Semi-Direct Visual Odometry for Monocular, Wide-angle, and Multi-Camera Systems

Christian Forster, Zichao Zhang, Michael Gassner, Manuel Werlberger, Davide Scaramuzza

***Abstract*—Direct methods for Visual Odometry (VO) have gained popularity due to their capability to exploit information from all image gradients in the image. However, low computational speed as well as missing guarantees for optimality and consistency are limiting factors of direct methods, where established feature-based methods instead succeed at. Based on these considerations, we propose a Semi-direct VO (SVO) that uses direct methods to track and triangulate pixels that are characterized by high image gradients but relies on proven feature-based methods for joint optimization of structure and motion. Together with a robust probabilistic depth estimation algorithm, this enables us to efficiently track pixels lying on weak corners and edges in environments with little or high-frequency texture. We further demonstrate that the algorithm can easily be extended to multiple cameras, to track edges, to include motion priors, and to enable the use of very large field of view cameras, such as fisheye and catadioptric ones. Experimental evaluation on benchmark datasets shows that the algorithm is significantly faster than the state of the art while achieving highly competitive accuracy.**

Supplementary Material

Video of the experiments: http://rpg.ifi.uzh.ch/svo2

I. Introduction

Estimating the six degrees-of-freedom motion of a camera merely from its stream of images has been an active field of research for several decades [1–6]. Today, state-of-the-art visual SLAM (V-SLAM) and visual odometry (VO) algorithms run in real-time on smart-phone processors and approach the accuracy, robustness, and efficiency that is required to enable various interesting applications. Examples comprise the robotics and automotive industry, where the ego-motion of a vehicle must be known for autonomous operation. Other applications are virtual and augmented reality, which requires precise and low latency pose estimation of mobile devices.

The central requirement for the successful adoption of vision-based methods for such challenging applications is to obtain highest *accuracy* and *robustness* with a limited computational budget. The most accurate camera motion estimate is obtained through joint optimization of structure (*i.e.*, landmarks) and motion (*i.e.,* camera poses). For *feature-based methods*, this is an established problem that is commonly known as *bundle adjustment* [7] and many solvers exist, which address the underlying non-linear least-squares problem efficiently [8–11]. Three aspects are key to obtain highest accuracy when using sparse feature correspondence and bundle adjustment: (1) long feature tracks with minimal feature drift, (2) a large number of uniformly distributed features in the image plane, and (3) reliable association of new features to old landmarks (*i.e.*, loop-closures).

The probability that many pixels are tracked reliably, *e.g.,* in scenes with little or high frequency texture (such as sand [12] or asphalt [13]), is increased when the algorithm is not restricted to use local point features (*e.g.,* corners or blobs) but may track edges [14] or more generally, all pixels with gradients in the image, such as in dense [15] or semi-dense approaches [16]. Dense or semi-dense algorithms that operate directly on pixel-level intensities are also denoted as *direct methods* [17]. Direct methods minimize the *photometric error* between corresponding pixels in contrast to feature-based methods, which minimize the *reprojection error*. The great advantage of this approach is that there is no prior step of data association: this is implicitly given through the geometry of the problem. However, joint optimization of dense structure and motion in real-time is still an open research problem, as is the optimal and *consistent* [18, 19] fusion of direct methods with complementary measurements (*e.g.,* inertial). In terms of efficiency, previous direct methods are computationally expensive as they require a semi-dense [16] or dense [15] reconstruction of the environment, while the dominant cost of feature-based methods is the extraction of features and descriptors, which incurs a high constant cost per frame.

In this work, we propose a VO algorithm that combines the advantages of direct and feature-based methods. We introduce the *sparse image alignment* algorithm (Sec. V), an efficient direct approach to estimate frame-to-frame motion by minimizing the photometric error of features lying on intensity corners and edges. The 3D points corresponding to features are obtained by means of robust recursive Bayesian depth estimation (Sec. VI). Once feature correspondence is established, we use bundle adjustment for refinement of the structure and the camera poses to achieve highest accuracy (Sec. V-B). Consequently, we name the system *semi-direct* visual odometry (SVO).

Our implementation of the proposed approach is exceptionally fast, requiring only 2.5 milliseconds to estimate the pose of a frame on a standard laptop computer, while achieving comparable accuracy with respect to the state of the art on benchmark datasets. The improved efficiency is due to three reasons: firstly, SVO extracts features only for selected keyframes in a parallel thread, hence, decoupled from hard

The authors are with with the Robotics and Perception Group, University of Zurich, Switzerland. Contact information: {forster, zzhang, gassner, werlberger, sdavide}@ifi.uzh.ch. This research was partially funded by the Swiss National Foundation (project number 200021-143607, Swarm of Flying Cameras), the National Center of Competence in Research Robotics (NCCR), the UZH Forschungskredit, and the SNSF-ERC Starting Grant.

real-time constraints. Secondly, the proposed direct tracking algorithm removes the necessity for robust data association. Finally, contrarily to previous direct methods, SVO requires only a sparse reconstruction of the environment.

This paper extends our previous work [20], which was also released as open source software.[1] The novelty of the present work is the generalization to wide FoV lenses (Sec. VII), multi-camera systems (Sec. VIII), the inclusion of motion priors (Sec. IX) and the use of edgelet features. Additionally, we present several new experimental results in Sec. XI with comparisons against previous works.

II. Related Work

Methods that simultaneously recover camera pose and scene structure, can be divided into two classes:

a) Feature-based: The standard approach to solve this problem is to extract a sparse set of salient image features (e.g. corners, blobs) in each image; match them in successive frames using invariant feature descriptors; robustly recover both camera motion and structure using epipolar geometry; and finally, refine the pose and structure through reprojection error minimization. The majority of VO and V-SLAM algorithms [6] follow a variant of this procedure. A reason for the success of these methods is the availability of robust feature detectors and descriptors that allow matching images under large illumination and view-point changes. Feature descriptors can also be used to establish feature correspondences with old landmarks when closing loops, which increases both the accuracy of the trajectory after bundle adjustment [7, 21] and the robustness of the overall system due to re-localization capabilities. This is also where we draw the line between VO and V-SLAM: While VO is only about incremental estimation of the camera pose, V-SLAM algorithms, such as [22], detect loop-closures and subsequently refine large parts of the map.

The disadvantage of feature-based approaches is their low speed due to feature extraction and matching at every frame, the necessity for robust estimation techniques that deal with erroneous correspondences (*e.g.,* RANSAC [23], M-estimators [24]), and the fact that most feature detectors are optimized for speed rather than precision. Furthermore, relying only on well localized salient features (*e.g.,* corners), only a small subset of the information in the image is exploited.

In SVO, features are extracted only for selected keyframes, which reduces the computation time significantly. Once extracted, a *direct* method is used to track features from frame to frame, resulting in outlier-free and sub-pixel precise matches. Apart from well localized corner features, this allows tracking and mapping any pixel with non-zero intensity gradient.

b) Direct methods: Direct methods estimate structure and motion directly by minimizing an error measure that is based on the image's pixel-level intensities [17]. The local intensity gradient magnitude and direction is used in the optimization compared to feature-based methods that consider only the distance to a feature-location. Pixel correspondence is given directly by the geometry of the problem, eliminating the need for robust data association techniques. However, this makes the approach dependent on a good initialization that must lie in the basin of attraction of the cost function.

Using a direct approach, the six degree of freedom (DoF) motion of a camera can be recovered by *image-to-model alignment*, which is the process of aligning the observed image to a view synthesized from the estimated 3D map. Early direct VO methods tracked and mapped few—sometimes manually selected—planar patches [25–29]. By estimating the surface normals of the patches [30], they could be tracked over a wide range of viewpoints. In [31], the local planarity assumption was relaxed and direct tracking with respect to arbitrary 3D structures computed from stereo cameras was proposed. For RGB-D cameras, where a dense depth-map for each image is given by the sensor, dense image-to-model alignment was subsequently introduced in [32–34]. In conjunction with dense depth registration this has become the standard in camera tracking for RGB-D cameras [35–38]. With DTAM [15], a direct method was introduced that computes a dense depthmap from a single moving camera in real-time. The camera pose is found through direct whole image alignment using the depthmap. However, inferring a dense depthmaps from monocular images is computationally intensive and is typically addressed using GPU parallelism, such as in the open-source REMODE algorithm [39]. Early on it was realized that only pixels with an intensity gradient provide information for motion estimation [40]. In this spirit, a *semi-dense* approach was proposed in [41] where the depth is only estimated for pixels with high intensity gradients. In our experimental evaluation in Sec. XI-A we show that it is possible to reduce the number of tracked pixels even more for frame-to-frame motion estimation without any noticeable loss in robustness or accuracy. Therefore, we propose the *sparse* image-to-model alignment algorithm that uses only sparse pixels at corners and along image intensity gradients.

A disadvantage of direct methods is that joint optimization of dense structure and motion in real-time is still an open research problem. For this reason, the standard approach is to estimate the latest camera pose with respect to a previously accumulated dense map and subsequently, given a set of estimated camera poses, update the dense map [15, 42]. Clearly, this separation of tracking and mapping only results in optimal accuracy when the output of each stage yields the optimal estimate. Other algorithms optimize a graph of poses but do not allow a deformation of the structure once triangulated [16]. Contrarily, some algorithms ignore the camera poses and instead allow non-rigid deformation of the 3D structure [36, 38]. The obtained results are accurate and visually impressive, however, a thorough probabilistic treatment is missing when processing measurements, separating tracking and mapping, or fixating and removing states. To the best of our knowledge, it is therefore currently not possible to obtain accurate covariance estimates from dense VO. Hence, the *consistent fusion* [18, 43] with complementary sensors (*e.g.*, inertial) is currently not possible. In the proposed work, we use direct methods only to establish feature correspondence. Subsequently, bundle adjustment can be used for joint optimization of structure and motion where it is also possible to include inertial measurements as we have demonstrated in previous work [44].

[1] http://github.com/uzh-rpg/rpg_svo

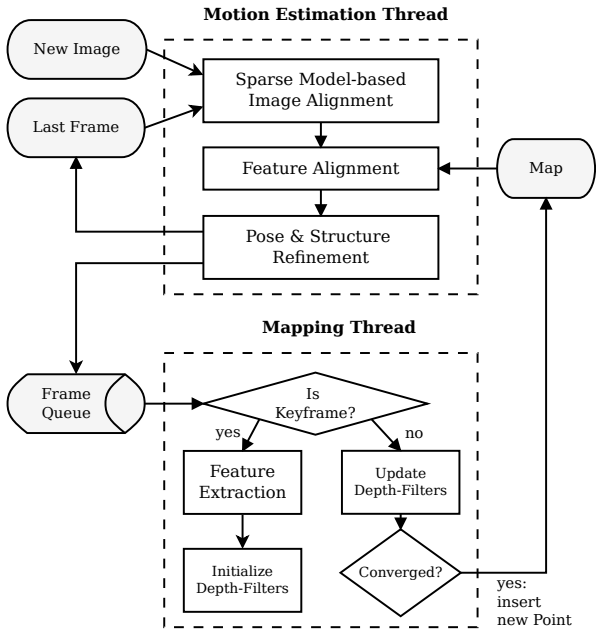


Fig. 1: Tracking and mapping pipeline

III. System Overview

Figure 1 provides an overview of the proposed approach. We use two parallel threads (as in [21]), one for estimating the camera motion, and a second one for mapping as the environment is being explored. This separation allows fast and constant-time tracking in one thread, while the second thread extends the map, decoupled from hard real-time constraints.

The motion-estimation thread implements the proposed semi-direct approach to motion estimation. Our approach is divided into three steps: sparse image alignment, relaxation, and refinement (Fig. 1). Sparse image alignment estimates frame-to-frame motion by minimizing the intensity difference of features that correspond to the projected location of the same 3D points. A subsequent step relaxes the geometric constraint to obtain sub-pixel feature correspondence. This step introduces a reprojection error, which we finally refine by means of bundle adjustment.

In the mapping thread, a probabilistic depth-filter is initialized for each feature for which the corresponding 3D point is to be estimated. New depth-filters are initialized whenever a new keyframe is selected for corner pixels as well as for pixels along intensity gradient edges. The filters are initialized with a large uncertainty in depth and undergo a recursive Bayesian update with every subsequent frame. When a depth filter's uncertainty becomes small enough, a new 3D point is inserted in the map and is immediately used for motion estimation.

IV. Notation

The intensity image recorded from a moving camera C at timestep k is denoted with $\mathrm{I}_k^{\mathrm{C}} : \Omega^{\mathrm{C}} \subset \mathbb{R}^2 \mapsto \mathbb{R}$, where Ω^{C} is the image domain. Any 3D point $\boldsymbol{\rho} \in \mathbb{R}^3$ maps to the image coordinates $\mathbf{u} \in \mathbb{R}^2$ through the camera projection model: $\mathbf{u} = \pi(\boldsymbol{\rho})$. Given the inverse scene depth $\rho > 0$ at pixel $\mathbf{u} \in \mathcal{R}_k^{\mathrm{C}}$, the position of a 3D point is obtained using the back-projection model $\boldsymbol{\rho} = \pi_\rho^{-1}(\mathbf{u})$. Where we denote with $\mathcal{R}_k^{\mathrm{C}} \subseteq \Omega$ those pixels for which the depth is known at time k in camera C. The projection models are known from prior calibration [45].

