Image and Vision Computing 30 (2012) 65–77

Contents lists available at SciVerse ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

Editors Choice Article

Visual SLAM: Why filter?☆

Hauke Strasdat [a,*], J.M.M. Montiel [b], Andrew J. Davison [a]

[a] Department of Computing, Imperial College London, UK
[b] Instituto de Investigacion en Ingeniera de Aragon (I3A), Universidad de Zaragoza, Spain

ARTICLE INFO

Article history:
Received 2 August 2011
Received in revised form 13 December 2011
Accepted 17 February 2012

Keywords:
SLAM
Structure from motion
Bundle adjustment
EKF
Information filter
Monocular vision
Stereo vision

ABSTRACT

While the most accurate solution to off-line structure from motion (SFM) problems is undoubtedly to extract as much correspondence information as possible and perform batch optimisation, sequential methods suitable for live video streams must approximate this to fit within fixed computational bounds. Two quite different approaches to real-time SFM – also called visual SLAM (simultaneous localisation and mapping) – have proven successful, but they sparsify the problem in different ways. Filtering methods marginalise out past poses and summarise the information gained over time with a probability distribution. Keyframe methods retain the optimisation approach of global bundle adjustment, but computationally must select only a small number of past frames to process.
In this paper we perform a rigorous analysis of the relative advantages of filtering and sparse bundle adjustment for sequential visual SLAM. In a series of Monte Carlo experiments we investigate the accuracy and cost of visual SLAM. We measure accuracy in terms of entropy reduction as well as root mean square error (RMSE), and analyse the efficiency of bundle adjustment versus filtering using combined cost/accuracy measures. In our analysis, we consider both SLAM using a stereo rig and monocular SLAM as well as various different scenes and motion patterns. For all these scenarios, we conclude that keyframe bundle adjustment outperforms filtering, since it gives the most accuracy per unit of computing time.

1. Introduction

Live motion and structure estimation from a single moving video camera have potential applications in domains such as robotics, wearable computing, augmented reality and the automotive sector. This research area has a long history dating back to work such as [21], but recent years – through advances in computer processing power as well as algorithms – have seen great progress and several standout demonstration systems have been presented. Two methodologies have been prevalent: filtering approaches [1,3,9,17,6] that fuse measurements from all images sequentially by updating probability distributions over features and camera pose parameters; and bundle adjustment (BA) methods that perform batch optimisation over selected images from the live stream, such as a sliding window [37,40], or in particular spatially distributed keyframes [27,49,31] which permit drift-free long-term operation. Both approaches were used for stereo vision [40,44,31] as well as for monocular vision [9,40,37,27,17,6,49].

Understanding of the generic character of localisation and reconstruction problems has recently matured significantly. In particular, recently a gap has been bridged between the structure from motion (SFM) research area in computer vision, whose principles were derived from photogrammetry, and the simultaneous localisation and mapping (SLAM) sub-field of mobile robotics research — hence the somewhat unfortunate dual terminology. The essential character of these two problems, estimating sensor motion by modelling the previously unknown but static environment, is the same, but the motivation of researchers has historically been different. SFM tackled problems of 3D scene reconstruction from small sets of images, and projective geometry and optimisation have been the prevalent methods of solution. In SLAM, on the other hand, the classic problem is to estimate the motion of a moving robot in real-time as it continuously observes and maps its unknown environment with sensors which may or may not include cameras. Here sequential filtering techniques have been to the fore.

It has taken the full adoption of Bayesian methods for both to be able to be understood with a unified single language and a full cross-over of methodologies to occur. Some approaches such as [33,19,26,25,45] aim at pulling together the best of both approaches. There remains, however, the fact that in the specific problem of real-time monocular camera tracking, the best systems have been strongly tied to one approach or the other. The question of why, and whether one approach is clearly superior to the other, needs resolving to guide future research in this important application area.

2. Filtering versus bundle adjustment

The general problem of SLAM/SFM can be posed in terms of inference on a graph [13]. We represent the variables involved by the

☆ This paper has been recommended for acceptance by Ian Reid. Editor's Choice Articles are invited and handled by a select rotating 12 member Editorial Board committee.

* Corresponding author.
E-mail addresses: strasdat@doc.ic.ac.uk (H. Strasdat), josemari@unizar.es (J.M.M. Montiel), ajd@doc.ic.ac.uk (A.J. Davison).

0262-8856/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2012.02.009

Markov random field shown in Fig. 1(a). The variables of interest are T_i, each a vector of parameters representing a historic position of the camera, and $\mathbf{x}_j$, each a vector of parameters representing the position of a feature, assumed to be static. These are linked by image feature measurements $\mathbf{z}_{ij}$ – the observation of feature $\mathbf{x}_j$ from pose T_i – represented by edges in the graph. In real-time SLAM, this network will continuously grow as new pose and measurement variables are added at every time step, and new feature variables will be added whenever new parts of a scene are explored for the first time.

Although various parametric and non-parametric inference techniques have been applied to SFM and SLAM problems (such as particle filters [47,16]), the most generally successful methods in both filtering and optimisation have assumed Gaussian distributions for measurements and ultimately state-space estimation; equivalently we could say that they are least-squares methods which minimises in the reprojection error. BA in SFM, or the extended Kalman filter (EKF) and variants in SLAM all manipulate the same types of matrices representing Gaussian means and covariances. The clear reason is the special status of the Gaussian as the central distribution of probability theory which makes it the most efficient way to represent uncertainty in a wide range of practical inference. We therefore restrict our analysis to this domain.

A direct application of optimal BA to sequential SLAM would involve finding the full maximum likelihood solution to the graph in Fig. 1(a) from scratch as it grew at every new time-step. The computational cost would clearly get larger at every frame, and quickly out of hand. In inference suitable for real-time implementation, we therefore face two key possibilities in order to avoid computational explosion.

In the *filtering* approach illustrated in Fig. 1(b), all poses other than the current one are marginalised out after every frame. Features, which may be measured again in the future, are retained. The result is a graph that stays relatively compact; it will not grow arbitrarily with time, and will not grow at all during repeated movement in a restricted area, adding persistent feature variables only when new areas are explored. The downside is that the graph quickly becomes fully inter-connected, since every elimination of a past pose variable causes fill-in with new links between every pair of feature variables to which it was joined. Joint potentials over all of these mutually-interconnected variables must therefore be stored and updated. The computational cost of propagating joint distributions scales poorly with the number of variables involved, and this is the main drawback of filtering: in SLAM, the number of features in the map will be severely limited. The standard algorithm for filtering using Gaussian probability distributions is the EKF, where the dense interconnections between features are manifest in a single joint density over features stored by a mean vector and large covariance matrix.

The other option is to retain BA's *optimisation* approach, solving the graph from scratch time after time as it grows, but to sparsify it by removing all but a small subset of past poses. In some applications it is sensible for the retained poses to be in a sliding window of the most recent camera positions, but more generally they are a set of intelligently or heuristically chosen *keyframes* (see Fig. 1(c)). The other poses, and all the measurements connected to them, are not marginalised out as in the filter, but simply discarded — they do not contribute to estimates. Compared to filtering, this approach will produce a graph that has more elements (since many past poses are retained), but importantly for inference the lack of marginalisation means that it will remain sparsely inter-connected. The result is that graph optimisation remains relatively efficient, even if the number of features in the graph and measured from the keyframes is very high. The ability to incorporate more feature measurements counters the information lost from the discarded frames. Note that BA-type optimisation methods are usually referred to as *smoothing* in the robotics community [13].

So the key question is whether it makes sense to summarise the information gained from historic poses and measurements by joint probability distributions in state space and propagate these through time (filtering), or to discard some of those measurements in such a way that repeated optimisation from scratch becomes feasible (keyframe BA), and propagating a probability distribution through time is unnecessary. Comparisons of filtering and BA have been presented in the past, but mainly focused on loop closures [12]. In particular, the fact that the EKF led to inconsistencies due to linearisation issues has been studied well in the past [24]. These results led to a series of submapping techniques [7,17,43] which are motivated not only by the inconsistencies in filters once uncertainty is large but also by the fact that a filter's cost increases, typical quadratically, with the map size. Similarly, several techniques were introduced to reduce the computational complexity of real-time BA using segment-wise optimisation [31], feature marginalisation followed by pose-graph optimisation [29,49], incremental smoothing [26] or relative/topological representations [46]. Thus, it is possible to achieve linear or even constant-time complexity for large scale visual SLAM. However, it remained unclear whether filtering or BA should be used for the building block of SLAM: very local motion estimates.

