SVO: Fast Semi-Direct Monocular Visual Odometry

Christian Forster, Matia Pizzoli, Davide Scaramuzza

ICRA 2014

Abstract

- We propose a semi-direct monocular visual odometry algorithm that is precise, robust, and faster than current state-of-the-art methods.

- The algorithm is applied to micro-aerial-vehicle state estimation in GPS-denied environments and runs at **55 frames per second** on the onboard embedded computer and at **more than 300 frames per second** on a consumer laptop.

- We call our approach SVO (Semi-direct Visual Odometry) and release our implementation as open-source software.

https://github.com/uzh-rpg/rpg_svo

Content

1.Introduction

- To our knowledge, all monocular Visual Odometry (VO) systems for MAVs are **feature-based**.
- In RGB-D and stereo-based SLAM systems however, **direct methods**—based on photometric error minimization—are becoming increasingly popular.

- In this work, we propose a **semi-direct** VO that **combines the success-factors of feature-based methods (**tracking many features, parallel tracking and mapping, keyframe selection**) with the accurracy and speed of direct methods.**

1.Introduction

A. Taxonomy of Visual Motion Estimation Methods

a) Feature-Based Methods:

The standard approach is to:

1. extract a sparse set of salient image features (e.g. points, lines) in each image;

2.match them in successive frames using invariant feature descriptors;

3.robustly recover both camera motion and structure using epipolar geometry;

4.finally, refine the pose and structure through reprojection error minimization.

1.Introduction

a) Feature-Based Methods:

The disadvantage of feature-based approaches is the reliance on detection and matching thresholds, the neccessity for robust estimation techniques to deal with wrong correspondences, and the fact that most feature detectors are optimized for speed rather than precision**.**

1.Introduction

b) Direct Methods:

Direct methods estimate structure and motion directly from intensity values in the image.

The local intensity gradient magnitude and direction is used in the optimisation compared to feature-based methods that consider only the distance to some feature-location.

1.Introduction

b) Direct Methods:

Outperform feature-based methods in terms of robustness in scenes with little texture or in the case of camera-defocus and motion blur.

The computation of the photometric error is more intensive than the reprojection error, However, since direct methods operate directly on the intensitiy values of the image, the time for feature detection and invariant descriptor computation can be saved.

1.Introduction

- **B. Related Work**

Most monocular VO algorithms for MAVs rely on PTAM. PTAM is a feature-based SLAM algorithm that achieves robustness through tracking and mapping many (hundreds) of features.

1.Introduction

Early direct monocular SLAM methods tracked and mapped few—sometimes manually selected—planar patches.

With DTAM, a novel direct method was introduced that computes a dense depthmap for each keyframe through minimisation of a global, spatially-regularised energy functional. This approach is computationally very intensive and only possible through heavy GPU parallelization.

1.Introduction

- **C. Contributions and Outline**

The proposed Semi-Direct Visual Odometry (SVO) algorithm uses feature-correspondence; however, feature-correspondence is an implicit result of direct motion estimation rather than of explicit feature extraction and matching.

Feature extraction is only required when a keyframe is selected to initialize new 3D points.

The advantage is **increased speed** due to the lack of feature-extraction at every frame and **increased accuracy** through subpixel feature correspondence.

1.Introduction

- The contributions of this paper are:

(1) a novel semi-direct VO pipeline that is faster and more accurate than the current state-of-the-art for MAVs

(2) the integration of a probabilistic mapping method that is robust to outlier measurements.

2.System Overview

Figure 1 provides an overview of SVO. The algorithm uses **two parallel threads** (as in PTAM) :

a) one for estimating the camera motion,

b) a second one for mapping as the environment is being explored.