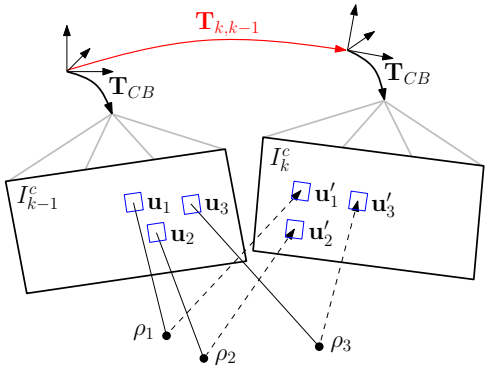


Fig. 2: Changing the relative pose $\mathrm{T}_{k,k-1}$ between the current and the previous frame implicitly moves the position of the reprojected points in the new image $\mathbf{u}'_i$. Sparse image alignment seeks to find $\mathrm{T}_{k,k-1}$ that minimizes the photometric difference between image 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.

The position and orientation of the world frame W with respect to the k^{th} camera frame is described by the rigid body transformation $\mathrm{T}_{k\mathrm{W}} \in \mathrm{SE}(3)$ [46]. A 3D point ${}_{\mathrm{W}}\boldsymbol{\rho}$ that is expressed in world coordinates can be transformed to the k^{th} camera frame using: ${}_{k}\boldsymbol{\rho} = \mathrm{T}_{k\mathrm{W}}\ {}_{\mathrm{W}}\boldsymbol{\rho}$.

V. Motion Estimation

In this section, we describe the proposed semi-direct approach to motion estimation, which assumes that the position of some 3D points corresponding to features in previous frames are known from prior depth estimation.

A. Sparse Image Alignment

Image to model alignment estimates the incremental camera motion by minimizing the intensity difference (*photometric error*) of pixels that observe the same 3D point.

To simplify a later generalization to multiple cameras, we introduce a *body frame* B that is rigidly attached to the camera frame C with known extrinsic calibration $\mathrm{T}_{\mathrm{CB}} \in \mathrm{SE}(3)$ (see Fig. 2). Our goal is to estimate the incremental motion of the body frame $\mathrm{T}_{kk-1} \doteq \mathrm{T}_{\mathrm{B}_k\mathrm{B}_{k-1}}$ such that the photometric error is minimized:

$$\mathrm{T}^{\star}_{kk-1} = \arg\min_{\mathrm{T}_{kk-1}} \sum_{\mathbf{u}\in\bar{\mathcal{R}}^{\mathrm{C}}_{k-1}} \frac{1}{2}\|\mathbf{r}_{\mathrm{I}^{\mathrm{C}}_{\mathbf{u}}}(\mathrm{T}_{kk-1})\|^2_{\Sigma_{\mathrm{I}}}, \qquad (1)$$

where the photometric residual $\mathbf{r}_{\mathrm{I}^{\mathrm{C}}_{\mathbf{u}}}$ is defined by the intensity difference of pixels in subsequent images $\mathrm{I}^{\mathrm{C}}_k$ and $\mathrm{I}^{\mathrm{C}}_{k-1}$ that observe the same 3D point $\boldsymbol{\rho}_{\mathbf{u}}$:

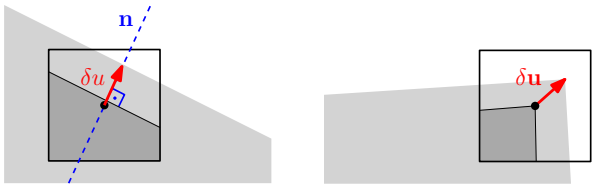
$$\mathbf{r}_{\mathrm{I}^{\mathrm{C}}_{\mathbf{u}}}(\mathrm{T}_{kk-1}) \doteq \mathrm{I}^{\mathrm{C}}_k\big(\pi(\mathrm{T}_{\mathrm{CB}}\mathrm{T}_{kk-1}\ \boldsymbol{\rho}_{\mathbf{u}})\big) - \mathrm{I}^{\mathrm{C}}_{k-1}\big(\pi(\mathrm{T}_{\mathrm{CB}}\ \boldsymbol{\rho}_{\mathbf{u}})\big). \qquad (2)$$

The 3D point $\boldsymbol{\rho}_{\mathbf{u}}$ (which is expressed in the reference frame B_{k-1}) can be computed for pixels with known depth by means of back-projection:

$$\boldsymbol{\rho}_{\mathbf{u}} = \mathrm{T}_{\mathrm{BC}}\ \pi_\rho^{-1}(\mathbf{u}), \qquad \forall\ \mathbf{u} \in \mathcal{R}^{\mathrm{C}}_{k-1}, \qquad (3)$$

(a) Sparse (b) Semi-Dense (c) Dense

Fig. 3: An image from the *ICL-NUIM* dataset (Sec. XI-B3) with pixels used for image-to-model alignment (marked in green for corners and magenta for edgelets) for sparse, semi-dense, and dense methods. Dense approaches (c) use every pixel in the image, semi-dense (b) use just the pixels with high intensity gradient, and the proposed sparse approach (a) uses selected pixels at corners or along intensity gradient edges.



(a) Edge alignment. (b) Corner alignment.

Fig. 4: Different alignment strategies for corners and edgelets. The alignment of an edge feature is restricted to the normal direction $\mathbf{n}$ of the edge.

However, the optimization in Eq. (1) includes only a subset of those pixels $\bar{\mathcal{R}}_{k-1}^{\mathrm{c}} \subseteq \mathcal{R}_{k-1}^{\mathrm{c}}$, namely those for which the back-projected points are also visible in the image $\mathtt{I}_k^{\mathrm{c}}$:

$$\bar{\mathcal{R}}_{k-1}^{\mathrm{c}} = \left\{ \mathbf{u} \;\middle|\; \mathbf{u} \in \mathcal{R}_{k-1}^{\mathrm{c}} \;\wedge\; \pi\left(\mathtt{T}_{\mathrm{CB}} \mathtt{T}_{k k-1} \mathtt{T}_{\mathrm{BC}}\; \pi_\rho^{-1}(\mathbf{u})\right) \in \Omega^{\mathrm{c}} \right\}.$$

Image to model alignment has previously been used in the literature to estimate camera motion. Apart from minor variations in the formulation, the main difference among the approaches is the source of the depth information as well as the region $\mathcal{R}_{k-1}^{\mathrm{c}}$ in image $\mathtt{I}_k^{\mathrm{c}}$ for which the depth is known. As discussed in Section II, we denote methods that know and exploit the depth for all pixels in the reference view as *dense* methods [15]. Converseley, approaches that only perform the alignment for pixels with high image gradients are denoted *semi-dense* [41]. In this paper, we propose a novel *sparse* image alignment approach that assumes known depth only for corners and features lying on intensity edges. Fig. 3 summarizes our notation of dense, semi-dense, and sparse approaches.

To make the sparse approach more robust, we propose to aggregate the photometric cost in a small patch centered at the feature pixel. Since the depth for neighboring pixels is unknown, we approximate it with the same depth that was estimated for the feature.

To summarize, sparse image alignment solves the non-linear least squares problem in Eq. (1) with $\mathcal{R}_{k-1}^{\mathrm{c}}$ corresponding to small patches centered at corner and edgelet features with known depth. This optimization can be solved efficiently using standard iterative non-linear least squares algorithms such as Levenberg-Marquardt. More details on the optimization, including the analytic Jacobians, are provided in the Appendix.

B. Relaxation and Refinement

Sparse image alignment is an efficient method to estimate the incremental motion between subsequent frames. However, to minimize drift in the motion estimate, it is paramount to register a new frame to the oldest frame possible. One approach is to use an older frame as reference for image alignment [16]. However, the robustness of the alignment cannot be guaranteed as the distance between the frames in the alignment increases (see experiment in Section XI-A). We therefore propose to relax the geometric constraints given by the reprojection of 3D points and to perform an individual 2D alignment of corresponding feature patches. The alignment of each patch in the new frame is performed with respect to a reference patch from the frame where the feature was first extracted; hence, the oldest frame possible, which should maximally minimize feature drift. However, the 2D alignment generates a reprojection error that is the difference between the projected 3D point and the aligned feature position. Therefore, in a final step, we perform bundle adjustment to optimize both the 3D point's position and the camera poses such that this reprojection error is minimized.

In the following, we detail our approach to feature alignment and bundle adjustment. Thereby, we take special care of features lying on intensity gradient edges.

2D feature alignment minimizes the intensity difference of a small image patch $\mathcal{P}$ that is centered at the projected feature position $\mathbf{u}'$ in the newest frame k with respect to a reference patch from the frame r where the feature was first observed (see Fig. 4). To improve the accuracy of the alignment, we apply an affine warping $\mathbf{A}$ to the reference patch, which is computed from the estimated relative pose $\mathtt{T}_{kr}$ between the reference frame and the current frame [21]. For corner features, the optimization computes a correction $\delta\mathbf{u}^\star \in \mathbb{R}^2$ to the predicted feature position $\mathbf{u}'$ that minimizes the photometric cost:

$$\mathbf{u}'^\star = \mathbf{u}' + \delta\mathbf{u}^\star, \quad \text{with} \quad \mathbf{u}' = \pi\left(\mathtt{T}_{\mathrm{CB}}\; \mathtt{T}_{kr}\; \mathtt{T}_{\mathrm{BC}}\; \pi_\rho^{-1}(\mathbf{u})\right) \tag{4}$$

$$\delta\mathbf{u}^\star = \arg\min_{\delta\mathbf{u}} \sum_{\Delta\mathbf{u}\in\mathcal{P}} \frac{1}{2} \left\| \mathtt{I}_k^{\mathrm{c}}\left(\mathbf{u}' + \delta\mathbf{u} + \Delta\mathbf{u}\right) - \mathtt{I}_r^{\mathrm{c}}(\mathbf{u} + \mathbf{A}\Delta\mathbf{u}) \right\|^2,$$

where $\Delta\mathbf{u}$ is the iterator variable that is used to compute the sum over the patch $\mathcal{P}$. This alignment is solved using the inverse compositional Lucas-Kanade algorithm [47].

For features lying on intensity gradient edges, 2D feature alignment is problematic because of the aperture problem — features may drift along the edge. Therefore, we limit the degrees of freedom in the alignment to the normal direction to the edge. This is illustrated in Fig. 4a, where a warped reference feature patch is schematically drawn at the predicted position in the newest image. For features on edges, we therefore optimize for a scalar correction $\delta u^\star \in \mathbb{R}$ in the direction of the edge normal $\mathbf{n}$ to obtain the corresponding feature position $\mathbf{u}'^\star$ in the newest frame:

$$\mathbf{u}'^\star = \mathbf{u}' + \delta u^\star \cdot \mathbf{n}, \quad \text{with} \tag{5}$$

$$\delta u^\star = \arg\min_{\delta u} \sum_{\Delta\mathbf{u}\in\mathcal{P}} \frac{1}{2} \left\| \mathtt{I}_k^{\mathrm{c}}\left(\mathbf{u}' + \delta u \cdot \mathbf{n} + \Delta\mathbf{u}\right) - \mathtt{I}_r^{\mathrm{c}}(\mathbf{u} + \mathbf{A}\Delta\mathbf{u}) \right\|^2.$$

This is similar to previous work on VO with edgelets, where feature correspondence is found by sampling along the normal direction for abrupt intensity changes [14, 48–52]. However,

in our case, sparse image alignment provides a very good initialization of the feature position, which directly allows us to follow the intensity gradient in an optimization.

After feature alignment, we have established feature correspondence with subpixel accuracy. However, feature alignment violated the epipolar constraints and introduced a reprojection error $\delta\mathbf{u}$, which is typically well below 0.5 pixels. Therefore, in the last step of motion estimation, we refine the camera poses and landmark positions $\mathcal{X} = \{\mathtt{T}_{k\mathrm{w}}, \boldsymbol{\rho}_i\}$ by minimizing the squared sum of reprojection errors:

$$\mathcal{X}^\star = \arg\min_{\mathcal{X}} \sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{L}_k^C} \frac{1}{2}\|\mathbf{u}'^\star_i - \pi\big(\mathtt{T}_{\mathrm{CB}}\ \mathtt{T}_{k\mathrm{W}}\ \boldsymbol{\rho}_i\big)\|^2 \qquad (6)$$
$$+ \sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{L}_k^E} \frac{1}{2}\|\mathbf{n}_i^\mathsf{T}\big(\mathbf{u}'^\star_i - \pi\big(\mathtt{T}_{\mathrm{CB}}\ \mathtt{T}_{k\mathrm{W}}\ \boldsymbol{\rho}_i\big)\big)\|^2$$

where $\mathcal{K}$ is the set of all keyframes in the map, $\mathcal{L}_k^C$ the set of all landmarks corresponding to corner features, and $\mathcal{L}_k^E$ the set of all edge features that were observed in the k^{th} camera frame. The reprojection error of edge features is projected along the edge normal because the component along the edge cannot be determined.

The optimization problem in Eq. (6) is a standard bundle adjustment problem that can be solved in real-time using iSAM2 [9]. In [44] we further show how the objective function can be extended to include inertial measurements.

While optimization over the whole trajectory in Eq. (6) results in the most accurate results (see Sec. XI-B), we found that for many applications (*e.g.* for state estimation of micro aerial vehicles [20, 53]) it suffices to only optimize the latest camera pose and the 3D points separately.

VI. Mapping

In the previous section, we assumed that the depth at sparse feature locations in the image is known. In this section, we describe how the mapping thread estimates this depth for newly detected features. Therefore, we assume that the camera poses are known from the motion estimation thread.