In our conference paper which the current article extends [48], we compared filtering versus BA for monocular SLAM in terms of accuracy and computational cost. The analysis was performed using covariance back-propagation starting from the ground truth solution and assuming the best for filtering — that the accuracy of BA and filtering is identical. The main result was: Increasing the number of observations N increases the accuracy, while increasing the number of intermediate keyframes M only has a minor effect. Considering the cost of BA (linear in N) to the cost of filtering (cubic in N), it becomes clear that BA is the more efficient technique — especially if high accuracy is required. In this work, we affirm this result while generalising our previous work along several dimensions: First, we implement and analyse the full SLAM pipeline including monocular bootstrapping, feature initialisation, and motion-only estimation. In particular, we implement a state of the art filter and analyse its accuracy compared to BA. Second, we extend our analysis to stereo SLAM. Third, and most important, we lift the assumption that all points are visible in all frames and investigate a more realistic scenario where there is only a partial scene overlap.

3. Defining an experimental setup

Hence, there are two main classes of real-time visual SLAM systems capable of consistent local mapping. The first class is based on

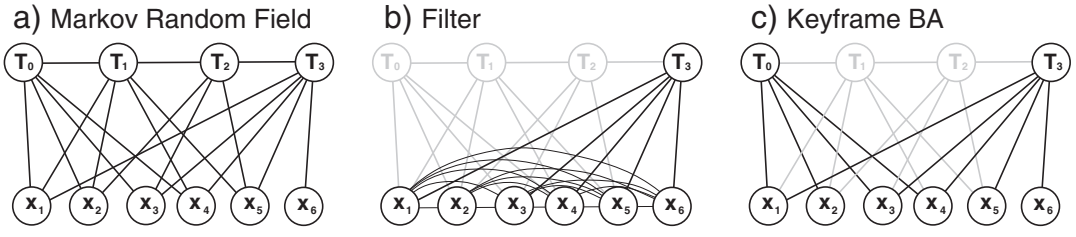

Fig. 1. (a) SLAM/SFM as Markov random field without representing the measurements explicitly. (b) and (c) visualise how inference progressed in a filter and with keyframe-based optimisation.

filtering. An early approach was developed by Chiuso et al. [3], while Davison et al.'s MonoSLAM [9,11] followed a similar method with developments such as active feature measurement and local loop closure. Several enhancements – mainly improving the parameterisation – were suggested [36,42,6]. Probably the best representative of this class is the approach of Eade and Drummond [17] which builds a map of locally filtered sub-maps. The other class is based on keyframe BA, introduced and mainly dominated by Klein and Murray's parallel tracking and mapping (PTAM) framework [28].

For defining an experimental setup, we keep these two successful representatives, PTAM and Eade and Drummond's system, in mind. These systems are similar in many regards, incorporating parallel processes to solve local metric mapping, appearance-based loop closure detection and background global map optimisation over a graph. They are very different at the very local level, however, in exactly the way that we wish to investigate, in what constitutes the fundamental building block of their mapping processes. In PTAM, it is the keyframe, a historical pose of the camera where a large number of features are matched and measured. Only information from these keyframes goes into the final map — all other frames are used locally for tracking but that information is ultimately discarded. Klein and Murray's key observation which permits real-time operation is that BA over keyframes does not have to happen at frame-rate. In their implementation, BA runs in one thread on a multi-core machine, completing as often as possible, while a second tracking thread does operate at frame-rate with the task of pose estimation of the current camera position with respect to the fixed map defined by the nearest keyframe. In Eade and Drummond's system, the building block is a 'node', which is a filtered probabilistic sub-map of the locations of features. Measurements from all frames are digested in this sub-map, but the number of features it contains is consequently much smaller. The spacing of keyframes in PTAM and Eade and Drummond's nodes is decided automatically in both cases, but turns out to be similar. Essentially, during a camera motion between two neighbouring keyframes or nodes, a high fraction of features in the image will remain observable. So in our simulations, we aim to isolate this very local part of the general mapping process: the construction of a building block which is a few nodes or the motion between a few keyframes.[1] Thus, we wish to analyse both accuracy and computational cost. As a measure of accuracy, we consider only the error between the start and end point of a camera motion. This is appropriate as it measures how much camera uncertainty grows with the addition of each building block to a large map.

For our comparison, we apply a state of the art sparse BA approach using the Schur-complement, and a sparse Cholesky solver [30]. It is less obvious what kind of filter variant to use. The standard EKF is fundamentally different from the BA formulation of SLAM, but has well-known limitations. However, there is a broad middle ground between filtering and BA/smoothing (see Section 7.2). Indeed, if one tries to define the best possible filter by modifying the standard approach, one would converge more and more towards BA. Therefore, it is important to define more precisely what we understand by a filter. Our concept of a filter is a cluster of related properties:

1. Explicit representation of uncertainties: A set of parameters is represented using a multivariate normal distribution.
2. Marginalisation: Temporary/outdated parameters are marginalised out in order to keep the state representation compact.
3. Covariance: Joint covariance can be recovered from the filter representation without increasing the overall algorithmic complexity.

[1] Note that in both systems, as opposed to MonoSLAM [9,11] and derived work, no motion prior is enforced. In this sense, their formulation is therefore largely equivalent to standard BA. Therefore, we also won't incorporate motion priors in our analysis, but simply define the log-likelihood to minimise in terms of reprojection error only.

It is obvious that property 1 is the core property of the filter concept. Property 2 is very common in visual SLAM, since filters are often applied at frame-rate. Each single frame produces a new pose estimate. In order to avoid an explosion in the state space, past poses are marginalised out. Still, property 3 is a crucial characteristic which distinguishes BA from filtering: It is possible to calculate the covariance of the BA problem using covariance propagation, but this would increase the algorithmic complexity of BA significantly. There are two fundamentally different approaches of Gaussian filters. The standard approach is the EKF, which represent the uncertainty using a covariance matrix Σ. It is easy to see that the EKF fulfils all the three properties defined above. Its dual is the extended information filter that represents the uncertainty using the inverse covariance or information matrix $\Lambda = \Sigma^{-1}$. In the SLAM community, the EKF and its variants are particularly popular since its computational complexity is $O(K^2)$ while it is in general $O(K^3)$ for the information filter, with K being the total number of features in the map. Since we only consider the local building block of SLAM the computational complexity is dominated by the number of visible features N, leading to a complexity of $O(N^3)$ for both filter types. Thus, both approaches are largely equivalent for our purpose. Indeed we choose the information matrix representation. The reason is twofold: first, the information filter approach is conceptually more appropriate for our comparison since the relation between filtering and BA becomes more obvious — both are non-linear least-squares methods. Second, the information form allows us to include variables without any prior into the state space. Thus, we can include new poses without any motion prior, and also we are able to represent infinite depth uncertainty for monocular inverse-depth features. In particular, we follow Eade and Drummond [17] as well as Sibley et al. [44] and employ the *Gauss–Newton filter*. It iteratively solves the normal equations using the Cholesky method and therefore is the dual of the iterative EKF [2]. Furthermore, note that the Gauss–Newton filter is algebraically equivalent to the classic square root information filter (SRIF) [14]. The SRIF never constructs the normal equations explicitly and solves the problem using an orthogonal decomposition on the square root form. While performing Gauss–Newton using Cholesky's decomposition is less numerically stable than performing the orthogonal decomposition method, its computation is more efficient, and therefore, it is the standard approach for real-time least-squares problems nowadays. Indeed, a sufficient numerical stability of even rank-deficient problems can be archived by applying a robust variant of Cholesky – such as the pivoted $L^\top DL$ decomposition used in the *Eigen* matrix library[2] – and the Levenberg–Marquardt damping term, called *Tikhonov regularisation* [50] in this context.

