ESTIMATING CAMERA-MOTION FLOW FROM GYROSCOPE AND OPTICAL FLOW

Anonymous authors
Paper under double-blind review

ABSTRACT

Camera-motion flow captures image displacement caused by camera rotation and translation while excluding independent object motion. We estimate this flow by refining gyroscope-derived rotation flow using optical flow from a pretrained estimator. Two learned scores guide reconstruction from the residual between these two flows: motion compatibility identifies useful motion evidence, and scene proximity indicates nearness to the camera. Distant, compatible residuals guide global homography correction; nearby, compatible residuals guide neural parallax recovery. Our method achieves state-of-the-art accuracy on GHOF and GHOF-Cam, reducing point matching error and endpoint error by 15.4% and 28.3%, respectively, relative to the best baseline on each benchmark. Zero-shot transfer of the camera-motion modules to MPI-Sintel improves estimation over both input flows, including on dynamic pixels. Applications to video stabilization and robotic navigation further demonstrate improved stabilization and camera-pose-based reconstruction over baseline methods.

1 INTRODUCTION

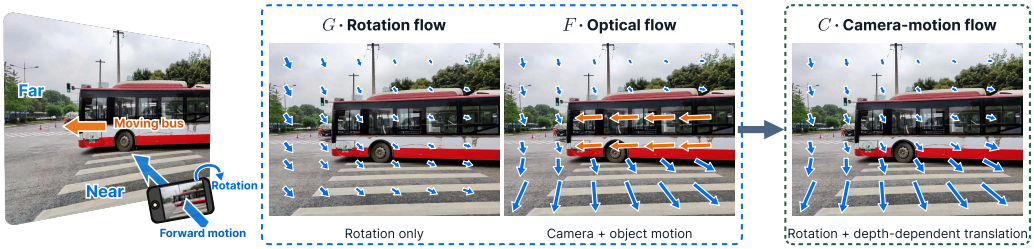


Figure 1: Estimating camera-motion flow C from gyroscope-derived rotation flow G and optical flow F. Blue backgrounds indicate the inputs (G, F), and the green background indicates the target C. For a camera moving forward and rotating, G captures depth-independent rotation-induced motion, while F also includes depth-dependent translational motion and independent motion of the bus. C retains camera-induced motion while excluding the bus's independent motion.

Estimating camera-induced motion between video frames is a fundamental step in computational photography and robotics. It supports tasks such as trajectory smoothing in video stabilization (Liu et al., 2013) and camera pose estimation in robotic navigation (Liu et al., 2022a). These applications require separating camera motion from independent object motion, since moving objects can distort the estimated camera trajectory. Even in a static scene, camera translation induces depth-dependent image displacements, giving rise to parallax. Our target is *camera-motion flow*: the dense 2D displacement induced by camera rotation and translation through scene geometry, excluding independent object motion (Figure 1).

Existing image-based methods estimate motion from correspondences between video frames. Classical ones fit global or spatially varying transformations to image correspondences (Lowe, 2004; Grundmann et al., 2012; Liu et al., 2016), while learning-based methods represent camera motion through combinations of local homography or motion bases (Liu et al., 2023; Li et al., 2025). Optical-flow-based approaches filter dense correspondences to obtain spatially coherent motion (Liu

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.

et al., 2014). These methods rely on image content to establish correspondences or predict motion, making them vulnerable to low light, motion blur, and adverse weather (Li et al., 2021).

Another line of work incorporates sensor measurements into motion estimation. Gyroscopes measure angular velocity independently of image content, enabling rotation estimation under challenging imaging conditions. Prior work combines gyroscope measurements with images to estimate optical flow and homography (Li et al., 2021; 2024b). However, neither output fully represents dense camera-motion flow: a global homography cannot generally capture translation-induced motion across varying depths, while optical flow also includes independent object motion. Gyroscope-derived rotation alone is likewise incomplete because it contains no camera translation. The challenge is thus to estimate dense camera-motion flow across varying scene depths while remaining robust to independent object motion and challenging imaging conditions.

Figure 1 illustrates the motion inputs: gyroscope-derived flow G captures depth-independent rotation, while optical flow F also includes depth-dependent translation and independent object motion. We recover camera-motion flow C by retaining the camera-induced components. Our central idea is to progressively refine G using the residual $F - G$, where F is estimated by a pretrained optical flow network (Wang et al., 2024b). This residual contains camera-induced translation, independent object motion, and noise in both the rotation flow and optical flow. To determine which residuals are useful and how to use them for camera-motion estimation, we introduce two learned guidance scores: motion compatibility, which indicates how useful each residual is for estimating camera motion, and scene proximity, which indicates how close each scene point is to the camera.

We reconstruct camera-motion flow through global correction followed by parallax recovery. Distant regions exhibit less parallax, making their compatible residuals useful for estimating a global homography correction A. This correction refines the gyroscope-derived rotation flow and approximates the image motion induced by camera translation in distant regions. Compatible residuals in nearby regions then guide a neural predictor that recovers the depth-dependent parallax P, yielding $C = G + A + P$. A compact guidance network predicts motion compatibility and scene proximity from G and F. We use depth information to supervise the guidance network during training. At inference, the network predicts both scores directly from G and F, without estimating depth.

We evaluate our method on sparse background correspondences from GHOF (Li et al., 2024b) and dense flow from GHOF-Cam (Li et al., 2025). The benchmarks cover regular and low-light conditions, as well as challenging weathers including fog, rain, and snow. They also include diverse camera and object motions. Our method achieves state-of-the-art (SOTA) accuracy on both benchmarks, reducing point matching error and endpoint error by approximately 15.4% and 28.3%, respectively, relative to the previous SOTA. To assess generalization and evaluate camera-motion flow in moving-object regions, we transfer our GHOF-trained camera-motion modules to MPI-Sintel (Butler et al., 2012), using synthesized rotation flow (G) and ground-truth camera-motion flow. With SEA-RAFT supplying optical flow (F), our method achieves lower overall endpoint error than both input flows, with a 73% reduction on dynamic pixels relative to F.

We further demonstrate the practical benefits of our method in video stabilization and robotic navigation. On the dataset introduced by Deep-FVS (Shi et al., 2022), our method improves stabilization quality over gyroscope-only and optical-flow-only baselines within the same stabilization pipeline. On sequences recorded with a quadruped robot, we evaluate all methods' estimated poses using the same per-frame depth maps and reconstruction procedure. Our poses yield more consistent point-cloud reconstructions than those of visual and visual–inertial baselines.

Our contributions are:

- We propose a framework that estimates dense camera-motion flow from gyroscope measurements and optical flow, progressively refining rotation flow through global correction and neural parallax recovery.
- We introduce motion compatibility and scene proximity scores to identify useful residuals and guide their contribution to camera-motion estimation. The scores are learned with depth supervision and predicted from motion inputs alone at inference.
- We achieve state-of-the-art accuracy on GHOF and GHOF-Cam, demonstrate zero-shot transfer of our camera-motion modules to MPI-Sintel with improved estimation, and show practical benefits in video stabilization and camera pose estimation.

2 Related Work

Gyroscope-guided estimation. Gyroscopes measure angular velocity independently of visual content, providing rotation information for video stabilization (Karpenko et al., 2011), egomotion estimation with optical flow (Bloesch et al., 2014), and image (Yang et al., 2025) and video deblurring (Rim et al., 2026). These applications exploit a motion signal that remains available when visual matching is unreliable. GyroFlow (Li et al., 2021) incorporates gyroscope-derived motion fields into unsupervised optical flow learning, while GyroFlow+ (Li et al., 2024b) estimates both homography and optical flow. The former output provides global alignment; the latter includes independent object motion. Neither directly represents dense camera-induced motion across varying scene depths while excluding object motion. Our contribution is to recover dense camera-motion flow from these complementary measurements, retaining gyroscope-derived rotation as an explicit reference and selecting camera-related evidence from pretrained optical flow.

Homography estimation. A homography models image motion induced by a planar scene or pure camera rotation. Classical methods match SIFT (Lowe, 2004) or ORB (Rublee et al., 2011) features and robustly fit a homography using RANSAC (Fischler & Bolles, 1981) or MAGSAC++ (Barath et al., 2020). Learned features and descriptors (DeTone et al., 2018; Tian et al., 2019) improve correspondence estimation, while deep networks estimate homographies through supervised (DeTone et al., 2016; Shao et al., 2021; Cao et al., 2022; Li et al., 2024a) or unsupervised learning (Zhang et al., 2020; Ye et al., 2021; Hong et al., 2022). These approaches address correspondence quality, outlier handling, and transformation estimation within a global model. A single homography cannot generally capture translation-induced motion across varying depths (Hartley & Zisserman, 2004); we therefore use it for global correction before recovering parallax.

Spatially varying motion estimation. Multi-homography models describe motion in different image regions for image stitching (Zaragoza et al., 2013), rolling-shutter correction (Grundmann et al., 2012), and video stabilization, alongside related mesh-based representations (Liu et al., 2013; 2016). Learned methods estimate local homographies using motion bases (Liu et al., 2023), while CamFlow represents dense camera motion with physical and stochastic bases (Li et al., 2025). RANSAC-Flow combines coarse homography alignment with learned pixelwise refinement for generic image alignment (Shen et al., 2020). These works establish motion models beyond a single homography, but greater flexibility alone does not determine whether a residual comes from camera translation or object motion. Our method recovers C from sensor-derived rotation flow G and pretrained optical flow F, without a multi-homography scene representation. Its global and local corrections are guided by compatibility and proximity rather than unrestricted agreement with image correspondences.

Camera motion and optical flow. Dense optical flow methods, including PWC-Net (Sun et al., 2018), RAFT (Teed & Deng, 2020), UFlow (Jonschkowski et al., 2020), FlowFormer (Huang et al., 2022), and SEA-RAFT (Wang et al., 2024b), provide motion evidence but do not themselves separate camera and object motion. SteadyFlow enforces spatial coherence in optical flow for video stabilization (Liu et al., 2014). GeoNet takes a geometric approach, constructing rigid flow from predicted depth and camera pose and estimating residual flow to account for non-rigid motion (Yin & Shi, 2018). GlobalFlowNet distills an optical flow network into a spatially smooth global-motion estimator using low-pass filtering (James et al., 2023). Our camera-motion flow target is closely related to rigid flow: both describe image displacement induced by camera motion through scene geometry. The distinction lies in how we estimate it. We recover C directly from gyroscope-derived rotation flow G and pretrained optical flow F, without estimating depth or full camera pose at inference. Our reconstruction preserves translation-induced parallax while suppressing independent object motion; depth is used only to construct training supervision.

3 Method

3.1 Overview

Given gyroscope-derived rotation flow G and optical flow F estimated from an image pair using SEA-RAFT (Wang et al., 2024b), our method estimates camera-motion flow C. As illustrated in Figure 2, we progressively refine G using the residual $F - G$. Two learned scores determine which

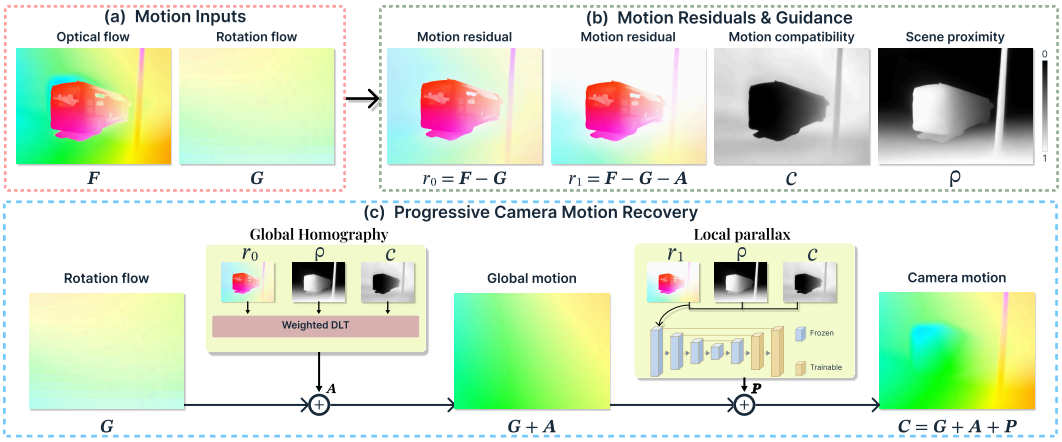


Figure 2: Overview of our method. (a) Optical flow F and gyroscope-derived rotation flow G. (b) Motion residuals r_0, r_1 and predicted guidance scores c, ρ. (c) Guided by these scores, we predict local parallax and refit the global homography once to obtain $C = G + A + P$. Brighter guidance values indicate higher compatibility or proximity.

residuals are useful and how they contribute to camera-motion estimation: motion compatibility c indicates how useful each residual is for estimating camera motion, and scene proximity ρ indicates how close each scene point is to the camera. Both scores lie in $[0, 1]$, with larger values indicating more useful residuals and nearer scene points, respectively. Compatible residuals in distant regions guide global homography correction, while those in nearby regions guide recovery of depth-dependent parallax. Combining these components yields:

$$C = G + A + P, \tag{1}$$

where A is a global homography correction and P captures spatially varying parallax. Combining these components lets downstream tasks select their motion representation; for example, video stabilization smooths the global component $G + A$, while pose estimation uses the dense flow $G + A + P$ to retain depth-dependent correspondences.

3.2 Motion Inputs

Figure 2(a) illustrates the two motion inputs, F and G. A gyroscope measures 3-axis angular velocity, enabling camera rotation estimation independently of image content (Karpenko et al., 2011). We integrate these measurements over time and use camera intrinsics to convert the estimated rotations into pixel displacements, yielding rotation flow G. This computation accounts for rolling shutter, where image rows are exposed at different times; details are provided in Appendix A. However, G is subject to gyroscope measurement errors (Xu et al., 2026; Liu et al., 2022b), and it captures only rotation and therefore lacks parallax caused by camera translation. We use optical flow F to refine G and recover this missing parallax.

We use pretrained SEA-RAFT (Wang et al., 2024b) to estimate F from two consecutive frames. This flow captures per-pixel displacements caused by camera rotation and translation, as well as independent object motion. Because it relies on visual content, F can be less reliable in low-light or adverse weather conditions, in textureless regions, or under motion blur. Occluded pixels pose an additional challenge because they have no visible correspondence in the next frame. We therefore selectively use motion in F that is compatible with camera motion to refine G and recover parallax.

3.3 Global Homography

The first correction stage estimates a global homography from the residual r_0 shown in Figure 2(b). At each pixel $\boldsymbol{p}$:

$$r_0(\boldsymbol{p}) = F(\boldsymbol{p}) - G(\boldsymbol{p}). \tag{2}$$

We fit a homography to the residual $F - G$ to estimate the global correction A. Not all residuals are equally useful: nearby scene points may carry parallax, while independent object motion and flow errors can distort the fit. A guidance network takes G and F as inputs and predicts motion

compatibility c and scene proximity ρ, while providing shared motion features Z for parallax prediction. Its architecture and training are described in Section 3.5. We retain pixels whose residual magnitude is below a threshold and weight them using c and ρ to favor distant, compatible residuals. The weighting scheme is detailed in Appendix B.

Each retained pixel provides a correspondence between its original position $\boldsymbol{p}$ and the residual-displaced position $\boldsymbol{p} + r_0(\boldsymbol{p})$. We fit a homography to these correspondences using weighted Direct Linear Transform (DLT) (Hartley & Zisserman, 2004). Applying this transformation to every pixel and subtracting its original position yields the homography correction A. Adding A to G gives the global motion estimate $G + A$. The remaining residual guides parallax recovery in the next stage.

3.4 Local Parallax

After the initial global fit, the remaining residual is

$$r_1(\boldsymbol{p}) = F(\boldsymbol{p}) - G(\boldsymbol{p}) - A(\boldsymbol{p}), \tag{3}$$

where A is fitted without a parallax term. This residual contains parallax, independent object motion, and flow errors. A lightweight network g_ϕ takes r_1 together with the guidance network's features Z and scores c, ρ to predict parallax. Providing r_1 explicitly lets the predictor account for the motion already removed by the global fit. We weight the prediction by scene proximity:

$$P = \rho \odot g_\phi(Z, r_1, \rho, c), \tag{4}$$

where ϕ denotes the network parameters and $\odot$ denotes elementwise multiplication. Proximity weighting reduces the prediction's contribution in distant regions.