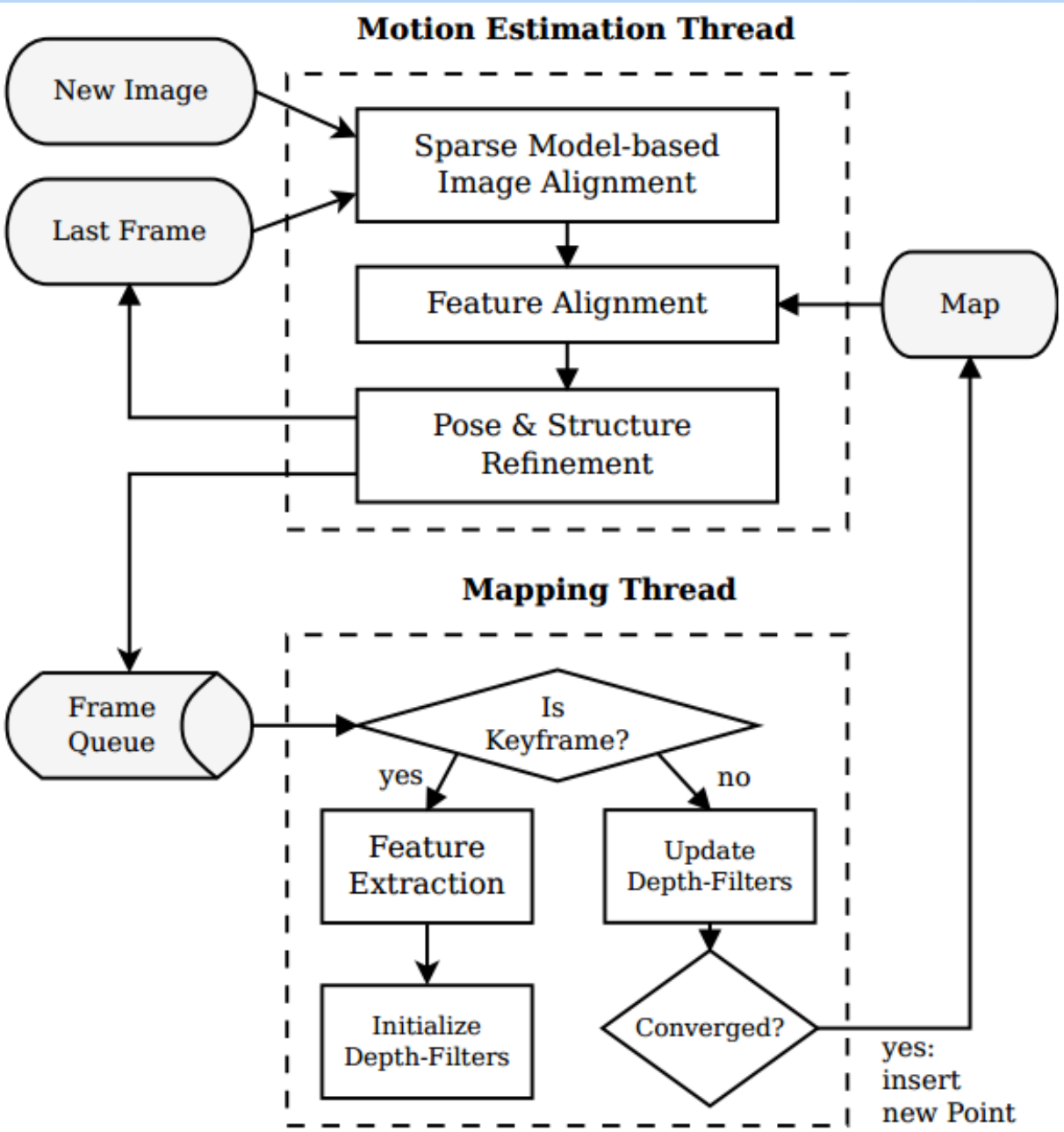


Fig. 1: Tracking and mapping pipeline

2.System Overview

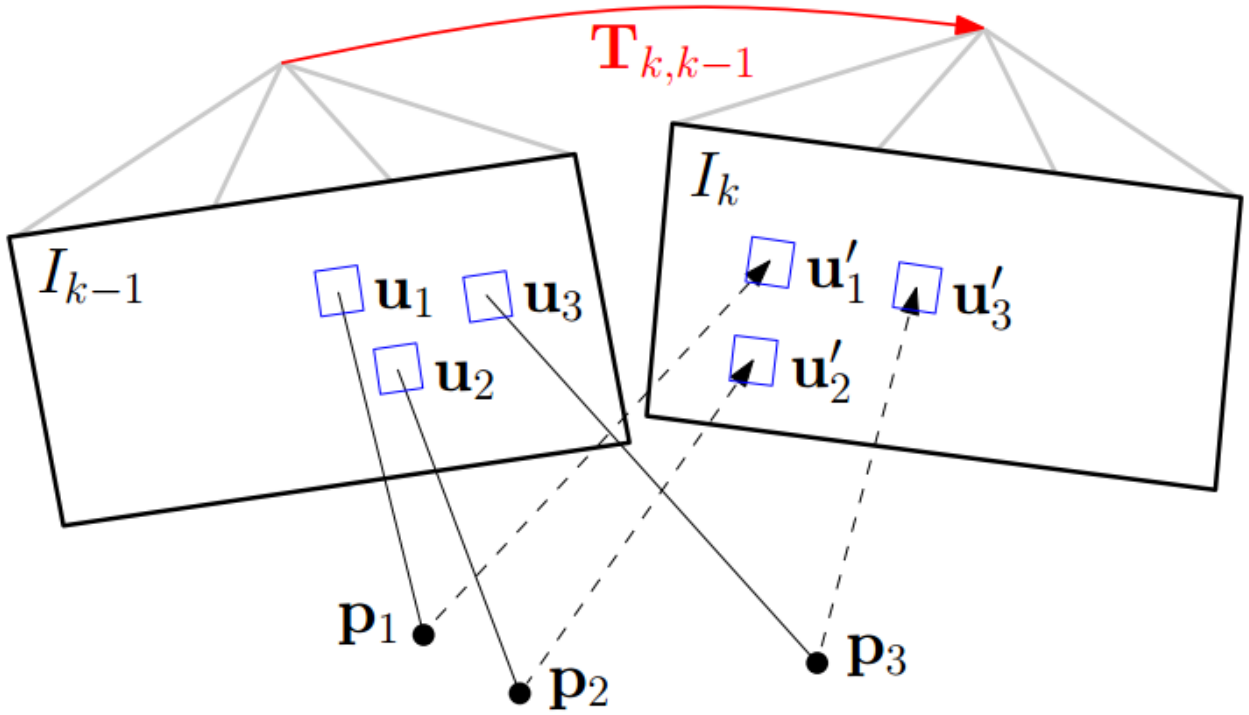


Fig. 2: Changing the relative pose $\mathbf{T}_{k,k-1}$ between the current and the previous frame implicitly moves the position of the reprojected points in the new image $\mathbf{u}'_i$. Sparse image alignment seeks to find $\mathbf{T}_{k,k-1}$ that minimizes the photometric difference between image patches corresponding to the same 3D point (blue squares). Note, in all figures, the parameters to optimize are drawn in red and the optimization cost is highlighted in blue.

2.System Overview

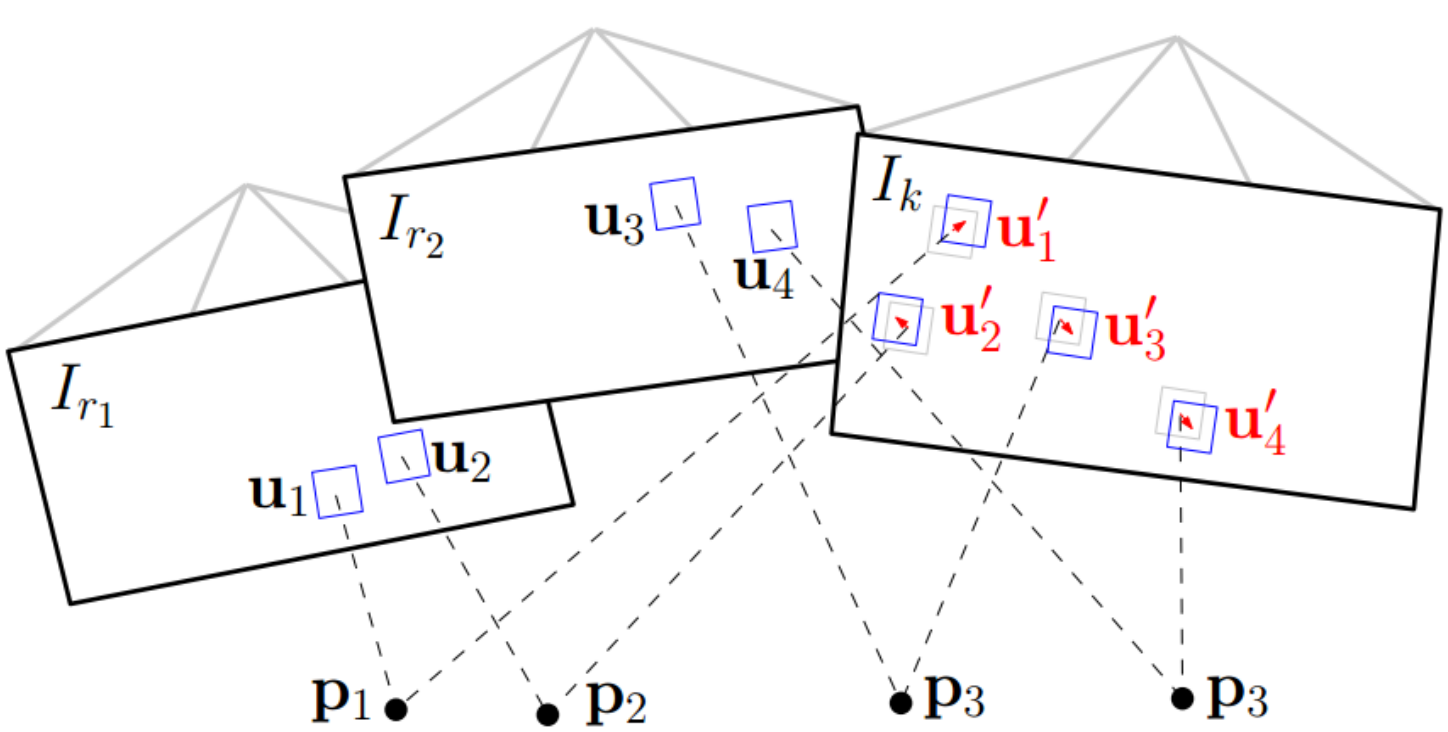


Fig. 3: Due to inaccuracies in the 3D point and camera pose estimation, the photometric error between corresponding patches (blue squares) in the current frame and previous keyframes r_i can further be minimised by optimising the 2D position of each patch individually.

2.System Overview

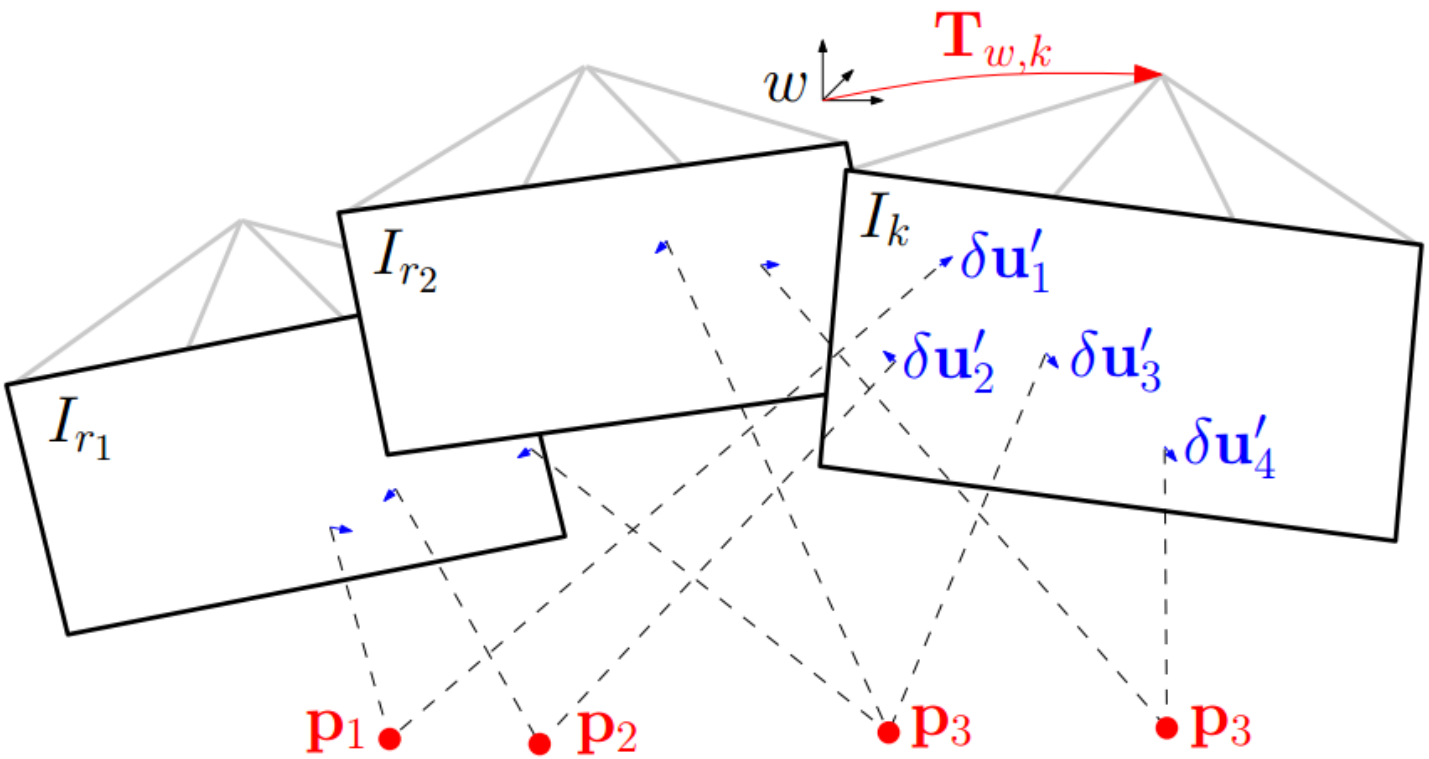


Fig. 4: In the last motion estimation step, the camera pose and the structure (3D points) are optimized to minimize the reprojection error that has been established during the previous feature-alignment step.