4. Preliminaries

4.1. Gauss–Newton and Levenberg–Marquardt

In a general state estimation problem, we would like to estimate a vector of parameters $\mathbf{y}$ given a vector of measurements $\mathbf{z}$, where we know the form of the likelihood function $p(\mathbf{z}|\mathbf{y})$. The most probable solution is the set of values $\mathbf{y}$ which maximises this likelihood, which is equivalent to minimising the negative log-likelihood $-logp(\mathbf{z}|\mathbf{y})$. Under the assumption that the likelihood distribution $p(\mathbf{z}|\mathbf{y})$ is Gaussian, the negative log-likelihood $\chi^2(\mathbf{y}) := -logp(\mathbf{z}|\mathbf{y})$ has a quadratic form:

$$\chi^2(\mathbf{y}) = (\mathbf{z} - \hat{\mathbf{z}}(\mathbf{y}))^\top \Lambda_{\mathbf{z}} (\mathbf{z} - \hat{\mathbf{z}}(\mathbf{y})), \tag{1}$$

where $\Lambda_{\mathbf{z}}$ is the information matrix or inverse of the covariance matrix of the likelihood distribution, and $\hat{\mathbf{z}}(\mathbf{y})$ is the measurement function which predicts the distribution of measurements $\mathbf{z}$ given a set of

[2] http://eigen.tuxfamily.org/dox/TutorialLinearAlgebra

parameters $\mathbf{y}$. Since χ^2 is a quadratic function and $\mathbf{d} := \mathbf{z} - \hat{\mathbf{z}}$ approximates zero at its minimum, Gauss–Newton optimisation is applicable. All estimation described in this paper is done using a common variant of Gauss–Newton called Levenberg–Marquardt (LM), which employs the augmented *normal equation*:

$$\left(\mathrm{J}_{\mathbf{d}}^{\top} \Lambda_{\mathbf{z}} \mathrm{J}_{\mathbf{d}} + \mu \mathrm{I}\right) \delta = -\mathrm{J}_{\mathbf{d}}^{\top} \Lambda_{\mathbf{z}} \mathbf{d}. \tag{2}$$

Here, $\mathrm{J}_{\mathbf{d}}$ is the Jacobian of $\mathbf{d}$ and μ the LM damping term.

4.2. Gauss–Newton filter

Let us assume we would like to estimate a parameter $\mathbf{y}$ over time. At each time step $1,\ldots,t$, we observe a set of measurements $\mathbf{z}_1,\ldots,\mathbf{z}_t$. Assuming a Gaussian distribution, the following recursive update scheme is applied:

$$\chi^2(\mathbf{y}_t) = (\mathbf{y}_t - \mathbf{y}_{t-1})^{\top} \Lambda_{\mathbf{y}_{t-1}} (\mathbf{y}_t - \mathbf{y}_{t-1}) + (\mathbf{z}_t - \hat{\mathbf{z}}(\mathbf{y}_t))^{\top} \Lambda_{\mathbf{z}} (\mathbf{z}_t - \hat{\mathbf{z}}(\mathbf{y}_t)). \tag{3}$$

This quadratic energy has of two components. The left summand is a *regulariser* which ensures that the state estimate $\mathbf{y}_t$ stays close to its prior distribution $\langle \mathbf{y}_{t-1}, \Lambda_{\mathbf{y}_{t-1}} \rangle$. The right summand is a *data term* which makes sure that the measurement error $\mathbf{z}_t - \hat{\mathbf{z}}(\mathbf{y}_t)$ is minimised. The information matrix is updated using uncertainty propagation:

$$\Lambda_{\mathbf{y}_t} = \Lambda_{\mathbf{y}_{t-1}} + \mathrm{J}_{\mathbf{d}_t}^{\top} \Lambda_{\mathbf{z}} \mathrm{J}_{\mathbf{d}_t} \tag{4}$$

[22, pp. 141].

5. Formulation of visual SLAM

5.1. Camera poses

We represent poses T as members of the Lie group SE3 [20], which consists of a 3×3 rotation matrix R and a translation 3-vector $\mathbf{t}$. There exists a minimal parameterisation $\omega \in \mathbb{R}^6$ which is represented in the tangent space of SE3 around the identity. Mapping from the tangent space to the manifold SE3 is done using the exponential map $\exp_{\mathbf{SE3}}(\omega) = \boldsymbol{T}$. Since $exp_{\mathbf{SE3}}$ is surjective (=onto), there exists an inverse relation $log_{\mathbf{SE3}}$. It can be shown that the Newton method has a quadratic convergence rate not only for Euclidean vector spaces but also if we optimise over general Lie groups [32]. During optimisation, incremental updates δ are calculated in the tangent space and mapped back onto the manifold: $\mathrm{T} \leftarrow exp_{\mathbf{SE3}}(\delta) \cdot \mathrm{T}$.

5.2. Monocular and stereo camera models

For monocular SLAM, we use the standard pinhole camera model $\hat{z}_m(\mathrm{T}, \mathbf{x})$ where T is the camera pose and $\mathbf{x}$ is a point in the world.

In stereo SLAM, each observation $\mathbf{z}_s = (\,u_l \quad v_l \quad u_r\,)^{\top}$ is a 3-vector, where the first two components u_l, v_l are the pixel measurements in the left camera, which is the reference frame. The third component u_r is the column measurement in the right camera frame. We assume all images are undistorted and rectified thanks to prior calibration. Furthermore, we assume independent Gaussian measurement noise for the monocular and stereo vision: $\Sigma_{\mathbf{z}_m} = \mathrm{diag}(\sigma_z^2, \sigma_z^2)$ and $\Sigma_{\mathbf{z}_s} = \mathrm{diag}(\sigma_z^2, \sigma_z^2, \sigma_z^2)$.

In the following, we mainly concentrate on stereo SLAM. Extensions to monocular SLAM are discussed in Section 5.5.

5.3. BA-SLAM

In BA, we optimise simultaneous for structure and motion by minimising the reprojection error:

$$\chi^2(\mathbf{y}) = \sum_{\mathbf{z}_{i,j} \in \mathcal{Z}_{0:i}} \left(z_{i,j} - \hat{z}\left(\mathrm{T}_i, x_j\right)\right)^2 \tag{5}$$

with respect to $\mathbf{y} = (\mathrm{T}_1, \ldots, \mathrm{T}_i, \mathcal{X})^{\top}$ with $\mathcal{X}$ being the set of all points $\mathbf{x}_j$. The first frame T_0 is typically fixed in order to eliminate the underlying gauge freedom. We employ the standard approach of BA based on LM and the Schur-complement [51,18], and we implement it using the $\mathbf{g}^2\mathbf{o}$ framework [30].

BA is just the core of the full SLAM pipeline. We use the following scheme which is summarised in Table 1. In the first frame, we initialise the 3D points $\mathbf{x}_j \in \mathcal{X}$ from the set of initial measurements $\mathcal{Z}_0$. Inspired by Mei et al. [34], we select a set $\mathcal{X}$ of N points from a larger set of scene point candidates using a quadtree to ensure that the corresponding 2D observations $\mathbf{z}_{0j} \in \mathcal{Z}_0$ are spread approximately equal across the image. For each time step, that is for each new keyframe, four steps are performed. First, we optionally initialised new 3D points in case some old features left the field of view. Using the quadtree, we initialise new points where the feature density. Second, the current pose T_i is estimated using motion-only BA. Thus, we minimise the reprojection error:

$$\chi^2(\mathrm{T}_i) = \sum_{\mathbf{z}_j \in \mathcal{Z}_i} \left(\mathbf{z}_j - \hat{\mathbf{z}}\left(\mathrm{T}_i, \mathbf{x}_j\right)\right)^2 \tag{6}$$

with respect to the current camera T_i. We simply initialise the current pose to the previous pose $\mathrm{T}_i = \mathrm{T}_{i-1}$, however, one can also use a motion model as in [27,11]. Third, we perform structure-only BA by minimising:

$$\chi^2(\mathcal{X}) = \sum_{\mathbf{z}_{i,j} \in \mathcal{Z}_{0:i}} \left(\mathbf{z}_{i,j} - \hat{\mathbf{z}}\left(\mathrm{T}_i, \mathbf{x}_j\right)\right)^2 \tag{7}$$

with respect to the set of points $\mathcal{X}$. Finally, we perform joint optimisation of structure and motion as formalised in Eq. (5).

