$

These terms are inserted into the blocks corresponding to the inverse-depth and intrinsic variables, respectively.

The resulting damped Gauss–Newton system is:

$$\big(\mathbf{J}^\top \widetilde{\mathbf{W}} \mathbf{J} + \mathbf{H}_d + \mathbf{H}_\theta + \lambda_{\mathrm{lm}} \mathbf{I}\big)\Delta = -\big(\mathbf{J}^\top \widetilde{\mathbf{W}} \mathbf{r} + \mathbf{g}_d + \mathbf{g}_\theta\big), \tag{45}$$

where λ_{lm} is the damping coefficient.

The complete increment vector is:

$$\Delta = \left[\delta\boldsymbol{\xi}_1^\top, \ldots, \delta\boldsymbol{\xi}_{N_t}^\top, \delta\mathbf{d}_1^\top, \ldots, \delta\mathbf{d}_{N_t}^\top, \delta\boldsymbol{\theta}^\top\right]^\top. \tag{46}$$

When the camera intrinsics are fixed, $\delta\boldsymbol{\theta}$ and the corresponding Jacobian, gradient, and Hessian blocks are removed.

After solving Eq. equation 45, the pose variables are updated using a right-multiplicative Lie-group update:

$$\mathbf{T}_i \leftarrow \mathbf{T}_i \exp\left(\delta\boldsymbol{\xi}_i^\wedge\right). \tag{47}$$

The inverse-depth maps are updated additively:

$$\mathbf{d}_i \leftarrow \mathbf{d}_i + \delta\mathbf{d}_i. \tag{48}$$

When jointly optimized, the camera intrinsics are updated as:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \delta\boldsymbol{\theta}. \tag{49}$$

Low-frequency information enters this optimization exclusively through the target-weight measurements stored on selected active edges. When a candidate passes the state-consistency gating procedure described in Sec. 3.3, it refreshes the current measurement $\mathcal{M}_{ij}$ on the corresponding edge. Rejected candidates leave the existing measurement unchanged, while all other active edges retain their current high-frequency measurements.

Importantly, refreshing an edge changes neither the graph topology nor the optimized variable set. Each active edge contributes exactly one dense target-weight residual, regardless of whether its current measurement was generated by the high-frequency stream or accepted from the low-frequency stream. The same bundle-adjustment objective and solver therefore optimize one persistent SLAM state using a unified set of high-frequency and accepted low-frequency constraints.

A.3 Experimental Details

We evaluate all methods at a common set of ground-truth timestamps for consistent comparison. Estimated poses are associated with the corresponding ground-truth poses by timestamp, and all trajectories are aligned to the ground truth using a similarity transformation in $\mathrm{Sim}(3)$. All reported trajectory errors are then computed from the aligned trajectories under the same evaluation protocol.

All methods are evaluated on the same input sequences and ground-truth trajectories. Unless otherwise specified, we use the original image resolution and the default configuration of each baseline, while preserving its original keyframe-selection and frame-processing strategy.

Table 8: Runtime performance comparison on the Replica dataset. All methods are evaluated on the same NVIDIA A100 (40 GB) GPU.

Method	FPS ↑	Peak GPU [MB] ↓
SLAM-Former	**8.154**	27693
VGGT-SLAM 2.0	3.909	**6523**
F²SLAM (Ours)	7.399	15443

Table 9: Sensitivity to the geometric consistency threshold on Replica.

τ_{geo}	ATE RMSE ↓
2.0	0.25
3.0	**0.23**
4.0	0.24

Table 10: Ablation study of Flow3r target motion on 7-Scenes. We report Sim(3)-aligned ATE RMSE (m). Lower is better.

Method	Opt. Intr.	Chess	Fire	Heads	Office	Pumpkin	RedKitchen	Stairs	Avg.
w/ target motion 600	✓	0.036	0.024	0.014	0.092	0.143	0.048	0.025	0.054
w/ target motion 800 (Ours)	✓	**0.036**	**0.024**	**0.014**	**0.087**	**0.140**	**0.041**	**0.020**	**0.051**
w/ target motion 1000	✓	0.036	0.025	0.014	0.087	0.140	0.041	0.020	0.051

For F²SLAM, we use the same configuration across all datasets. Keyframes are inserted with an image-space motion threshold of $\tau_{\text{kf}} = 40$. The low-frequency stream is triggered at a target motion of 800 and uses a maximum context size of 20 keyframes. For the window-size ablation, this maximum is varied from 4 to 40 while all other parameters remain fixed. For state-consistent admission, we set $\tau_{\text{sup}} = 0.05$, $\tau_{\text{geo}} = 3.0$, and $\eta_{\text{low}} = 0.25$. Accepted low-frequency measurements remain locked while active and are removed after 20 frontend updates. The high-frequency graph uses 3 temporal and 2 proximity neighbors, and dense BA performs 2 iterations per graph update.