The depth at a single pixel is estimated from multiple observations by means of a recursive Bayesian *depth filter*. New depth filters are initialized at intensity corners and along gradient edges when the number of tracked features falls below some threshold and, therefore, a keyframe is selected. Every depth filter is associated to a reference keyframe r, where the initial depth uncertainty is initialized with a large value. For a set of previous keyframes[2] as well as every subsequent frame with known relative pose $\{\mathtt{I}_k, \mathtt{T}_{kr}\}$, we search for a patch along the epipolar line that has the highest correlation (see Fig. 5). Therefore, we move the reference patch along the epipolar line and compute the zero mean sum of squared differences. From the pixel with maximum correlation, we triangulate the depth measurement $\tilde{\rho}_i^k$, which is used to update

[2]In the previous publication of SVO [20] and in the open source implementation we suggested to update the depth filter only with newer frames $k > r$, which works well for down-looking cameras in micro aerial vehicle applications. However, for forward motions, it is beneficial to update the depth filters also with previous frames $k < r$, which increases the performance with forward-facing cameras.

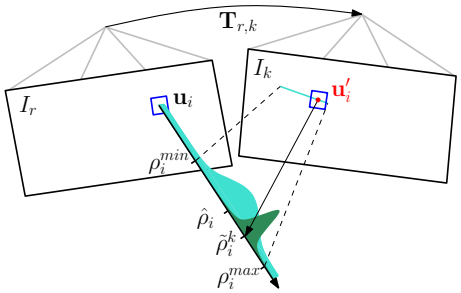


Fig. 5: Probabilistic depth estimate $\hat{\rho}_i$ for feature i in the reference frame r. The point at the true depth projects to similar image regions in both images (blue squares). Thus, the depth estimate is updated with the triangulated depth $\tilde{\rho}_i^k$ computed from the point $\mathbf{u}'_i$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

the depth filter. If enough measurements were obtained such that uncertainty in the depth is below a certain threshold, we initialize a new 3D point at the estimated depth, which subsequently can be used for motion estimation (see system overview in Fig. 1). This approach for depth estimation also works for features on gradient edges. Due to the aperture problem, we however skip measurements where the edge is parallel to the epipolar line.

Ideally, we would like to model the depth with a non-parametric distribution to deal with multiple depth hypotheses (top rows in Fig. 6). However, this is computationally too expensive. Therefore, we model the depth filter according to [54] with a two dimensional distribution: the first dimension is the inverse depth ρ [55], while the second dimension γ is the inlier probability (see bottom rows in Fig. 6). Hence, a measurement $\tilde{\rho}_i^k$ is modeled with a *Gaussian + Uniform* mixture model distribution: an inlier measurement is normally distributed around the true inverse depth ρ_i while an outlier measurement arises from a uniform distribution in the interval $[\rho_i^{\min}, \rho_i^{\max}]$:

$$p(\tilde{\rho}_i^k|\rho_i,\gamma_i) = \gamma_i \mathcal{N}\big(\tilde{\rho}_i^k|\rho_i,\tau_i^2\big) + (1-\gamma_i)\mathcal{U}\big(\tilde{\rho}_i^k|\rho_i^{\min},\rho_i^{\max}\big), \quad (7)$$

where τ_i^2 the variance of a good measurement that can be computed geometrically by assuming a disparity variance of one pixel in the image plane [39].

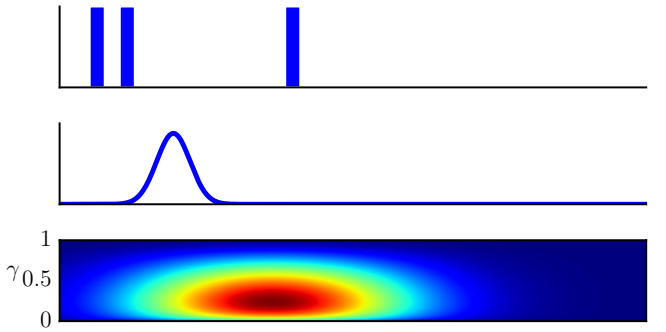
Assuming independent observations, the Bayesian estimation for ρ on the basis of the measurements $\tilde{\rho}_{r+1}, \ldots, \tilde{\rho}_k$ is given by the posterior

$$p(\rho,\gamma|\tilde{\rho}_{r+1},\ldots,\tilde{\rho}_k) \propto p(\rho,\gamma)\prod_k p(\tilde{\rho}_k|\rho,\gamma), \qquad (8)$$

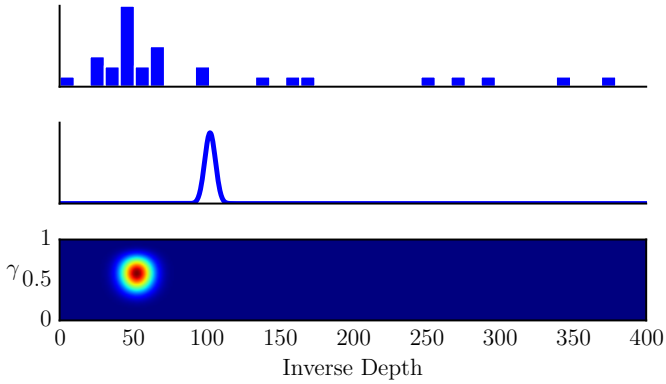
with $p(\rho,\gamma)$ being a prior on the true inverse depth and the ratio of good measurements supporting it. For incremental computation of the posterior, the authors of [54] show that (8) can be approximated by the product of a Gaussian distribution for the depth and a Beta distribution for the inlier ratio:

$$q(\rho,\gamma|a_k,b_k,\mu_k,\sigma_k^2) = Beta(\gamma|a_k,b_k)\mathcal{N}(\rho|\mu_k,\sigma_k^2), \qquad (9)$$

where a_k and b_k are the parameters controlling the Beta distribution. The choice is motivated by the fact that the $Beta \times Gaussian$ is the approximating distribution mini-



(a) After 3 measurements with 70% inlier probability.



(b) After 30 measurements with 70% inlier probability.

Fig. 6: Illustration of posterior distributions for depth estimation. The histogram in the top rows show the measurements affected by outliers. The distribution in the middle rows show the posterior distribution when modeling the depth with a single variate Gaussian distribution. The bottom rows show the posterior distribution of the proposed approach that is using the model from [54]. The distribution is bi-variate and models the inlier probability (vertical axis) together with the inverse depth (horizontal axis).

mizing the Kullback-Leibler divergence from the true posterior (8). Upon the k-th observation, the update takes the form

$$p(\rho,\gamma|\tilde{\rho}_{r+1},\dots,\tilde{\rho}_k) \approx q(d,\gamma|a_{k-1},b_{k-1},\mu_{k-1},\sigma^2_{k-1}) \cdot p(\tilde{\rho}_k|d,\gamma)\cdot \text{const}, \quad (10)$$

and the authors of [54] approximated the true posterior (10) with a $Beta \times Gaussian$ distribution by matching the first and second order moments for $\hat{d}$ and γ. The updates formulas for a_k, b_k, μ_k and σ^2_k are thus derived and we refer to the original work in [54] for the details on the derivation.

Fig. 6 shows a small simulation experiment that highlights the advantage of the model proposed in [54]. The histogram in the top rows show the measurements that are corrupted by 30% outlier measurements. The distribution in the middle rows show the posterior distribution when modeling the depth with a single variate Gaussian distribution as used for instance in [41]. Outlier measurements have a huge influence on the mean of the estimate. The figures in the bottom rows show the posterior distribution of the proposed approach that is using the model from [54] with the inlier probability drawn in the vertical axis. As more measurements are received at the same depth, the inlier probability increases. In this model, the mean of the estimate is less affected by outliers while the inlier probability is informative about the confidence of the estimate. Fig. 7 shows qualitatively the importance of robust depth estimation in self-similar environments, where outlier matches are frequent.

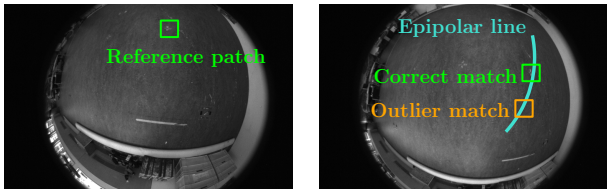


Fig. 7: Illustration of the epipolar search to estimate the depth of the pixel in the center of the reference patch in the left image. Given the extrinsic and intrinsic calibration of the two images, the epipolar line that corresponds to the reference pixel is computed. Due to self-similar texture, erroneous matches along the epipolar line are frequent.

In [39] we demonstrate how the same depth filter can be used for *dense* mapping.

VII. Large Field of View Cameras

To model large optical distortion, such as fisheye and catadioptric (see Fig. 8), we use the camera model proposed in [56], which models the projection $\pi(\cdot)$ and unprojection $\pi^{-1}(\cdot)$ functions with polynomials. Using the Jacobians of the camera distortion in the sparse image alignment and bundle adjustment step is sufficient to enable motion estimation for large FoV cameras.

For estimating the depth of new features (*c.f.,* Sec. VI), we need to sample pixels along the epipolar line. For distorted images, the epipolar line is curved (see Fig. 7). Therefore, we regularly sample the *great circle*, which is the intersection of the epipolar plane with the unit sphere centered at the camera pose of interest. The angular resolution of the sampling corresponds approximately to one pixel in the image plane. For each sample, we apply the camera projection model $\pi(\cdot)$ to obtain the corresponding pixel coordinate on the curved epipolar line.

VIII. Multi-Camera Systems

The proposed semi-direct camera motion estimation starts directly with an optimization of the relative pose T_{kk-1}. Since in Sec. V-A we already introduced a body frame B that is rigidly attached to the camera, it is now straightforward to generalize sparse image alignment to multiple rigidly attached and synchronized cameras. Let us assume that given a camera rig with M cameras (see Fig. 9). The extrinsic calibration of the individual cameras $c \in \mathrm{C}$ with respect to the body frame T_{CB} is assumed to be known from prior extrinsic calibration[3]. To use multiple cameras, we only need to add an extra summation in the cost function of Eq. (1) to use the information from all images for sparse image alignment:

$$\mathsf{T}^{\star}_{kk-1} = \arg\min_{\mathsf{T}_{kk-1}} \sum_{c\in\mathrm{C}} \sum_{\mathbf{u}\in\mathcal{R}^c_{k-1}} \frac{1}{2}\|\mathbf{r}_{\mathrm{I}^c_{\mathbf{u}}}(\mathsf{T}_{kk-1})\|^2_{\Sigma_{\mathrm{I}}}. \quad (11)$$

The same summation is necessary in the bundle adjustment step to sum the reprojection errors from all cameras. The remaining steps of feature alignment and mapping are independent of how many cameras are used. To summarize, the

[3]We use the calibration toolbox *Kalibr* [45], which is available at https://github.com/ethz-asl/kalibr

(a) Perspective (b) Fisheye (c) Catadioptric

Fig. 8: Different optical distortion models that are supported by SVO.

only modification to enable the use of multiple cameras is to refer the optimizations to a central body frame, which requires us to include the extrinsic calibration $\mathtt{T}_{\mathrm{CB}}$ in the Jacobians as shown in the Appendix.

IX. Motion Priors

In feature-poor environments, during rapid motions, or in case of dynamic obstacles it can be very helpful to employ a motion prior. A motion prior is an additional term that is added to the cost function in Eq. (11), which penalizes motions that are not in agreement with the prior estimate. Thereby, "jumps" in the motion estimate due to unconstrained degrees of freedom or outliers can be suppressed. In a car scenario for instance, a constant velocity motion model may be assumed as the inertia of the car prohibits sudden changes from one frame to the next. Other priors may come from additional sensors such as gyroscopes, which allow us to measure the incremental rotation between two frames.

Let us assume that we are given a relative translation prior $\tilde{\mathbf{p}}_{kk-1}$ (*e.g.*, from a constant velocity assumption) and a relative rotation prior $\tilde{\mathtt{R}}_{kk-1}$ (*e.g.*, from integrating a gyroscope). In this case, we can employ a motion prior by adding additional terms to the cost of the sparse image alignment step:

$$\begin{aligned}\mathtt{T}^{\star}_{kk-1} = \arg\min_{\mathtt{T}_{kk-1}} \sum_{c\in\mathrm{C}} \sum_{\mathbf{u}\in\bar{\mathcal{R}}^{c}_{k-1}} \frac{1}{2}\|\mathbf{r}_{\mathrm{I}^{c}_{\mathbf{u}}}(\mathtt{T}_{kk-1})\|^2_{\Sigma_{\mathrm{I}}} \\ + \frac{1}{2}\|\mathbf{p}_{kk-1} - \tilde{\mathbf{p}}_{kk-1}\|^2_{\Sigma_{\mathbf{p}}} \\ + \frac{1}{2}\|\log(\tilde{\mathtt{R}}^{\mathsf{T}}_{kk-1}\mathtt{R}_{kk-1})^{\vee}\|^2_{\Sigma_{\mathtt{R}}},\end{aligned} \tag{12}$$

where the covariances $\Sigma_{\mathbf{p}}, \Sigma_{\mathtt{R}}$ are set according to the uncertainty of the motion prior and the variables $(\mathbf{p}_{kk-1}, \mathtt{R}_{kk-1}) \doteq \mathtt{T}_{kk-1}$ are the current estimate of the relative position and orientation (expressed in body coordinates B). The *logarithm map* maps a rotation matrix to its rotation vector (see Eq. (18)). Note that the same cost function can be added to the bundle adjustment step. For further details on solving Eq. (12), we refer the interested reader to the Appendix.

X. Implementation Details

In this section we provide additional details on various aspects of our implementation.

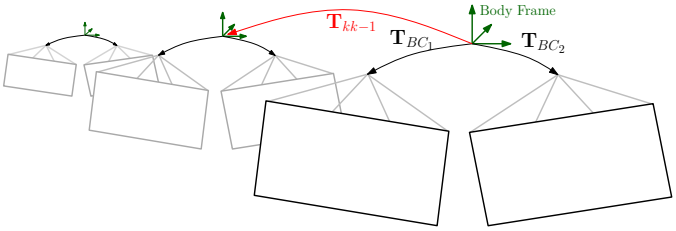


Fig. 9: Visual odometry with multiple rigidly attached and synchronized cameras. We know the relative pose of each camera to the body frame $\mathtt{T}_{\mathrm{BC}_j}$ from extrinsic calibration and the goal is to estimate the relative motion of the body frame $\mathtt{T}_{kk-1}$.