2.System Overview

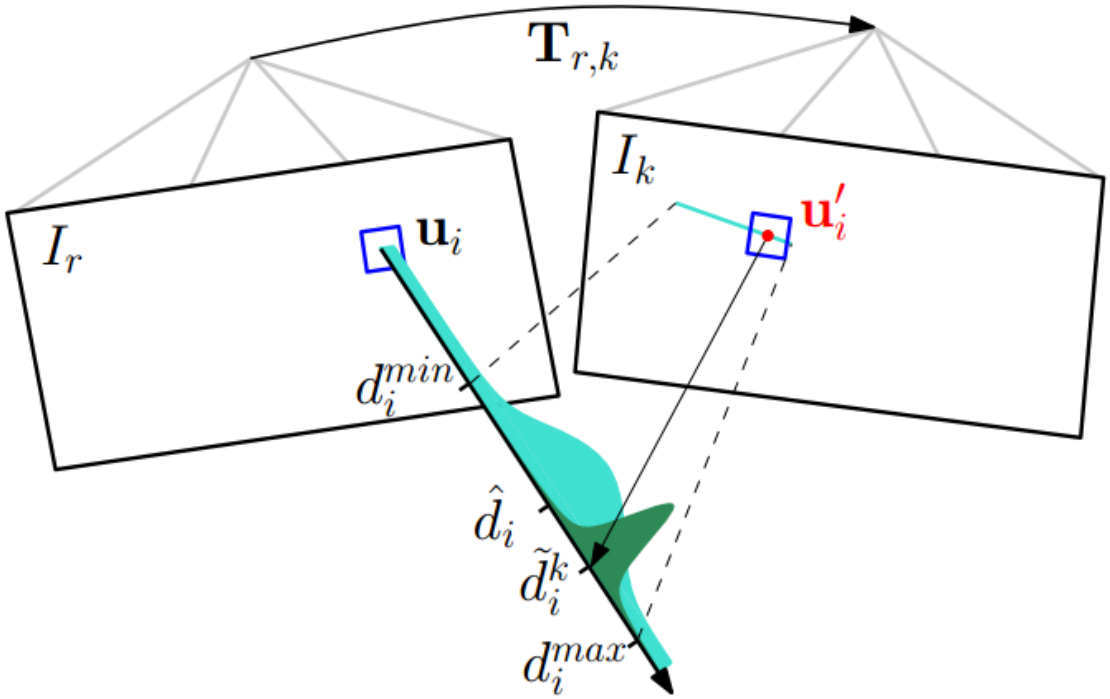


Fig. 5: Probabilistic depth estimate $\hat{d}_i$ for feature i in the reference frame r. The point at the true depth projects to similar image regions in both images (blue squares). Thus, the depth estimate is updated with the triangulated depth $\tilde{d}_i^k$ computed from the point $\mathbf{u}'_i$ of highest correlation with the reference patch. The point of highest correlation lies always on the epipolar line in the new image.

3.Notation

- The intensity image collected at timestep **k** is denoted with, $I_k : \Omega \subset \mathbb{R}^2 \mapsto \mathbb{R}$ where Ω is the image domain.
- Any 3D point $\mathbf{p} = (x, y, z)^\top$ on the visible scene surface S ⊂ R3 maps to the image coordinates $\mathbf{u} = (u, v)^\top \in \Omega$ through the camera projection model $\pi : \mathbb{R}^3 \mapsto \mathbb{R}^2$:

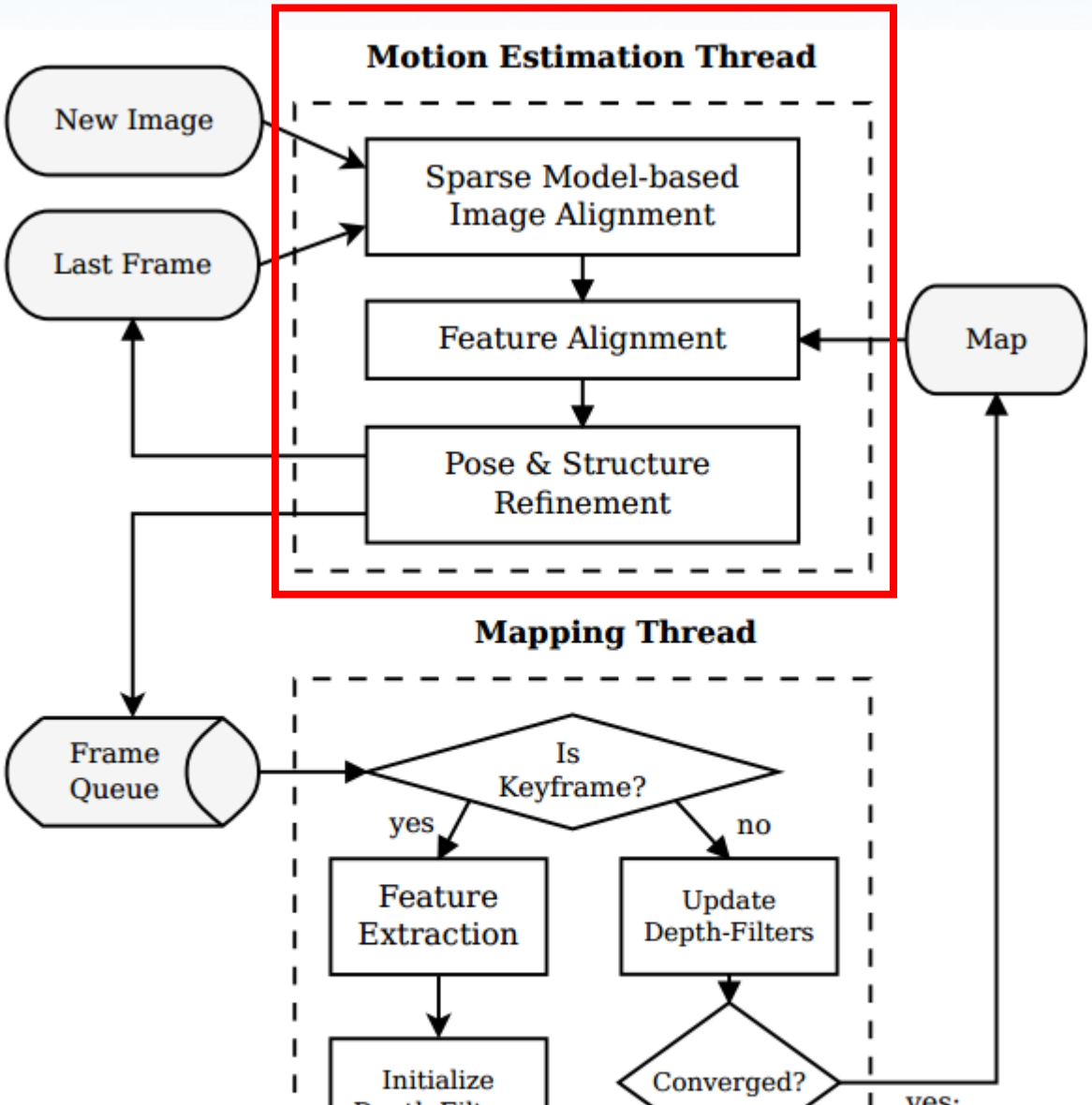
$$\mathbf{u} = \pi({}_k\mathbf{p})$$

where the prescript **k** denotes that the point coordinates are expressed in the camera frame of reference **k**.

- The 3D point corresponding to an image coordinate u can be recovered, given the inverse projection function π^{-1} and the depth $d_\mathbf{u} \in \mathcal{R}$:

$${}_k\mathbf{p} = \pi^{-1}(\mathbf{u}, d_\mathbf{u})$$

4.Motion Estimation



4.Motion Estimation

- **A. Sparse Model-based Image Alignment**

The maximum likelihood estimate of the rigid body transformation $\mathbf{T}_{k,k-1}$ between two consecutive camera poses minimizes the negative log-likelihood of the intensity residuals:

$$\mathbf{T}_{k,k-1} = \arg\min_{\mathbf{T}} \iint_{\bar{\mathcal{R}}} \rho\left[\delta I(\mathbf{T},\mathbf{u})\right] d\mathbf{u}.$$

The intensity residual δI is defined by the photometric difference between pixels observing the same 3D point.