A.4 Runtime Analysis

For SLAM systems, computational efficiency is also an important consideration. We therefore compare the runtime performance of different methods on the Replica dataset. All methods are evaluated on the same NVIDIA A100 GPU with 40 GB of memory under the same hardware environment.

We report the average processing frame rate in frames per second (FPS) together with peak GPU memory usage. FPS is computed based on the number of processed input frames and the corresponding processing time, and the reported results are averaged over the evaluated Replica sequences.

As shown in Tab. 8, F²SLAM achieves competitive computational efficiency, reaching 7.399 FPS with moderate GPU memory usage. Compared with SLAM-Former, our method achieves a similar processing speed while reducing peak GPU memory consumption. It also provides substantially higher processing throughput than VGGT-SLAM 2.0. These results demonstrate that F²SLAM maintains an effective balance between localization performance and computational efficiency.

A.5 Ablation on Low-frequency Temporal Support and Gating

We study two parameters that control the temporal support of the low-frequency stream: the target motion used to trigger inference and the number of keyframes included in each window. The target-motion threshold determines when sufficient camera displacement has accumulated for a low-frequency update, whereas the window size controls how much temporal context is jointly processed.

Target motion. Tab. 10 shows that increasing the target motion from 600 to 800 reduces the average ATE RMSE from 0.054 m to 0.051 m. The improvement is most evident on Office, RedKitchen, and Stairs, where a larger motion span provides additional viewpoint diversity beyond the short-range constraints of the high-frequency stream. This suggests that the low-frequency update is most useful when it captures sufficiently separated observations rather than repeatedly processing nearly redundant local views.

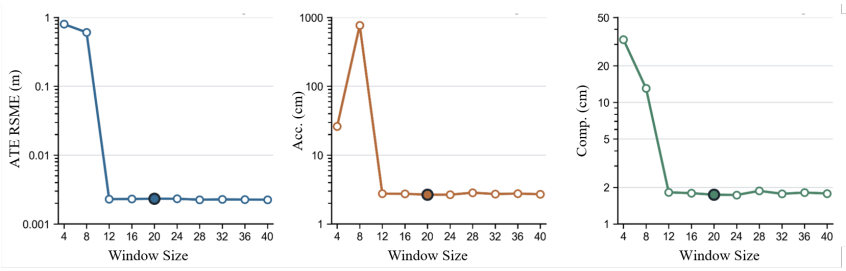


Figure 4: Ablation on the low-frequency context window size. Lower is better for all metrics. Log-scaled y-axes retain a continuous line plot while keeping the large errors at sizes 4 and 8 visible. We use a logarithmic coordinate system to better show the overall trend.

Increasing the threshold further to 1000 does not provide a consistent additional gain, with the average ATE remaining at 0.051 m. At such larger motion spans, the stronger geometric baseline is likely balanced by more severe viewpoint, occlusion, and appearance changes, which can reduce the reliability of the predicted measurements. Importantly, performance does not noticeably degrade, indicating that the candidate gating mechanism prevents unreliable low-frequency measurements from strongly perturbing the active graph. We therefore adopt 800 as the default target motion.

Window size. Fig. 4 evaluates maximum context sizes from 4 to 40 keyframes while keeping the data, optimizer, trigger threshold, and gating rules fixed. The two smallest settings produce catastrophic failures because their temporal support is insufficient to establish reliable multi-view geometry. With only a few keyframes, the window provides limited baseline and viewpoint diversity, making correspondence and depth predictions poorly constrained. The resulting low-frequency measurements are therefore noisy and can be rejected inconsistently by the state-consistency gate. This effect is especially severe in the smallest windows, where a small change in the sampled views can substantially alter the available geometric evidence.

Once the window reaches a moderate size (12 keyframes and above), it contains enough overlapping yet non-redundant views to support stable correspondence, depth, and confidence estimation. Candidate gating then suppresses residual outliers before they refresh graph factors, which explains the sharp recovery and the relatively flat, non-monotonic curves in the remaining range. Adding more keyframes beyond this regime mostly introduces redundant observations and does not yield a consistent geometric gain. We therefore use 20 keyframes as the default context size: it attains the best Accuracy while remaining close to the best completion and within the stable ATE regime. This is a balanced choice across the three metrics rather than a claim of uniformly optimal performance.