A. Initialization

The algorithm is bootstrapped to obtain the pose of the first two keyframes and the initial map using the 5-point relative pose algorithm from [57].

B. Sparse Image Alignment

For sparse image alignment, we use a patch size of 4×4 pixels. In the experimental section we demonstrate that the sparse approach with such a small patch size achieves comparable performance to semi-dense and dense methods in terms of robustness when the inter-frame distance is small, which typically is true for frame-to-frame motion estimation. In order to cope with large motions, we apply the sparse image alignment algorithm in a coarse-to-fine scheme. Therefore, the image is halfsampled to create an image pyramid of five levels. The photometric cost is then optimized at the coarsest level until convergence, starting from the initial condition $\mathtt{T}_{kk-1} = \mathbf{I}_{4\times4}$. Subsequently, the optimization is continued at the next finer level to improve the precision of the result. To save processing time, we stop after convergence on the third level, at which stage the estimate is accurate enough to initialize feature alignment. To increase the robustness against dynamic obstacles, occlusions and reflections, we additionally employ a robust cost function [24, 34].

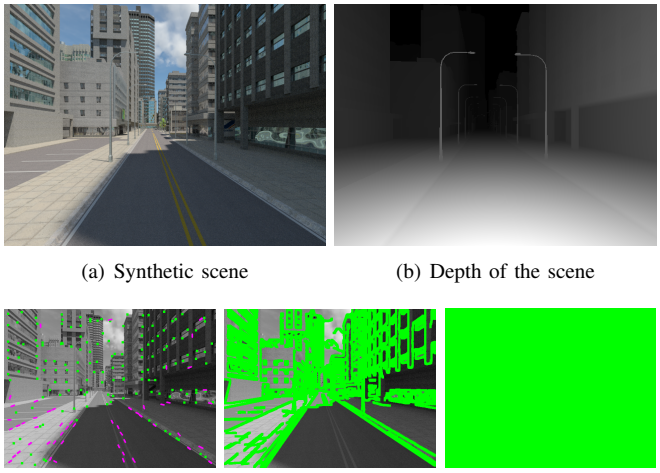
C. Feature Alignment

For feature alignment we use a patch-size of 8×8 pixels. Since the reference patch may be multiple frames old, we employ an affine illumination model to cope with illumination changes [58]. For all experiments we limit the number of matched features to 180 in order to guarantee a constant cost per frame.

D. Mapping

In the mapping thread, we divide the image in cells of fixed size (e.g., 32×32 pixels). For every keyframe a new depth-filter is initialized at the FAST corner [59] with highest score in the cell unless there is already a 2D-to-3D correspondence present. In cells where no corner is found, we detect the pixel with highest gradient magnitude and initialize an edge feature. This results in evenly distributed features in the image.

To speed up the depth-estimation we only sample a short range along the epipolar line; in our case, the range corresponds to twice the standard deviation of the current depth estimate. We use a 8×8 pixel patch size for the epipolar search.



(a) Synthetic scene (b) Depth of the scene

(c) Sparse (d) Semi-Dense (e) Dense

Fig. 10: An image from the *Urban Canyon* dataset [60] (Sec. XI-A) with pixels used for image-to-model alignment (marked in green) for sparse, semi-dense, and dense methods. Dense approaches use every pixel in the image, semi-dense use just the pixels with high intensity gradient, and the proposed sparse approach uses selected pixels at corners or along intensity gradient edges.

XI. Experimental Evaluation

We implemented the proposed VO system in C++ and tested its performance in terms of accuracy, robustness, and computational efficiency. We first compare the proposed sparse image alignment algorithm against semi-dense and dense image alignment algorithms and investigate the influence of the patch size used in the sparse approach. Finally, in Sec. XI-B we compare the full pipeline in different configurations against the state of the art on nine different dataset sequences.

A. Image Alignment: From Sparse to Dense

In this section we evaluate the robustness of the proposed sparse image alignment algorithm (Sec. V-A) and compare its performance to semi-dense and dense image alignment alternatives. Additionally, we investigate the influence of the patch-size that is used for the sparse approach.

The experiment is based on a synthetic dataset with known camera motion, depth and calibration [60].[4] The camera performs a forward motion through an urban canyon as the excerpt of the dataset in Fig. 10a shows. The dataset consists of 2500 frames with 0.2 meters distance between frames and a median scene depth of 12.4 meters. For the experiment, we select a reference image $\mathtt{I}_r$ with known depth (see Fig. 10b) and estimate the relative pose $\mathtt{T}_{rk}$ of 60 subsequent images $k \in \{r+1, \ldots, r+60\}$ along the trajectory by means of image to model alignment. For each image pair $\{\mathtt{I}_r, \mathtt{I}_k\}$, the alignment is repeated 800 times with initial perturbation that is sampled uniformly within a 2 m range around the true value. We perform the experiment at 18 reference frames along the trajectory. The alignment is considered converged when the estimated relative pose is closer than 0.1 meters from the ground-truth. The goal of this experiment is to study the magnitude of the perturbation from which image to model

[4]The *Urban Canyon* dataset [60] is available at http://rpg.ifi.uzh.ch/fov.html

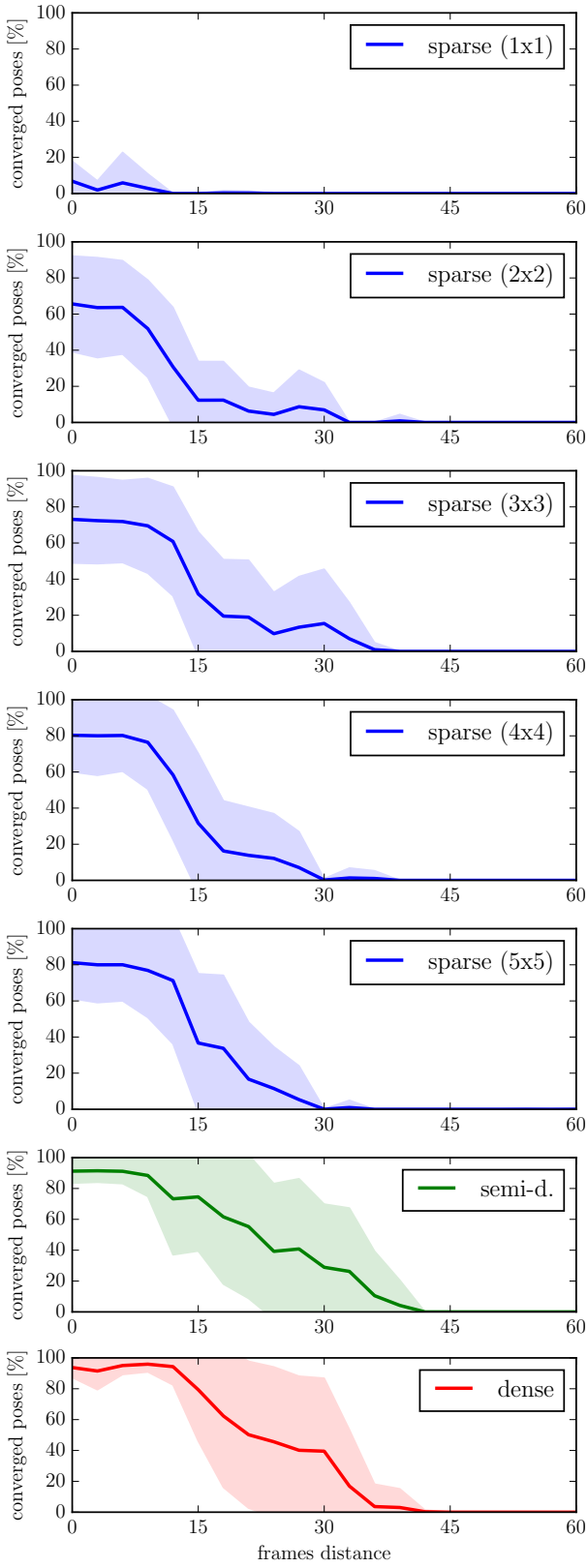


Fig. 11: Convergence probability of the model-based image alignment algorithm as a function of the distance to the reference image and evaluated for sparse image alignment with patch sizes ranging from 1×1 to 5×5 pixels, semi-dense, and dense image alignment. The colored region highlights the 68% confidence interval.

alignment is capable to converge as a function of the distance to the reference image. The performance in this experiment is a measure of robustness: successful pose estimation from large initial perturbations shows that the algorithm is capable of dealing with rapid camera motions. Furthermore, large distances between the reference image $\mathtt{I}_r$ and test image $\mathtt{I}_k$ simulates the performance at low camera frame-rates.

For the sparse image alignment algorithm, we extract 100 FAST corners in the reference image (see Fig. 10c) and initialize the corresponding 3D points using the known depth-map from the rendering process. We repeat the experiment with patch-sizes ranging from 1×1 pixels to 5×5 pixels. We evaluate the semi-direct approach (as proposed in the LSD framework [41]) by using pixels along intensity gradients (see Fig. 10d). Finally, we perform the experiment using all pixels in the reference image as proposed in DTAM [15].

The results of the experiment are shown in Fig. 11. Each plot shows a variant of the image alignment algorithm with the vertical axis indicating the percentage of converged trials and the horizontal axis the frame index counted from the reference frame. We can observe that the difference between semi-dense image alignment and dense image alignment is marginal. This is because pixels that exhibit no intensity gradient are not informative for the optimization as their Jacobians are zero [40]. We suspect that using all pixels becomes only useful when considering motion blur and image defocus, which is out of the scope of this evaluation. In terms of sparse image alignment, we observe a gradual improvement when increasing the patch size to 4×4 pixels. A further increase of the patch size does not show improved convergence and will eventually suffer from the approximations adopted by not warping the patches according to the surface orientation.

Compared to the semi-dense approach, the sparse approaches do not reach the same convergence radius, particularly in terms of distance to the reference image. For this reason SVO uses sparse image alignment only to align with respect to the previous image (*i.e.,* $k = r + 1$), in contrast to LSD [41] which aligns with respect to the last keyframe.

In terms of computational efficiency, we note that the complexity scales linearly with the number of pixels used in the optimization. The plots show that we can trade-off using a high frame rate camera and a sparse approach with a lower frame-rate camera and a semi-dense approach. The evaluation of this trade-off would ideally incorporate the power consumption of both the camera and processors, which is out of the scope of this evaluation.

B. Real and Synthetic Experiments

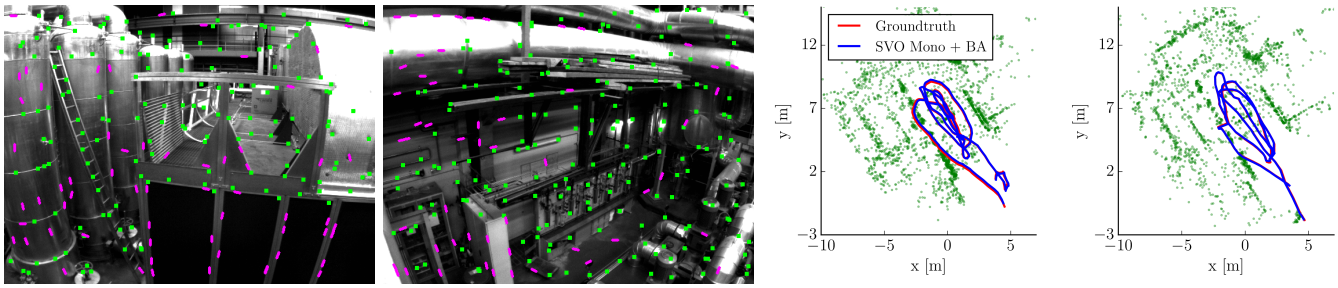
In this section, we compare the proposed algorithm against the state of the art on real and synthetic datasets. Therefore, we present results of the proposed pipeline on the EUROC benchmark [61], the TUM RGB-D benchmark dataset [62], the synthetic ICL-NUIM dataset [37], and our own dataset that compares different field of view cameras. A selection of these experiments, among others (*e.g.,* from the KITTI benchmark), can also be viewed in the video attachment of this paper.

1) Euroc Datasets: The EUROC dataset [61] consists of stereo images and inertial data that was recorded using a VI-Sensor [63] that was mounted on a micro aerial vehicle and flown inside a machine hall. Extracts from the dataset are shown in Fig. 12a and 12b. The dataset provides a precise ground-truth trajectory that was obtained using a Leica MS50 laser tracking system. In Table I we present results of various monocular and stereo configurations of the proposed algorithm on the first two trajectories of the dataset. The trajectories are 65 and 58 meters long respectively.

We compare our algorithm against the open-source versions of ORB-SLAM [22] and LSD-SLAM [16] on these datasets with their default parameter settings. In contrast to SVO, ORB-SLAM and LSD-SLAM detect loop-closures and subsequently perform a global optimization. Hence, during loop-closure refinements, the latest camera pose may undergo significant "jumps". For this reason, we base the evaluation on the final poses of all keyframes in the trajectory. We observed that the initial poses of LSD-SLAM are not accurate due to the initialization procedure and therefore discard the first 350 frames in the evaluation for LSD-SLAM.