4.Motion Estimation

- **A. Sparse Model-based Image Alignment**

$$\delta I(\mathbf{T},\mathbf{u}) = I_k\left(\pi\left(\mathbf{T}\cdot\pi^{-1}(\mathbf{u},d_{\mathbf{u}})\right)\right) - I_{k-1}(\mathbf{u}) \quad \forall\, \mathbf{u}\in\bar{\mathcal{R}}$$

$$\bar{\mathcal{R}} = \left\{\mathbf{u} \;\middle|\; \mathbf{u}\in\mathcal{R}_{k-1} \wedge \pi\left(\mathbf{T}\cdot\pi^{-1}(\mathbf{u},d_{\mathbf{u}})\right)\in\Omega_k\right\}.$$

the depth du is known at time $k-1$

back-projected points are visible in the current image

4.Motion Estimation

- **A. Sparse Model-based Image Alignment**

For the sake of simplicity, we assume in the following that the intensity residuals are normally distributed with unit variance. The negative log likelihood minimizer then corresponds to the least squares problem:

$$\rho[.] \hat{=} \frac{1}{2} \| . \|^2$$

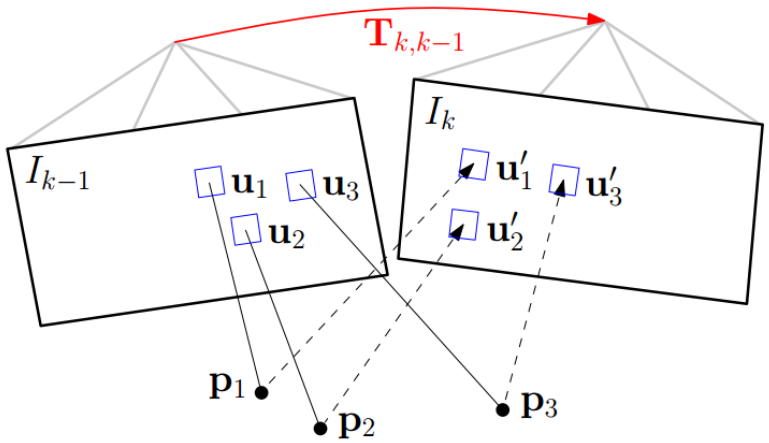
In practice, the distribution has heavier tails due to occlusions and thus, a robust cost function must be applied.

4.Motion Estimation

- **A. Sparse Model-based Image Alignment**

We denote small patches of 4 × 4 pixels around the feature point with the vector $\mathbf{I}(\mathbf{u}_i)$. We seek to find the camera pose that minimizes the photometric error of all patches (see Figure 2)

$$\mathbf{T}_{k,k-1} = \arg\min_{\mathbf{T}_{k,k-1}} \frac{1}{2} \sum_{i \in \bar{\mathcal{R}}} \| \delta\mathbf{I}(\mathbf{T}_{k,k-1}, \mathbf{u}_i) \|^2 .$$



4.Motion Estimation

- **A. Sparse Model-based Image Alignment**

$$\mathbf{T}_{k,k-1} = \arg\min_{\mathbf{T}_{k,k-1}} \frac{1}{2} \sum_{i \in \bar{\mathcal{R}}} \| \delta\mathbf{I}(\mathbf{T}_{k,k-1}, \mathbf{u}_i) \|^2 .$$

we solve it in an iterative Gauss-Newton procedure.

Given an estimate of the relative transformation $\hat{\mathbf{T}}_{k,k-1}$, an incremental update **T(ξ)** to the estimate can be parametrised with a twist $\xi \in \mathfrak{se}(3)$.

4.Motion Estimation

$$\mathbf{T}_{k,k-1} = \arg\min_{\mathbf{T}_{k,k-1}} \frac{1}{2} \sum_{i \in \bar{\mathcal{R}}} \| \delta\mathbf{I}(\mathbf{T}_{k,k-1}, \mathbf{u}_i) \|^2 .$$

- **A. Sparse Model-based Image Alignment**

We use the ***inverse compositional*** **formulation** [27] of the intensity residual, which computes the update step T(ξ) for the reference image at time *k* − 1:

$$\delta\mathbf{I}(\xi, \mathbf{u}_i) = \mathbf{I}_k\left(\pi\left(\hat{\mathbf{T}}_{k,k-1} \cdot \mathbf{p}_i\right)\right) - \mathbf{I}_{k-1}\left(\pi\left(\mathbf{T}(\xi) \cdot \mathbf{p}_i\right)\right)$$

Given an estimate of the relative transformation

incremental update

with $\mathbf{p}_i = \pi^{-1}(\mathbf{u}_i, d_{\mathbf{u}_i})$.

The inverse of the update step is then applied to the current estimate

$$\hat{\mathbf{T}}_{k,k-1} \longleftarrow \hat{\mathbf{T}}_{k,k-1} \cdot \mathbf{T}(\xi)^{-1} .$$

[27] S. Baker and I. Matthews, “Lucas-Kanade 20 Years On: A Unifying Framework: Part 1,” *International Journal of Computer Vision*, vol. 56, no. 3, pp. 221–255, 2002.

4.Motion Estimation

- **A. Sparse Model-based Image Alignment**

$$\delta \mathbf{I}(\xi, \mathbf{u}_i) = \mathbf{I}_k\left(\pi\left(\hat{\mathbf{T}}_{k,k-1} \cdot \mathbf{p}_i\right)\right) - \mathbf{I}_{k-1}\left(\pi\left(\mathbf{T}(\xi) \cdot \mathbf{p}_i\right)\right)$$

Note that we do not warp the patches for computing speed reasons. This assumption is valid in case of **small frame-to-frame motions** and for **small patch-sizes**.

4.Motion Estimation

$$\delta \mathbf{I}(\xi, \mathbf{u}_i) = \mathbf{I}_k\Big(\pi\big(\hat{\mathbf{T}}_{k,k-1} \cdot \mathbf{p}_i\big)\Big) - \mathbf{I}_{k-1}\Big(\pi\big(\mathbf{T}(\xi) \cdot \mathbf{p}_i\big)\Big), \quad (8)$$

$$\mathbf{T}_{k,k-1} = \arg\min_{\mathbf{T}_{k,k-1}} \frac{1}{2} \sum_{i \in \bar{\mathcal{R}}} \| \delta \mathbf{I}(\mathbf{T}_{k,k-1}, \mathbf{u}_i) \|^2 . \quad (7)$$

To find the optimal update step T(ξ), we compute the derivative of (7) and set it to zero:

$$\sum_{i \in \bar{\mathcal{R}}} \nabla \delta \mathbf{I}(\xi, \mathbf{u}_i)^\top \ \delta \mathbf{I}(\xi, \mathbf{u}_i) = 0. \quad (10)$$

To solve this system, we linearize around the current state

$$\delta \mathbf{I}(\xi, \mathbf{u}_i) \approx \delta \mathbf{I}(0, \mathbf{u}_i) + \nabla \delta \mathbf{I}(0, \mathbf{u}_i) \cdot \xi \quad (11)$$

4.Motion Estimation

$$\sum_{i \in \bar{\mathcal{R}}} \nabla \delta \mathbf{I}(\xi, \mathbf{u}_i)^\top \; \delta \mathbf{I}(\xi, \mathbf{u}_i) = 0. \qquad (10)$$

$$\delta \mathbf{I}(\xi, \mathbf{u}_i) \approx \delta \mathbf{I}(0, \mathbf{u}_i) + \nabla \delta \mathbf{I}(0, \mathbf{u}_i) \cdot \xi \qquad (11)$$

By inserting (11) into (10) and by stacking the Jacobians in a matrix J, we obtain the normal equations:

$$\mathbf{J}^T \mathbf{J} \; \xi = -\mathbf{J}^T \delta \mathbf{I}(0)$$

which can be solved for the update twist ξ.

4.Motion Estimation

$$\mathbf{J}^T\mathbf{J}\,\xi = -\mathbf{J}^T\delta\mathbf{I}(0)$$

Note that by using the inverse compositional approach**, the Jacobian can be precomputed as it remains constant over all iterations** ,which results in a significant speedup.

$$\mathbf{J}_i := \nabla\delta\mathbf{I}(0,\mathbf{u}_i)$$

$$\frac{\partial\delta\mathbf{I}(\xi,\mathbf{u}_i)}{\partial\xi} = \frac{\partial\mathbf{I}_{k-1}(\mathbf{a})}{\partial\mathbf{a}}\Big|_{\mathbf{a}=\mathbf{u}_i} \cdot \frac{\partial\pi(\mathbf{b})}{\partial\mathbf{b}}\Big|_{\mathbf{b}=\mathbf{p}_i} \cdot \frac{\partial\mathbf{T}(\xi)}{\partial\xi}\Big|_{\xi=0} \mathbf{p}_i$$

4.Motion Estimation

- **B. Feature Alignment**

  The last step aligned the camera with respect to the previous frame.

  Through back-projection, the found relative pose $\mathbf{T}_{k,k-1}$ implicitly defines an initial guess for the feature positions of all visible 3D points in the new image. **Due to inaccuracies in the 3D points' positions and, thus, the camera pose, this initial guess can be improved.**

  To reduce the drift, the camera pose should be aligned with respect to the map, rather than to the previous frame.