Overall, these ablations show that F^{2}SLAM requires a minimum amount of low-frequency temporal support, after which performance is relatively insensitive to the exact window size. This robustness is enabled by candidate gating and measurement-level factor refresh, which allow measurements conditioned on broader multi-view context to update the graph while suppressing unreliable updates.

Gating threshold. Tab. 9 evaluates the geometric consistency threshold τ_{geo} around its default value of 3.0. Performance remains stable across the tested range. A stricter threshold may reject useful feed-forward measurements, whereas a more permissive threshold may admit less reliable geometric evidence. The default setting provides a balanced operating point between measurement coverage and geometric consistency.

Overall, these ablations show that F^{2}SLAM remains robust to moderate variations in its low-frequency configuration. The target-motion and context-window experiments demonstrate stable behavior across different temporal support, while the gating analysis indicates that state-consistent admission does not rely on a narrowly tuned threshold.

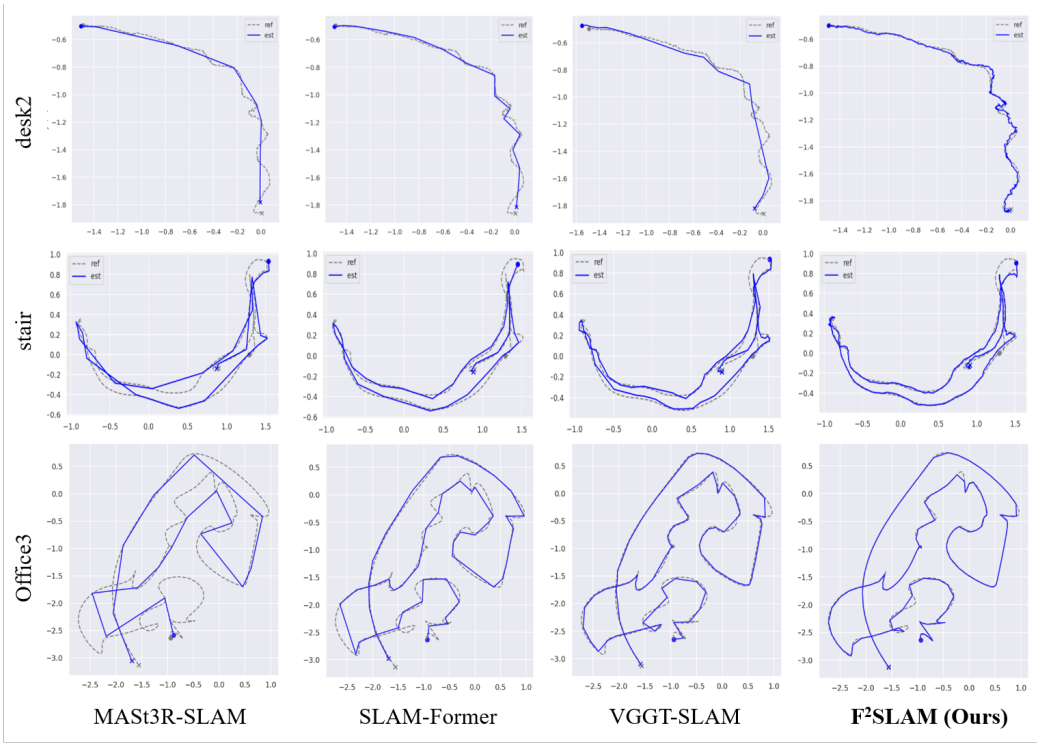


Figure 5: Qualitative trajectory comparison on representative sequences. All estimated trajectories are aligned to the ground truth using the same evaluation protocol. Compared with competing methods, F^{2}SLAM follows the reference trajectory more closely throughout the sequence and exhibits less accumulated drift.

A.6 Trajectory Visualization

We provide qualitative trajectory comparisons to complement the quantitative ATE evaluation. The visualizations examine trajectory accuracy from three perspectives: overall sequence-level behavior, two-dimensional projected motion, and full three-dimensional geometry.