To understand the influence of the proposed extensions of SVO, we run the algorithm in various configurations. "SVO Mono" (only corners) uses only the images that were recorded with the left camera of the sensor. "SVO Mono + Prior" indicates that we use measurements from the gyroscope as priors in the image alignment step as we discussed in Sec. IX. In the next setting we additionally use edgelet features in combination with corner features. In these first three settings, we only optimize the latest pose; conversely, the keyword "Bundle Adjustment" indicates that results were obtained by optimizing the whole history of keyframes by means of the incremental smoothing algorithm iSAM2 [9]. Therefore, we insert and optimize every new keyframe in the iSAM2 graph when a new keyframe is selected. In this setting, we do neither use motion priors nor edgelets. Since SVO is a visual odometry, it does not not detect loop-closures and only maintains a small local map of the last keyframes. To provide a fair comparison with ORB-SLAM and LSD-SLAM, we deactivated their capability to detect large loop closures via image retrieval. Additionally, we provide results using both image streams of the stereo camera. Therefore, we apply the approach introduced in Sec. VIII to estimate the motion of a multi-camera system.

To obtain a measure of accuracy of the different approaches, we align the final trajectory of keyframes with the ground-truth trajectory using the least-squares approach proposed in [64]. Since scale cannot be recovered using a single camera, we also rescale the estimated trajectory to best fit with the ground-truth trajectory. Subsequently, we compute the Euclidean distance between the estimated and ground-truth keyframe poses and compute the mean, median, and Root Mean Square Error (RMSE) in meters. We chose the absolute trajectory error measure instead of relative drift metrics [62] because the final trajectory in ORB-SLAM consists only of a sparse set of keyframes, which makes drift measures on relatively short trajectories less expressive. The reported results are averaged over three runs.



(a) (b) (c) Machine Hall 1 (d) Machine Hall 2

Fig. 12: Figures (a) and (b) show excerpts of the EuRoC dataset [61] with tracked corners marked in green and edgelets marked in magenta. Figures (c) and (d) show the reconstructed trajectory and pointcloud on the first two trajectories of the dataset.

	Machine Hall 1			Machine Hall 2			Timing		
	Mean	Median	RMSE [m]	Mean	Median	RMSE [m]	Mean [ms]	St.D.	CPU@20 fps
SVO Mono	0.224	0.198	0.269	0.531	0.356	0.652	2.53	0.42	55 ±10 %
SVO Mono + Prior	0.199	0.131	0.270	0.345	0.314	0.408	**2.32**	**0.40**	**70 ± 8** %
SVO Mono + Prior + Edgelet	0.171	0.149	0.201	0.368	0.318	0.425	2.51	0.52	73 ± 7 %
SVO Mono + Bundle Adjustment	**0.048**	**0.042**	**0.057**	**0.061**	**0.060**	**0.072**	5.25	10.89	72 ±13 %
SVO Stereo	0.096	0.092	0.104	0.064	0.063	0.070	4.70	1.31	90 ± 6 %
SVO Stereo + Prior	0.071	0.066	0.078	0.067	0.059	0.072	**3.86**	**0.86**	**90 ± 7** %
SVO Stereo + Prior + Edgelet	0.072	0.060	0.083	0.072	0.062	0.077	4.12	1.11	91 ± 7 %
SVO Stereo + Bundle Adjustment	**0.039**	**0.037**	**0.043**	**0.046**	**0.042**	**0.053**	7.61	19.03	96 ±13 %
ORB Mono SLAM (No loop closure)	0.105	0.126	0.114	0.175	0.209	0.190	29.81	5.67	187 ±32 %
LSD Mono SLAM (No loop closure)	0.111	0.107	0.125	0.388	0.357	0.428	23.23	5.87	236 ±37 %

TABLE I: Absolute translation errors in meters after 6 DoF alignment with the ground-truth trajectory and timing measurements on laptop computer with Intel Core i7-4810MQ CPU (2.80 GHz) averaged over three runs of the EUROC Machine Hall 01 dataset. Loop closure detection and optimization was deactived for ORB and LSD SLAM to allow a fair comparison with SVO. The first and second column report mean and standard devitation of the processing time. Since all algorithms use multi-threading, the third column reports the average CPU load when providing new images at a constant rate of 20 Hz.

	Thread	Intel i7 [ms]	Jetson TX1 [ms]
Sparse image alignment	1	0.66	2.54
Feature alignment	1	1.04	1.40
Optimize pose & landmarks	1	0.42	0.88
Extract features	2	1.64	5.48
Update depth filters	2	1.80	2.97

TABLE II: Mean time consumption in milliseconds by individual components of SVO Mono on the EUROC Machine Hall 1 dataset. We report timing results on a laptop with Intel Core i7 (2.80 GHz) processor and on the NVIDIA Jetson TX1 ARM processor.

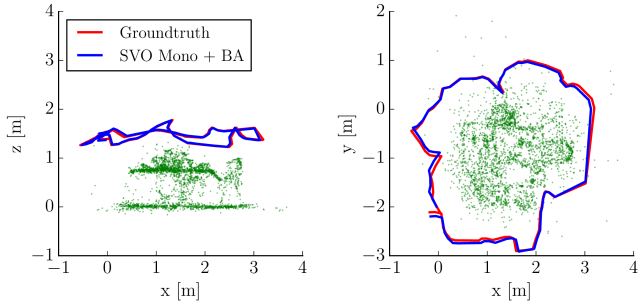
The results show that using a stereo camera in general results in much higher accuracy. Apart from the additional visual measurements, the main reason for the improved results is that the stereo system does not drift in scale and inter camera triangulations allow to quickly initialize new 3D landmarks in case of on-spot rotations. Notice that SVO with bundle adjustment is twice as accurate as ORB and LSD SLAM.

However, the power of the less accurate configurations of SVO becomes obvious when analyzing the timing and processor usage of the approaches, which are reported in the right columns of Table I. In the table, we report the mean time to process a single frame in milliseconds and the standard deviation over all measurements. Since all algorithms make use of multi-threading and the time to process a single frame may therefore be misleading, we additionally report the CPU usage (continuously sampled during execution) when providing new images at a constant rate of 20 Hz to the algorithm. All measurements are averaged over 3 runs of the first EUROC dataset and computed on the same laptop computer (Intel Core i7-2760QM CPU). In Table II, we further report the average time consumption of individual components of SVO on the laptop computer and an NVIDIA TX1 ARM processor, which is popular in mobile robotics applications. The results show that the SVO approach is up to ten times faster than ORB-SLAM and LSD-SLAM and requires only a fourth of the CPU usage. The reason for this significant difference is that SVO does not extract features and descriptors in every frame, as in ORB-SLAM, but does so only for keyframes in the concurrent mapping thread. Additionally, ORB-SLAM—being a SLAM approach—spends most of the processing time in finding matches to the map (see Table I in [65]), which in theory results in a pose-estimate without drift in an already mapped area. Contrarily, in the first three configurations of SVO, we estimate only the pose of the latest camera frame with respect to the local map. Compared to LSD-SLAM, SVO is faster because it operates on significantly less numbers of pixels, hence, also does not result in a semi-dense reconstruction of the environment. This, however, could be achieved in a parallel process as we have shown in [39, 53, 66]. We further remark that the bundle adjustment version has a significantly higher standard deviation in the timing as we update iSAM2 at every keyframe, which takes approximately 10 milliseconds longer than processing a regular frame. Using a motion prior further helps to improve the efficiency as the sparse-image-alignment optimization can be initialized closer to the solution, and therefore needs less iterations to converge.

An edgelet provides only a one-dimensional constraint in the image domain, while a corner provides a two-dimensional

(a) fr2 desk (b) fr2 xyz

Fig. 13: Impressions from the TUM RGB-D benchmark dataset [62] with tracked corners in green and edgelets in magenta.



(a) fr2_desk (side) (b) fr2_desk (top)

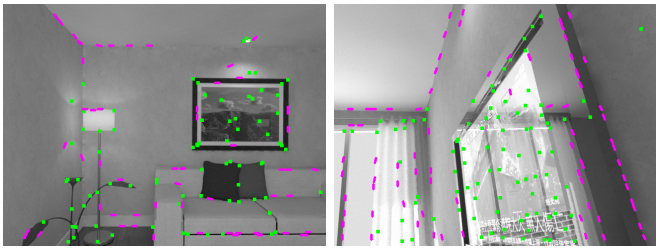
Fig. 14: Estimated trajectory and pointcloud of the TUM "fr2_desk" dataset.

	fr2_desk RMSE [cm]	fr2_xyz RMSE [cm]
SVO Mono (with edgelets)	9.7	1.1
SVO Mono + Bundle Adjustment	6.7	0.8
LSD-SLAM [16] ◯	4.5	1.5
ORB-SLAM [22] ◯	**0.9**	0.3
PTAM [21]	× / ×	**0.2** / 24.3
Semi-Dense VO [41]	13.5	3.8
Direct RGB-D VO [34] ★	1.8	1.2
Feature-based RGB-D SLAM [67] ★ ◯	9.5	2.6

TABLE III: Results on the TUM RGB-D benchmark dataset [62]. Results for [16, 34, 41, 67] were obtained from [16] and for PTAM we report two results that were published in [22] and [41] respectively. Algorithms marked with ★ use a depth-sensor, and ◯ indicates loop-closure detection. The symbol × indicates that tracking the whole trajectory did not succeed.

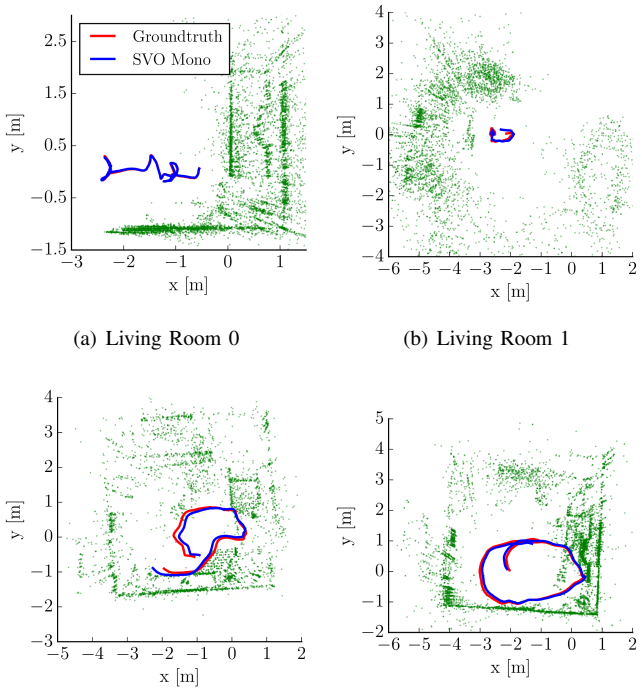
constraint. Therefore, whenever sufficient corners can be detected, the SVO algorithm prioritizes the corners. Since the environment in the EUROC dataset is well textured and provides many corners, the use of edgelets does not improve the accuracy. However, the edgelets bring a benefit in terms of robustness when the texture is such that no corners are present.

2) TUM Datasets: A common dataset to evaluate visual odometry algorithms is the TUM Munich RGB-D benchmark [62]. The dataset was recorded with a Microsoft Kinect RGB-D camera, which provides images of worse quality (e.g. rolling shutter, motion blur) than the VI-Sensor EUROC dataset. Fig. 13 shows excerpts from the "`fr2_desk`" and "`fr2_xyz`" datasets which have a trajectory length of 18.8 m and 7 m respectively. Groundtruth is provided by a motion capture system. Table III shows the results of the proposed algorithm (averaged over three runs) and comparisons against related works. The resulting trajectory and the recovered landmarks are shown in Fig. 14. The results from related works



(a) (b)

Fig. 15: Impressions from the synthetic ICL-NUIM dataset [37] with tracked corners marked in green and edgelets in magenta.



(a) Living Room 0 (b) Living Room 1

(c) Living Room 2 (d) Living Room 3

Fig. 16: Results on the ICL-NUIM [37] noisy synthetic living room dataset.

	lr_kt0 RMSE	lr_kt1 RMSE	lr_kt2 RMSE	lr_kt3 RMSE
SVO Mono (with edgelets)	**0.02**	0.07	0.1	**0.07**
LSD SLAM	×	×	×	×
ORB SLAM [22] ◯	×	×	0.03	0.12
DVO [34] ★	0.29	0.12	0.47	0.54
FOVIS [68] ★	2.05	1.87	1.49	1.47
ICP [69] ★	0.07	**0.005**	**0.01**	0.36
ICP+RGB-D [70] ★	0.39	0.021	0.12	0.86

TABLE IV: Results on the ICL-NUIM Dataset [37]. Algorithms marked with ★ use a depth-sensor, and ◯ indicates loop-closure detection. The symbol × indicates that tracking the whole trajectory did not succeed.

were obtained from the evaluation in [22] and [16]. We argue that the better performance of ORB-SLAM and LSD-SLAM is due to the capability to detect loop-closures.

3) ICL-NUIM Datasets: The ICL-NUIM dataset [37] is a synthetic dataset that aims to benchmark RGB-D, visual odometry and SLAM algorithms. The dataset consists of four trajectories of length 6.4 m, 1.9 m, 7.3 m, and 11.1 m. The synthesized images are corrupted by noise to simulate real

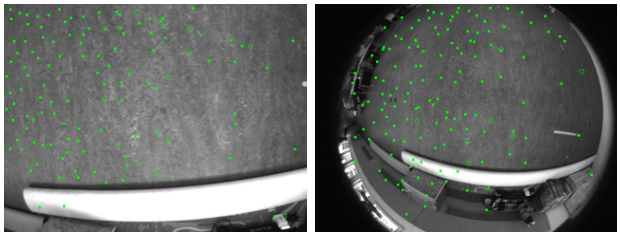
camera images. Ground-truth and calibration are provided by the dataset. Most reported results on this dataset use the synthetsized measurements from the depth sensor together with the rendered images. Indeed, the datsets are very challenging for purely vision-based odometry due to difficult texture and frequent on-spot rotations as can be seen in the excerpts from the dataset in Fig. 15.