We train g_ϕ offline on pseudo-targets from an analytical solver guided by the predicted c and ρ. These scores are learned using depth estimates from pretrained Pi3 (Wang et al., 2026). Neither Pi3 nor the pseudo-target solver is required at inference. Training details are provided in Appendix C. As illustrated in Figure 2(c), we predict P once and then refit A to $F - G - P$ using the weighted homography fitting in Section 3.3. We keep P fixed during this refit and output $C = G + A + P$.

3.5 Learning Motion Compatibility and Scene Proximity

We train a compact guidance network to predict motion compatibility c and scene proximity ρ from G and F. Frozen Pi3 (Wang et al., 2026) provides depth estimates to construct the proximity target and guide an analytical teacher for camera-motion flow. Agreement between optical flow and the teacher flow defines the compatibility target. Both predictions are trained with an L1 loss, without requiring Pi3 at inference. The network takes the concatenation of G, F, and $F - G$ as input. A shared U-Net encoder–decoder extracts features Z, and two convolutional heads with sigmoid outputs predict c and ρ. The same features feed the parallax predictor. Target construction and training details are provided in Appendix B.

4 Experiments

4.1 Datasets and Evaluation Metrics

We train our guidance network and parallax predictor on the GHOF (Li et al., 2024b) training split and evaluate camera-motion flow using sparse background correspondences and dense static-region annotations. GHOF provides synchronized images and gyroscope measurements under regular (RE), foggy (FOG), low-light (LL), rainy (RAIN), and snowy (SNOW) conditions. We measure point matching error (PME), the Euclidean distance between background points displaced by the estimated motion and their annotated matches. GHOF-Cam (Li et al., 2025) extends GHOF with dense motion annotations. We measure endpoint error (EPE) between the estimated flow and the reference optical flow on valid static pixels, where optical flow represents camera-induced motion. Both metrics are reported in pixels; lower is better.

MPI-Sintel (Butler et al., 2012) is a synthetic optical-flow dataset derived from the animated film Sintel, with Albedo, Clean, and Final rendering passes. We use 1,041 frame pairs from the Final training split, which includes motion blur, defocus blur, and atmospheric effects, to evaluate zero-shot transfer of our camera-motion modules. Its depth and camera annotations allow us to construct camera-motion flow ground truth and evaluate EPE over both static and dynamic regions.

Table 1: Camera-motion estimation on GHOF and GHOF-Cam. PME measures annotated background point matching error; EPE measures flow error on valid static pixels. Both are in pixels (↓). * denotes fine-tuning on GHOF. Best and second-best results are **bold** and underlined.

	GHOF: sparse PME ↓						GHOF-Cam: dense EPE ↓					
Method	AVG	RE	FOG	LL	RAIN	SNOW	AVG	RE	FOG	LL	RAIN	SNOW
Identity $\mathbf{I}_{3\times3}$	6.33	4.94	7.24	8.09	5.48	5.89	5.22	3.65	6.69	5.88	4.90	4.96
SIFT (Lowe, 2004)	4.80	0.59	4.47	12.10	0.62	6.20	2.82	0.60	2.43	7.09	0.61	3.37
SOSNet (Tian et al., 2019)	5.27	0.72	5.56	12.95	0.77	6.37	3.29	1.23	3.52	6.59	0.69	4.42
SPSG (Sarlin et al., 2020)	4.47	3.54	2.21	10.66	0.83	5.10	3.07	3.99	1.57	6.88	0.79	2.16
LoFTR (Sun et al., 2021)	3.04	2.64	1.33	6.71	0.56	3.94	0.94	0.69	0.30	1.89	0.57	1.26
DHN (DeTone et al., 2016)	6.61	6.04	6.02	7.68	6.99	6.32	4.64	4.56	5.25	3.82	5.33	4.24
LocalTrans (Shao et al., 2021)	5.72	4.06	6.49	5.95	5.78	6.34	5.44	5.71	4.55	6.90	5.39	4.64
IHN (Cao et al., 2022)	8.17	7.10	8.71	9.34	6.57	9.13	4.59	3.85	4.99	7.37	2.85	3.88
RealSH (Jiang et al., 2023)	1.72	1.60	0.88	4.42	0.43	1.28	1.30	1.44	0.96	2.74	0.47	0.88
DMHomo (Li et al., 2024a)	1.75	0.64	0.85	4.16	0.39	2.74	1.08	0.71	0.51	2.50	0.41	1.26
DMHomo* (Li et al., 2024a)	0.70	0.60	0.47	1.35	0.47	0.63	0.70	0.81	0.40	1.13	0.49	0.65
CAHomo (Zhang et al., 2020)	3.87	4.10	3.84	6.99	1.27	3.17	2.81	2.02	2.03	4.56	2.84	2.61
BasesHomo (Ye et al., 2021)	2.28	2.02	1.43	4.90	0.78	2.29	1.74	1.39	0.97	4.12	0.66	1.58
HomoGAN (Hong et al., 2022)	1.95	1.73	0.60	3.95	0.47	3.02	1.16	1.61	0.47	1.94	0.45	1.35
HomoGAN* (Hong et al., 2022)	0.79	0.75	0.52	1.39	0.59	0.69	0.73	0.81	0.42	1.16	0.60	0.68
MeshFlow (Liu et al., 2016)	1.89	0.88	2.64	3.16	0.57	2.20	2.15	1.09	2.21	5.57	0.44	1.69
HM-Mix (Grundmann et al., 2012)	4.15	3.42	2.35	6.41	3.78	4.80	4.35	1.02	4.03	8.75	1.53	6.42
RANSAC-Flow (Shen et al., 2020)	3.11	3.18	2.43	4.92	2.17	2.83	3.26	2.81	3.14	5.12	2.21	3.04
CamFlow (Li et al., 2025)	1.23	1.15	0.96	2.69	0.40	0.93	1.10	1.08	0.74	2.15	0.46	1.05
CamFlow* (Li et al., 2025)	0.64	0.55	0.39	1.15	0.41	0.68	0.64	0.71	0.31	1.04	0.45	0.71
VGGT (Wang et al., 2025)	1.96	1.55	1.07	3.31	1.80	2.09	1.88	1.24	1.08	3.63	1.64	1.67
DUSt3R (Wang et al., 2024a)	5.56	4.65	6.28	7.52	4.46	4.91	4.68	3.18	5.39	6.59	3.97	4.29
Pi3 (Wang et al., 2026)	2.78	2.75	2.16	4.30	2.36	2.32	2.13	2.43	1.77	2.61	2.11	1.72
GyroFlow+* (Li et al., 2024b)	0.52	0.28	**0.23**	1.04	0.46	**0.56**	0.60	0.81	**0.19**	0.91	0.51	0.57
Ours	**0.44**	**0.24**	0.25	**0.80**	**0.33**	**0.56**	**0.43**	**0.38**	0.20	**0.66**	**0.32**	**0.56**

4.2 Comparison Methods

We compare with homography-based and spatially varying motion estimators. Homography baselines include methods that fit homography from correspondences (SIFT (Lowe, 2004), SOSNet (Tian et al., 2019), SuperPoint–SuperGlue (SPSG) (DeTone et al., 2018; Sarlin et al., 2020), and LoFTR (Sun et al., 2021)) and learning estimators (DHN (DeTone et al., 2016), LocalTrans (Shao et al., 2021), IHN (Cao et al., 2022), RealSH (Jiang et al., 2023), DMHomo (Li et al., 2024a), CAHomo (Zhang et al., 2020), BasesHomo (Ye et al., 2021), and HomoGAN (Hong et al., 2022)). Spatially varying baselines include MeshFlow (Liu et al., 2016), Homography Mixture (HM-Mix) (Grundmann et al., 2012), RANSAC-Flow (Shen et al., 2020), and CamFlow (Li et al., 2025).

Additional comparisons include 3D geometry models (DUSt3R (Wang et al., 2024a), VGGT (Wang et al., 2025), and Pi3 (Wang et al., 2026)) and the gyroscope-guided method GyroFlow+ (Li et al., 2024b). The identity warp serves as a no-motion baseline.

We follow the sparse and dense evaluation protocols of GHOF and GHOF-Cam (Li et al., 2025), respectively. Methods marked with * in Table 1 are fine-tuned on the GHOF training split; the remaining comparison methods use their released or pretrained configurations.

4.3 Quantitative Comparison

4.3.1 Sparse Background Correspondences

Table 1 reports PME on GHOF. Our method achieves the lowest average PME of 0.44, reducing the error by approximately 15.4% relative to GyroFlow+ (0.52). It improves RE (0.28 to 0.24), LL (1.04 to 0.80), and RAIN (0.46 to 0.33). Both methods reach 0.56 in SNOW, while our error is slightly higher in FOG (0.25 versus 0.23). Performance varies substantially across imaging conditions. SIFT achieves a PME of 0.59 in RE, but its error rises to 12.10 in LL. GHOF-fine-tuned HomoGAN and DMHomo perform better in LL, with errors of 1.39 and 1.35, respectively. Among visual-only methods without GHOF fine-tuning, CamFlow achieves the lowest average PME of 1.23, compared with 1.96 for VGGT. Fine-tuning further improves CamFlow to 0.64.

4.3.2 DENSE STATIC-REGION FLOW

Table 1 reports EPE on valid static pixels in GHOF-Cam. Our method achieves the lowest average EPE of 0.43, reducing the error by approximately 28.3% relative to GyroFlow+ (0.60). Among spatially varying baselines, CamFlow achieves an average EPE of 1.10, outperforming MeshFlow, HM-Mix, and RANSAC-Flow, and improves to 0.64 after GHOF fine-tuning. The 3D geometry models VGGT and DUSt3R yield higher errors of 1.88 and 4.68, respectively. Rigid flow reconstructed directly from Pi3 reaches 2.13 EPE and 2.78 PME, showing that its predicted geometry and poses alone do not provide accurate camera-motion flow under this protocol (Appendix B).

4.4 GENERALIZATION TO SYNTHETIC SCENES

We transfer the guidance network and parallax predictor pretrained on GHOF to MPI-Sintel (Butler et al., 2012), without fine-tuning these modules. SEA-RAFT remains the pretrained optical-flow input estimator. On 1,041 final-pass frame pairs, we construct camera-motion ground truth C^* from depth and camera poses and synthesize GT rotation flow. Our method achieves an overall EPE of 4.76, compared with 12.20 for F and 5.93 for G. On dynamic pixels, ours achieves 9.61 EPE versus 35.65 for F and 10.24 for G: reductions of 73.0% and 6.2%. Dynamic error is lower than F in all scenes containing dynamic pixels. These results support suppressing independent object motion and generalizability of our method. Appendix F describes the EPE and reports overall and dynamic-region results in Table 6.

4.5 QUALITATIVE COMPARISON

We compare alignment in scenes containing large moving vehicles and static structures at different depths under regular and degraded imaging conditions. Following prior visualization practices (Li et al., 2025; Hong et al., 2022), we visualize alignment using color ghosting and heat maps. Separated colored contours indicate residual image motion, and brighter heat-map responses indicate larger alignment errors in static regions. Independent object motion is expected to remain after camera-motion compensation, so reduced ghosting on a moving object does not necessarily indicate a better camera-motion estimate.

4.5.1 COMPARISON WITH HOMOGRAPHY-BASED METHODS

Figure 3 compares our method with SIFT (Lowe, 2004), BasesHomo (Ye et al., 2021), HomoGAN (Hong et al., 2022), and DMHomo (Li et al., 2024a) under low-light, snowy, and regular conditions. The first two rows include large moving vehicles. Low light and motion blur obscure background details in the first row, while snowy regions provide little texture in the second. The compared methods show visible misalignment in the highlighted static regions, whereas our method achieves better alignment. In the third row, our method aligns both the distant background and the nearby road despite their different depths.

4.5.2 COMPARISON WITH DENSE MOTION BASELINES

Figure 4 compares our method with MeshFlow (Liu et al., 2016), VGGT (Wang et al., 2025), CamFlow (Li et al., 2025), and GyroFlow+ (Li et al., 2024b). The first two rows show forward camera motion with large moving vehicles. In the first row, MeshFlow aligns the nearby static region but leaves errors in the distant background. GyroFlow+ aligns the distant background but leaves errors in the nearby region. Our method aligns both regions. In the second row, MeshFlow achieves static-region alignment comparable to ours, but the reduced ghosting on the moving bus suggests that it also compensates for part of the bus's independent motion. In the third row, all methods achieve reasonable alignment under foggy conditions, while our method shows smaller alignment errors in the highlighted static regions.

4.6 APPLICATIONS

Our method supports different requirements: stabilization uses $G + A$ for global smoothing, while camera pose estimation uses $G + A + P$ for dense correspondences across scene depths.

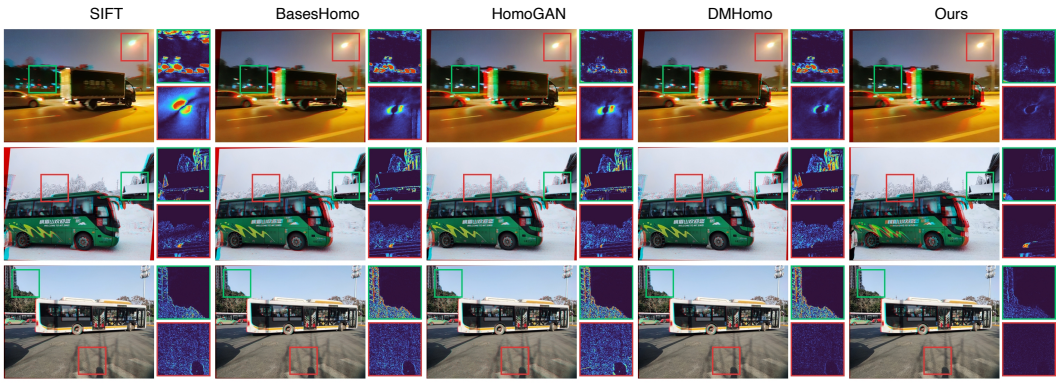


Figure 3: Qualitative comparison with homography-based methods. Each column shows one method. Colored ghosting reveals misalignment; green and red boxes mark static regions with corresponding alignment heat maps on the right. Brighter responses indicate larger alignment errors.

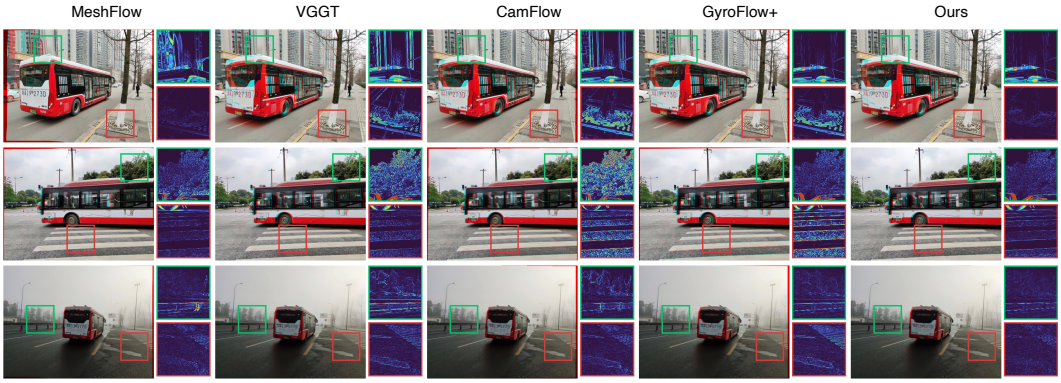


Figure 4: Qualitative comparison with dense motion baselines. Each column shows one method. Green and red boxes mark static regions with corresponding alignment heat maps on the right; brighter responses indicate larger alignment errors. Ghosting on moving objects can remain after camera-motion compensation because their independent motion is not removed.