Fig. 5 provides an overall comparison across representative sequences. Several competing methods recover short-term motion reasonably well, but small local errors accumulate into visible global deformation, endpoint offset, or trajectory-scale inconsistency. In contrast, F^{2}SLAM remains close to the ground truth over a larger portion of each sequence and preserves the global trajectory structure more faithfully. This shows that the improvement is not limited to isolated poses or final endpoint alignment, but extends across the complete keyframe trajectory.

Fig. 6 projects the trajectories onto a representative two-dimensional plane, making accumulated translation and orientation errors easier to identify. Although the baselines often reproduce the coarse motion pattern, their estimates progressively deviate around turns and long trajectory segments. F^{2}SLAM more accurately preserves both the local turning structure and the global path shape, resulting in smaller endpoint displacement and less systematic drift.

Fig. 7 further compares the trajectories in full three-dimensional space. This view exposes vertical drift, out-of-plane deviations, and geometric distortions that may appear small in a two-dimensional projection. The baseline trajectories exhibit larger spatial inconsistency as the sequence progresses, whereas F^{2}SLAM remains more coherent in all three dimensions and better preserves the overall motion geometry.

These qualitative results support the quantitative findings and clarify the source of the improvement. High-frequency factors maintain accurate local tracking, while accepted low-frequency measurements predicted from broader multi-view context repeatedly correct accumulated global drift within the persistent graph. As a result, F^{2}SLAM achieves not only lower endpoint error, but also better