5.4. Filter-SLAM

For filtering, it is especially important that the state representation is as 'linear' as possible. It proved to be useful, especially but not exclusively for monocular SLAM, to represent 3D points using anchored inverse depth coordinates [5]. Our effort is to combine the most successful approaches. We represent points using the inverse depth formulation of Eade [15]. As in Pietzsch et al. [42], the bundle of points which were initialised at the same time is associated with its common anchor frame A_k. There is a function $a(j) = k$ which assigns an anchor frame index k for each point index j. Thus, our anchored inverse depth representation ψ is defined as:

$$\psi_j := \mathrm{inv_d}\left(\mathrm{A}_{a(j)} \mathbf{x}_j\right) \quad \text{with } \mathrm{inv_d}(\mathbf{a}) = \frac{1}{a_3}(a_1, a_2, 1)^{\top}. \tag{8}$$

Table 1
BA-SLAM pipeline.

$\mathcal{X} \leftarrow \mathrm{initialise_p oints}(\mathcal{Z}_0)$
for each keyframe/time step $i = 1$ to M **do**
  if a number of $n \geq 1$ points left field of view **then**
    $\mathcal{X} \leftarrow \mathcal{X} \cup \mathrm{initialise_n new_p oints}(\mathcal{Z}_i, n)$
  end if
  $\mathrm{T}_i \leftarrow \mathrm{motion_o nly_B A}(\mathcal{X}, \mathcal{Z}_i)$
  $\mathcal{X} \leftarrow \mathrm{structure_o nly_B A}(\mathrm{T}_{0:i}, \mathcal{X}, \mathcal{Z}_{0:i})$
  $\mathrm{T}_{1:i}, \mathcal{X} \leftarrow \mathrm{full_B A}(\mathrm{T}_{1:i}, \mathcal{X}, Z_{0:i})$
end for

Hence, the reprojection error of point ψ_j in the current frame T_i equals:

$$\mathbf{d}_j = \mathbf{z}_j - \hat{\mathbf{z}}\left(T_i, A_{a(j)}^{-1}\psi_j\right). \tag{9}$$

We represent the map state Φ as a set of points and their corresponding anchor frames:

$$\Phi = (\psi_1, \ldots, \psi_N, A_0, \ldots, A_k)^\top. \tag{10}$$

The first frame $T_0 = A_0$ is considered as the fixed origin and therefore discarded from the state representation. If all points are visible in all keyframes, we anchor all points to the origin T_0 and the map representation simplifies to $\Phi = (\psi_1, \ldots, \psi_N)^\top$. As motivated above, we perform filtering using a Gauss–Newton Filter. Thus, we minimise the following sum of squares function,

$$\chi^2(\Phi_i, T_i) = (\Phi_i \ominus \Phi_{i-1})^\top \Lambda_{\Phi_{i-1}} (\Phi_i \ominus \Phi_{i-1}) + \sum_{\mathbf{z}_j \in \mathcal{Z}_i} \mathbf{d}_j^\top \Lambda_{\mathbf{z}} \mathbf{d}_j, \tag{11}$$

wrt. to the map Φ_i and the current camera pose T_i. Here, $\Phi_{i-1}, \Lambda_{\Phi_{i-1}}$ is Gaussian map prior. Differences between two poses are calculated in the tangent space of SE3:

$$A^{[i]} \ominus A^{[i-1]} := log_{\mathbf{SE3}}\left(\left(A^{[i]}\right)^{-1} \cdot A^{[i-1]}\right). \tag{12}$$

Since we do not impose a motion prior on T_i, the prior joint information over the current pose T_i and the map Φ_{i-1} is:

$$\Lambda_{i-1} := \begin{pmatrix} \Lambda_{\Phi_{i-1}} & \Lambda_{\Phi_{i-1},T_i}^\top \\ \Lambda_{\Phi_{i-1},T_i} & \Lambda_{T_i} \end{pmatrix} = \begin{pmatrix} \Lambda_{\Phi_{i-1}} & O_{3n\times 6} \\ O_{6\times 3n} & O_{6\times 6} \end{pmatrix}. \tag{13}$$

Following Eq. (4), we calculate the update of the information matrix:

$$\Lambda_i = \Lambda_{i-1} + D^\top \begin{bmatrix} \Sigma_{\mathbf{z}}^{-1} & & \\ & \ddots & \\ & & \Sigma_{\mathbf{z}}^{-1} \end{bmatrix} D. \tag{14}$$

D is the sparse Jacobian of the stacked reprojection function:

$$\mathbf{d} = \left(\mathbf{d}_1^\top, \ldots, \mathbf{d}_N^\top\right)^\top \tag{15}$$

with respect to the pose T_i, to the points $\psi_1, \ldots, \psi_N$ and to the corresponding anchor frames $\{A_{a(j)} | j = 1, \ldots, N\}$.

The whole filter-SLAM pipeline is sketched in Table 2. In the first frame, the inverse depth points ψ are initialised from the stereo observation $\mathbf{z}_s$:

$$\psi = \begin{pmatrix} \frac{u_l - p_u}{f} & \frac{v_l - p_v}{f} & \frac{u_l - u_r}{fb} \end{pmatrix}^\top, \tag{16}$$

Table 2
Filter-SLAM pipeline.

$T_0 = A_0$; $k \leftarrow 0$
$\langle \Phi_0, \Lambda_{\Phi_0} \rangle \leftarrow$ Initialise map using Eqs. (16) and (17).
for each time step $i = 1$ to M **do**
  if a number of $n \geq 1$ points left field of view **then**
    $k \leftarrow k+1$; $A_k \leftarrow T_{i-1}$; $\Phi_i \leftarrow (\Phi_{i-1}, A_k)^\top$; $\Lambda_{\Phi_i} \leftarrow \Lambda_{i-1}$
    Marginalise out the n invisible points from Φ_i, Λ_{Φ_i}.
    Initialise n new inverse depth points anchored to A_k.
  else
    $\Phi_i \leftarrow \Phi_{i-1}$; $\Lambda_{\Phi_i} \leftarrow \Lambda_{\Phi_{i-1}} - \Lambda_{T_{i-1},\Phi i-1}{}^\top \Lambda_{T_{i-1}}{}^{-1} \Lambda_{T_{i-1},\Phi i-1}$.
  end if
  $\Sigma_{\Phi_i} \leftarrow \Lambda_{\Phi_i}{}^{-1}$ {calculate covariance, optionally}
  $T_i \leftarrow$ Motion-only BA (Eq. (6)) or using map prior $\langle \Phi_i, \Sigma_{\Phi_i} \rangle$.
  $(T_i, \Phi_i) \leftarrow$ Joint filter update by minimising Eq. (11).
  $\Lambda_i \leftarrow$ Augment information matrix and update it (Eqs. (13) and (14)).
end for

with f being the focal length and b being the baseline of the stereo camera. We initial the corresponding information matrix as:

$$\Lambda_\psi = \left(\frac{\partial \mathbf{z}_s}{\partial \psi} \Sigma_{\mathbf{z}} \frac{\partial \mathbf{z}_s}{\partial \psi}^\top\right)^{-1} \quad \text{with} \quad \frac{\partial \mathbf{z}_s}{\partial \psi} = \begin{pmatrix} \frac{1}{f} & 0 & 0 \\ 0 & \frac{1}{f} & 0 \\ \frac{1}{fb} & 0 & -\frac{1}{fb} \end{pmatrix}. \tag{17}$$

At each time step i, we do the following: first, we decide whether we want to initialise new points. If this is the case, we define the previous estimated pose T_{i-1} as the new anchor frame A_k and argument the map state accordingly $\Phi_i = (\Phi_{i-1}, A_k)^\top$. Then, we marginalise out n old points from the filter state and replace them with n new points anchored to A_k. As in BA-SLAM, a quadtree is used for point initialisation. Otherwise, we marginalise out the pose T_{i-1} from Λ_{i-1}:

$$\Lambda_{\Phi_{i-1}} = \Lambda_{\Phi_{i-1}} - \Lambda_{T_{i-1},\Phi_{i-1}}^\top \Lambda_{T_{i-1}}^{-1} \Lambda_{T_{i-1},\Phi_{i-1}}. \tag{18}$$

Next, we approximate the new camera pose T_i given the previous map Φ_{i-1}. In traditional filter-based SLAM implementations, this step is often omitted. However, in case of large camera displacements (e.g., due to low filter frequency) it is desirable to approximate the camera motion before applying the joint filter update. There are two possibilities to estimate the camera pose given a known map. One can either do motion-only BA by minimising Eq. (6). Here we assume that the points are accurately known. In the case that there is a significant uncertainty in the map, and a model of this uncertainty is available, we can do better. As described by Eade [15, pp. 126], we can estimate a pose given a Gaussian map prior. The effect is that taking account of the 3D uncertainty in point positions will weight their impact on camera motion estimation, and better accuracy will be obtained because accurately located points will be trusted more than uncertain ones. The pros and cons of these two approaches are analysed in Section 6.3. Finally, we perform the joint filter estimate and update the information matrix as discussed above.