4.6.1 Video Stabilization

On Deep-FVS (Shi et al., 2022), we fine-tune on 16 training videos and evaluate on 34 test videos. Within the same stabilization pipeline, we compare rotation flow (G-only), SEA-RAFT optical flow (Wang et al., 2024b) (F-only), and our method (Ours). Figure 5 shows frames at the same time index: F-only distorts the moving person and nearby background, whereas Ours better preserves their geometry. We quantify stability by the high-frequency jitter of camera paths estimated from the stabilized videos, and non-rigid distortion by the deviation of mesh warps from their best-fitting homographies. Larger deviations indicate more local deformation. On the 34 test videos, Ours reduces jitter from 12.2 px with G-only to 7.2 px, matching F-only, while reducing the non-rigid residual from 5.11 px with F-only to 0.40 px. Ours produces no mesh foldovers, whereas F-only produces at least one flipped mesh triangle in 33.6% of frames. These results show improved stability over G-only and less distortion than F-only. Appendix G provides more detail and examples.

4.6.2 Robotic Navigation

A quadruped robot navigates from point A to point B while synchronized RGB-D and IMU measurements are recorded. We use the camera-motion flow C to estimate the onboard camera's motion, retaining parallax for pose estimation across scene depths. We obtain relative rotation from gyroscope and estimate translation from 3D–2D correspondences constructed using depth and camera-motion flow. Implementation details are provided in Appendix H. We assess pose estimates through a controlled reconstruction experiment. We compare our pose estimates with those from the visual method DPVO (Teed et al., 2023) and the visual-inertial method DynamicVINS (Liu et al., 2022a).



Figure 5: Video stabilization on Deep-FVS. Columns show frames at the same time index from the input video and the gyroscope-only (G-only), SEA-RAFT optical-flow-only (F-only), and our stabilization results (Ours). F-only distorts the person and nearby background, while our method better preserves their geometry.



Figure 6: Qualitative comparison of point-cloud reconstructions using camera poses estimated by DynamicVINS (Liu et al., 2022a), DPVO (Teed et al., 2023), and Ours, with an RGB reference view on the left. All reconstructions use the same per-frame depth maps and reconstruction procedure; only the estimated camera poses differ. Sharper static structures and fewer duplicated contours indicate more consistent alignment across frames.

For each method, we accumulate the same per-frame point clouds using its estimated poses, keeping the depth maps and reconstruction procedure fixed. Better point-cloud alignment indicates more consistent pose estimates. In Figure 6, our method produces sharper static structures, including thin chair legs, and fewer duplicated contours than the baselines. These results suggest that our camera-motion flow provides useful correspondences for camera pose estimation in robotic navigation.

4.7 Ablation Study

Comparison with input motion fields. Table 3 in Appendix E compares G, SEA-RAFT flow F (Wang et al., 2024b), and $G + A + P$. Rotation flow yields 0.53 PME and 0.64 EPE; optical flow yields 1.74 and 0.89. Ours reaches 0.44 PME and 0.43 EPE. Its error is lower than that of G in every condition for both metrics. Thus selected optical-flow evidence improves the rotation-based estimate despite the higher errors of F alone.

Progressive camera-motion reconstruction. Global correction $G + A$ reduces EPE from 0.64 to 0.49 and PME from 0.53 to 0.43. Adding neural parallax gives 0.43 EPE and 0.44 PME, improving dense alignment while maintaining similar background matching accuracy. Using only $G + P$ with the same neural predictor configuration yields 0.51 EPE and 0.54 PME.

Motion compatibility and scene proximity. Removing ρ raises EPE from 0.43 to 0.50 and PME from 0.44 to 0.57. Removing c increases EPE to 0.94 and PME to 1.43. Thus, motion compatibility is important for rejecting unsuitable residuals, while scene proximity improves dense alignment and background matching by limiting the parallax contribution. Appendix E.1 reports the full results using Table 1 as the full-model reference. A separate comparison with robust fitting without learned guidance is provided in Table 5, where RANSAC affine fitting achieves 0.53 EPE and 0.53 PME.

5 Conclusion

We presented a framework for estimating dense camera-motion flow from gyroscope-derived rotation flow and optical flow from a pretrained estimator. Motion compatibility and scene proximity guide global homography correction and neural parallax recovery to capture camera-induced motion while suppressing independent object motion. Our method achieves state-of-the-art accuracy on GHOF and GHOF-Cam, and zero-shot transfer of the camera-motion modules to MPI-Sintel improves estimation over both input flows. In video stabilization, it reduces jitter relative to rotation flow while introducing less distortion than optical flow. In robotic navigation, its estimated camera poses yield more consistent point-cloud reconstructions than the baselines.

AI USE STATEMENT

We used Claude to assist with literature searches, code implementation, data processing, and analysis of experimental results. We used Codex for code review, manuscript drafting and editing, and brainstorming figure designs. We also used ChatGPT for manuscript review, language polishing, and consistency checking. The authors reviewed all AI-assisted outputs, checked references against their sources, inspected and executed code, verified experimental results, and examined figures for accuracy. AI-generated ideas served as starting points for author-led writing and figure creation, with the authors implementing and revising the final presentation. The authors take full responsibility for the content of this paper and its accompanying artifacts.

REFERENCES

Daniel Barath, Jana Noskova, Maksym Ivashechkin, and Jiri Matas. Magsac++, a fast, reliable and accurate robust estimator. In *Proc. CVPR*, pp. 1304–1312, 2020.

Michael Bloesch, Sammy Omari, Péter Fankhauser, Hannes Sommer, Christian Gehring, Jemin Hwangbo, Mark A Hoepflinger, Marco Hutter, and Roland Siegwart. Fusion of optical flow and inertial measurements for robust egomotion estimation. In *Proc. IROS*, pp. 3102–3107, 2014.

Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, and Michael J. Black. A naturalistic open source movie for optical flow evaluation. In *Proc. ECCV*, pp. 611–625. Springer, 2012.

Si-Yuan Cao, Jianxin Hu, Zehua Sheng, and Hui-Liang Shen. Iterative deep homography estimation. In *Proc. CVPR*, pp. 1879–1888, 2022.

Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Deep image homography estimation. *arXiv preprint arXiv:1606.03798*, 2016.

Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proc. CVPRW*, pp. 224–236, 2018.

Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24 (6):381–395, 1981.

Matthias Grundmann, Vivek Kwatra, Daniel Castro, and Irfan Essa. Calibration-free rolling shutter removal. In *Proc. ICCP*, pp. 1–8, 2012.

Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

Paul W Holland and Roy E Welsch. Robust regression using iteratively reweighted least-squares. *Communications in Statistics - Theory and Methods*, 6(9):813–827, 1977.

Mingbo Hong, Yuhang Lu, Nianjin Ye, Chunyu Lin, Qijun Zhao, and Shuaicheng Liu. Unsupervised homography estimation with coplanarity-aware GAN. In *Proc. CVPR*, pp. 17663–17672, 2022.

Zhaoyang Huang, Xiaoyu Shi, Chao Zhang, Qiang Wang, Ka Chun Cheung, Hongwei Qin, Jifeng Dai, and Hongsheng Li. Flowformer: A transformer architecture for optical flow. In *Proc. ECCV*, pp. 668–685. Springer, 2022.

Jerin Geo James, Devansh Jain, and Ajit Rajwade. GlobalFlowNet: Video stabilization using deep distilled global motion estimates. In *Proc. WACV*, pp. 5078–5087, 2023.

Hai Jiang, Haipeng Li, Songchen Han, Haoqiang Fan, Bing Zeng, and Shuaicheng Liu. Supervised homography learning with realistic dataset generation. In *Proc. ICCV*, pp. 9806–9815, 2023.

Rico Jonschkowski, Austin Stone, Jonathan T. Barron, Ariel Gordon, Kurt Konolige, and Anelia Angelova. What matters in unsupervised optical flow. In *Proc. ECCV*, 2020.

Alexandre Karpenko, David Jacobs, Jongmin Baek, and Marc Levoy. Digital video stabilization and rolling shutter correction using gyroscopes. Technical Report CSTR 2011-03, Stanford University, 2011.

Haipeng Li, Kunming Luo, and Shuaicheng Liu. GyroFlow: Gyroscope-guided unsupervised optical flow learning. In *Proc. ICCV*, 2021.

Haipeng Li, Hai Jiang, Ao Luo, Ping Tan, Haoqiang Fan, Bing Zeng, and Shuaicheng Liu. DMHomo: Learning homography with diffusion models. *ACM Transactions on Graphics (TOG)*, 43(3):1–16, 2024a.

Haipeng Li, Kunming Luo, Bing Zeng, and Shuaicheng Liu. GyroFlow+: Gyroscope-guided unsupervised deep homography and optical flow learning. *International Journal of Computer Vision (IJCV)*, 132(6):2331–2349, 2024b.

Haipeng Li, Tianhao Zhou, Zhanglei Yang, Yi Wu, Yan Chen, Zijing Mao, Shen Cheng, Bing Zeng, and Shuaicheng Liu. Estimating 2D camera motion with hybrid motion basis. In *Proc. ICCV*, pp. 7624–7633, 2025.

Jianheng Liu, Xuanfu Li, Yueqian Liu, and Haoyao Chen. RGB-D inertial odometry for a resource-restricted robot in dynamic environments. *IEEE Robotics and Automation Letters (RA-L)*, 7(4): 9573–9580, 2022a.

Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. Bundled camera paths for video stabilization. *ACM Transactions on Graphics (TOG)*, 32(4), 2013.

Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. SteadyFlow: Spatially smooth optical flow for video stabilization. In *Proc. CVPR*, pp. 4209–4216, 2014.

Shuaicheng Liu, Ping Tan, Lu Yuan, Jian Sun, and Bing Zeng. Meshflow: Minimum latency online video stabilization. In *Proc. ECCV*, pp. 800–815, 2016.

Shuaicheng Liu, Haipeng Li, Zhengning Wang, Jue Wang, Shuyuan Zhu, and Bing Zeng. DeepOIS: Gyroscope-guided deep optical image stabilizer compensation. *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, 32(5):2856–2867, 2022b.

Shuaicheng Liu, Yuhang Lu, Hai Jiang, Nianjin Ye, Chuan Wang, and Bing Zeng. Unsupervised global and local homography estimation with motion basis learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 45(6):7885–7899, 2023.

David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004.

Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. IJCAI*, 1981.

Jaesung Rim, Woohyeok Kim, Haeyun Lee, Heemin Yang, Ke Wang, and Sunghyun Cho. Gyro-based deep video deblurring. In *Proc. CVPR*, 2026.

Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. In *Proc. ICCV*, pp. 2564–2571, 2011.

Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proc. CVPR*, pp. 4938–4947, 2020.

Ruizhi Shao, Gaochang Wu, Yuemei Zhou, Ying Fu, Lu Fang, and Yebin Liu. Localtrans: A multi-scale local transformer network for cross-resolution homography estimation. In *Proc. ICCV*, pp. 14890–14899, 2021.

Xi Shen, François Darmon, Alexei A. Efros, and Mathieu Aubry. Ransac-flow: Generic two-stage image alignment. In *Proc. ECCV*, pp. 618–637, 2020.

Jianbo Shi and Carlo Tomasi. Good features to track. In *Proc. CVPR*, 1994.

Zhenmei Shi, Fuhao Shi, Wei-Sheng Lai, Chia-Kai Liang, and Yingyu Liang. Deep online fused video stabilization. In *Proc. WACV*, pp. 865–873. IEEE, 2022.

Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *Proc. CVPR*, 2018.

Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. LoFTR: Detector-free local feature matching with transformers. In *Proc. CVPR*, pp. 8922–8931, 2021.

Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *Proc. ECCV*, pp. 402–419, 2020.

Zachary Teed, Lahav Lipson, and Jia Deng. Deep patch visual odometry. In *Proc. NeurIPS*, volume 36, pp. 39033–39051, 2023.

Yurun Tian, Xin Yu, Bin Fan, Fuchao Wu, Huub Heijnen, and Vassileios Balntas. SOSNet: Second order similarity regularization for local descriptor learning. In *Proc. CVPR*, pp. 11016–11025, 2019.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual geometry grounded transformer. In *Proc. CVPR*, pp. 5294–5306, 2025.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3d vision made easy. In *Proc. CVPR*, 2024a.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Permutation-equivariant visual geometry learning. In *Proc. ICLR*, 2026.

Yihan Wang, Lahav Lipson, and Jia Deng. SEA-RAFT: Simple, efficient, accurate RAFT for optical flow. In *Proc. ECCV*, pp. 36–54, 2024b.

Bo Xu, Zewen Xu, Yijia He, Zhanpeng Ouyang, Hao Wei, Yihong Wu, Jiancheng Li, and Hongdong Li. A rotation-translation decoupled solution for visual-inertial initialization and online spatial-temporal calibration. *IEEE Transactions on Robotics (TRO)*, 42:854–873, 2026.

Heemin Yang, Jaesung Rim, Seungyong Lee, Seung-Hwan Baek, and Sunghyun Cho. Gyro-based neural single image deblurring. In *Proc. CVPR*, pp. 23111–23120, 2025.

Nianjin Ye, Chuan Wang, Haoqiang Fan, and Shuaicheng Liu. Motion basis learning for unsupervised deep homography estimation with subspace projection. In *Proc. ICCV*, pp. 13117–13125, 2021.

Zhichao Yin and Jianping Shi. GeoNet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proc. CVPR*, pp. 1983–1992, 2018.

Julio Zaragoza, Tat-Jun Chin, Michael S Brown, and David Suter. As-projective-as-possible image stitching with moving dlt. In *Proc. CVPR*, pp. 2339–2346, 2013.

Jirong Zhang, Chuan Wang, Shuaicheng Liu, Lanpeng Jia, Nianjin Ye, Jue Wang, Ji Zhou, and Jian Sun. Content-aware unsupervised deep homography estimation. In *Proc. ECCV*, pp. 653–669, 2020.

A GYROSCOPE-DERIVED ROTATION FLOW

A gyroscope measures angular velocity $\boldsymbol{\omega} = (\omega_x, \omega_y, \omega_z)^\top$ in rad/s about the device's three axes. We first express these measurements in camera coordinates, accumulate rotations over time, and project the relative rotation into the image to obtain G.

From angular velocity to rotation. Let $\boldsymbol{\omega}_k$ be the angular velocity at timestamp t_k, expressed in camera coordinates via the calibrated camera–IMU rotation. Multiplying by the sampling interval gives a rotation vector $\Delta\boldsymbol{n}_k = \boldsymbol{\omega}_k\,\Delta t_k$, whose direction is the instantaneous rotation axis and whose magnitude is the rotation angle (in radians). We accumulate the camera orientation as a rotation matrix $\boldsymbol{Q}_k$ ($\boldsymbol{Q}_0 = \boldsymbol{I}$) using Rodrigues' formula:

$$\begin{aligned} &\boldsymbol{Q}_{k+1} = \boldsymbol{Q}_k\,\mathcal{R}(\Delta\boldsymbol{n}_k), \\ &\mathcal{R}(\boldsymbol{n}) = \boldsymbol{I} + \frac{\sin\theta}{\theta}[\boldsymbol{n}]_\times + \frac{1-\cos\theta}{\theta^2}[\boldsymbol{n}]_\times^2, \qquad \theta = \|\boldsymbol{n}\|, \end{aligned} \tag{5}$$

where θ is the rotation angle and $[\boldsymbol{n}]_\times$ is the 3×3 skew-symmetric matrix encoding the cross product $\boldsymbol{n}\times(\cdot)$. Interpolating between gyroscope samples gives the orientation $\boldsymbol{Q}(t)$ at arbitrary timestamps; the relative rotation from frame I_1 to I_2 is then $\boldsymbol{R}_{21} = \boldsymbol{Q}(t_2)^\top\boldsymbol{Q}(t_1)$.