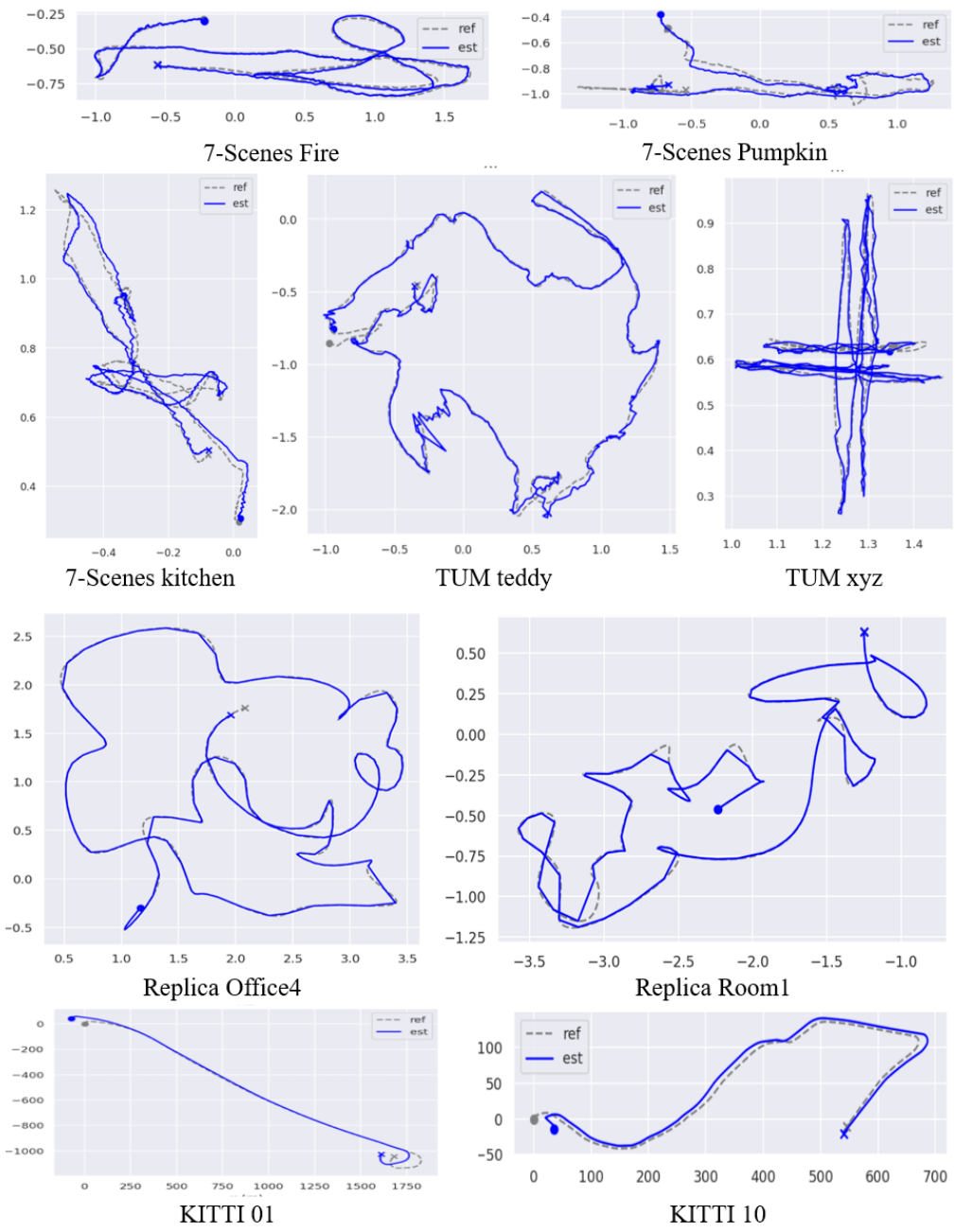


Figure 6: Two-dimensional trajectory projections on representative sequences. The projections emphasize global shape, turning behavior, and endpoint drift. F^{2}SLAM better preserves the reference path and avoids the progressive deviation observed in competing methods.

trajectory shape, stronger long-term consistency, and more stable motion estimates throughout the sequence.

A.7 Additional Qualitative Results

We provide additional qualitative results on Replica and KITTI to complement the quantitative evaluation in the main paper. Fig. 8 shows reconstructions of representative Replica scenes, including

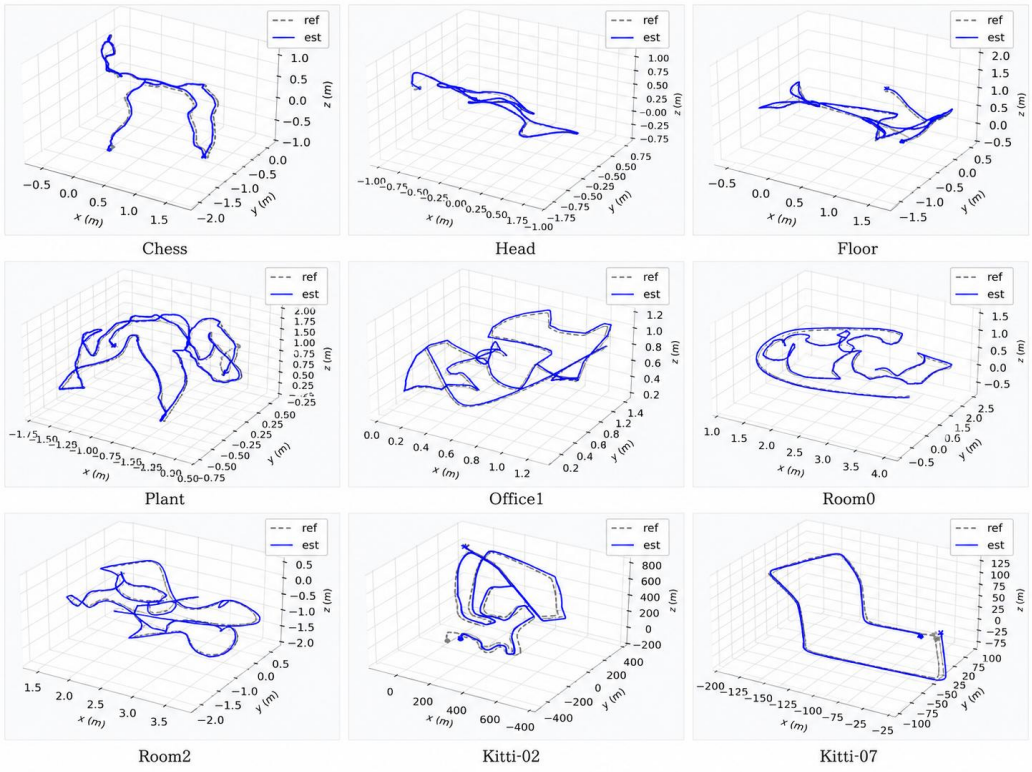


Figure 7: Three-dimensional trajectory visualization. The full spatial trajectories reveal vertical drift and geometric distortion that can be hidden by planar projections. F^{2}SLAM maintains a more coherent trajectory across all spatial directions.



Figure 8: Qualitative reconstruction results on representative Replica scenes. F^{2}SLAM recovers coherent room layouts and preserves major scene structures across long sequences.

Office0, *Room0*, and *Room1*. F^{2}SLAM recovers coherent room layouts and preserves the spatial arrangement of major scene structures across the sequences.