4.Motion Estimation

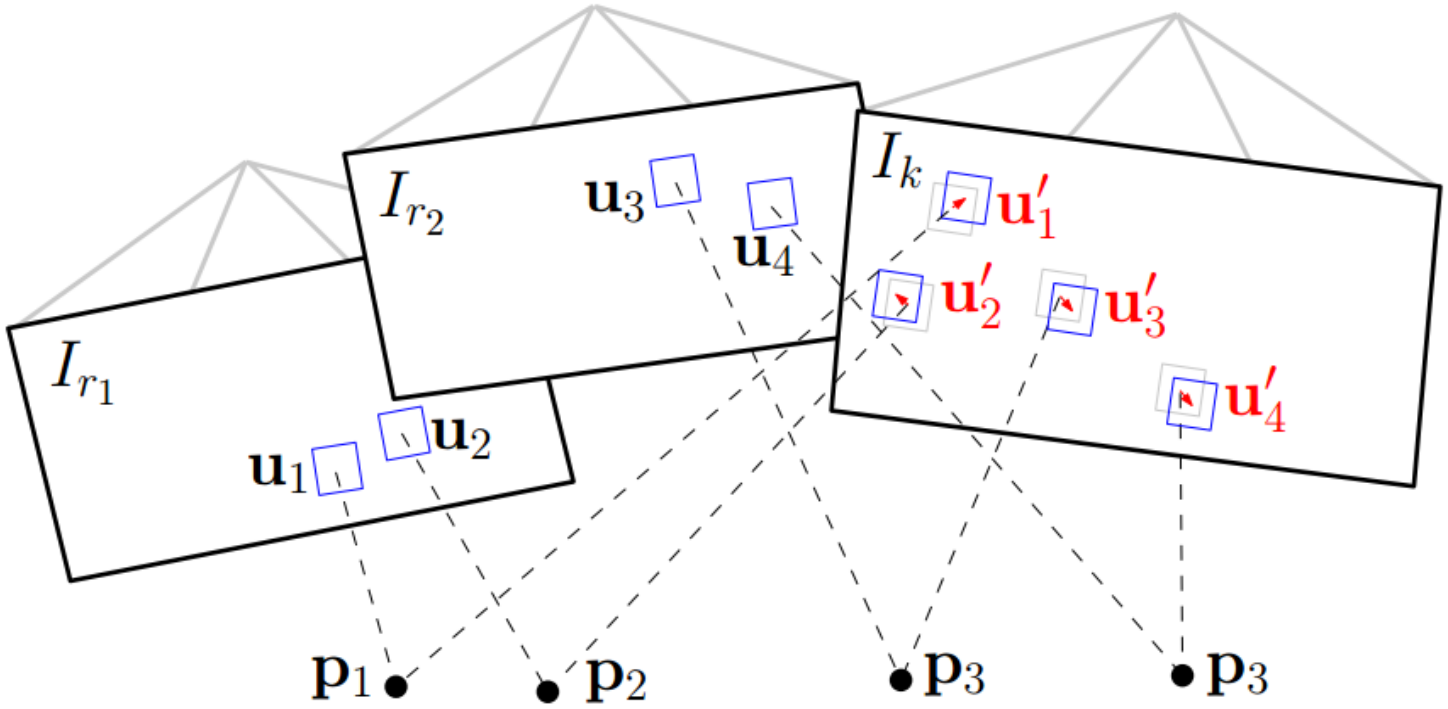


Fig. 3: Due to inaccuracies in the 3D point and camera pose estimation, the photometric error between corresponding patches (blue squares) in the current frame and previous keyframes r_i can further be minimised by optimising the 2D position of each patch individually.

4.Motion Estimation

$$\mathbf{u}_i' = \arg\min_{\mathbf{u}_i'} \frac{1}{2} \| \mathbf{I}_k(\mathbf{u}_i') - \mathbf{A}_i \cdot \mathbf{I}_r(\mathbf{u}_i) \|^2, \quad \forall\ i. \tag{13}$$

This alignment is solved using the inverse compositional Lucas-Kanade algorithm [27]. Contrary to the previous step, we apply an affine warping $\mathbf{A}_i$ to the reference patch, since a larger patch size is used (8×8 pixels) and the closest keyframe is typically farther away than the previous image.

This step can be understood as a relaxation step that violates the epipolar constraints to achieve a higher correlation between the feature-patches.

4.Motion Estimation

- **C. Pose and Structure Refinement**

In the previous step, we have established feature correspondence with subpixel accuracy at the cost of violating the epipolar constraints.

In particular, we have generated a reprojection residual

$$||\delta \mathbf{u}_i|| = ||\mathbf{u}_i - \pi(\mathbf{T}_{k,w}\ {}_w\mathbf{p}_i)|| \neq 0$$

which is around 0.3 pixels.

4.Motion Estimation

- **C. Pose and Structure Refinement**

  In this final step, we again optimize the camera pose $\mathbf{T}_{k,w}$ to minimize the reprojection residuals (see Figure 4):

$$\mathbf{T}_{k,w} = \arg\min_{\mathbf{T}_{k,w}} \frac{1}{2} \sum_i \| \mathbf{u}_i - \pi(\mathbf{T}_{k,w} \ {}_w\mathbf{p}_i) \|^2 .$$

  This is the well known problem of **motion-only** BA .

  Subsequently, we optimize the position of the observed 3D points through reprojection error minimization (**structure only BA**).

  Finally, it is possible to apply **local BA**, in which both the pose of all close keyframes as well as the observed 3D points are jointly optimized.

4.Motion Estimation

- **D. Discussion**

One could directly start with the second step and establish feature correspondence through Lucas-Kanade tracking of all feature-patches, followed by nonlinear pose refinement(BA).

While this would work,

1. The processing time would be higher. Tracking all features over large distances (e.g., 30 pixels) requires a larger patch and a pyramidal implementation.

2. Furthermore, some features might be tracked inaccurately, which would require outlier detection.

In SVO however, the sparse image alignment step satisfies implicitly the epipolar constraint and ensures that there are no outliers.

4.Motion Estimation

- **D. Discussion**

  One may also argue that the first step (sparse image alignment) would be sufficient to estimate the camera motion. In fact, this is what recent algorithms developed for RGB-D cameras do, however, by aligning the full depth-map rather than sparse patches.

  We found empirically that using the first step only results in significantly more drift compared to using all three steps together.

  The improved accuracy is due to the alignment of the new image with respect to the keyframes and the map, whereas sparse image alignment aligns the new frame only with respect to the previous frame.

4.Mapping

- Given an image and its pose $\{I_k, \mathbf{T}_{k,w}\}$ the mapping thread estimates the depth of 2D features for which the corresponding 3D point is not yet known.

The depth estimate of a feature is modeled with a probability distribution. Every subsequent observation is used to update the distribution in a Bayesian framework (see Figure 5) as in [28].

When the variance of the distribution becomes small enough, the depth-estimate is converted to a 3D point, the point is inserted in the map and immediately used for motion estimation.

[28] G. Vogiatzis and C. Hernandez, “Video-based, Real-Time Multi View Stereo,” *Image and Vision Computing*, vol. 29, no. 7, 2011.

4.Mapping

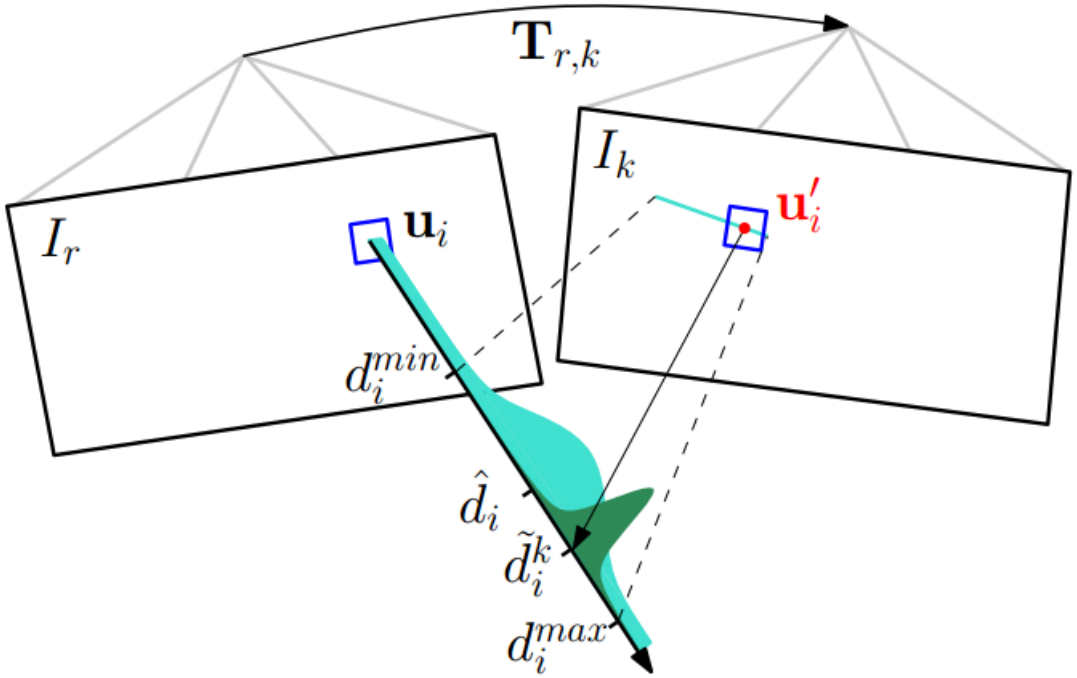


Fig. 5: Probabilistic depth estimate $\hat{d}_i$ for feature i in the reference frame r. The point at the true depth projects to similar image regions in both images (blue squares). Thus, the depth estimate is updated with the triangulated depth $\tilde{d}_i^k$ computed from the point $\mathbf{u}'_i$ of highest correlation with the reference patch. The point of highest correlation lies always on the epipolar line in the new image.