From rotation to image displacement. Under a calibrated pinhole model with intrinsic matrix $\boldsymbol{K}$ (encoding focal length and principal point), pure rotation induces the homography

$$\boldsymbol{H} = \boldsymbol{K}\boldsymbol{R}_{21}\boldsymbol{K}^{-1}. \tag{6}$$

Most smartphone cameras use a rolling shutter, which reads out rows sequentially rather than capturing the entire frame at once. To account for rotation during readout, we divide the image into horizontal bands and compute a rotation homography for each band using its exposure times in the two frames (Karpenko et al., 2011; Li et al., 2021; Liu et al., 2022b). Neighboring band rotations are interpolated via spherical linear interpolation (SLERP) to produce a smooth per-scanline homography $\boldsymbol{H}(y)$. The rotation flow is the resulting displacement field:

$$G(x,y) = \pi\big(\boldsymbol{H}(y)[x,y,1]^\top\big) - [x,y]^\top, \qquad \pi([u,v,w]^\top) = [u/w, v/w]^\top. \tag{7}$$

B GUIDANCE SUPERVISION AND INFERENCE

We construct offline pseudo-labels from depth and motion consistency, then train the guidance network to predict them from F and G. Depth supplies a proximity cue; agreement with an analytical camera-motion estimate supplies a compatibility cue. These targets encode geometric priors rather than ground-truth camera motion. The analytical teacher is used for supervision, while inference uses the learned scores and a weighted homography fit.

Parameter selection. The teacher hyperparameters reported below, including the depth scale factor, were selected using a separate validation set. We report their values and roles to specify the pseudo-label construction.

Depth to scene proximity. We run frozen Pi3 (Wang et al., 2026) on the image pair and extract depth as the coordinate along the camera optical axis in the first view's local point map. Pi3 camera poses and confidence outputs are not used. We map inverse depth to a bounded proximity score, emphasizing nearby structure and suppressing distant structure:

$$\rho_T(\boldsymbol{p}) = \left[\mathrm{clip}\left(\frac{1/\max(D(\boldsymbol{p}), 10^{-3}) - 1/d_{\mathrm{far}}}{1/d_{\mathrm{near}} - 1/d_{\mathrm{far}}}, 0, 1\right)\right]^\gamma. \tag{8}$$

Here D is the depth after bilinear resizing to the flow resolution and multiplication by 0.9410. We use $d_{\mathrm{near}} = 0.5$, $d_{\mathrm{far}} = 3.0$, and $\gamma = 2.5$ in this normalized depth convention, not in meters. The near and far thresholds set the transition range, and γ controls its shape. The score is a depth-based weighting cue, not a metric depth prediction. This use requires a near-versus-far cue mapped through clipping and the exponent γ, whereas the Pi3 rigid-flow baseline also relies on predicted camera poses and geometric reprojection. Its flow error therefore does not directly measure the quality of the proximity supervision.

Camera flow to compatibility. We define compatibility by how well optical flow agrees with an analytical camera-motion estimate. The teacher first estimates a global correction, then pools spatially consistent residuals to obtain a local field. Let $\widetilde{G}$ denote the perturbed training gyro field. We first fit a global homography correction A_T to $F - \widetilde{G}$ using DLT. Only pixels with residual magnitude below 0.8 px contribute, with equal weights; fewer than 200 selected pixels yields zero correction. Define $R_T = F - \widetilde{G} - A_T$.

To obtain a local residual field, we initially retain pixels with $\|R_T\|_2 < 3$ px and compute their normalized Gaussian-weighted average, with bandwidth 60 px and denominator regularizer 0.05. We then retain only those initial candidates whose discrepancy from this estimate is also below 3 px, and repeat the average once. Denote the resulting field by L_T. The residual checks reject large or locally inconsistent candidates, while normalized averaging pools the remaining evidence. The denominator regularizer limits the estimate where support is weak. The depth gate assembles the teacher camera-motion flow C^{teach}, which is used only to construct training labels and is distinct from the final prediction C:

$$C^{\text{teach}} = \widetilde{G} + A_T + \rho_T \odot L_T, \qquad c_T(\boldsymbol{p}) = \exp\left(-\frac{\|F(\boldsymbol{p}) - C^{\text{teach}}(\boldsymbol{p})\|_2}{\tau_c}\right), \tag{9}$$

with $\tau_c = 1$ px. A high score means that the observed flow agrees with the teacher's camera-motion explanation. Object motion, flow errors, and errors in the teacher can all reduce this score. Thus c_T is teacher-relative compatibility, not a ground-truth motion mask. Both supervision maps are fixed during training.

Guidance network. The input $[F - G, G, F]$ (six channels) is replicate-padded to a multiple of 8 and cropped back after decoding. The U-Net has three downsampling levels with 32, 64, and 128 channels and a 256-channel bottleneck. Each level uses two 3×3 convolutions, each followed by group normalization with 8 groups and ReLU; downsampling uses 2×2 max pooling and upsampling uses 2×2 transposed convolutions with skip connections. The last decoder feature map, with 32 channels at full resolution, is the shared feature Z. Two 1×1 convolutions followed by a sigmoid predict c and ρ. They are zero-initialized, so an untrained network outputs $c = \rho = 0.5$. The network has 1.93M parameters and is trained from scratch.

Global correction and inference. The guidance network supplies features Z and maps c, ρ. We use the hard-thresholded weighted DLT fit of Section 3.3. For a residual field r, we retain pixels with $\|r(\boldsymbol{p})\|_2 < \kappa$, where $\kappa = 0.8$ px, and assign weights $w_A(\boldsymbol{p}) = c(\boldsymbol{p})^4(1 - \rho(\boldsymbol{p}))$. The fourth power suppresses residuals with low compatibility. Weighted DLT fits a full projective homography H to the retained correspondences $\boldsymbol{p} \leftrightarrow \boldsymbol{p} + r(\boldsymbol{p})$, yielding the correction $A(\boldsymbol{p}) = \pi(H\widetilde{\boldsymbol{p}}) - \boldsymbol{p}$, where $\widetilde{\boldsymbol{p}}$ denotes homogeneous coordinates. The initial fit uses $r = F - G$; subsequent refits use $r = F - G - P$, with the same threshold and guidance weights.

At inference, we first fit $A^{(0)}$ to $F - G$ without parallax and use the remaining residual as an input to the parallax predictor. This gives $P_0 = \rho \odot U_{\phi_0}$. We then refit A once to $F - G - P_0$ with the same weights, keeping P_0 fixed. The final flow is $C = G + A + P_0$, and no network parameters are updated at test time.

Training settings. Optical flow is computed with the SEA-RAFT-M checkpoint trained on Tartan-C-T-TSKH-Spring (Wang et al., 2024b), at the native 600×800 resolution with 4 refinement iterations, and stored in half precision. Training and evaluation use the same setting. The guidance network trains on 9,568 image pairs at 600×800 resolution for 30 epochs, using AdamW with batch size 8, initial learning rate 3×10^{-4}, weight decay 10^{-4}, and cosine learning-rate decay. The loss is $\|c - c_T\|_1 + \|\rho - \rho_T\|_1$ with equal weights, averaged over pixels. We use no cropping, flipping, or online augmentation and keep the final epoch.

Each training gyro field receives one fixed, offline perturbation with a per-sample seed. We draw a severity $s \sim \mathcal{U}(0.2, 1)$. Both flow components are scaled by $1 + 0.15a$ with $a \sim \mathcal{U}(-s, s)$, and we add offsets $t_x, t_y \sim \mathcal{U}(-1.5s, 1.5s)$ px. We then add low-frequency noise: Gaussian noise sampled on a $1/16$-resolution grid, bilinearly upsampled and multiplied by $0.8s$. The perturbed field is stored in half precision. The same perturbed field is used for teacher-label construction and network input throughout training.

Pi3 rigid-flow baseline. For the Pi3 baseline in Table 1, we estimate focal length from the first view's predicted point map. With Pi3's camera-to-world poses P_0 and P_1, we transform the first-view points into the second camera using $P_1^{-1}P_0$, project them, and subtract the pixel grid to obtain rigid flow. We resize the flow to 600×800 and scale its displacement components accordingly. This baseline uses only Pi3 predictions, without gyroscope measurements or optical flow. It differs from our use of Pi3 for guidance supervision, which uses its depth predictions rather than its predicted poses.

C Parallax Prediction

This section details the offline training of the predictor in Section 3.4. Its supervision consists of analytical pseudo-targets, not ground-truth camera flow. The guidance network and its outputs remain frozen throughout this stage.

Predictor inputs. Let $A^{(0)}$ denote the homography correction fitted to $F - G$ without a parallax term, using the fit described in Appendix B. The predictor's input residual is $r_1^{(0)} = F - G - A^{(0)}$, computed in the same way during training and at the start of inference. The predictor works at $1/8$ resolution (75×100). Let Down denote area downsampling to this resolution and $\bar{r}_1^{(0)} = \mathrm{Down}(r_1^{(0)})$. We bound its magnitude with the pointwise mapping $\tilde{r}_1^{(0)} = \bar{r}_1^{(0)}/(1 + \|\bar{r}_1^{(0)}\|_2/\tau)$, where $\tau = 3\,\mathrm{px}$. This preserves direction and limits the influence of large input residuals; motion selection is handled by the guidance scores. The notation $g_\phi(Z, r_1, \rho, c)$ in Eq. (4) includes this residual preprocessing. Internally, the network receives the concatenation of $\mathrm{Down}(Z)$ (32 channels), $\tilde{r}_1^{(0)}$ (2 channels), $\mathrm{Down}(\rho)$, and $\mathrm{Down}(c)$, for 36 channels in total. The downsampled inputs are rounded to half precision, identically during training and inference.

The network consists of two blocks, each a 3×3 convolution with 32 output channels followed by group normalization with 8 groups and ReLU, and a zero-initialized 1×1 convolution that outputs two channels. It has 19,842 parameters. Its output is an ungated displacement field in full-resolution pixel units. Bilinear upsampling to 600×800 gives U_ϕ, yielding $P = \rho \odot U_\phi$. For the training loss below, let z denote the combined predictor inputs at its working resolution and $f_\phi(z)$ its prediction at that resolution.

Analytical pseudo-targets. The parallax pseudo-target is constructed by alternating global fitting with local, compatibility-weighted regression. The local update fits the residual through the proximity gate: its numerator pools $c\rho R$, while its denominator accumulates $c\rho^2$ and a regularizer. This accounts for the gate in the fitted displacement and limits estimates where local support is weak. With $w_A = c^4(1 - \rho)$ and $U_T^{(0)} = 0$, two rounds compute

$$H_T^{(j)} = \mathrm{Hom}\Big(F - G - \rho \odot U_T^{(j-1)}, w_A\Big), \tag{10}$$

$$R_T^{(j)} = F - G - A_{H_T^{(j)}}, \qquad U_T^{(j)} = \frac{\mathcal{G}_\sigma(c\rho R_T^{(j)})}{\mathcal{G}_\sigma(c\rho^2) + \lambda + \epsilon}, \quad j = 1, 2. \tag{11}$$

Here Hom denotes the same hard-thresholded weighted DLT fit as at inference ($\kappa = 0.8\,\mathrm{px}$), and $A_H(\boldsymbol{p}) = \pi(H\widetilde{\boldsymbol{p}}) - \boldsymbol{p}$ is its displacement field. $\mathcal{G}_\sigma$ denotes Gaussian smoothing. We use $\sigma = 45\,\mathrm{px}$, $\lambda = 0.025$, and $\epsilon = 10^{-6}$, with a unit-sum kernel truncated at four standard deviations and replicate padding. On training pairs, G is the perturbed field $\widetilde{G}$, and c, ρ are the frozen guidance network's predictions on $(F, \widetilde{G})$. All solver computations run at full resolution in double precision. The ungated field $U_{\mathrm{ref}} = U_T^{(2)}$ supplies the regression target. Thus, depth provides the proximity target, teacher-flow agreement provides the compatibility target, and the analytical solver provides the parallax target. The predictor learns to approximate this last stage from motion features; it is not supervised by ground-truth camera-motion flow.

Let $\bar{U}_{\mathrm{ref}} = \mathrm{Down}(U_{\mathrm{ref}})$ and $\bar{w} = \mathrm{Down}(c\rho)$. We train the predictor with the sample-normalized loss

$$\mathcal{L}_{\mathrm{init}} = \frac{1}{B}\sum_{i=1}^{B} \frac{\sum_{\mathbf{q}} \bar{w}_i(\mathbf{q}) \|f_\phi(z_i)(\mathbf{q}) - \bar{U}_{\mathrm{ref},i}(\mathbf{q})\|_1}{2\sum_{\mathbf{q}} \bar{w}_i(\mathbf{q}) + \epsilon}. \tag{12}$$

All predictor parameters are trained for 20 epochs using AdamW with learning rate 10^{-3}, weight decay 10^{-4}, $\beta = (0.9, 0.999)$, batch size 64, and cosine learning-rate decay to zero, without data augmentation. The pseudo-targets and inputs are precomputed once per training pair. Training takes about one minute on a single RTX 4090D. The last checkpoint provides the pretrained parameters ϕ_0 used at inference.

D Cost of the Teacher and the Student

Our training labels come from a teacher that requires a large 3D foundation model and several analytical stages, whereas the deployed student obtains c, ρ, and Z from one compact network. We compare the two on the same GHOF-Cam pairs. Both routes share the optical flow F and the rotation flow G.

Teacher. Pi3 (Wang et al., 2026) runs on the RGB image pair and provides depth, from which we compute ρ_T (Eq. (8)). The teacher camera-motion flow C^{teach} then yields c_T (Eq. (9)). Given c_T and ρ_T, the analytical solver of Eq. (10) computes $C = G + A + \rho_T \odot U_T^{(2)}$. It uses two global fits and two Gaussian kernel passes.

Student. The guidance network maps (F, G) to c, ρ, and Z in one forward pass. The student then performs one global fit, one forward pass of the parallax predictor, and one refit (Appendix B). It needs neither the images, nor Pi3, nor any Gaussian kernel regression.

Protocol. All timings use one RTX 4090D and report the median over 32 evenly spaced benchmark pairs after 5 warm-up pairs, with GPU synchronization. Pi3 runs as when the teacher labels were generated: bfloat16 autocast, an input resolution of 434×574, and the PyTorch implementation of RoPE. Our implementation runs the analytical stages in double precision. Consumer GPUs have low double-precision throughput, so we also report all analytical stages in single precision. For the solver, single precision leaves EPE and PME unchanged to four decimals. FLOPs count convolutions (including the Gaussian filters), matrix multiplications, and attention; pointwise operations and the small least-squares solves are excluded. Shared inputs are excluded from the totals. SEA-RAFT-M has 19.7M parameters and takes 891 GFLOPs and 33.2 ms per pair.

Table 2: Cost per image pair of the teacher and the student on GHOF-Cam (RTX 4090D). The optical flow F and rotation flow G are shared and excluded. Times are for single-precision analytical stages; values in parentheses use the double-precision stages of our implementation. Memory is the peak activation memory added by each stage.

Route	Stage	Params	GFLOPs	Time (ms)	Memory (MiB)
Teacher	Pi3 depth $\rightarrow \rho_T$	958.7M	6117.8	102.4	1921
	Teacher flow $\rightarrow c_T$	–	5.6	11.7 (205.2)	451
	Analytical solver $\rightarrow C$	–	4.3	11.8 (166.0)	81
	Total	**958.7M**	**6127.6**	**125.9 (473.6)**	–
Student	Guidance network $\rightarrow c, \rho, Z$	1.93M	135.9	9.5	401
	Global fit, predictor, refit $\rightarrow C$	0.02M	0.4	5.2 (16.0)	79
	Total	**1.95M**	**136.3**	**14.7 (25.4)**	–

Results. Table 2 shows that the student uses about $490\times$ fewer parameters and $45\times$ fewer FLOPs than the teacher, and runs $8.6\times$ faster ($18.6\times$ with the double-precision stages). Pi3 dominates the teacher's cost; it also needs the RGB images and 3.6 GiB of weights, whereas the student's weights occupy 7 MiB. Including the shared optical flow, a pair takes 159 ms with the teacher and 48 ms with the student. The accuracy remains close. On all 254 GHOF-Cam pairs, the teacher reaches an EPE of 0.415 and a PME of 0.443, whereas the feed-forward student reaches 0.43 EPE and 0.44 PME, as reported in Table 1.

E Additional Ablation Results

The full comparison with the input motion fields is provided in Table 3.

Table 3: Comparison with the input motion fields on GHOF and GHOF-Cam. PME is evaluated on background correspondences and EPE on valid static pixels. Both metrics are in pixels (↓).