Table IV reports the results of the proposed algorithm (averaged over three runs) and the results from other algorithms on the "living room" sequence. Similar to the previous datasets, we report the root mean square error after rotation, translation and scale alignment with the ground-truth trajectory. Fig. 16 shows the reconstructed maps and recovered trajectories. The maps are very noisy due to the fine grained texture of the scene. We also run ORB-SLAM and LSD-SLAM on the dataset. ORB-SLAM fails to initialize on the second sequence and we did not manage to obtain any results from LSD-SLAM on all datasets. The reason for the failures is lack of texture for initialization and frequent on-spot rotations. For SVO, this also required us to set a particularly low FAST corner detection threshold on this dataset (to 5 instead of 20). A lower threshold results in detection of many low-quality features. However, features are only used in SVO once their corresponding scene depth is sucessfully estimated by means of the robust depth filter described in Sec. VI. Hence, the process of depth estimation helps to identify the stable features with low score that can be reliably used for motion estimation.

In this dataset, we were not able to refine the results of SVO with bundle adjustment. The reason is that the iSAM2 backend is based on Gauss Newton which is very sensitive to underconstrained variables that render the linearized problem indeterminant. The frequent on-spot rotations and very low parallax angle triangulations result in many underconstrained variables. Using an optimizer that is based on Levenberg Marquardt or adding additional inertial measurements [44] would help in such cases.

For comparison, Table IV also shows the results reported in [37] of algorithms that also use the depth measurements.

4) Circle Dataset: In the last experiment, we want to demonstrate the usefulness of wide field of view lenses for VO. We recorded the dataset with a micro aerial vehicle that we flew in a motion capture room and commanded it to fly a perfect circle with downfacing camera. Subsequently, we flew the exact same trajectory again with a wide fisheye camera. Excerpts from the dataset are shown in Fig. 17. We run SVO (without bundle adjustment) on both datasets and show the resulting trajectories in Fig. 18. To run SVO on the fisheye images, we use the modifications described in Sec. VII. While the recovered trajectory from the perspective camera slowly drifts over time, the result on the fisheye camera perfectly overlaps with the groundtruth trajectory. We also run ORB-SLAM and LSD-SLAM on the trajectory with the perspective images. The result of ORB-SLAM is as close to the ground-truth trajectory as the SVO fisheye result. However, if we deactivate loop-closure detection (shown result) the trajectory drifts more than SVO. We were not able to run LSD-SLAM and ORB-SLAM on the fisheye images as the open source implementations do not support very large FoV cameras. Due to the difficult high-frequency texture of the floor, we were not able to initialize LSD-SLAM on this dataset. A more in-depth evaluation of the benefit of large FoV cameras for SVO is provided in [60].



(a) Perspective. (b) Fisheye.

Fig. 17: SVO tracking with (a) perspective and (b) fisheye camera lens.

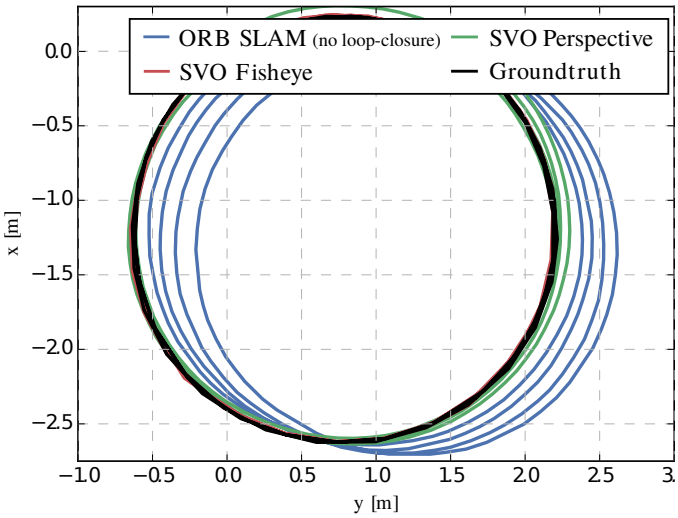


Fig. 18: Comparison of perspective and fisheye lense on the same circular trajectory that was recorded with a micro aerial vehicle in a motion capture room. The ORB-SLAM result was obtained with the perspective camera images and loop-closure was deactivated for a fair comparison with SVO. ORB-SLAM with a perspective camera and with loop-closure activated performs as good as SVO with a fisheye camera.

XII. Discussion

In this section we discuss the proposed SVO algorithm in terms of efficiency, accuracy, and robustness.

A. Efficiency

Feature-based algorithms incur a constant cost of feature and descriptor extraction per frame. For example, ORB-SLAM requires 11 milliseconds per frame for ORB feature extraction only [22]. This constant cost per frame is a bottleneck for feature-based VO algorithms. On the contrary, SVO does not have this constant cost per frame and benefits greatly from the use of high frame-rate cameras. SVO extracts features only for selected keyframes in a parallel thread, thus, decoupled from hard real-time constraints. The proposed tracking algorithm, on the other hand, benefits from high frame-rate cameras: the sparse image alignment step is automatically initialized closer to the solution and, thus, converges faster. Therefore,

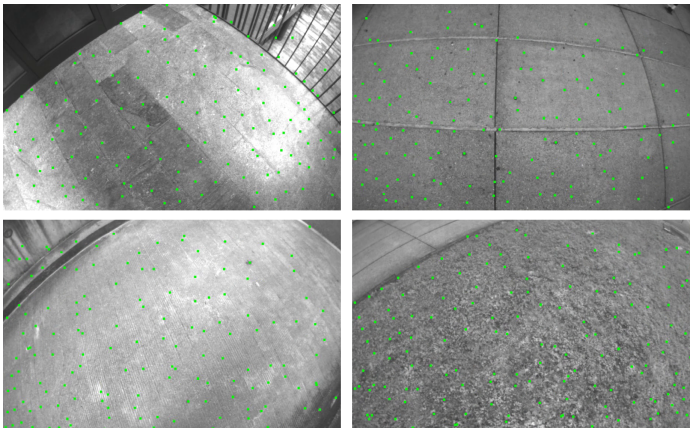


Fig. 19: Successful tracking in scenes of high-frequency texture.

increasing the camera frame-rate actually reduces the computational cost per frame in SVO. The same principle applies to LSD-SLAM. However, LSD-SLAM tracks significantly more pixels than SVO and is, therefore, up to an order of magnitude slower. To summarize, on a laptop computer with an Intel i7 2.8 GHz CPU processor ORB-SLAM and LSD-SLAM require approximately 30 and 23 milliseconds respectively per frame while SVO requires only 2.5 milliseconds (see Table I).

B. Accuracy

SVO computes feature correspondence with sub-pixel accuracy using direct feature alignment. Subsequently, we optimize both structure and motion to minimize the reprojection errors (see Sec. V-B). We use SVO in two settings: if highest accuracy is not necessary, such as for motion estimation of micro aerial vehicles [53], we only perform the refinement step (Sec. V-B) for the latest camera pose, which results in the highest frame-rates (*i.e.*, 2.5 ms). If highest accuracy is required, we use iSAM2 [9] to jointly optimize structure and motion of the whole trajectory. iSAM2 is an incremental smoothing algorithm, which leverages the expressiveness of factor graphs [8] to maintain sparsity and to identify and update only the typically small subset of variables affected by a new measurement. In an odometry setting, this allows iSAM2 to achieve the same accuracy as batch estimation of the whole trajectory, while preserving real-time capability. Bundle adjustment with iSAM2 is consistent [44], which means that the estimated covariance of the estimate matches the estimation errors (*e.g.*, are not over-confident). Consistency is a prerequisite for optimal fusion with additional sensors [18]. In [44], we therefore show how SVO can be fused with inertial measurements to achieve a drift that is approximately 0.1% of the traveled distance. LSD-SLAM, on the other hand, only optimizes a graph of poses and leaves the structure fixed once computed (up to a scale). The optimization does not capture correlations between the semi-dense depth estimates and the camera pose estimates. This separation of depth estimation and pose optimization is only optimal if each step yields the optimal solution.

C. Robustness

SVO is most robust when a high frame-rate camera is used (*e.g.*, between 40 and 80 frames per second). This increases the resilience to fast motions as it is demonstrated in the video attachment. A fast camera, together with the proposed robust depth estimation, allows us to track the camera in environments with repetitive and high frequency texture (*e.g.*, grass or asphalt as shown in Fig. 19). The advantage of the proposed probabilistic depth estimation method over the standard approach of triangulating points from two views only is that we observe far fewer outliers as every depth filter undergoes many measurements until convergence. Furthermore, erroneous measurements are explicitly modeled, which allows the depth to converge in highly self-similar environments.

A further advantage of SVO is that the algorithm starts directly with an optimization. Data association in sparse image alignment is directly given by geometry of the problem and therefore, no RANSAC [23] is required as it is typical in feature-based approaches. Starting directly with an optimization also simplifies greatly the use of multi-camera systems, which greatly improves resilience to on-spot rotations as the field of view of the system is enlarged and depth can be triangulated from inter-camera-rig measurements.

Finally, the use of gradient edge features (*i.e.*, edgelets) increases the robustness in areas where only few corner features are found. Our simulation experiments have shown that the proposed sparse image alignment approach achieves comparable performance as semi-dense and dense alignment in terms of robustness of frame-to-frame motion estimation.

XIII. Conclusion

In this paper, we proposed the semi-direct VO pipeline "SVO" that is significantly faster than the current state-of-the-art VO algorithms while achieving highly competitive accuracy. The gain in speed is due to the fact that features are only extracted for selected keyframes in a parallel thread and feature matches are established very fast and robustly with the novel sparse image alignment algorithm. Sparse image alignment tracks a set of features jointly under epipolar constrains and can be used instead of KLT-tracking [71] when the scene depth at the feature positions is known. We further propose to estimate the scene depth using a robust filter that explicitly models outlier measurements. Robust depth estimation and direct tracking allows us to track very weak corner features and edgelets. A further benefit of SVO is that it directly starts with an optimization, which allows us to easily integrate measurements from multiple cameras as well as motion priors. The formulation further allows using large FoV cameras with fisheye and catadioptric lenses. The SVO algorithm has further proven successful in real-world applications such as vision-based flight of quadrotors [53] or 3D scanning applications with smartphones.

Acknowledgments The authors gratefully acknowledge Henri Rebecq for creating the "Urban Canyon" datasets that can be accessed here: http://rpg.ifi.uzh.ch/fov.html

APPENDIX

In this section, we derive the analytic solution to the multi-camera sparse-image-alignment problem with motion prior.

Given a rig of M calibrated cameras $c \in \mathrm{C}$ with known extrinsic calibration $\mathtt{T}_{\mathrm{CB}}$, the goal is to estimate the incremental body motion $\mathtt{T}_{\mathrm{BB}-1}$ by minimizing the intensity residual $\mathbf{r}_{\mathtt{I}_i^{\mathrm{C}}}$ of corresponding pixels in subsequent images. Corresponding pixels are found by means of projecting a known point on the scene surface $\boldsymbol{\rho}_i \doteq {}_{\mathrm{B}-1}\boldsymbol{\rho}_i$ (prefix $\mathrm{B}-1$ denotes that the point is expressed in the previous frame of reference) into images of camera C that were recorded at poses k and $k-1$, which are denoted $\mathtt{I}_k^{\mathrm{C}}$ and $\mathtt{I}_{k-1}^{\mathrm{C}}$ respectively. To improve the convergence properties of the optimization (see Sec. XI-A), we accumulate the intensity residual errors in small patches $\mathcal{P}$ centered at the pixels where the 3d points project. Therefore, we use the iterator variable $\Delta\mathbf{u}$ to sum the intensities over a small patch $\mathcal{P}$. We further assume that a prior of the incremental body motion $\tilde{\mathtt{T}}_{kk-1} \doteq (\tilde{\mathtt{R}}, \tilde{\mathbf{p}})$ is given. The goal is to find the incremental camera rotation and translation $\mathtt{T}_{kk-1} \doteq (\mathtt{R}, \mathbf{p})$ that minimizes the sum of squared errors:

$$(\mathtt{R}^\star, \mathbf{p}^\star) = \arg\min_{(\mathtt{R},\mathbf{p})} C(\mathtt{R}, \mathbf{p}), \quad \text{with} \tag{13}$$

$$C(\mathtt{R}, \mathbf{p}) = \sum_{\mathrm{c}\in\mathrm{C}} \sum_{i=1}^{N} \sum_{\Delta\mathbf{u}\in\mathcal{P}} \frac{1}{2}\|\mathbf{r}_{\mathtt{I}^{\mathrm{C}}_{i,\Delta\mathbf{u}}}\|^2_{\Sigma_{\mathtt{I}}} + \frac{1}{2}\|\mathbf{r}_{\mathtt{R}}\|^2_{\Sigma_{\mathtt{R}}} + \frac{1}{2}\|\mathbf{r}_{\mathbf{p}}\|^2_{\Sigma_{\mathbf{p}}},$$

where N is the number of visible 3D points. We have furder defined the image intensity and prior residuals as:

$$\begin{aligned}
\mathbf{r}_{\mathtt{I}^{\mathrm{C}}_{i,\Delta\mathbf{u}}} &\doteq \mathtt{I}^{\mathrm{C}}_k\big(\pi(\mathtt{T}_{\mathrm{CB}}(\mathtt{R}\boldsymbol{\rho}_i + \mathbf{p})) + \Delta\mathbf{u}\big) - \mathtt{I}^{\mathrm{C}}_{k-1}\big(\pi(\mathtt{T}_{\mathrm{CB}}\ \boldsymbol{\rho}_i) + \Delta\mathbf{u}\big) \\
\mathbf{r}_{\mathtt{R}} &\doteq \log(\tilde{\mathtt{R}}^{\mathsf{T}}\mathtt{R})^\vee \\
\mathbf{r}_{\mathbf{p}} &\doteq \mathbf{p} - \tilde{\mathbf{p}}
\end{aligned} \tag{14}$$

For readability, we write the cost function in matrix form

$$C(\mathtt{R}, \mathbf{p}) = \mathbf{r}(\mathtt{R}, \mathbf{p})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\mathbf{r}(\mathtt{R}, \mathbf{p}), \tag{15}$$

where $\boldsymbol{\Sigma}$ is a block-diagonal matrix composed of the measurement covariances. Since the residuals are non-linear in $(\mathtt{R}, \mathbf{p})$, we solve the optimization problem in an iterative Gauss-Newton procedure [72]. Therefore, we substitute the following perturbations in the cost function:

$$\mathtt{R} \ \leftarrow \ \mathtt{R}\ \exp(\delta\boldsymbol{\phi}^\wedge), \qquad \mathbf{p} \ \leftarrow \ \mathbf{p} + \mathtt{R}\delta\mathbf{p}, \tag{16}$$

where the hat operator $(.)^\wedge$ forms a 3×3 skew-symmetric matrix from a vector in $\mathbb{R}^3$.