4.Mapping

- The main advantage of the proposed methods over the standard approach of triangulating points from two views is that:

  we observe far fewer outliers as every filter undergoes many measurements until convergence. Furthermore, erroneous measurements are explicitly modeled, which allows the depth to converge even in highly-similar environments.

4.Mapping

- Figure 6 demonstrates how little motion is required to significantly reduce the uncertainty in depth.

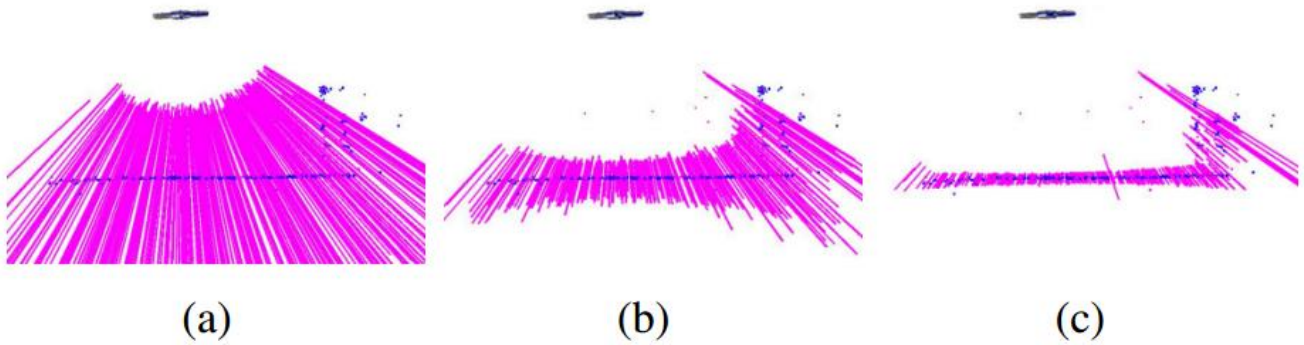


Fig. 6: Very little motion is required by the MAV (seen from the side at the top) for the uncertainty of the depth-filters (shown as mangenta lines) to converge.

6. Implementation Details

The algorithm is bootstrapped to obtain the pose of the first two keyframes and the initial map. The inital map is triangulated from the first two views.

In order to cope with large motions, we apply the sparse image alignment algorithm in a coarse-to-fine scheme.

The algorithm keeps for efficiency reasons a fixed number of *keyframes* in the map, which are used as reference for feature-alignment and for structure refinement. A keyframe is selected if the Euclidean distance of the new frame relative to all keyframes exceeds 12% of the average scene depth.

6. Implementation Details

- In the mapping thread, we divide the image in cells of fixed size (e.g., 30 × 30 pixels). A new depth-filter is initialized at the **FAST** corner with highest Shi-Tomasi score in the cell unless there is already a 2D-to-3D correspondence present.

  This results in evenly distributed features in the image.

7. Experimental Results

- Experiments were performed on datasets recorded from a downward-looking camera attached to a MAV and sequences from a handheld camera.

- The video was processed on both a laptop and on an embedded platform that is mounted on the MAV.

7. Experimental Results

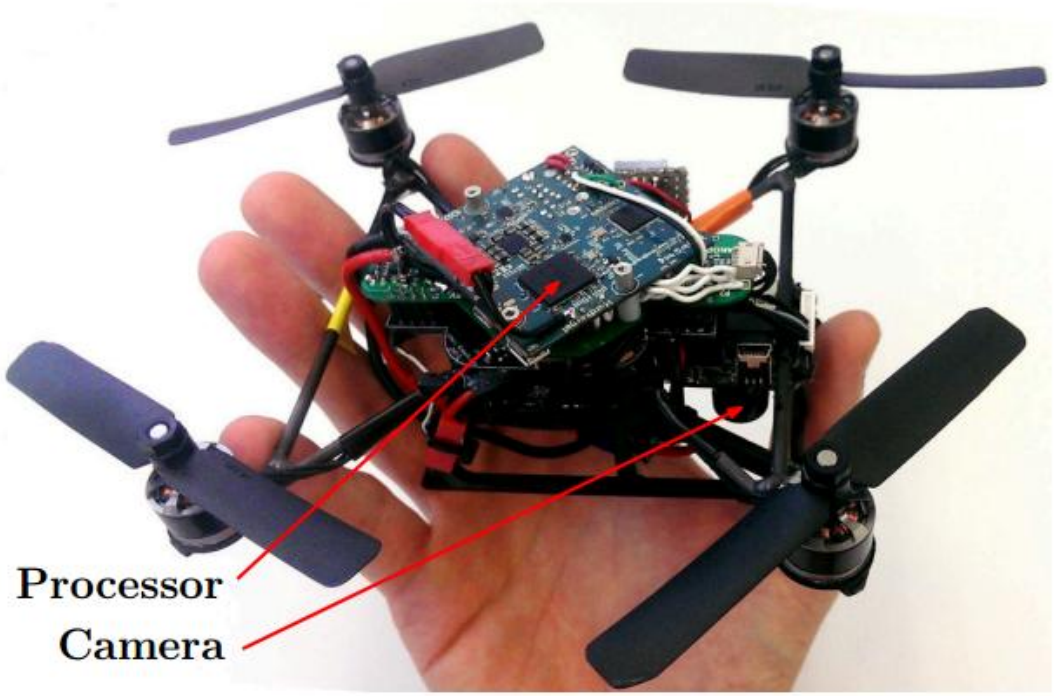


Fig. 17: "Nano+" by KMel Robotics, customized with embedded processor and downward-looking camera. SVO runs at 55 frames per second on the platform and is used for stabilization and control.

7. Experimental Results

- The experiments on the consumer laptop were run with two different parameters' settings, one optimised for speed and one for accuracy (Table I).
- On the embedded platform only the *fast* parameters' setting is used.

	Fast	*Accurate*
Max number of features per image	120	200
Max number of keyframes	10	50
Local Bundle Adjustment	no	yes

TABLE I: Two different parameter settings of SVO.

7. Experimental Results

- We compare the performance of SVO with the modified PTAM algorithm of [2]. The reason we do not compare with the original version of PTAM [16] is because it does not handle large environments and is not robust enough in scenes of high-frequency texture [2]. The version of [2] solves these problems and constitutes to our knowledge the best performing monocular SLAM algorithm for MAVs.

[2] S. Weiss, M. W. Achtelik, S. Lynen, M. C. Achtelik, L. Kneip, M. Chli, and R. Siegwart, Monocular Vision for Long-term Micro Aerial Vehicle State Estimation: A Compendium," *Journal of Field Robotics*, vol. 30, no. 5, 2013.

[16] G. Klein and D. Murray, "Parallel Tracking and Mapping for Small AR Workspaces," *IEEE and ACM International Symposium on Mixed and Augmented Reality*, pp. 1–10, Nov. 2007.

7. Experimental Results

- *A. Accuracy*

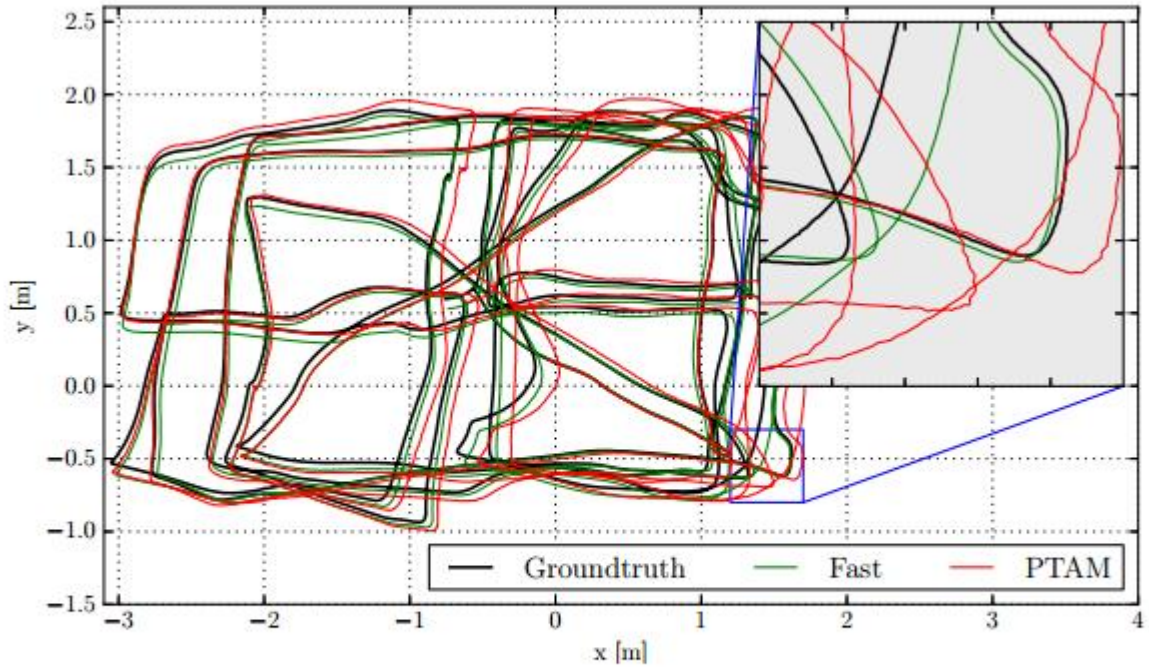


Fig. 7: Comparison against the ground-truth of SVO with the *fast* parameter setting (see Table I) and of PTAM. Zooming-in reveals that the proposed algorithm generates a smoother trajectory than PTAM.