5.5. Monocular SLAM

5.5.1. Monocular bundle adjustment

For BA, the gauge freedom increases from 6 DoF to 7 DoF from stereo to monocular vision. Even after fixing the origin T_0, one dimension of scale gauge remains. We simply leave this one degree unfixed, since the damping term of LM can deal with gauge freedom effectively [23]. In BA-SLAM, new 3D points are triangulated between two consecutive keyframes using a set of independent filters [28,49].

5.5.2. Monocular filter

Since an anchored inverse depth representation was chosen for the filter, no substantial improvements are necessary when moving from stereo to monocular vision. As opposed to monocular BA, the monocular filter does not introduce a scale ambiguity. The reason is that a non-trivial map distribution $\langle \Phi, \Lambda_\Phi \rangle$ introduces a scale prior and therefore the degree of free gauge in Eq. (3) remains zero. This arbitrary scale factor is invented during bootstrapping (see Section 5.5.3). For monocular vision, new features ψ are initialised with infinite uncertainty along the feature depth ψ_3:

$$\psi = \begin{pmatrix} \frac{u - p_u}{f} & \frac{v - p_v}{f} & 1 \end{pmatrix}^\top \quad \text{and} \quad \Lambda_\psi = \text{diag}\left(\frac{f^2}{\sigma_z^2}, \frac{f^2}{\sigma_z^2}, 0\right). \tag{19}$$

5.5.3. Structure and motion bootstrapping

Unless there is any additional prior knowledge such as a known object in the scene, monocular SLAM requires a special bootstrapping

mechanism. We perform bootstrapping between three consecutive keyframes T_{b0}, T_{b1}, T_0. The standard approach relies on the 5-point algorithm [39], which however requires a RANSAC-like procedure. We instead employ an iterative optimisation, exploiting the fact that the consecutive keyframes share similar poses. First, we define T_{b0} as our fixed origin and apply monocular filtering between T_{b0} and T_{b1}. Let us assume without loss of generality that $T_{b0}=I$. Note that now Eq. (3) has one dimension of gauge freedom, since there is infinite uncertainty along all feature depths ψ_3. This scale freedom during optimisation is handled with the LM damping term. Afterwards, we check whether the estimated motion T_{b1} has sufficient parallax. To summarise, we have estimated $6+3N$ parameters, while the underlying problem only has $5+3N$ DoF. In order to avoid a rank-deficient map distribution, we convert the pose T_{b1} into a 5 DoF representation by enforcing the additional constraint on SE3 that the translation must be unity $|\mathbf{t}_{b1}| = 1$. First, we scale the whole state estimate – all inverse depth points ψ_j as well as the initial motion T_{b1} – such that $|\mathbf{t}_{b1}| = 1$. Afterwards, we perform uncertainty propagation (Eq. (14)) with a modified Jacobian D reflecting that the pose only has 5 DoF. Then, the 5 DoF pose is marginalised out. The resulting precision matrix Λ_p has full rank and enforces a scale prior (that the initial translation between T_{b0} andT_{b1} has unit length). Finally, we perform a standard monocular filter update (as described above) between frames T_{b1} and T_0 so that the resulting map is well initialised and can be used for either BA-SLAM or filter-SLAM.

6. Experiments

As motivated in Section 3, we analyse the performance of visual SLAM by evaluating local motion in a set of simulation experiments. We choose a camera with a resolution of 640×480 pixels and a focal length of $f=500$. Thus, the simulated camera has a horizontal view angle of 65.2° and a vertical view angle of 51.3°. We assume normally distributed measurement noise with a standard derivation of $\sigma_z = \frac{1}{2}$pixel. For the simulated stereo camera, we choose a baseline of 10 cm.

6.1. Four different settings

In our experiments, we consider four different scenes/motion patterns (see Fig. 2). In setting (i), the camera performs a motion sideways of 0.5 m while observing an approximately planar scene. Here, all points are visible in all frames, and therefore the number of points in the map equals the number of observations per frame. The number M of keyframes (intermediate keyframes plus end frame, excluding the first frame) is varied between 1 and 16; more specifically $M\in\{1, 2, 4, 8, 16\}$. The number of observations N is chosen from $N\in\{15, 30, 60, 120, 240\}$. In addition, we also consider $N=480$ for some specific cases.

The configuration of setting (i) is motivated in two ways. Firstly, it represents a situation of relatively detailed local scene reconstruction, essentially optimising the local environment of one view with the support of very nearby surrounding views, as might be encountered practically for instance in small scale augmented reality, or object model construction. Secondly, the estimation produced in this setting could be seen as a building block of a sub-mapping SLAM system. In particular it is very comparable to a single filter node of Eade and Drummond's SLAM framework [17]. Since no new points need to be initialised, all points are anchored to the fixed origin A_0.

The previous setting is very specific in the sense that all points are visible in all frames. In a typical visual odometry building block, there is only partial scene overlap. In each new frame of a sequence, some point projections leave the field of view while new points become visible. For setting (ii), we have chosen a translation of 1.1 m, so that the first and the final frames barely overlap. Therefore, at least one intermediate keyframe has to be used and we choose $M\in\{2, 4, 8, 16\}$.

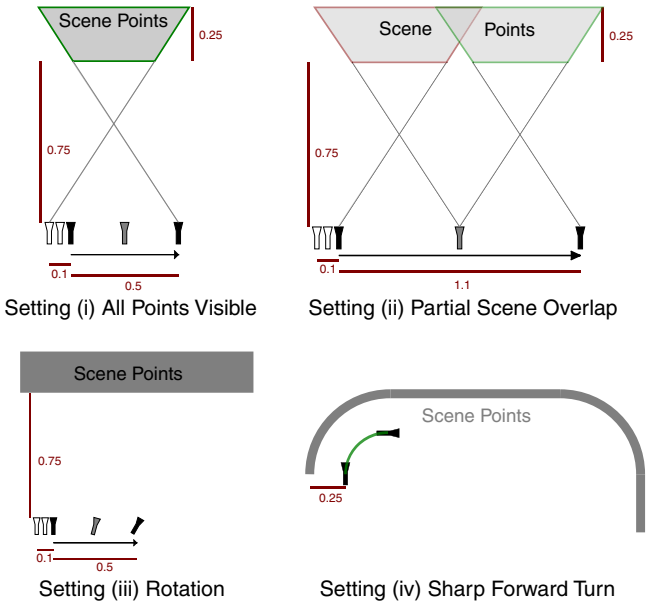

Fig. 2. Birds-eye view of different motion/scene settings. Black cameras represent start and end pose. Intermediate poses are presented in grey. Unfilled cameras indicate the poses used for monocular bootstrapping T_{b0}, T_{b1}. Scene points are initialised within the grey-shaded areas. In setting (i), all points are visible in all frames. In setting (ii), there is only a partial scene overlap. Here, we illustrate the case with a single intermediate camera ($M=2$). Some points are triangulated between the first and middle frames (right/red area), with others between middle and end frame (left/green area). In setting (iii), the camera performs a 30° rotation while still moving sideways. In setting (iv), the camera performs a sharp forward turn so that the scene points quickly leave the field of view. To avoid cluttering the figure, we do not show intermediate and bootstrapping poses here.