	GHOF: sparse PME ↓						GHOF-Cam: dense EPE ↓					
Method	AVG	RE	FOG	LL	RAIN	SNOW	AVG	RE	FOG	LL	RAIN	SNOW
Rotation flow G	0.53	0.33	0.26	1.06	0.44	0.57	0.64	0.88	0.28	0.95	0.53	0.58
SEA-RAFT (F) (Wang et al., 2024b)	1.74	2.56	0.73	3.52	0.61	1.29	0.89	0.24	0.55	2.59	0.39	0.68
Ours ($G + A + P$)	0.44	0.24	0.25	0.80	0.33	0.56	0.43	0.38	0.20	0.66	0.32	0.56

E.1 Separate Ablation of the Guidance Scores

We evaluate three variants: retaining only motion compatibility c, retaining only scene proximity ρ, and using uniform guidance with both scores removed. Each removed score is eliminated from the reconstruction stages rather than replaced by its mean. In particular, removing ρ also removes its gate on the parallax prediction, so that the residual is no longer attenuated by scene proximity. We evaluate the neural feed-forward estimate, the analytical solver, and global correction alone ($G+A$).

Table 4 reports neural ablation variants averaged over two random seeds and deterministic results for the analytical solver and $G + A$. We use the full model from Table 1, with EPE 0.43 and PME 0.44, as the neural reference for all relative comparisons below.

Scene proximity. Removing ρ raises neural EPE from 0.43 to 0.50 and PME from 0.44 to 0.57, increases of approximately 16% and 30%, respectively. With only c, the analytical solver yields 0.42 EPE and 0.55 PME; its full-guidance result is not reported in this table. For $G + A$, removing ρ reduces EPE from 0.49 to 0.47 but raises PME from 0.43 to 0.46. Thus, proximity guidance improves both dense alignment and background matching for the neural predictor, while its effect on EPE depends on the reconstruction method. The PME degradation is consistent with the proximity gate limiting residual motion in distant regions.

Motion compatibility. Removing c raises neural EPE to 0.94 and PME to 1.43; removing both scores gives 0.93 EPE and 1.63 PME. The analytical solver and global-only estimate also degrade substantially without compatibility guidance. These results support using c to limit the influence of residuals unsuitable for camera-motion estimation. Uniform guidance has the highest combined error J for each of the three outputs, although it does not have the highest error for every individual metric. For global correction, both ρ-only and uniform guidance perform worse than leaving G uncorrected (0.64 EPE and 0.53 PME), showing that fitting unsuitable residuals can degrade the rotation estimate. These variants isolate the role of the scores within our reconstruction procedure. Table 5 separately compares against robust fitting without learned guidance; RANSAC with an affine model yields 0.53 EPE and 0.53 PME.

E.2 Robust Global Fitting

We compare learned guidance with RANSAC (Fischler & Bolles, 1981) and MAGSAC-based robust fitting (Barath et al., 2020). We sample approximately 30,000 correspondences on a grid with a stride of four pixels at the 800×600 resolution. Each correspondence maps $\boldsymbol{p}$ to $\boldsymbol{p} + (F - G)(\boldsymbol{p})$. We fit an affine transformation or a homography using OpenCV's `estimateAffine2D` or `findHomography`, respectively, with thresholds of 0.5, 1, and 2 px, a maximum of 5,000 iterations, and confidence 0.999. These baselines use neither learned guidance weights nor depth and output $G + A$. We therefore compare them with our global-correction estimate $G + A$, without parallax recovery.

Table 5 reports the fitting configurations shown in this comparison. Our learned guidance yields lower EPE and PME than all four robust-fitting baselines. The difference is particularly clear in

Table 4: Separate ablation of motion compatibility and scene proximity. Neural ablation variants are means over two seeds; the full neural reference is taken from Table 1. Analytical and global-only results are deterministic. $J = (\mathrm{EPE} + \mathrm{PME})/2$. All values are in pixels (↓). Values are rounded for display. The full analytical solver generates the parallax pseudo-targets; its aggregate results are not reported here and are marked with dashes. The G reference is repeated across columns for comparison.

	Neural $G + A + P$			Analytical solver			Global $G + A$		
Guidance	EPE	PME	J	EPE	PME	J	EPE	PME	J
G (no correction)	0.64	0.53	0.59	0.64	0.53	0.59	0.64	0.53	0.59
c and ρ	0.43	0.44	0.44	–	–	–	0.49	0.43	0.46
c only	0.50	0.57	0.54	0.42	0.55	0.48	0.47	0.46	0.47
ρ only	0.94	1.43	1.19	0.94	1.44	1.19	0.91	1.04	0.98
Uniform	0.93	1.63	1.28	1.02	1.70	1.36	1.18	1.43	1.30

Table 5: Global correction using learned guidance or robust geometric fitting. All methods output $G + A$. EPE and PME are evaluated on GHOF-Cam and GHOF, respectively; LL PME is measured on the low-light subset. $J = (\mathrm{EPE} + \mathrm{PME})/2$. All errors are in pixels (↓).

Global fitting	J	EPE	PME	LL PME
Ours ($G + A$, learned guidance)	**0.46**	**0.49**	**0.43**	**0.80**
RANSAC affine, 1 px	0.53	0.53	0.53	1.04
MAGSAC affine, 0.5 px	0.58	0.54	0.62	1.18
RANSAC homography, 1 px	0.62	0.52	0.72	1.62
MAGSAC homography, 0.5 px	0.71	0.56	0.86	2.20

low light: our PME is 0.80, compared with 1.04 for RANSAC with an affine model and 1.62 for RANSAC with a homography. These results support using motion compatibility and scene proximity to select evidence for global camera-motion correction beyond geometric robust fitting alone.

F Camera-Motion Flow Evaluation on MPI-Sintel

Data and preprocessing. We evaluate on the MPI-Sintel training split (Butler et al., 2012), comprising 23 scenes, 1,064 frames, and 1,041 consecutive frame pairs. We use the final rendering pass, which includes motion blur and atmospheric effects. The original images have width 1024 and height 436. To match the model's input dimensions, we take a central crop of width 581 and height 436 (columns 221–801) and resize it to width 800 and height 600. Camera-motion ground truth, rotation flow, reference optical flow, depth, and evaluation masks are expressed on this target grid. Pixels marked invalid by Sintel are excluded. All reported errors are in pixels at the evaluation resolution.

Camera-motion ground truth. The target C^* represents the image displacement caused solely by camera motion. We back-project each first-frame pixel using the scene depth and hold the resulting 3D point fixed while moving the camera to its second-frame pose. If $X_1(\boldsymbol{p})$ is the point in first-camera coordinates and (R_{21}, t_{21}) maps those coordinates to the second camera, then

$$C^*(\boldsymbol{p}) = \pi(K_2\,[R_{21}X_1(\boldsymbol{p}) + t_{21}]) - \boldsymbol{p}. \tag{13}$$

Here K_2 is the intrinsic matrix on the target grid and π performs perspective division. For static scene points, this target coincides with optical flow. For points on moving objects, it retains the camera-induced displacement while excluding the object's own motion. As a coordinate-consistency check, the median Euclidean discrepancy between Sintel's reference optical flow and the constructed C^* over valid pixels is 0.019 px. Since these fields differ on moving objects, this median is a consistency check rather than a measure of agreement over the entire scene.

Motion inputs and model. Rotation flow G uses only ground-truth relative camera rotation. With world-to-camera rotations R_1 and R_2, we use $R_{21} = R_2 R_1^\top$ and compute

$$G(\boldsymbol{p}) = \pi\big(K_2 R_{21} K_1^{-1} \widetilde{\boldsymbol{p}}\big) - \boldsymbol{p}, \tag{14}$$

where $\widetilde{\boldsymbol{p}}$ denotes homogeneous pixel coordinates and the intrinsics account for cropping and resizing. No translation information enters G. This provides an ideal rotation input rather than a measurement from a physical gyroscope.

We use the GHOF-trained guidance network and parallax predictor with the same feed-forward pipeline as on GHOF-Cam (Appendix B). No camera-motion module is trained, fine-tuned, or tuned on Sintel, and no test-time parameter adaptation is performed. Optical flow F is computed by pretrained SEA-RAFT-M (Wang et al., 2024b) with four iterations on the resized image pairs. The zero-shot setting applies only to our GHOF-trained camera-motion modules. SEA-RAFT remains a pretrained input estimator whose training includes Sintel (Appendix B).

Metric and evaluation regions. Following Section 4.4, we evaluate all three outputs against C^* using EPE, the mean Euclidean flow error over the evaluated pixels. Let V_i denote the valid pixels in pair i and $e_i(\boldsymbol{p}) = \|\widehat{C}_i(\boldsymbol{p}) - C_i^*(\boldsymbol{p})\|_2$. For a set of evaluation pixels S_i, we compute

$$\mathrm{EPE}(S) = \frac{\sum_{i=1}^{N} \sum_{\boldsymbol{p} \in S_i} e_i(\boldsymbol{p})}{\sum_{i=1}^{N} |S_i|}, \tag{15}$$

with $N = 1{,}041$. We report this EPE over all valid pixels ($S_i = V_i$), including those on moving objects, and over the dynamic subset. To assess camera-motion recovery on independently moving regions, we use

$$S_i = V_i^{\mathrm{dyn}} = \{\boldsymbol{p} \in V_i : \|F_{\mathrm{GT},i}(\boldsymbol{p}) - C_i^*(\boldsymbol{p})\|_2 > 1\,\mathrm{px}\}, \tag{16}$$

where F_{GT} is Sintel's reference optical flow. Dynamic pixels account for 29.9% of valid pixels and occur in 907 pairs across 21 scenes. Pairs without dynamic pixels contribute no pixels to dynamic EPE. The same EPE definition, 800×600 grid, and invalid-pixel mask are used for both evaluation regions.

Table 6: Camera-motion flow estimation on MPI-Sintel final with ideal rotation inputs. EPE is measured against C^* over all valid pixels and over dynamic pixels, in pixels (↓). All valid pixels span 1,041 pairs; dynamic pixels occur in 907 pairs across 21 scenes. Ours uses feed-forward inference.

	EPE ↓	
Method	All valid pixels	Dynamic pixels
Optical flow F (SEA-RAFT)	12.20	35.65
Rotation flow G	5.93	10.24
Ours ($G + A + P$)	**4.76**	**9.61**

Results. Table 6 shows that our method reduces overall EPE by 61.0% relative to F and 19.7% relative to G. On dynamic regions, the reductions are 73.0% and 6.2%, respectively, with lower error than F in each of the 21 scenes containing dynamic pixels. The improvement over optical flow on moving objects supports recovery of camera-induced motion while suppressing independent object motion. The smaller gain over rotation flow shows that recovering the remaining camera-induced displacement on these regions is still challenging. For example, in `temple_2`, dynamic EPE remains approximately 157 px for G and 155 px for Ours. These results support transferring the GHOF-trained guidance network and parallax predictor to synthetic scenes with ideal camera rotations.

G Video Stabilization Details

This section details the video stabilization experiment in Section 4.6.1. Beyond the single sequence in Figure 5, we evaluate all 34 Deep-FVS test videos, plot the output camera paths, show spatio-

temporal scanline slices, and compare the methods frame by frame on six representative sequences, one per page.

Data. Deep-FVS (Shi et al., 2022) contains 50 hand-held smartphone videos at 1920×1080 with gyroscope measurements, 16 for training and 34 for testing, recorded at 30 to 120 fps. We use the dataset calibration for all videos: a focal length of $w/1.27$ for image width w and a rolling-shutter readout time of 15.98 ms. Video frames are matched to the sensor timestamps in order. We discard frame pairs whose rolling-shutter rows fall outside the gyroscope coverage; we never extrapolate the sensor logs.

Stabilization pipeline. We implement a SteadyFlow-style backend (Liu et al., 2014) in which only the motion input changes. Each input is converted to a dense forward displacement field between adjacent frames and sampled on a 200×150 mesh over the frame. At each mesh vertex, the camera path is the cumulative sum of these displacements. We smooth every vertex path with the same temporal Gaussian filter (standard deviation of 5 frames, truncated at ± 30 frames, one pass). The correction is the difference between the smoothed and the original path, and each frame is rendered by warping the mesh triangles to their corrected positions. We apply no crop and leave uncovered regions black, so that content loss and warping artifacts remain visible. The mesh, smoothing, and rendering are identical for all methods.

Motion inputs. G-only uses the gyroscope-derived rotation flow G, computed with per-row-band homographies to account for the rolling shutter. F-only uses SEA-RAFT optical flow (Wang et al., 2024b) on frames resized to 800×600, with four iterations. Our decomposition provides motion components that downstream applications can combine. Video stabilization uses $G + A$ to smooth global camera motion while preserving scene geometry. Smoothing the depth-dependent term P independently at each mesh vertex can introduce local distortion near depth discontinuities; we therefore reserve the full $G + A + P$ flow for applications such as pose estimation that need dense correspondences across depths. For reference, we also report the teacher's $G + A$, whose global correction is fitted with Pi3 depth.

Training settings. We initialize the guidance network with our pretrained weights and fine-tune it on the 16 Deep-FVS training videos. The labels follow Appendix B: ρ_T comes from Pi3 depth, and c_T is computed with the same two-round local field. The teacher's global correction A_T is a full homography fitted to the residual using the hard-thresholded DLT procedure of Appendix B, with depth weights $1 - \rho_T$. We train for 10 epochs using L1 losses on c and ρ, AdamW with learning rate 3×10^{-4} and weight decay 10^{-4}, batch size 8, and cosine decay. We do not apply synthetic gyroscope perturbations because the rotation flow already contains real sensor errors. At test time, the global correction is fitted with the same rules, but with the weights $c^4(1 - \rho)$ of Section 3 computed from the predicted maps, so that residuals incompatible with camera motion are excluded as well.

Evaluation. We measure stability on the output videos, independently of the motion inputs. For each stabilized video, we track Shi–Tomasi corners (Shi & Tomasi, 1994) with pyramidal Lucas–Kanade (Lucas & Kanade, 1981) at 960×540 inside the rendered region, keep the tracks that pass a forward–backward check, and fit a similarity transform between adjacent frames with RANSAC. Accumulating these transforms gives the output camera path. The jitter is the root mean square of this path after subtracting its Gaussian low-pass (standard deviation 0.2 s), in pixels at 1920×1080. We measure distortion directly on the mesh warps. The non-rigid residual is the root mean square distance between the warped mesh and its best-fitting homography, averaged over frames. The local stretch is the ratio s_1/s_2 of the singular values of the warp's Jacobian on each mesh cell; it equals 1 for warps that locally preserve shape, and we report its 99th percentile per frame, averaged over frames. Finally, we count the frames in which at least one mesh triangle folds over.

Results. Table 7 summarizes the test set. G-only reduces the jitter of the input only slightly: G describes the camera rotation alone, so the image motion that it misses stays in the stabilized video. F-only removes most of the jitter, but its mesh warps deviate from a homography by 5.11 px on average and fold over in a third of the frames. Because optical flow also contains independent object motion and depth-dependent parallax, the smoothed F-only paths differ between neighboring mesh

Table 7: Video stabilization on the 34 Deep-FVS test videos, averaged over videos. The jitter is measured on the output videos; the distortion metrics are measured on the mesh warps.

	Stability	Distortion		
Motion input	Jitter (px) ↓	Non-rigid (px) ↓	Stretch ↓	Foldover (%) ↓
Input	14.4	–	–	–
G-only	12.2	0.39	1.06	0.0
F-only	7.2	5.11	3.12	33.6
Teacher $G + A$	7.4	0.40	1.06	0.0
Ours	7.2	0.40	1.06	0.0

vertices. The teacher's $G + A$ removes nearly as much jitter as F-only while its warps stay as rigid as those of G-only. Ours matches the jitter of F-only, 7.2 px, with the rigidity of G-only. Its jitter is lower than that of G-only on all 34 videos and lower than that of the input on 33; the exception is a night street scene whose input barely shakes (1.07 px, against 1.08 px for ours). Its non-rigid residual is lower than that of F-only on all 34 videos, and none of its warps folds over. In the same night scene, the teacher's jitter is 5.1 px, whereas ours is 1.08 px.

Camera paths. Figures 7a and 7b plot the output camera paths of four test videos over the 8-second window with the strongest input shake. G-only follows much of the input jitter, especially vertically. The F-only and Ours paths are smooth and nearly coincide; their difference lies in distortion, not in stability.