7. Experimental Results

- *A. Accuracy*

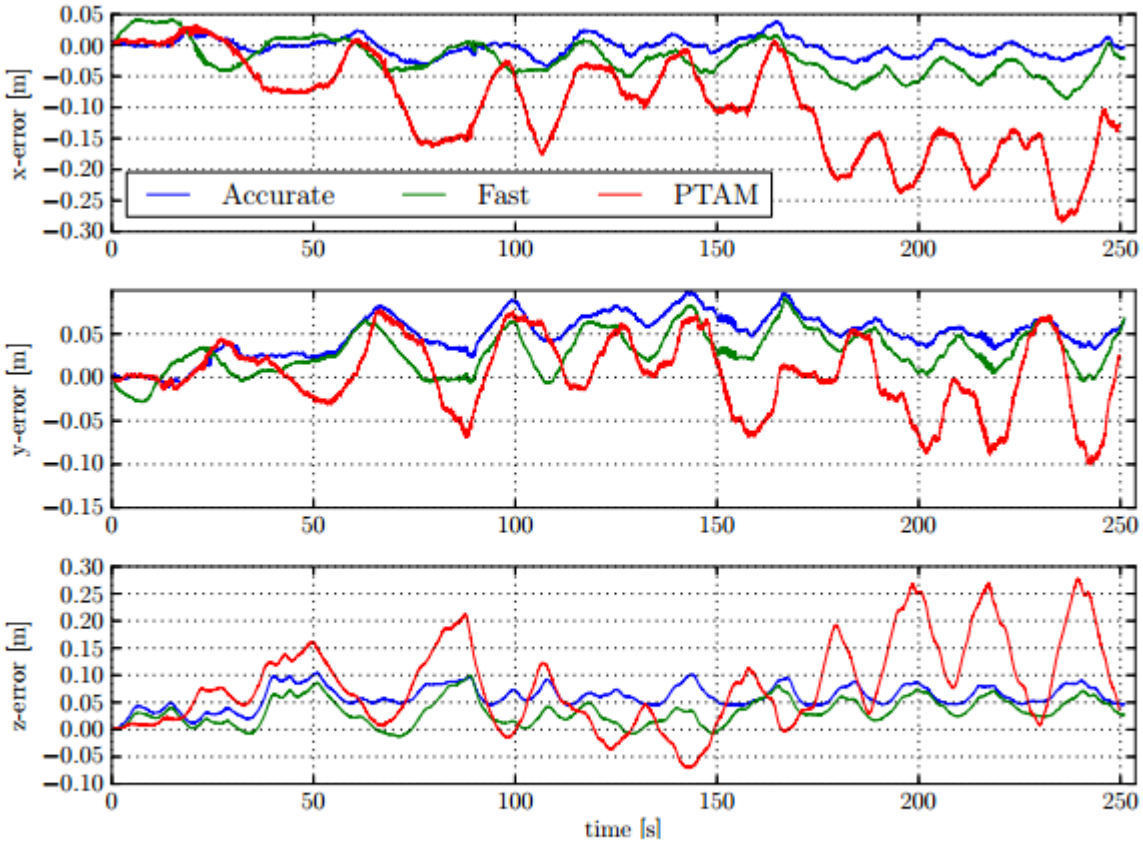


Fig. 8: Position drift of SVO with *fast* and *accurate* parameter setting and comparison against PTAM.

7. Experimental Results

- *A. Accuracy*

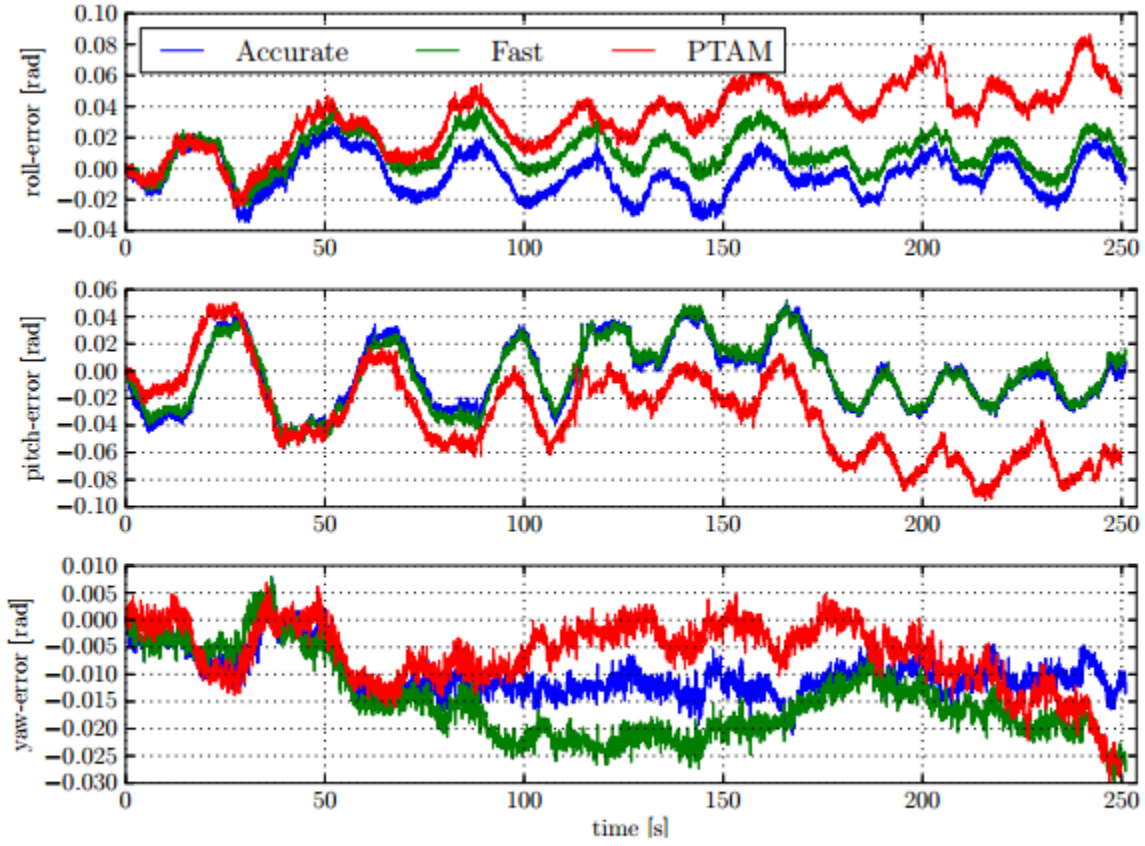


Fig. 9: Attitutde drifts of SVO with *fast* and *accurate* parameter setting and comparison against PTAM.

7. Experimental Results

- *B. Speed*

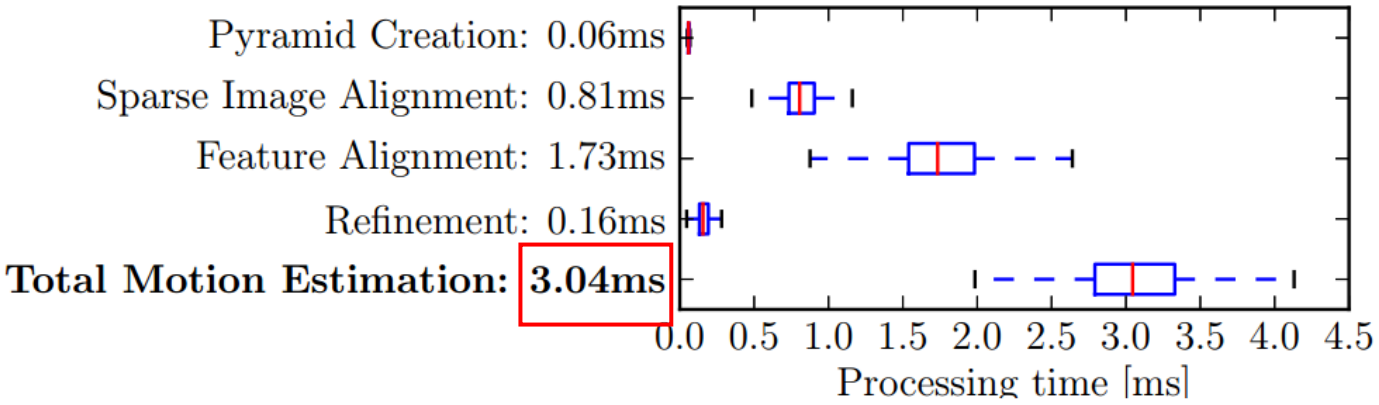


Fig. 13: Timing results on a laptop computer.