In the setting (iii), the camera performs a sideways motion plus rotation which leads to a partial scene overlap. Again, we choose $M\in\{2, 4, 8, 16\}$. In the final setting (iv), the camera performs a sharp forward turn. This setting is typical for a camera mounted on a robot which performs a sharp 90° turn in an indoor environment. This setting is especially hard for monocular SLAM: Scene points leave the field of view quickly while parallax is low due to the lack of translation. To achieve an acceptable level of robustness, we select $M\in\{4, 8, 16\}$.

For all optimisations (motion update, structure-only BA, full BA, joint filter update) we perform three LM iterations in settings (i and ii) and ten LM iterations in settings (iii and iv).

6.2. Accuracy of visual SLAM

We analyse the accuracy using the difference between the true final camera position $\mathbf{t}_{\text{true}}$ and the corresponding estimate $\mathbf{t}_{\text{est}}$:

$$\Delta t = t_{true} - t_{est}. \tag{20}$$

Note that while our estimation framework of course produces both translation and 3D rotation estimates for camera motion, our accuracy analysis is based purely on translation. We believe that this is valid since accurate translation clearly implies that rotation is also well estimated; and in this way avoid the ill-posed question of forming a single unified measure representing both rotation and translation accuracy. For each chosen number of frames and points $\langle M,N\rangle$, we perform a set of $k=500$ Monte Carlo trials. For setting (i) using stereo SLAM, the resulting plots are shown in Fig. 3(a). Approximately, the presented discrete error distributions appear to consist of samples from unimodal, zero-mean Gaussian-like distributions.

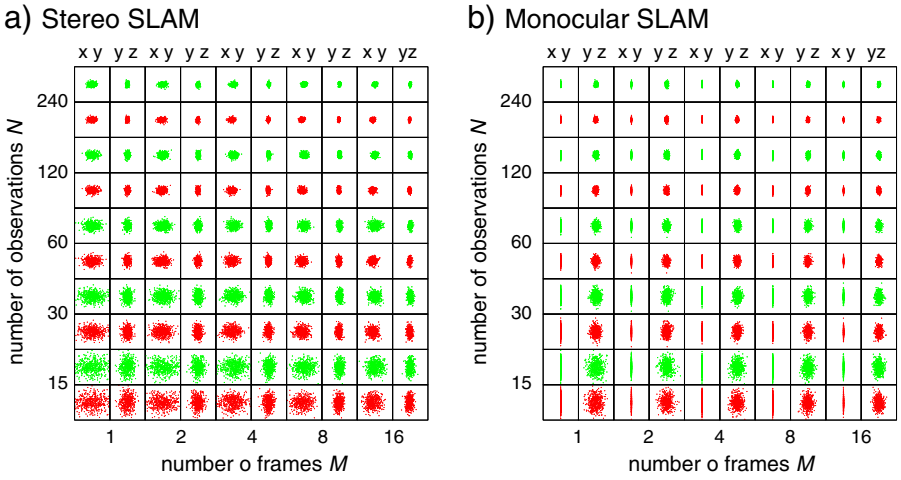

Fig. 3. End pose accuracy of stereo and monocular SLAM. BA results are shown in red (top rows), whereas filtering results are shown in green (below). The distributions are shown in a zero-centred 1.5 cm sector.

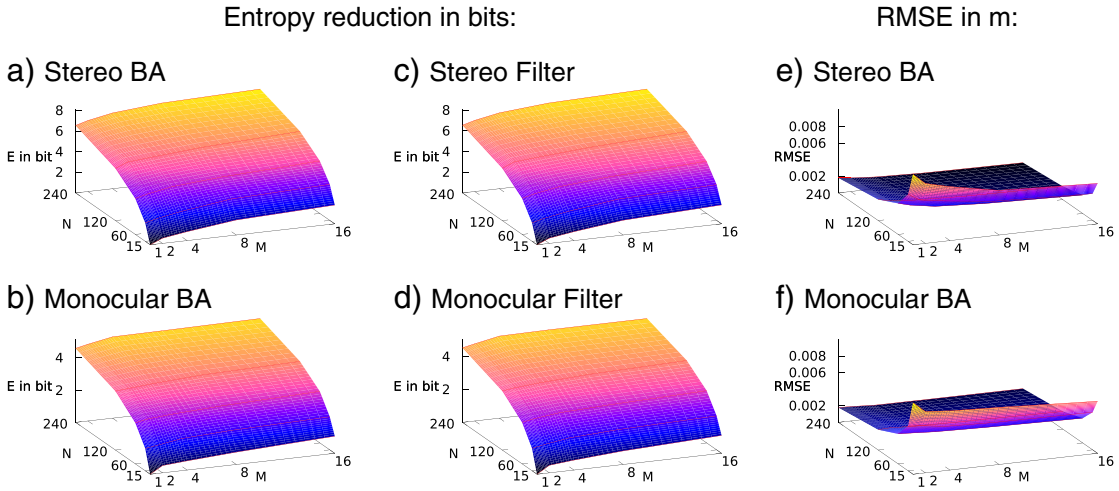

Fig. 4. Setting (i). Accuracy plots in terms of entropy reduction in bits and RMSE.

In the case of monocular SLAM, we can only estimate the translation modulo in an unknown scale factor. Therefore, we eliminate the scale ambiguity in our evaluation by normalising the estimated translation to the true scale:

$$\mathbf{t}^* = \frac{|\mathbf{t}_{\text{true}}|}{|\mathbf{t}_{\text{est}}|}\mathbf{t}_{\text{est}}. \tag{21}$$

Hence, all normalised estimates $\mathbf{t}^*$ lie on the sphere of radius $|\mathbf{t}_{\text{true}}|$. This explains why the projection of the error distribution onto the *xy* plane is elongated, with no uncertainty along the unknown scale dimension (here *x*-axis). Interestingly, error distributions in the *yz* plane for monocular and stereo SLAM are of similar shape and size. In order to have a minimal and Gaussian-like parameterisation of the monocular error distribution, we calculate the error in the tangent plane around the point $\mathbf{t}_{\text{true}}$:

$$\Delta\mathbf{t} = \phi_{\mathbf{t}_{\text{true}}}(\mathbf{t}^*). \tag{22}$$

Here, $\phi_{\mathbf{t}_{\text{true}}}$ is an orthogonal projection that maps points on the ball with radius $|\mathbf{t}_{\text{true}||}$ onto the tangent plane around $\mathbf{t}_{\text{true}}$ (so that $\mathbf{t}_{\text{true}}$ is mapped to $(0,0)^\top$).

We use two ways to describe the error distribution. Our first measure is based on information theory. We analyse the influence of different parameters $\langle M,N\rangle$ in terms of entropy reduction. Therefore, for each setting $\langle M,N\rangle$ we estimate the covariance matrix $\Sigma_{\langle M,N\rangle}$ of the translation error distribution $\Delta\mathbf{t}$. Then, we can compute the entropy reduction in bits,

$$E = \frac{1}{2}log_2\left(\frac{det\left(\Sigma_{M_{\text{min}},15}\right)}{det\left(\Sigma_{M,N}\right)}\right) \tag{23}$$

in relation to the least accurate case where only the minimal number of frames M_{min}[3] and 15 points are used for SLAM. Thus, geometrically the measure E describes the ratio of the volumes of the two ellipsoids $\Sigma_{\langle M_{\text{min}},15\rangle}$ and $\Sigma_{\langle M,N\rangle}$ on a log scale. This measure, which is described in detail in A, is only meaningful if both distributions share approximatively the same mean.

The influence of the parameters $\langle M,N\rangle$ in setting (i) is illustrated in Fig. 4(a–d). As can be seen in all plots (monocular vs. stereo, filtering vs. BA), increasing the number of features leads to a significant entropy reduction. On the other hand, increasing the number of intermediate frames has only a minor influence. This is the single most important result of our analysis. Also, we can see that the accuracy of our

[3] This is $M_{\text{min}}=1$ for setting (i) and $M_{\text{min}}=2$ for settings (ii and iii), and $M_{\text{min}}=4$ for setting (iv).