Scanline slices. Figures 8–10 show spatio-temporal slices. We select vertical and horizontal scanlines on a reference frame, at different depths, and stack the pixels along each scanline over five seconds of the stabilized video. High-frequency shake appears as saw-tooth edges, intentional camera motion as smooth waves, and warping artifacts as bent or broken structures. The input and G-only slices are serrated; the F-only and Ours slices are smooth.

Per-case comparisons. Figures 11–16 compare the stabilized frames of six test videos at three time instants, together with the local stretch of the F-only and Ours warps on the input frame. The cases cover a large moving vehicle, a dancing person, a person running up a staircase, a hand reaching for a car door close to the camera, strong parallax filmed from a moving car, and a truck crossing a night street. In each case, the F-only warp concentrates on the moving object or the near structure, where it stretches, tears, or folds the content, and leaves black holes. Our warp is nearly uniform over the frame.

H Robotic Navigation Details

This section details the robotic navigation experiment in Section 4.6.2. We describe the recording setup, how camera poses are obtained from camera-motion flow, and how the reconstructions are formed. We then extend Figure 6 with the estimated trajectories, an additional baseline, and further sequences.

Setup. We mount an Orbbec Gemini 336L RGB-D camera, which contains a built-in IMU, on a Unitree Go2 quadruped robot and record synchronized color, depth, and IMU streams with their sensor timestamps. The gyroscope measurements are transformed into the color camera frame using the fixed camera–IMU extrinsics provided with the camera.

Camera pose estimation. For each pair of consecutive frames, we obtain the relative rotation $\boldsymbol{R}_{21}$ by integrating the gyroscope measurements as in Appendix A and evaluating the accumulated orientation at the two color-frame timestamps; no rolling-shutter correction is applied here. We then estimate only the translation $\boldsymbol{t}$. We sample pixels $\boldsymbol{p}$ of the first frame that have valid depth and reliable camera-motion correspondences, and back-project them using the depth map D_1, giving 3D points $\boldsymbol{x} = D_1(\boldsymbol{p})\,\boldsymbol{K}^{-1}\tilde{\boldsymbol{p}}$, where $\tilde{\boldsymbol{p}}$ denotes homogeneous coordinates. The camera-motion flow

gives the corresponding pixel $\boldsymbol{q} = \boldsymbol{p} + C(\boldsymbol{p})$ in the second frame. Let (u, v) be the normalized image coordinates of $\boldsymbol{q}$, and let $\boldsymbol{y} = \boldsymbol{R}_{21}\boldsymbol{x}$. Requiring $\boldsymbol{x}$ to project to $\boldsymbol{q}$ under $(\boldsymbol{R}_{21}, \boldsymbol{t})$ yields two equations that are linear in $\boldsymbol{t}$:

$$t_x - u\, t_z = u\, y_z - y_x, \qquad t_y - v\, t_z = v\, y_z - y_y. \tag{17}$$

We stack these equations over the sampled pixels and solve for $\boldsymbol{t}$ by iteratively reweighted least squares with the Huber loss (Holland & Welsch, 1977). Because C excludes independent object motion, Eq. (17) remains valid for pixels on moving objects: $\boldsymbol{q}$ is where the point would appear if it were static, and the equation involves only first-frame depth. Two geometric checks further improve robustness. A local depth consistency test discards unreliable 3D points, particularly at depth boundaries. After the initial solve, we transform the 3D points into the second frame with the estimated $(\boldsymbol{R}_{21}, \boldsymbol{t})$, compare their depths with the second-frame depth map at $\boldsymbol{q}$, suppress inconsistent points, and re-estimate $\boldsymbol{t}$. These checks reduce the influence of depth discontinuities, correspondence errors, and residual dynamic content. Since the sensor depth is metric, $\boldsymbol{t}$ is recovered at metric scale without separate scale estimation. Finally, we compose the relative poses into a camera trajectory. The pipeline is purely frame-to-frame: it uses no keyframes, sliding-window optimization, or loop closure.

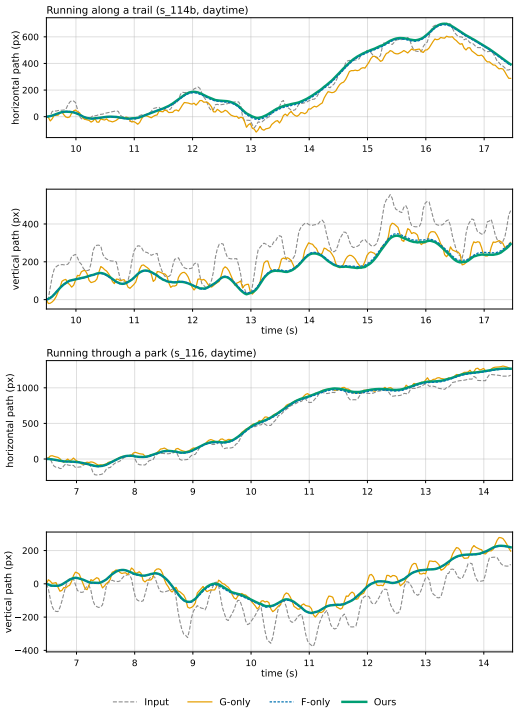
Reconstruction. Using each method's poses, we transform every per-frame point cloud into the first camera's coordinate system and accumulate them into a local reconstruction. All reconstructions use the same per-frame depth maps and reconstruction procedure; only the estimated camera poses differ. Pose errors between frames place the same surface at different positions, which appears as blurred structures and duplicated contours.

Baselines. Section 4.6.2 compares with the monocular visual odometry method DPVO (Teed et al., 2023) and the visual–inertial method DynamicVINS (Liu et al., 2022a). Here Pi3X denotes the Pi3 (Wang et al., 2026) model refined on the robot recordings, with the same architecture and parameter count as the depth teacher. It predicts camera poses jointly with scene geometry from images. For all baselines, we use only the estimated poses and discard any predicted geometry.

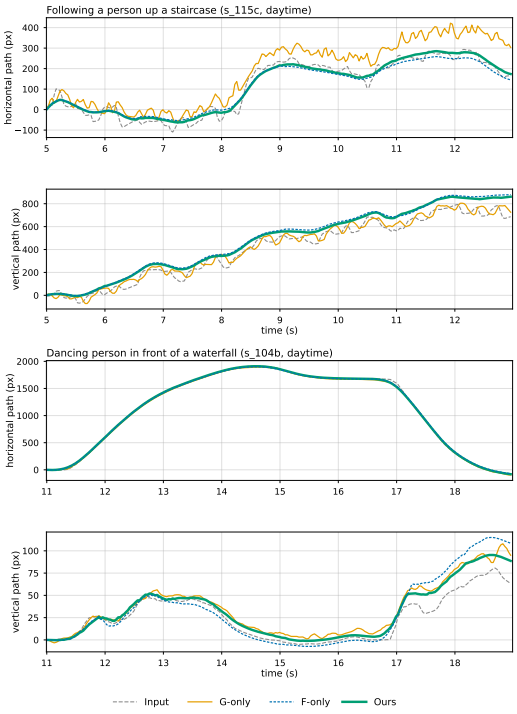
Results. Figure 17 shows the trajectories and reconstructions. The reconstruction with DPVO poses shows noticeable scale-related misalignment; DPVO is a monocular method whose translation is determined only up to scale. DynamicVINS produces a relatively stable trajectory, but its reconstruction still shows accumulated drift. Our reconstruction shows sharper static structures than these two baselines, including thin chair legs, and fewer duplicated contours. Pi3X yields a reconstruction of quality comparable to ours.

Comparison with Pi3X. The Pi3 model used for Pi3X has 958.7M parameters; refinement on the robot data does not change this count. Our flow pipeline, including SEA-RAFT, has 21.7M parameters, about $44\times$ fewer. For inference cost, the controlled GHOF-Cam pair measurements in Appendix D give 102 ms for a Pi3 forward pass at 434×574 and 48 ms for our flow pipeline at 600×800, using single-precision analytical stages. These timings measure pairwise inference, excluding robot-data refinement and our subsequent pose solve, which adds gyroscope integration and robust least squares.

Additional sequences. Figures 18a–19b compare with DynamicVINS and DPVO on another view and three further sequences. In Figure 18a, our reconstruction again resolves the thin chair legs, and the cardboard box shows fewer duplicated contours. In Figures 18b and 19a, which include several people near the robot, the text and logo printed on the ground remain legible in our reconstruction but are blurred or duplicated with the baselines. In Figure 19b, the desks, monitors, and seated people in the background remain distinct in our reconstruction but are blurred with the baselines. Since the depth maps and reconstruction procedure are shared, these differences reflect how consistently each method estimates poses across frames.



(a) Two running sequences.



(b) Two sequences with a moving person.

Figure 7: Output camera paths estimated from the stabilized videos by feature tracking. Each sequence shows horizontal (top) and vertical (bottom) paths over the 8-second window with the strongest input shake. The F-only path is dashed and drawn over Ours.

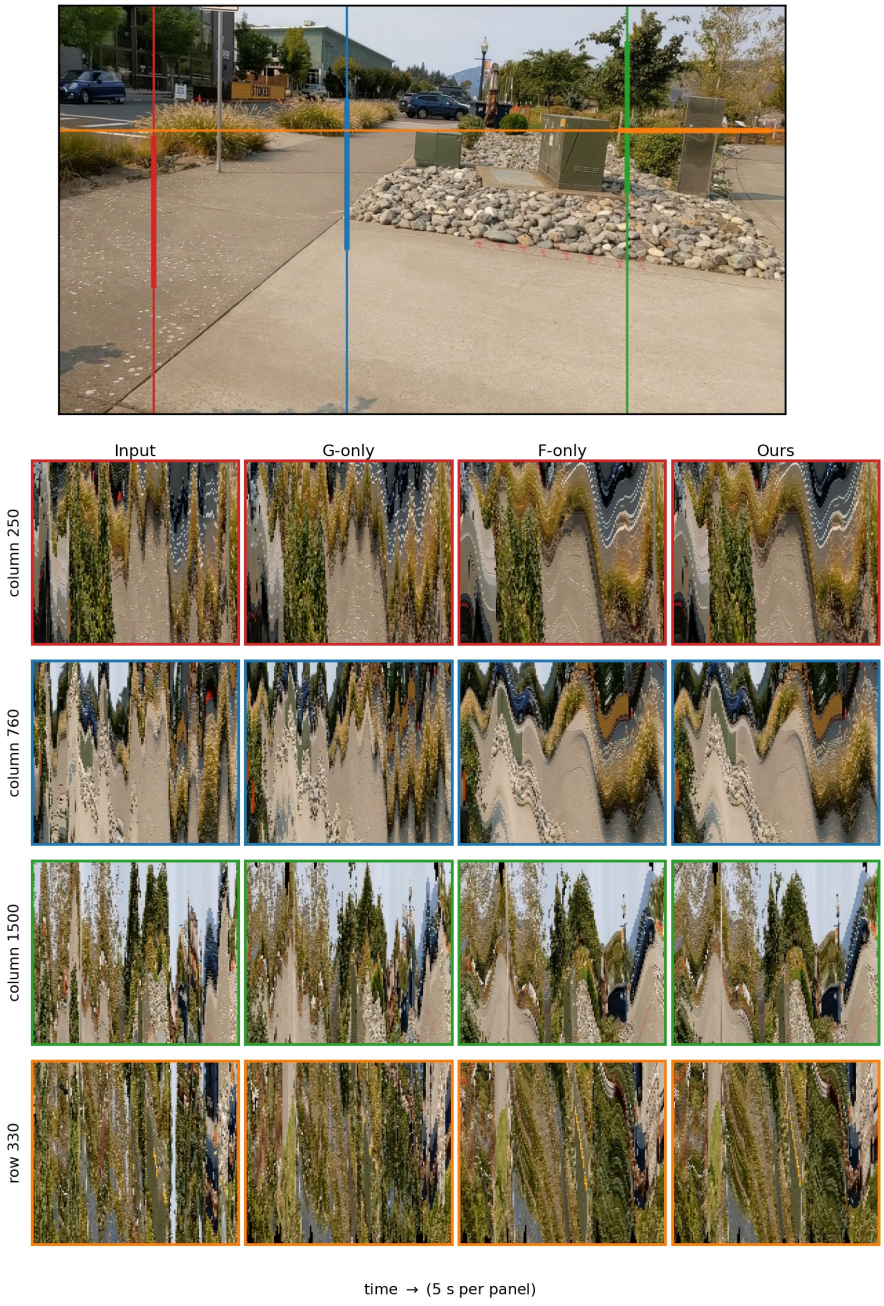


Figure 8: Running sequence (s_114b, daytime). The reference frame at the top marks the selected scanlines. Below, spatio-temporal slices compare the input, G-only, F-only, and Ours over five seconds, with time running left to right. Saw-tooth edges in the input and G-only slices indicate residual shake; F-only and Ours produce smoother traces.

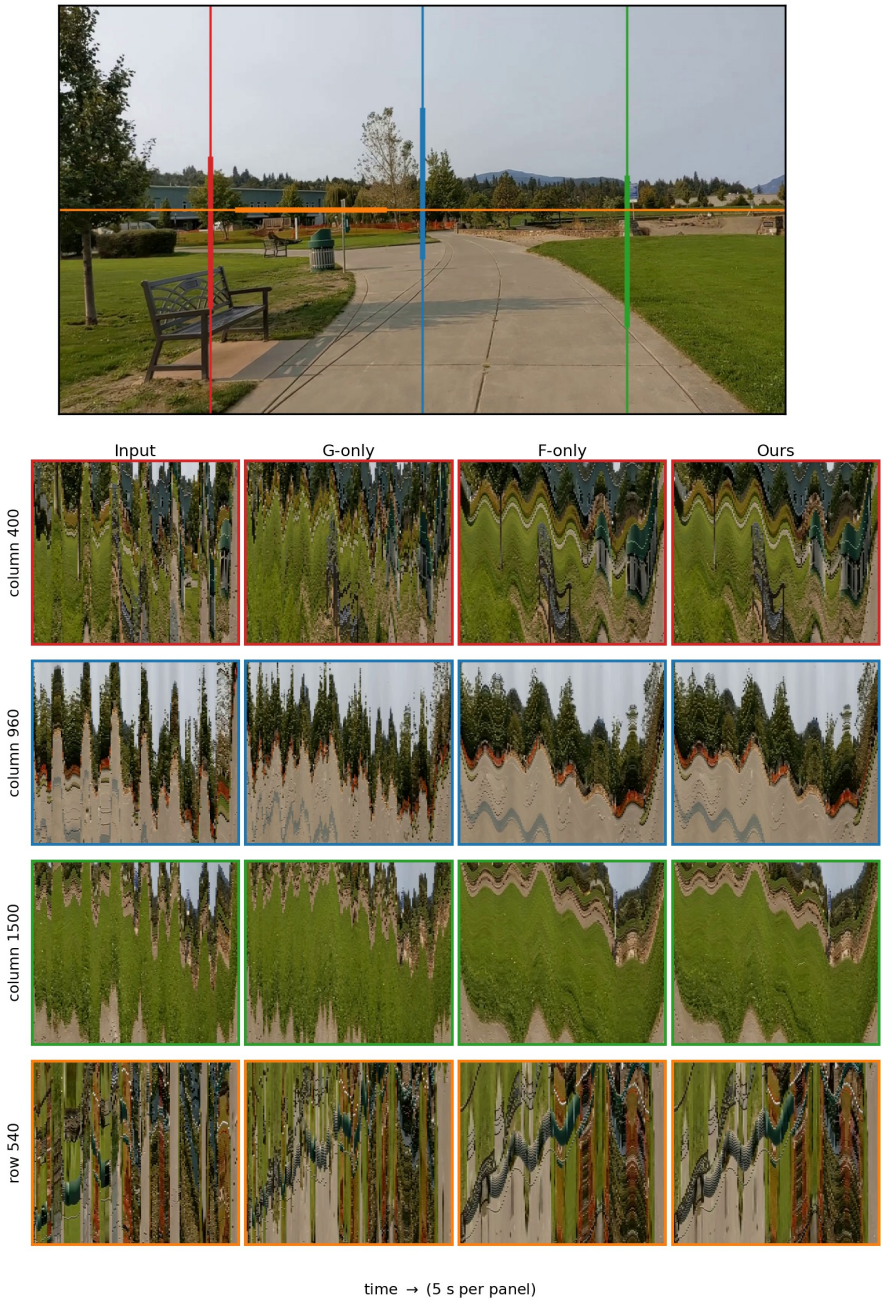


Figure 9: Park sequence (s_116, daytime). The reference frame at the top marks the selected scanlines. Below, spatio-temporal slices compare the input, G-only, F-only, and Ours over five seconds, with time running left to right. Saw-tooth edges in the input and G-only slices indicate residual shake; F-only and Ours produce smoother traces.