As it is common practice for optimizations involving rotations [44, 72], we use the exponential map $\exp(\cdot)$ to perturb the rotation in the tangent space of $\mathrm{SO}(3)$ which avoids singularities and provides a minimal parametrization of the rotation increment. The *exponential map* (at the identity) $\exp : \mathfrak{so}(3) \to \mathrm{SO}(3)$ associates a 3×3 skew-symmetric matrix to a rotation and coincides with the standard matrix exponential (Rodrigues' formula):

$$\exp(\boldsymbol{\phi}^\wedge) = \mathbf{I} + \frac{\sin(\|\boldsymbol{\phi}\|)}{\|\boldsymbol{\phi}\|}\boldsymbol{\phi}^\wedge + \frac{1-\cos(\|\boldsymbol{\phi}\|)}{\|\boldsymbol{\phi}\|^2}\left(\boldsymbol{\phi}^\wedge\right)^2. \tag{17}$$

The inverse relation is the *logarithm map* (at the identity), which associates $\mathtt{R} \in \mathrm{SO}(3)$ to a skew symmetric matrix:

$$\log(\mathtt{R}) = \frac{\varphi\cdot(\mathtt{R}-\mathtt{R}^{\mathsf{T}})}{2\sin(\varphi)} \text{ with } \varphi = \cos^{-1}\left(\frac{\mathrm{tr}\,(\mathtt{R})-1}{2}\right). \tag{18}$$

Note that $\log(\mathtt{R})^\vee = \mathbf{a}\varphi$, where $\mathbf{a}$ and φ are the rotation axis and the rotation angle of $\mathtt{R}$, respectively.

Substituting the perturbations makes the residual errors a function defined on a vector space. This allows us to linearize the quadratic cost at the current estimate, form the *normal equations*, and solve them for the optimal perturbations:

$$\mathbf{J}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\mathbf{J}\ [\delta\boldsymbol{\phi}^{\mathsf{T}}\ \delta\mathbf{p}^{\mathsf{T}}]^{\mathsf{T}} = -\mathbf{J}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\mathbf{r}(\mathtt{R}, \mathbf{p}), \tag{19}$$

where we introduced the variable $\mathbf{J}$, which stacks all Jacobian matrices from the linearization. The solution is subsequently used to update our estimate in $(\mathtt{R}, \mathbf{p})$ according to (16). This procedure is repeated until the norm of the update vectors is sufficiently small, which indicates convergence.

In the following, we show how to linearize the residuals to obtain the Jacobians. Therefore, we substitute the perturbations in the residuals and expand:

$$\begin{aligned}
&\mathbf{r}_{\mathtt{R}}(\mathtt{R}\ \exp(\delta\boldsymbol{\phi}^\wedge)) \\
&\quad = \log(\tilde{\mathtt{R}}^{\mathsf{T}}\mathtt{R}\ \exp(\delta\boldsymbol{\phi}^\wedge))^\vee \overset{(a)}{\simeq} \mathbf{r}_{\mathtt{R}}(\mathtt{R}) + \mathtt{J}_r^{-1}(\log(\tilde{\mathtt{R}}^{\mathsf{T}}\mathtt{R})^\vee)\delta\boldsymbol{\phi}
\end{aligned} \tag{20}$$

$$\begin{aligned}
&\mathbf{r}_{\mathbf{p}}(\mathbf{p} + \mathtt{R}\delta\mathbf{p}) \\
&\quad = (\mathbf{p} + \mathtt{R}\delta\mathbf{p}) - \tilde{\mathbf{p}} = \mathbf{r}_{\mathbf{p}}(\mathbf{p}) + \mathtt{R}\delta\mathbf{p}
\end{aligned} \tag{21}$$

$$\begin{aligned}
&\mathbf{r}_{\mathtt{I}^{\mathrm{C}}_i}(\mathtt{R}\ \exp(\delta\boldsymbol{\phi}^\wedge)) \\
&\quad = \mathtt{I}^{\mathrm{C}}_k\big(\pi(\mathtt{T}_{\mathrm{CB}}(\mathtt{R}\ \exp(\delta\boldsymbol{\phi}^\wedge)\boldsymbol{\rho}_i + \mathbf{p}))\big) - \mathtt{I}^{\mathrm{C}}_{k-1}\big(\pi(\mathtt{T}_{\mathrm{CB}}\ \boldsymbol{\rho}_i)\big) \\
&\quad \overset{(b)}{\simeq} \mathtt{I}^{\mathrm{C}}_k\big(\pi(\mathtt{T}_{\mathrm{CB}}(\mathtt{R}\boldsymbol{\rho}_i + \mathbf{p}))\big) - \mathtt{I}^{\mathrm{C}}_{k-1}\big(\pi(\mathtt{T}_{\mathrm{CB}}\exp(\delta\boldsymbol{\phi}^\wedge)^{-1}\boldsymbol{\rho}_i)\big) \\
&\quad \overset{(c)}{\simeq} \mathtt{I}^{\mathrm{C}}_k\big(\pi(\mathtt{T}_{\mathrm{CB}}(\mathtt{R}\boldsymbol{\rho}_i + \mathbf{p}))\big) - \mathtt{I}^{\mathrm{C}}_{k-1}\big(\pi(\mathtt{T}_{\mathrm{CB}}(\mathbf{I} - \delta\boldsymbol{\phi}^\wedge)\boldsymbol{\rho}_i)\big) \\
&\quad \overset{(d)}{=} \mathtt{I}^{\mathrm{C}}_k\big(\pi(\mathtt{T}_{\mathrm{CB}}(\mathtt{R}\boldsymbol{\rho}_i + \mathbf{p}))\big) - \mathtt{I}^{\mathrm{C}}_{k-1}\big(\pi(\mathtt{T}_{\mathrm{CB}}\boldsymbol{\rho}_i + \mathtt{T}_{\mathrm{CB}}\boldsymbol{\rho}_i^\wedge\delta\boldsymbol{\phi})\big) \\
&\quad \overset{(e)}{\simeq} \mathbf{r}_{\mathtt{I}^{\mathrm{C}}_i}(\mathtt{R}) - \frac{\partial\mathtt{I}^{\mathrm{C}}_{k-1}(\mathbf{u})}{\partial\mathbf{u}}\Big|_{\mathbf{u}=\pi({}_c\boldsymbol{\rho}_i)}\frac{\partial\pi(\boldsymbol{\rho})}{\partial\boldsymbol{\rho}}\Big|_{\boldsymbol{\rho}={}_c\boldsymbol{\rho}_i}\mathtt{R}_{\mathrm{CB}}\boldsymbol{\rho}_i^\wedge\delta\boldsymbol{\phi}
\end{aligned} \tag{22}$$

$$\begin{aligned}
&\mathbf{r}_{\mathtt{I}^{\mathrm{C}}_i}(\mathbf{p} + \mathtt{R}\delta\mathbf{p}) \\
&\quad = \mathtt{I}^{\mathrm{C}}_k\big(\pi(\mathtt{T}_{\mathrm{CB}}(\mathtt{R}\boldsymbol{\rho}_i + \mathbf{p} + \mathtt{R}\delta\mathbf{p}))\big) - \mathtt{I}^{\mathrm{C}}_{k-1}\big(\pi(\mathtt{T}_{\mathrm{CB}}\ \boldsymbol{\rho}_i)\big) \\
&\quad \overset{(b)}{\simeq} \mathtt{I}^{\mathrm{C}}_k\big(\pi(\mathtt{T}_{\mathrm{CB}}(\mathtt{R}\boldsymbol{\rho}_i + \mathbf{p}))\big) - \mathtt{I}^{\mathrm{C}}_{k-1}\big(\pi(\mathtt{T}_{\mathrm{CB}}(\boldsymbol{\rho}_i - \delta\mathbf{p}))\big) \\
&\quad \overset{(e)}{\simeq} \mathbf{r}_{\mathtt{I}^{\mathrm{C}}_i}(\mathtt{R}) + \frac{\partial\mathtt{I}^{\mathrm{C}}_{k-1}(\mathbf{u})}{\partial\mathbf{u}}\Big|_{\mathbf{u}=\pi({}_c\boldsymbol{\rho}_i)}\frac{\partial\pi(\boldsymbol{\rho})}{\partial\boldsymbol{\rho}}\Big|_{\boldsymbol{\rho}={}_c\boldsymbol{\rho}_i}\mathtt{R}_{\mathrm{CB}}\delta\mathbf{p}
\end{aligned} \tag{23}$$

In step (a), we have used a first-order expansion of the matrix logarithm:

$$\log\left(\exp(\boldsymbol{\phi}^\wedge)\ \exp(\delta\boldsymbol{\phi}^\wedge)\right)^\vee \approx \boldsymbol{\phi} + \mathtt{J}_r^{-1}(\boldsymbol{\phi})\delta\boldsymbol{\phi}, \tag{24}$$

which holds for small values of $\delta\boldsymbol{\phi}$. The term $\mathtt{J}_r^{-1}$ is the inverse of the *right Jacobian of* $\mathrm{SO}(3)$ [72, 73]:

$$\mathtt{J}_r^{-1}(\boldsymbol{\phi}) = \mathbf{I} + \frac{1}{2}\boldsymbol{\phi}^\wedge + \left(\frac{1}{\|\boldsymbol{\phi}\|^2} + \frac{1+\cos(\|\boldsymbol{\phi}\|)}{2\|\boldsymbol{\phi}\|\sin(\|\boldsymbol{\phi}\|)}\right)(\boldsymbol{\phi}^\wedge)^2.$$

In step (b), we invert the perturbation and apply it to the reference frame. This trick stems from the *inverse compositional* [47] formulation, which allows us to keep the term

containing the perturbation constant such that the Jacobian of the intensity residual remains unchanged over all iterations, greatly improving compuational efficiency. In (c), we first used that $\exp(\delta\boldsymbol{\phi}^\wedge)^{-1} = \exp(-\delta\boldsymbol{\phi}^\wedge)$ and subsequently used the first-order approximation of the exponential map:

$$\exp(\delta\boldsymbol{\phi}) \simeq \mathbf{I} + \delta\boldsymbol{\phi}^\wedge. \tag{25}$$

For step (d), we used a property of skew symmetric matrices

$$\delta\boldsymbol{\phi}^\wedge\boldsymbol{\rho} = -\boldsymbol{\rho}^\wedge\delta\boldsymbol{\phi}. \tag{26}$$

Finally, in step (e), we perform a Taylor expansion around the perturbation. The term $\frac{\partial \mathtt{I}^{\mathtt{C}}_{k-1}(\mathbf{u})}{\partial \mathbf{u}}$ denotes the image derivative at pixel $\mathbf{u}$ and $\frac{\partial \pi(\boldsymbol{\rho})}{\partial \boldsymbol{\rho}}$ is the derivative of the camera projection function, which for standard pinhole projection with focal length (f_x, f_y) and camera center (c_x, c_y) takes the form

$$\frac{\partial \pi(\boldsymbol{\rho})}{\partial \boldsymbol{\rho}} = \begin{bmatrix} f_x \frac{x}{z} & 0 & -\frac{c_x}{z^2} \\ 0 & f_y \frac{y}{z} & -\frac{c_y}{z^2} \end{bmatrix} \quad \text{with } \boldsymbol{\rho} = [x, y, z]^\mathsf{T}. \tag{27}$$

To summarize, the Jacobians of the residuals are:

$$\begin{aligned}
\frac{\partial \mathbf{r}_{\mathtt{R}}}{\partial \delta\boldsymbol{\phi}} &= \mathtt{J}_r^{-1}(\mathrm{Log}(\tilde{\mathtt{R}}^\mathsf{T}\mathtt{R})) \\
\frac{\partial \mathbf{r}_{\mathbf{p}}}{\partial \delta\mathbf{p}} &= \mathtt{R} \\
\frac{\partial \mathbf{r}_{\mathtt{I}^{\mathtt{C}}_i}}{\partial \delta\boldsymbol{\phi}} &= -\frac{\partial \mathtt{I}^{\mathtt{C}}_{k-1}(\mathbf{u})}{\partial \mathbf{u}}\Big|_{\mathbf{u}=\pi({}_c\boldsymbol{\rho}_i)} \frac{\partial \pi(\boldsymbol{\rho})}{\partial \boldsymbol{\rho}}\Big|_{\boldsymbol{\rho}={}_c\boldsymbol{\rho}_i} \mathtt{R}_{\mathtt{CB}} \boldsymbol{\rho}_i^\wedge \\
\frac{\partial \mathbf{r}_{\mathtt{I}^{\mathtt{C}}_i}}{\partial \delta\mathbf{p}} &= \frac{\partial \mathtt{I}^{\mathtt{C}}_{k-1}(\mathbf{u})}{\partial \mathbf{u}}\Big|_{\mathbf{u}=\pi({}_c\boldsymbol{\rho}_i)} \frac{\partial \pi(\boldsymbol{\rho})}{\partial \boldsymbol{\rho}}\Big|_{\boldsymbol{\rho}={}_c\boldsymbol{\rho}_i} \mathtt{R}_{\mathtt{CB}}
\end{aligned} \tag{28}$$

References

[1] S. Ullman. *The Interpretation of Visual Motion*. MIT Press: Cambridge, MA, 1979.