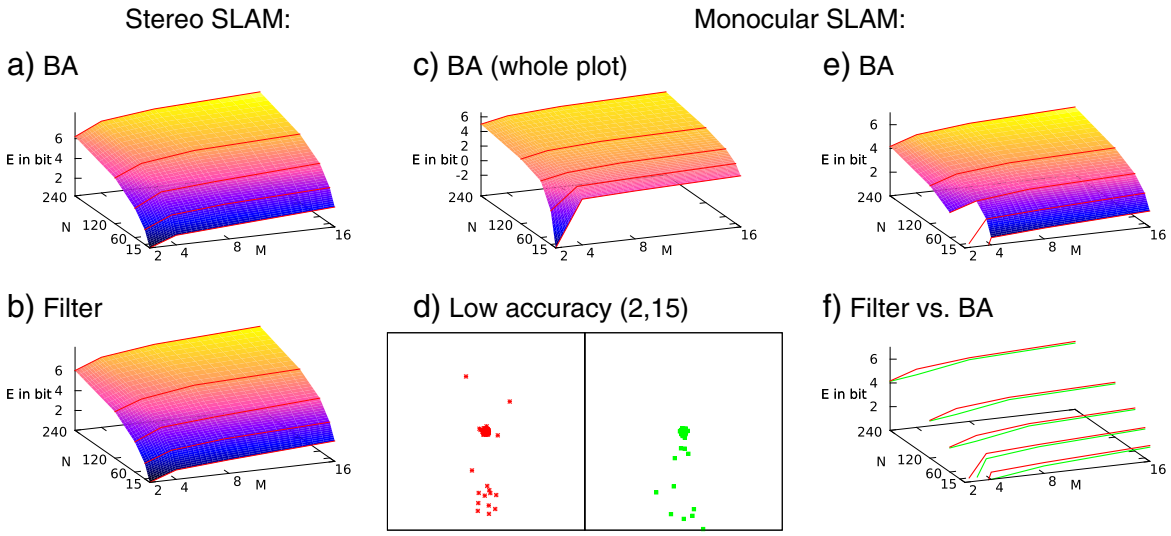

Fig. 5. Setting (ii). Accuracy plots in terms of entropy reduction in bits (a–c, e, f). Plot (d) illustrates the error distribution for the low robustness case $\langle M,N\rangle = \langle 2,15\rangle$. For both BA (left, red) and filtering (right, green), the distributions for this lowest accuracy case contain outliers, i.e. complete SLAM estimation failures, and this explains the discontinuities in the otherwise smooth plots (c, e, f) in the low accuracy corner. Even though we show a range of one metre, a significant portion of outliers lies outside this range.

filter is in fact very close to the accuracy of BA, confirming that we have chosen the filter parameterisation well.

The accuracy results for setting (ii), where the camera still moves sideways but now over a distance such that there is hardly any scene overlap between the first and last frames, are shown in Fig. 5(a, b). The plots for stereo SLAM look similar to setting (i). The whole accuracy plot for monocular BA is shown in Fig. 5(c). Note that for the low accuracy cases $\langle 2,15\rangle$ and $\langle 2,30\rangle$ the estimation is not very robust, and SLAM fails occasionally. Thus, the corresponding error distributions are heavy tailed/non-Gaussian as shown in Fig. 5(d), and therefore the entropy reduction measure is not fully meaningful. Therefore, we excluded these two cases from the subsequent analysis and defined $\langle 4,15\rangle$ as the minimal base case. A corresponding accuracy plot is shown in Fig. 5(e). The characteristic pattern we saw before is repeated: increasing the number of points is the most significant way to increase accuracy. Meanwhile, increasing the number of frames has the main effect of increasing robustness — i.e., avoiding complete failures. Once robustness is achieved, a further increase in M has only a minor effect on accuracy. Finally, as we can see in Fig. 5(f), monocular BA leads to marginally better accuracy than filtering, especially for small M.

In general, the accuracy plots for setting (i) and setting (ii) show a similar pattern. However, there is a significant difference between

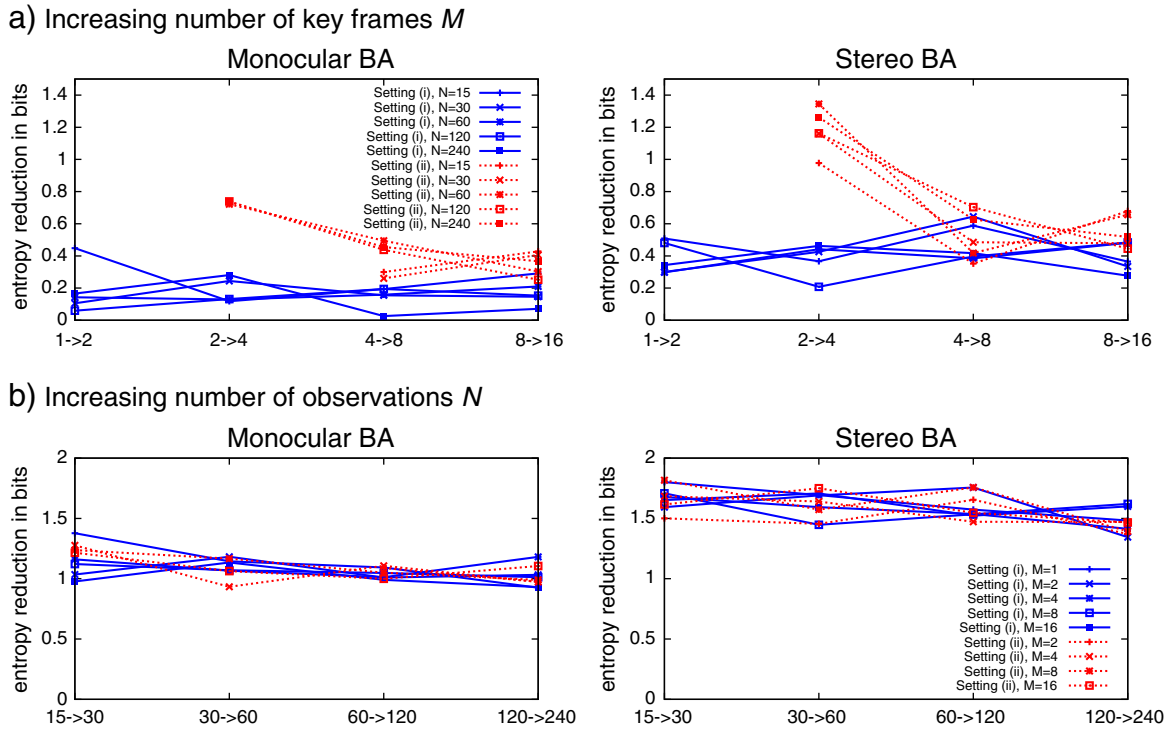

Fig. 6. Relative entropy reduction when (a) we double the number of intermediate frames and when (b) we double the number of observations. Note the difference between setting (i) (blue, connected lines) and setting (ii) (red, dotted lines).