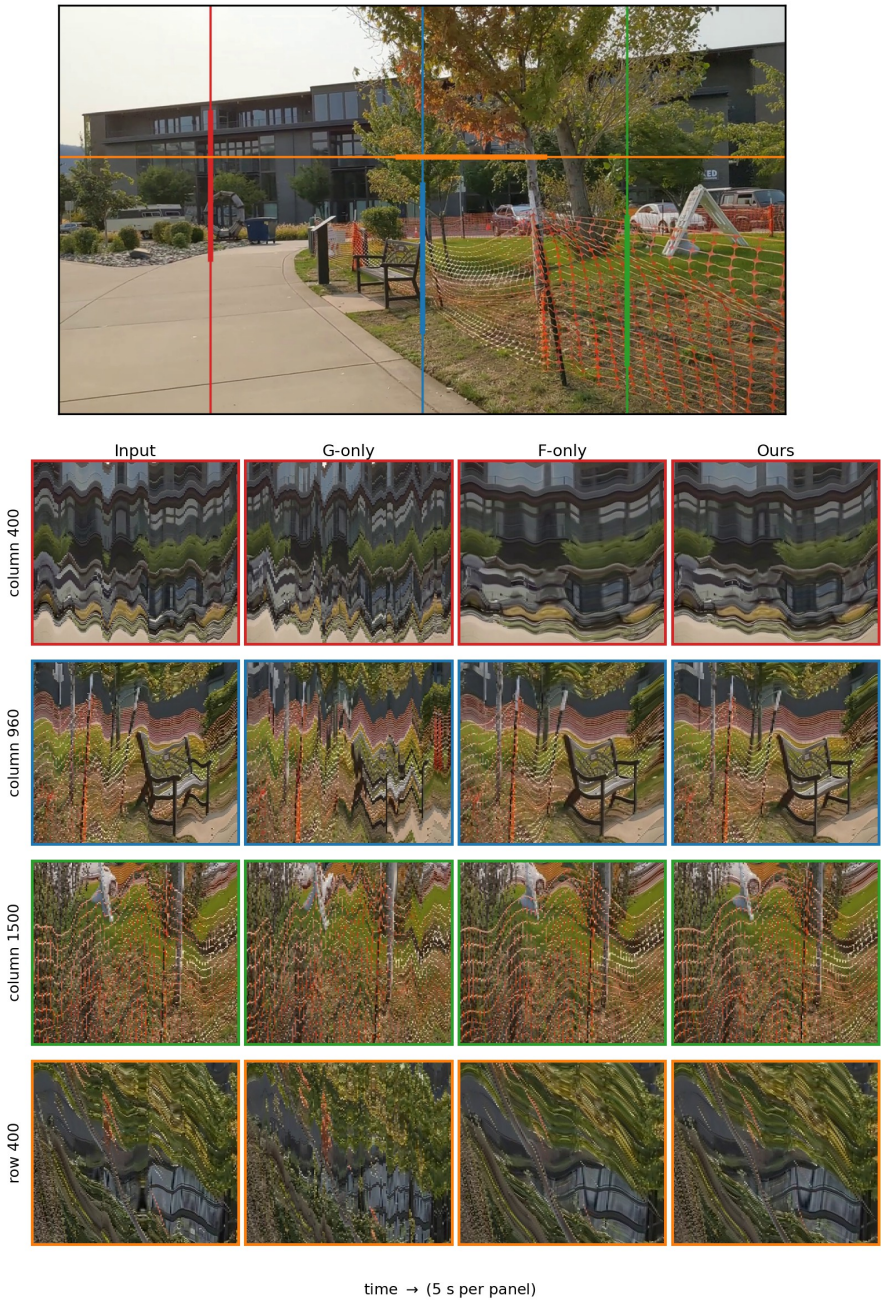


Figure 10: Walking sequence by a building (s_106, daytime). The reference frame at the top marks scanlines across the building and nearby ground. Below, spatio-temporal slices compare the input, G-only, F-only, and Ours over five seconds, with time running left to right. Building floor lines are serrated in the input and G-only slices and smoother in the F-only and Ours slices.



Figure 11: A large moving vehicle (s_103c, daytime). A van approaches and passes the camera. Rows: input, the three stabilized results, and the local stretch s_1/s_2 of the F-only and Ours warps on the input frame. F-only tears the side of the van and leaves black holes, because the smoothed path follows the van; our warp stays uniform.

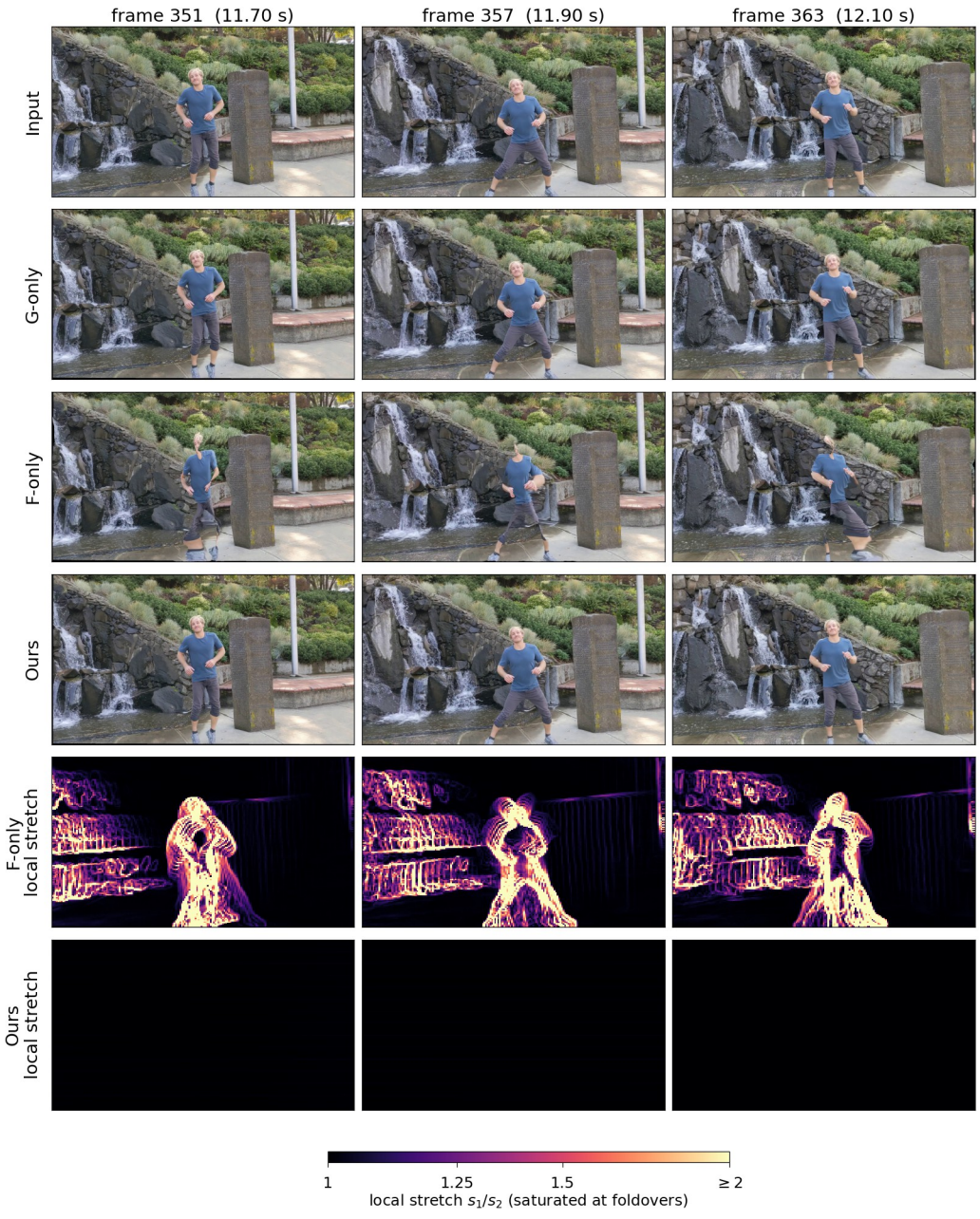


Figure 12: A dancing person in front of a waterfall (s_104b, daytime), displayed as in Figure 11. F-only stretches the head, arms, and legs of the person; the waterfall also produces local stretch.

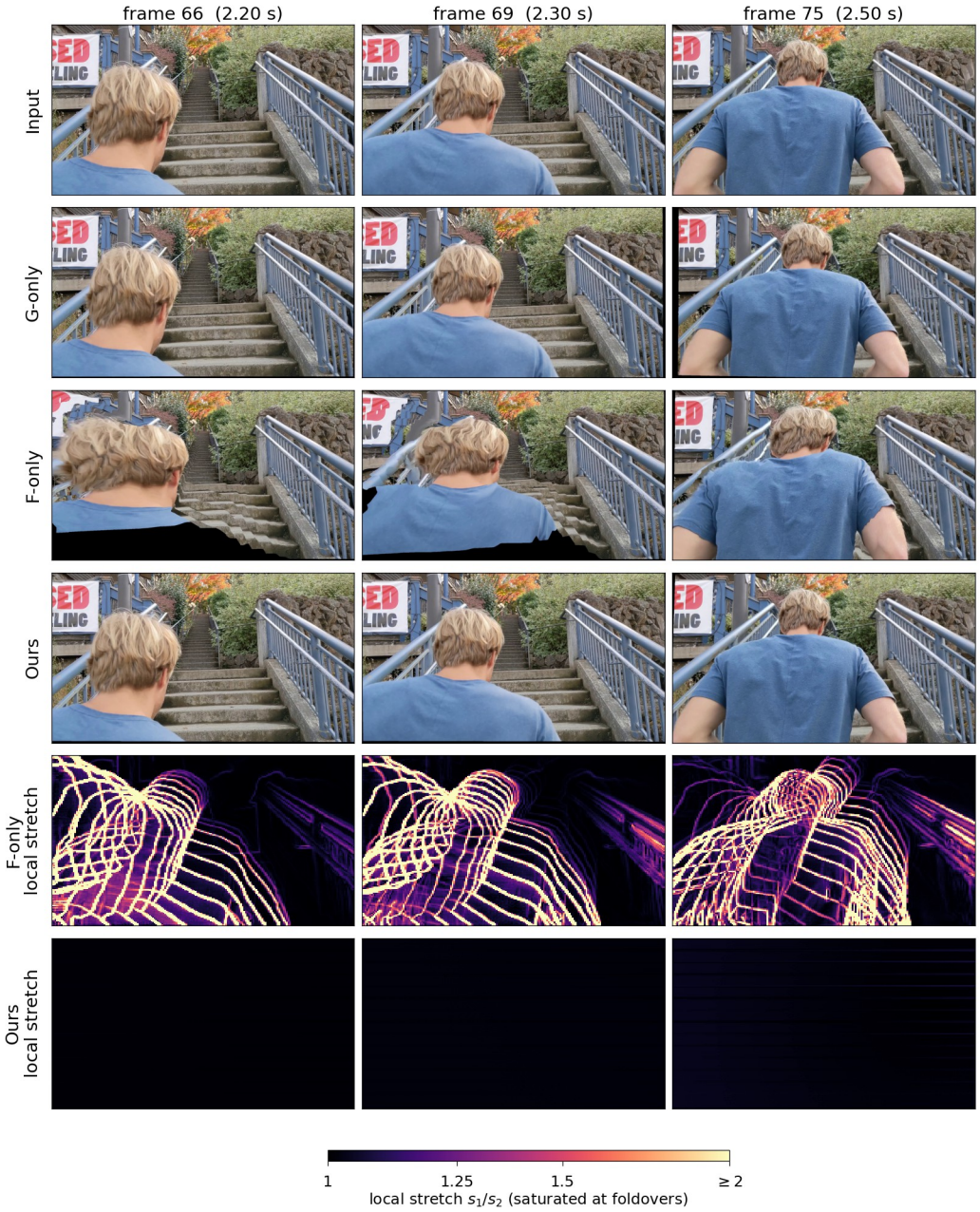


Figure 13: Following a person up a staircase (s_115c, daytime), displayed as in Figure 11. The person fills much of the frame; F-only bends the steps and the railing around the person and leaves large black regions.

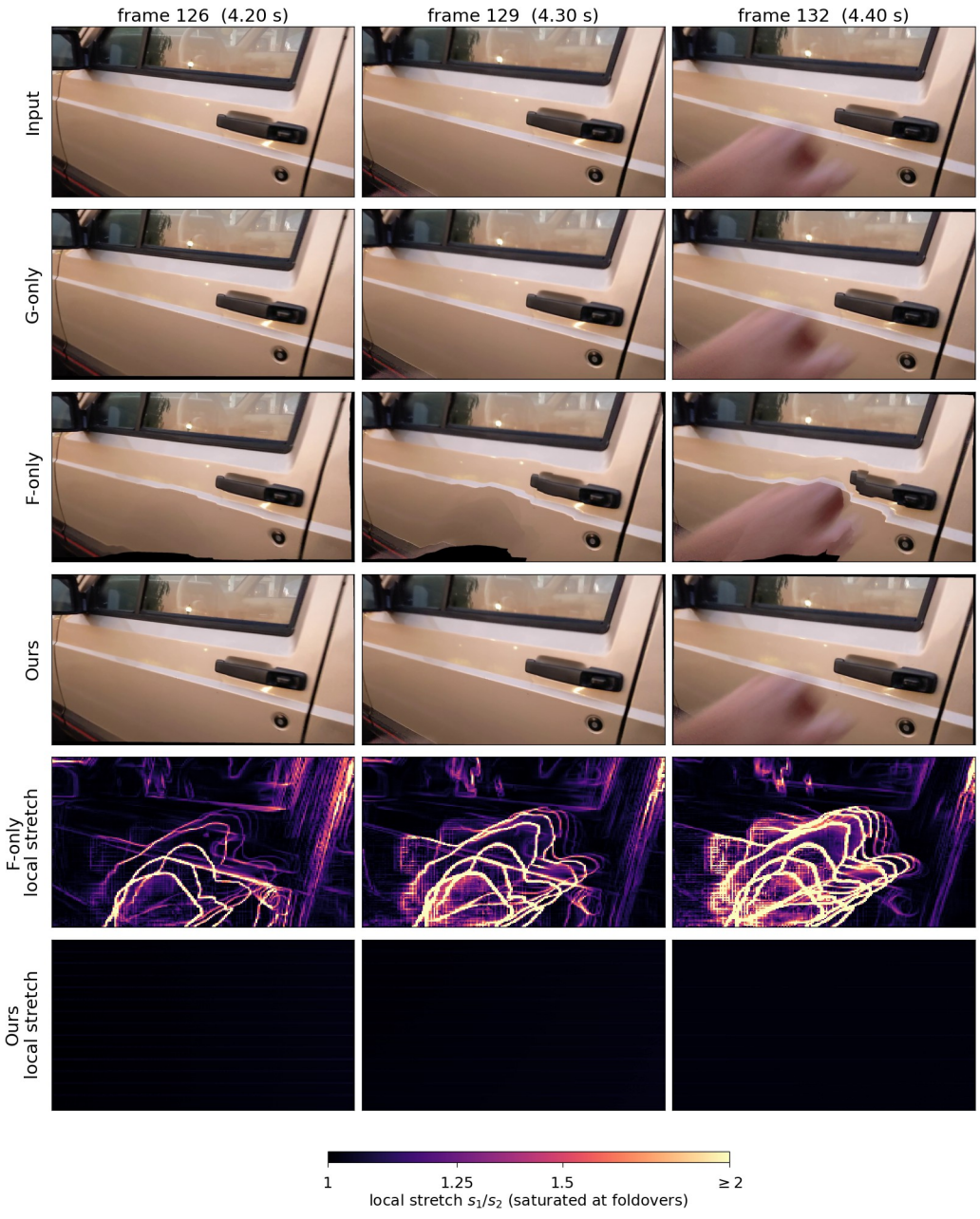


Figure 14: A hand reaching for a car door close to the camera (s_110, night), displayed as in Figure 11. F-only tears the door panel along the moving hand.