[2] C. Tomasi and T. Kanade. Shape and motion from image streams: a factorization method. *Int. J. Comput. Vis.*, (7597):137–154, 1992.

[3] A. Chiuso, P. Favaro, H. Jin, and S. Soatto. Structure from motion causally integrated over time. *IEEE Trans. Pattern Anal. Machine Intell.*, 24(4):523–535, Apr 2002.

[4] D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, volume 1, pages 652–659, June 2004.

[5] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Trans. Pattern Anal. Machine Intell.*, 29(6):1052–1067, June 2007.

[6] D. Scaramuzza and F. Fraundorfer. Visual odometry [tutorial]. Part I: The first 30 years and fundamentals. *IEEE Robotics Automation Magazine*, 18(4):80 –92, December 2011. ISSN 1070-9932. doi: 10.1109/MRA.2011.943233.

[7] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment – a modern synthesis. In W. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, volume 1883 of *LNCS*, pages 298–372. Springer Verlag, 2000.

[8] F. Dellaert and M. Kaess. Square Root SAM: Simultaneous localization and mapping via square root information smoothing. *Int. J. of Robotics Research*, 25(12):1181–1203, December 2006.

[9] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J.J. Leonard, and F. Dellaert. iSAM2: Incremental smoothing and mapping using the Bayes tree. *Int. J. of Robotics Research*, 31:217–236, February 2012.

[10] A. Agarwal, K. Mierle, and Others. Ceres solver. http://ceres-solver.org.

[11] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A general framework for graph optimization. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Shanghai, China, May 2011.

[12] M. Maimone, Y. Cheng, and L. Matthies. Two years of visual odometry on the mars exploration rovers. *J. of Field Robotics*, 24(3):169–186, 2007. ISSN 1556-4967. doi: 10.1002/rob.20184. URL http://dx.doi.org/10.1002/rob.20184.

[13] S. Lovegrove, A. J. Davison, and J. Ibañez Guzmán. Accurate visual odometry from a rear parking camera. *IEEE Intelligent Vehicles Symposium, Proceedings*, pages 788–793, 2011. ISSN 1931-0587. doi: 10.1109/IVS.2011.5940546.

[14] G. Klein and D. Murray. Improving the agility of keyframe-based SLAM. In *Eur. Conf. on Computer Vision (ECCV)*, pages 802–815, 2008.

[15] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. DTAM: Dense tracking and mapping in real-time. In *Int. Conf. on Computer Vision (ICCV)*, pages 2320–2327, November 2011.

[16] J. Engel, J. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *Eur. Conf. on Computer Vision (ECCV)*, 2014.

[17] M. Irani and P. Anandan. All about direct methods. In *Proc. Workshop Vis. Algorithms: Theory Pract.*, pages 267–277, 1999.

[18] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan. *Estimation with Applications To Tracking and Navigation*. John Wiley and Sons, 2001.

[19] G. P. Huang, A. I. Mourikis, and S. I. Roumeliotis. Observability-based rules for designing consistent EKF SLAM estimators. *Int. J. of Robotics Research*, 29:502–528, May 2010.

[20] C. Forster, M. Pizzoli, and D. Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 15–22, 2014. URL http://dx.doi.org/10.1109/ICRA.2014.6906584.

[21] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *IEEE and ACM Int. Sym. on Mixed and Augmented Reality (ISMAR)*, pages 225–234, Nara, Japan, November 2007.

[22] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Trans. Robotics*, 31(5):1147–1163, 2015.

[23] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981. ISSN 0001-0782. doi:

http://doi.acm.org/10.1145/358669.358692.

[24] K. MacTavish and T. D. Barfoot. At all costs: A comparison of robust cost functions for camera correspondence outliers. In *Conf. on Computer and Robot Vision (CRV)*, 2015.

[25] H. Jin, P. Favaro, and S. Soatto. A semi-direct approach to structure from motion. *The Visual Computer*, 19(6): 377–394, 2003.

[26] S. Benhimane and E. Malis. Integration of euclidean constraints in template based visual tracking of piecewise-planar scenes. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2006.

[27] G. Silveira, E. Malis, and P. Rives. An efficient direct approach to visual slam. *IEEE Trans. Robotics*, 2008.

[28] C. Mei, S. Benhimane, E. Malis, and P. Rives. Efficient homography-based tracking and 3-d reconstruction for single-viewpoint sensors. *IEEE Trans. Robotics*, 24(6): 1352–1364, December 2008. ISSN 1552-3098. doi: 10.1109/TRO.2008.2007941.

[29] A. Pretto, E. Menegatti, and E. Pagello. Omnidirectional dense large-scale mapping and navigation based on meaningful triangulation. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 3289–3296. IEEE, May 2011. ISBN 978-1-61284-386-5. doi: 10.1109/ICRA.2011.5980206.

[30] N. D. Molton, A. J. Davison, and I. D. Reid. Locally planar patch features for real-time structure from motion. In *British Machine Vision Conf. (BMVC)*. BMVC, September 2004.

[31] A.I. Comport, E. Malis, and P. Rives. Real-time quadrifocal visual odometry. *Int. J. of Robotics Research*, 29 (2-3):245–266, January 2010. ISSN 0278-3649. doi: 10.1177/0278364909356601.

[32] M. Meilland, A. Comport, and P. Rives. Real-time dense visual tracking under large lighting variations. In *British Machine Vision Conf. (BMVC)*, 2011. ISBN 1-901725-43-X. doi: 10.5244/C.25.45.

[33] T. Tykkala, C. Audras, and A.I. Comport. Direct iterative closest point for real-time visual odometry. In *Int. Conf. on Computer Vision (ICCV)*, 2011.

[34] C. Kerl, J. Sturm, and D. Cremers. Robust odometry estimation for rgb-d cameras. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2013.

[35] M. Meilland and A.I. Comport. On unifying key-frame and voxel-based dense visual SLAM at large scales. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Tokyo, Japan, 3-8 November 2013. IEEE/RSJ.

[36] T. Whelan, M. Kaess, H. Johannsson, M.F. Fallon, J.J. Leonard, and J.B. McDonald. Real-time large scale dense RGB-D SLAM with volumetric fusion. *Int. J. of Robotics Research*, 2014.

[37] A. Handa, T. Whelan, J.B. McDonald, and A.J. Davison. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Hong Kong, China, May 2014.

[38] T. Whelan, S. Leutenegger, R. F. Salas-Moreno, B. Glocker, and A. J. Davison. ElasticFusion: Dense SLAM without a pose graph. In *Robotics: Science and Systems (RSS)*, Rome, Italy, July 2015.

[39] M. Pizzoli, C. Forster, and D. Scaramuzza. REMODE: Probabilistic, monocular dense reconstruction in real time. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 2609–2616, 2014. URL http://dx.doi.org/10.1109/ICRA.2014.6907233.

[40] F. Dellaert and R. Collins. Fast image-based tracking by selective pixel integration. In *ICCV Workshop on Frame-Rate Vision*, 1999.

[41] J. Engel, J. Sturm, and D. Cremers. Semi-dense visual odometry for a monocular camera. In *Int. Conf. on Computer Vision (ICCV)*, 2013.

[42] P. Ondruska, P. Kohli, and S. Izadi. Mobilefusion: Real-time volumetric surface reconstruction and dense tracking on mobile phones. In *IEEE and ACM Int. Sym. on Mixed and Augmented Reality (ISMAR)*, Fukuoka, Japan, October 2015.

[43] D. G. Kottas, J. A. Hesch, S. L. Bowman, and S. I. Roumeliotis. On the consistency of vision-aided inertial navigation. In *Int. Sym. on Experimental Robotics (ISER)*, 2012.

[44] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza. On-manifold preintegration theory for fast and accurate visual-inertial navigation. December 2015. URL http://arxiv.org/pdf/1512.02363v1.pdf.

[45] P. Furgale, J. Rehder, and R. Siegwart. Unified temporal and spatial calibration for multi-sensor systems. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2013.

[46] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry. *An Invitation to 3-D Vision: From Images to Geometric Models*. Springer Verlag, 2005.

[47] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *Int. J. Comput. Vis.*, 56(3):221–255, 2004.

[48] C. Harris and C. Stennett. RAPiD - a video-rate object tracker. In *British Machine Vision Conf. (BMVC)*, pages 73–78, 1990.

[49] T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Trans. Pattern Anal. Machine Intell.*, 24:932–946, 2002.

[50] A. I. Comport, E. Marchand, and F. Chaumette. A real-time tracker for markerless augmented reality. In *IEEE and ACM Int. Sym. on Mixed and Augmented Reality (ISMAR)*, 2003.

[51] L. Vacchetti, V. Lepetit, and P. Fua. Combining edge and texture information for real-time accurate 3d camera tracking. In *IEEE and ACM Int. Sym. on Mixed and Augmented Reality (ISMAR)*, 2004.

[52] G. Reitmayr and T.W. Drummond. Going out: robust model-based tracking for outdoor augmented reality. In *IEEE and ACM Int. Sym. on Mixed and Augmented Reality (ISMAR)*, pages 109–118, October 2006.

[53] M. Faessler, F. Fontana, C. Forster, E. Mueggler, M. Pizzoli, and D. Scaramuzza. Autonomous, vision-based flight and live dense 3D mapping with a quadrotor MAV. *J. of Field Robotics*, pages 1556–4967, 2015. URL http://dx.doi.org/10.1002/rob.21581.

[54] G. Vogiatzis and C. Hernández. Video-based, real-time multi view stereo. *Image Vision Comput.*, 29(7):434–441, 2011.

[55] J. Civera, A.J. Davison, and J. Montiel. Inverse depth parametrization for monocular slam. *IEEE Trans. Robotics*, 24(5), 2008.

[56] D. Scaramuzza, A. Martinelli, and R. Siegwart. A flexible technique for accurate omnidirectional camera calibration and structure from motion. In *Int. Conf. on Computer Vision Systems (ICVS)*, pages 45–45, 2006.

[57] D. Nister. An efficient solution to the five-point relative pose problem. *IEEE Trans. Pattern Anal. Machine Intell.*, 26(6):756–777, 2004.

[58] H. Jin, P. Favaro, and S. Soatto. Real-time feature tracking and outlier rejection with changes in illumination. *Int. Conf. on Computer Vision (ICCV)*, 1, 2001. doi: 10.1109/ICCV.2001.937588.

[59] E. Rosten, R. Porter, and T. Drummond. Faster and better: A machine learning approach to corner detection. *IEEE Trans. Pattern Anal. Machine Intell.*, 32 (1):105–119, January 2010. ISSN 0162-8828. doi: 10.1109/TPAMI.2008.275.

[60] Z. Zhang, H. Rebecq, C. Forster, and D Scaramuzza. Benefit of large field-of-view cameras for visual odometry. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2016.

[61] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. Achtelik, and R. Siegwart. The EuRoC MAV datasets. *Int. J. of Robotics Research*, 2015. URL http://projects.asl.ethz.ch/datasets/doku.php?id=kmavvisualinertialdatasets.

[62] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Oct. 2012.

[63] J. Nikolic, J. Rehder, M. Burri, P. Gohl, S. Leutenegger, P. Furgale, and R. Siegwart. A synchronized visual-inertial sensor system with FPGA pre-processing for accurate real-time SLAM. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014.

[64] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Trans. Pattern Anal. Machine Intell.*, 13(4), 1991.

[65] Raúl Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. `arXiv:1502.00956`, February 2015.

[66] C. Forster, M. Faessler, F. Fontana, M. Werlberger, and D. Scaramuzza. Continuous on-board monocular-vision–based aerial elevation mapping for quadrotor landing. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 111–118, 2015. URL http://dx.doi.org/10.1109/ICRA.2015.7138988.

[67] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard. An evaluation of the RGB-D SLAM system. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2012.

[68] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy. Visual odometry and mapping for autonomous flight using an RGB-D camera. In *Proc. of the Int. Symp. of Robotics Research (ISRR)*, Flagstaff, USA, August 2011.

[69] R. A. Newcombe, A. J. Davison, S. Izadi, P. Kohli, O. Hilliges, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, and A. Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *IEEE and ACM Int. Sym. on Mixed and Augmented Reality (ISMAR)*, pages 127–136, Basel, Switzerland, October 2011.

[70] T. Whelan, H. Johannsson, M. Kaess, J. J. Leonard, and J. B. McDonald. Robust real-time visual odometry for dense RGB-D mapping. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Karlsruhe, Germany, May 2013.

[71] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Int. Joint Conf. on Artificial Intelligence*, pages 121–130, 1981.

[72] T. D. Barfoot. *State Estimation for Robotics - A Matrix Lie Group Approach*. Cambridge University Press, 2015.

[73] G. S. Chirikjian. *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications (Applied and Numerical Harmonic Analysis)*. Birkhauser, 2012.