Fig. 9 visualizes estimated trajectories overlaid on the reconstructed scene for representative KITTI Odometry sequences, complementing the ATE-based analysis in Appendix A.6. These outdoor sequences contain long camera motions, limited image overlap, and repeated road structures.

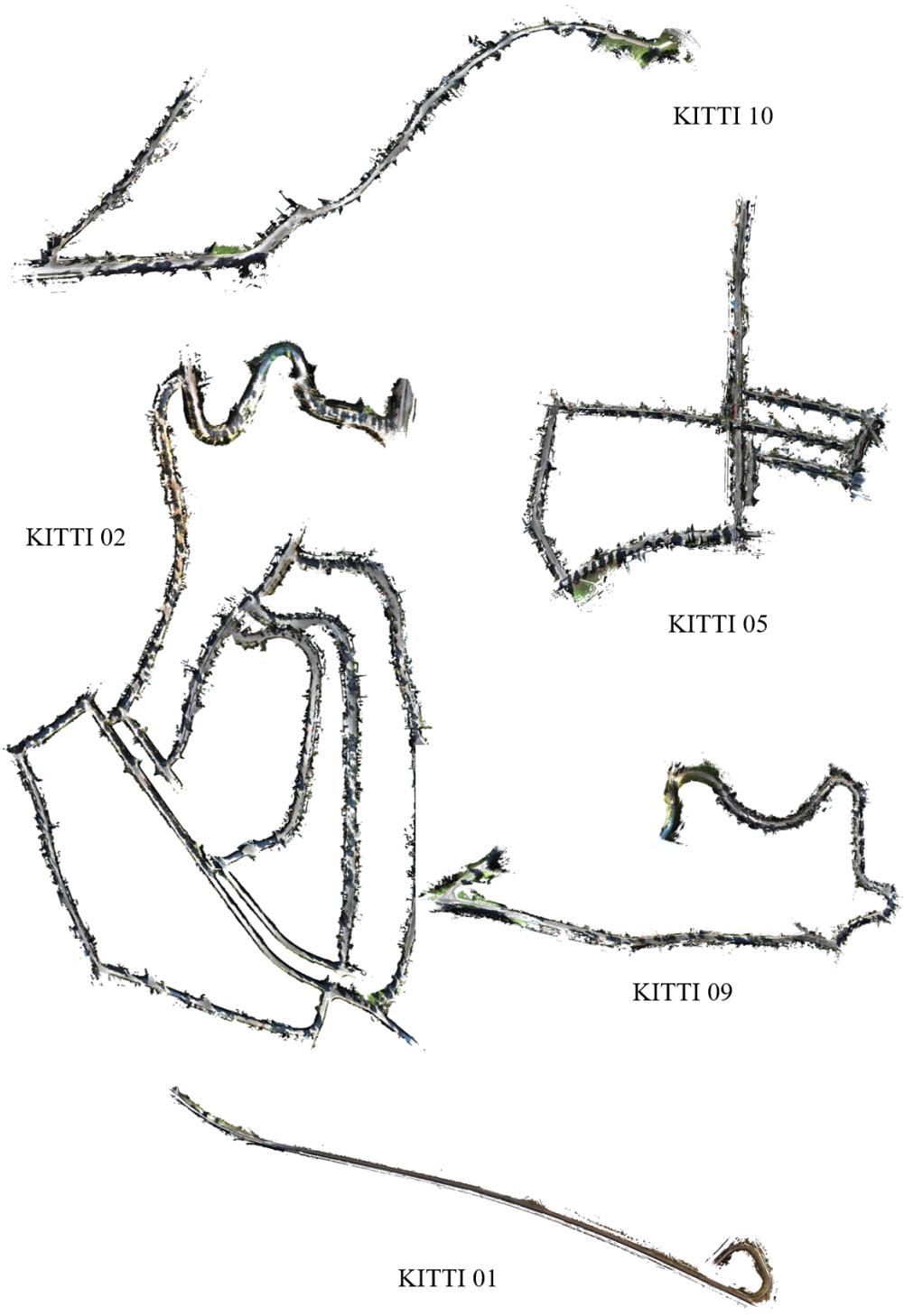


Figure 9: Qualitative reconstruction results on representative KITTI Odometry sequences. F^{2}SLAM maintains improved global consistency and reduces accumulated drift over long driving sequences.

By jointly optimizing high-frequency tracking factors and low-frequency wide-baseline factors, F^{2}SLAM maintains global trajectory consistency and reduces accumulated drift.