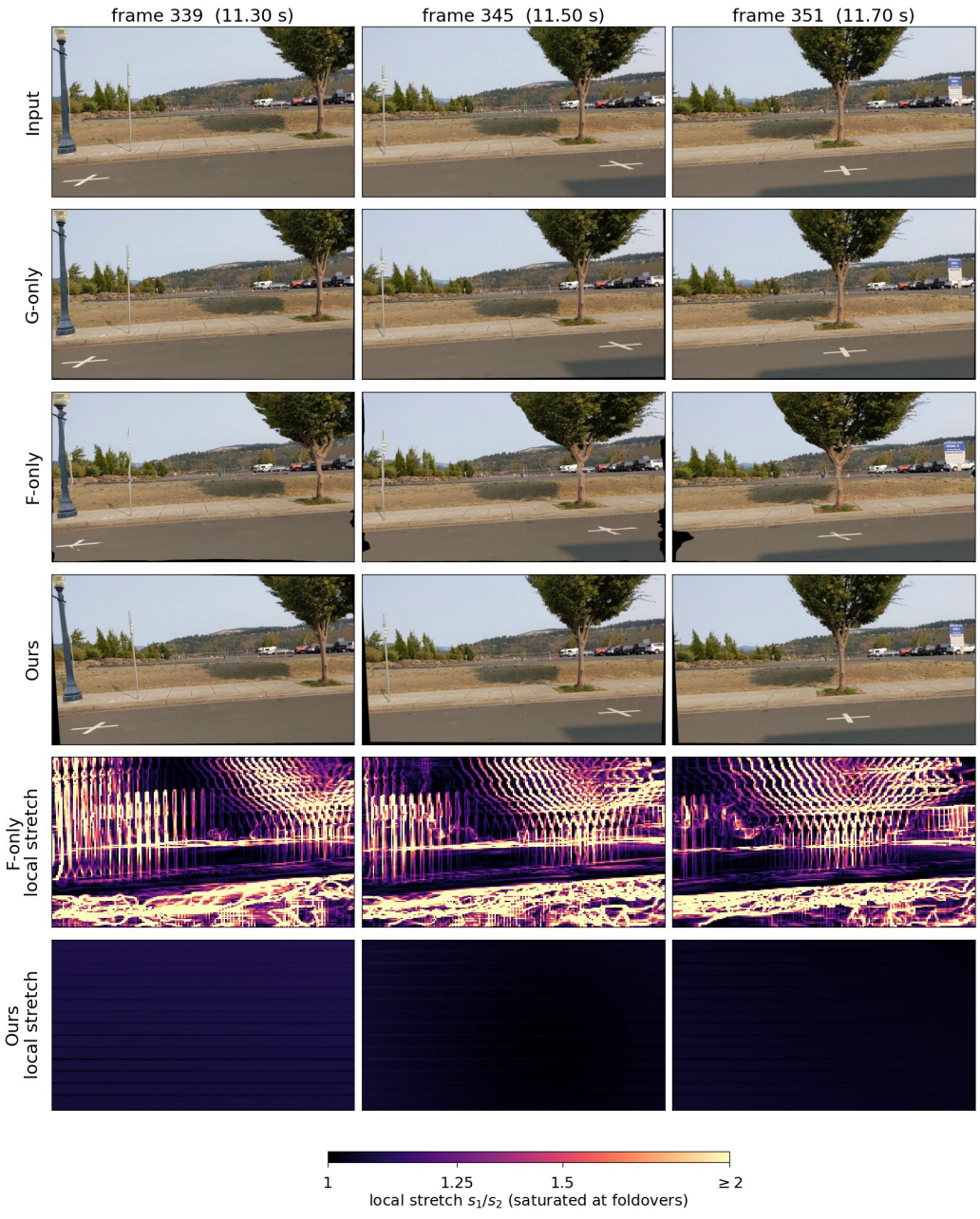


Figure 15: Strong parallax filmed from a moving car (s_118, daytime), displayed as in Figure 11. Near poles and tree trunks move much faster than the background; F-only bends them.

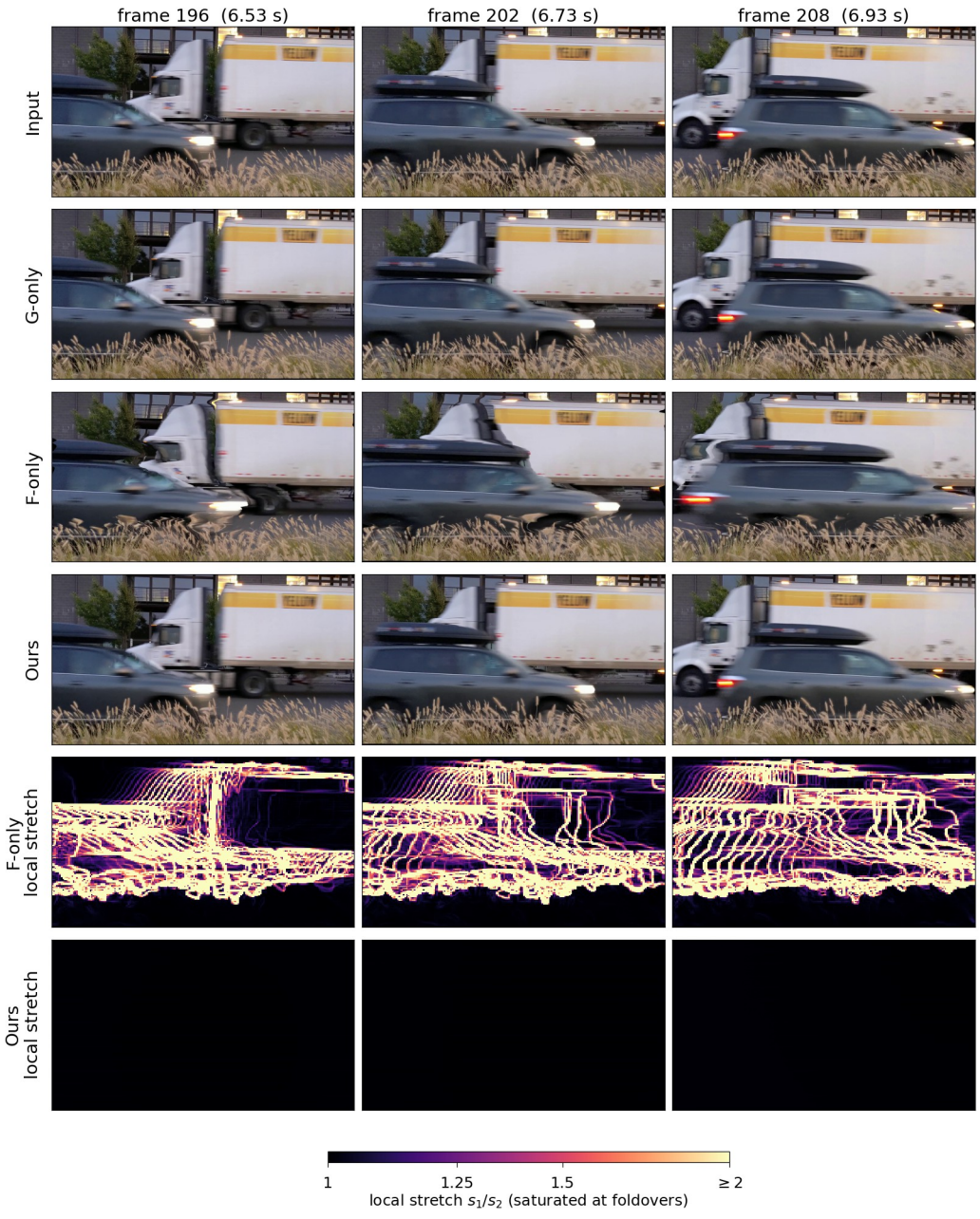


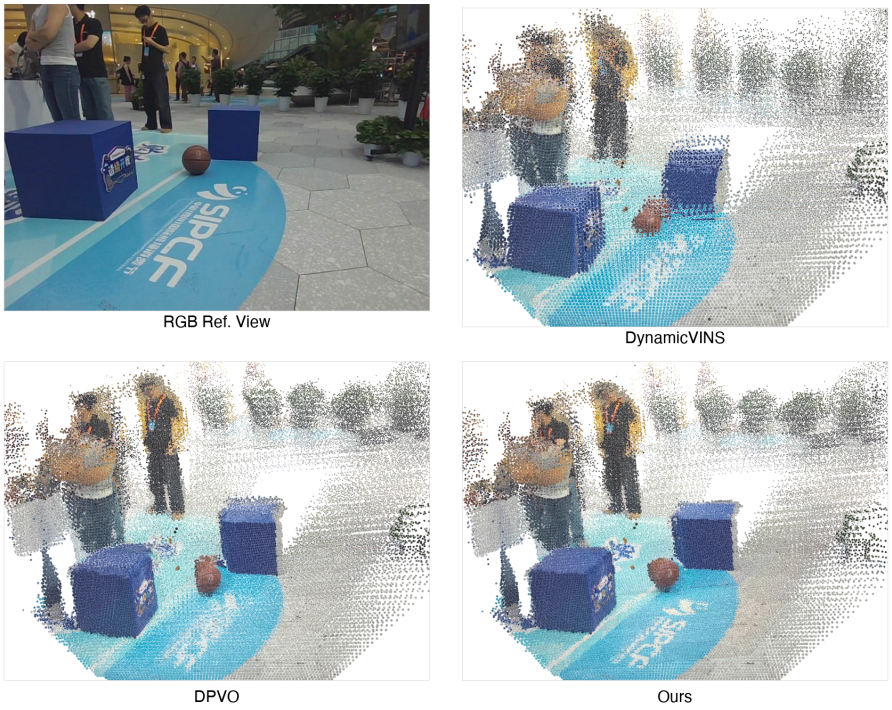
Figure 16: A truck crossing a night street (s_103c, night), displayed as in Figure 11. F-only deforms the truck and the car in front of it, while Ours better preserves their geometry.



Figure 17: Extended comparison of camera poses for robotic navigation. Top row: the RGB reference view, and the trajectories estimated by DPVO (orange), Pi3X (blue), DynamicVINS (red), and Ours (green), overlaid on the point cloud reconstructed by Ours. Middle and bottom rows: the point clouds accumulated with the poses of Ours, DPVO, Pi3X, and DynamicVINS. For visualization, each trajectory is aligned to Ours by an $\mathrm{SE}(3)$ transformation. All reconstructions use the same per-frame depth maps and reconstruction procedure; only the estimated camera poses differ. Sharper static structures and fewer duplicated contours indicate more consistent alignment across frames.

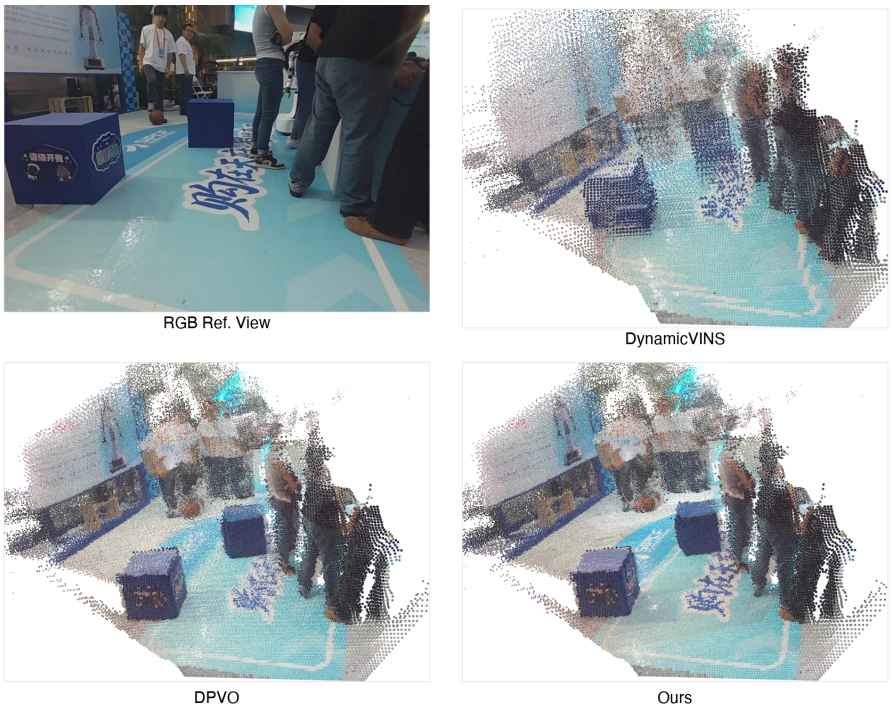


(a) Another view of the indoor sequence in Figure 6. From left to right and top to bottom: RGB reference, DynamicVINS, DPVO, and Ours.

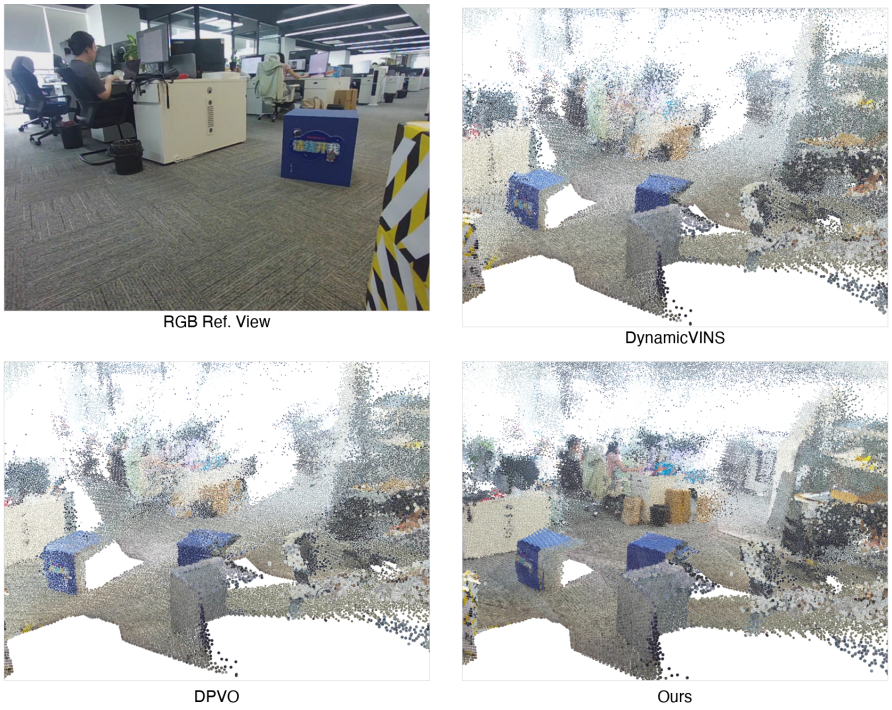


(b) Sequence with people near the robot. Our reconstruction preserves sharper text and markings on the ground.

Figure 18: Additional point-cloud reconstructions for robotic navigation. Each case compares the RGB reference with point clouds accumulated using poses from DynamicVINS, DPVO, and Ours. All reconstructions use the same per-frame depth maps and reconstruction procedure; only the estimated camera poses differ.



(a) Further sequence with people near the robot. Ground text and logos remain more distinct in our reconstruction.



(b) Additional indoor sequence. Desks and monitors in the background are sharper in our reconstruction.

Figure 19: Additional point-cloud reconstructions for robotic navigation. Each case compares the RGB reference with point clouds accumulated using poses from DynamicVINS, DPVO, and Ours. All reconstructions use the same per-frame depth maps and reconstruction procedure; only the estimated camera poses differ.