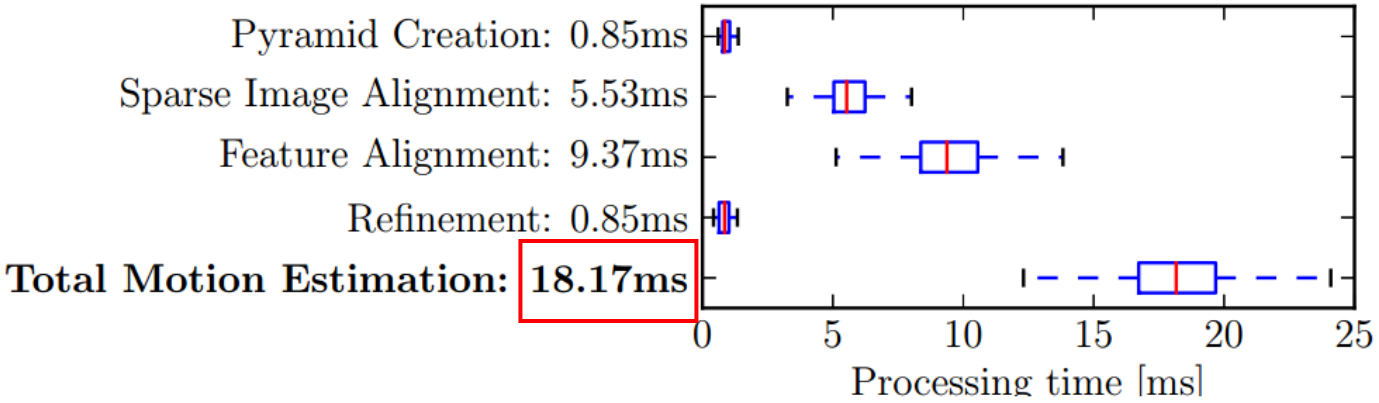


Fig. 14: Timing results on the embedded platform.

7. Experimental Results

- *C. Robustness*

- The speed and accuracy of SVO is partially due to the depth-filter, which produces only a minimal number of outlier 3D points. Also the robustness is due to the depth filter: precise, high frame-rate tracking allows the filter to converge even in scenes of repetitive and high-frequency texture (e.g., asphalt, grass), as it is best demonstrated in
the video accompanying this paper.

Screenshots of the video are shown in Figure 15

7. Experimental Results

- *C. Robustness*

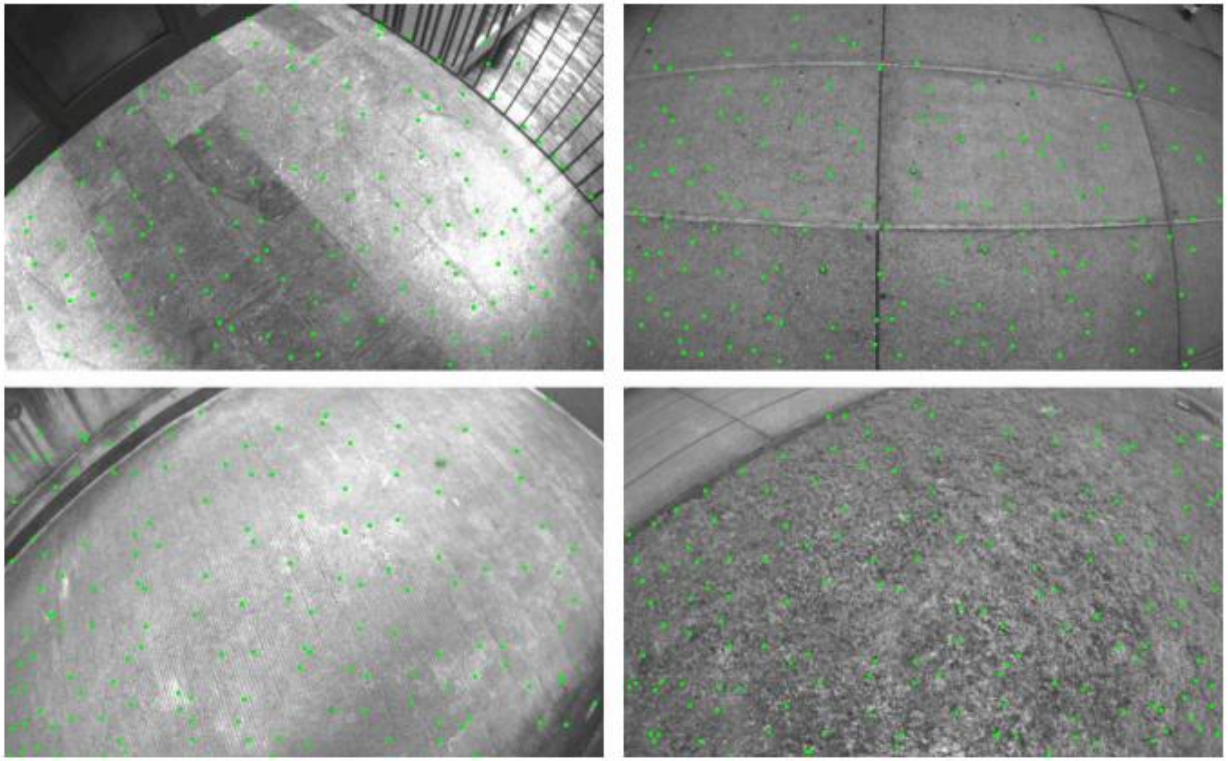


Fig. 15: Successful tracking in scenes of high-frequency texture.

7. Experimental Results

- *C. Robustness*

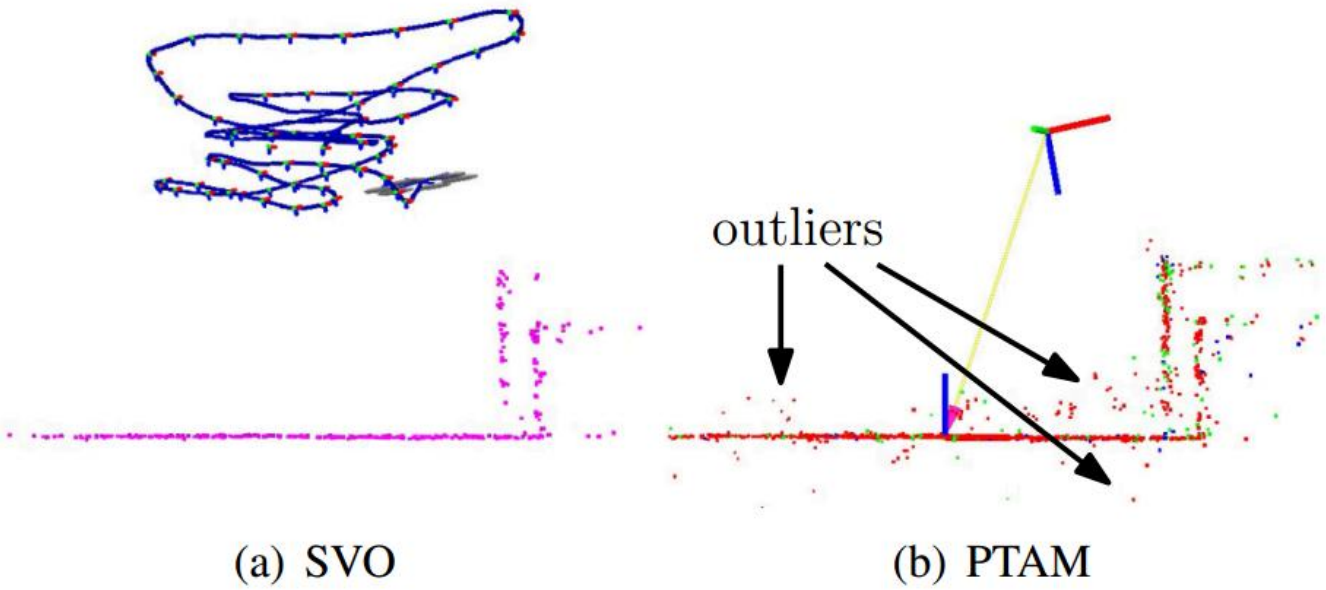


Fig. 16: Sideview of a piecewise-planar map created by SVO and PTAM. The proposed method has fewer outliers due to the depth-filter.

8. Conclusion

- In this paper, we proposed the semi-direct VO pipeline “SVO” that is precise and faster than the current state-of-the-art.

- The gain in speed is due to the fact that feature-extraction and matching is not required for motion estimation. Instead, a direct method is used, which is based directly on the image intensities.

- High frame rate motion estimation, combined with an outlier resistant probabilistic mapping method, provides increased robustness in scenes of little, repetitive, and high frequency-texture.

谢 谢