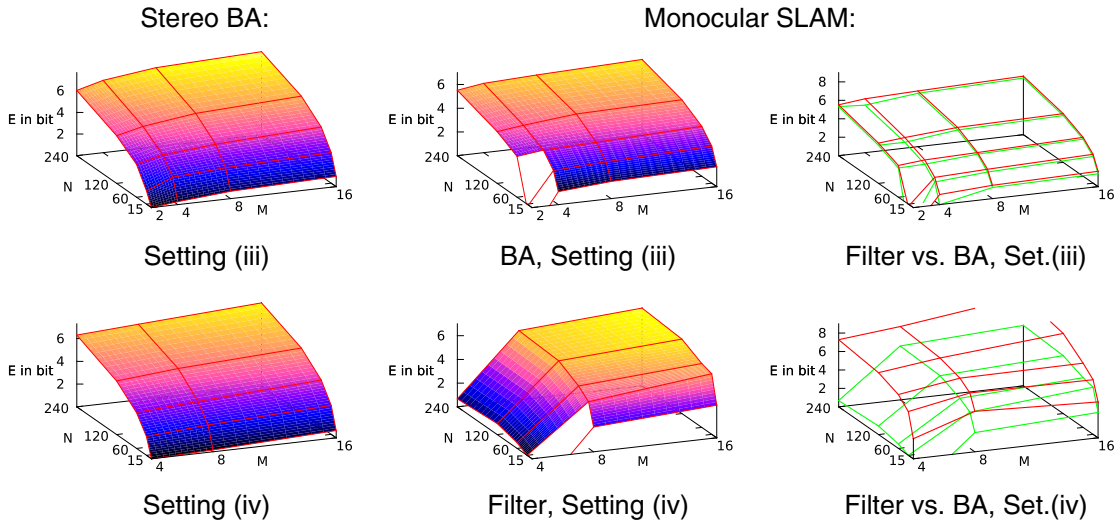

Fig. 7. Accuracy plots in terms of entropy reduction in bits. The stereo filter leads to very similar result than stereo BA and is therefore not shown here.

setting (i) and setting (ii). Let us consider the relative entropy reduction when we double the number of intermediate frames, i.e., comparing $\Sigma_{\langle M,N\rangle}$ with $\Sigma_{\langle 2M,N\rangle}$. From Fig. 6(a), one can clearly see that setting (ii) benefits more from the increased number of keyframes than setting (i). This effect is especially prominent for monocular SLAM. While all points are visible in all frames in setting (i), the scene overlap is larger for more closely placed keyframes in setting (ii). Increasing the number of observations per frame has a similar impact on both settings (Fig. 6(b)).

The second error measure we use is the root mean square error (RMSE):

$$R = \sqrt{\frac{1}{k}\sum_{k=0}^{k} \Delta \mathbf{t}_k^2} \tag{24}$$

where $k = 500$ is the number of Monte Carlo trials. Compared to the entropy reduction, this is a measure which is not relative but absolute. It is still meaningful for non-Gaussian and non-zero-centred error distributions. The RMSE for setting (i) is illustrated in Fig. 4(e, f). In the case that the error distributions are zero-mean Gaussians, entropy reduction and RMSE behave very similarly: they are anti-monotonic to each other. Our main reason for concentrating on entropy reduction is to make our analysis comparable to [48], in which the experiments were performed using covariance propagation and other error metrics such as RMSE were not applicable.

Accuracy plots for the two motion cases with rotational components, settings (iii and iv), are shown in Fig. 7. One can see that the result of setting (iii) is comparable to setting (ii). This is not surprising since both settings lead to a similar amount of scene overlap. Again, the two low accuracy cases $\langle 2,15\rangle$ and $\langle 2,30\rangle$ lead to unstable results and are excluded. For both rotational cases, setting (iii) and setting (iv), the stereo filter approaches the accuracy of stereo BA. However, for the difficult case, monocular vision in setting (iv), the results are different. BA leads to significant better results than filtering. Especially for a low number of frames, the performance of the filter is worse. We only removed the very inaccurate case $\langle 2,15\rangle$, since it is not practical to excluding all non-robust cases. Even for many features and frames, e.g. $\langle 16,240\rangle$, the error distributions are slightly heavy tailed. This low level of robustness might also explain the slightly chaotic, non-monotonic behaviour of the accuracy plots. Thus, conclusions drawn from setting (iv) have to be dealt with care.

6.3. The cost and accuracy of motion-only estimation for filter-SLAM

As described in Section 5.4, when performing filter-SLAM there are two main options to perform motion-only estimation. Either one can do motion-only BA by minimising Eq. (6) or one can also consider the map uncertainty. While motion-only BA is linear in the number of points N, pose estimation using a Gaussian map prior is cubic in N due to the inversion of innovation matrix S. In an approximated but much more efficient version of this algorithm, the innovation matrix S and its inverse are only calculated once. For stereo SLAM, we can usually measure the 3D points precisely so that motion-only BA leads to accurate results. However, for a monocular filter where the point depth is

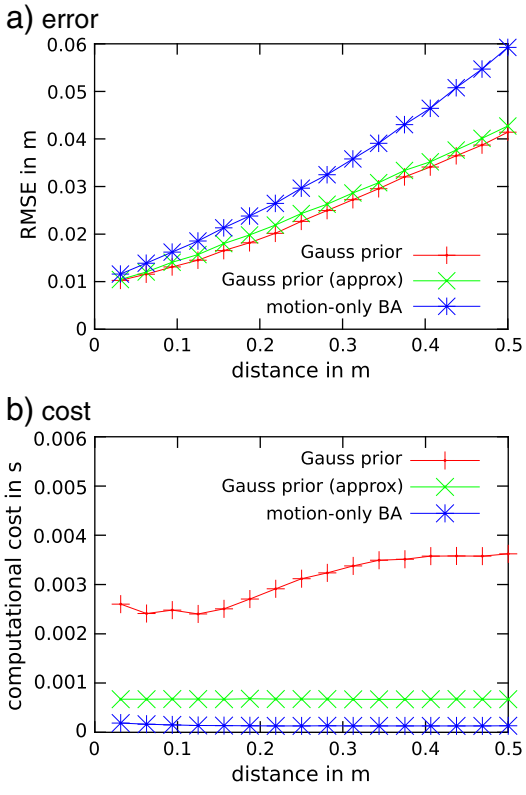

Fig. 8. Pose update given known map estimated by monocular filter.

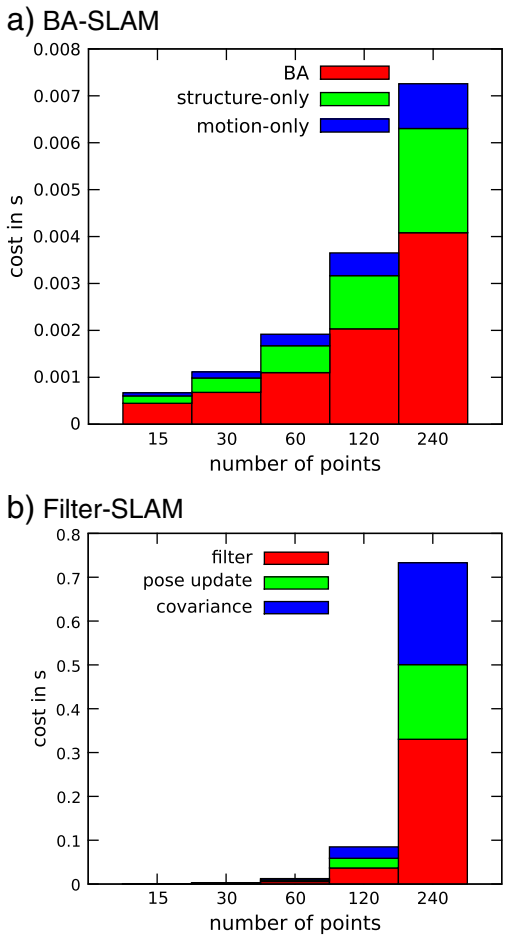

Fig. 9. Computational cost of monocular SLAM.

uncertain, it is beneficial to consider this uncertainty explicitly (Fig. 8(a)). Considering map uncertainty in pose estimation leads to a significant increase in computation time (Fig. 8(b)). In the monocular filtering experiments, we use the approximated version of the algorithm.

6.4. The cost of visual SLAM

Under the assumption that all points are visible in all frames, the cost of BA is $O(NM^2+M^3)$, where the first term reflects the Schur complement, while the second term represents the cost of solving the reduced linear system [18]. The costs of structure-only and motion-only estimation are both linear in the number of points. In filtering, the filter update is cubic in the number of observations, which leads to $O(MN^3)$ for the whole trajectory. The cost of pose update given a map is either linear or cubic (see previous section). The cost of the whole SLAM pipelines for varying number of points N is shown in Fig. 9. Here we illustrate the case of $M=1$, setting (i) and monocular SLAM.

6.5. Trade-off of accuracy versus cost

We would like to analyse the efficiency of BA and filtering for visual SLAM by trading off accuracy against computational cost.[4] First, we do this using the combined accuracy/cost measure that we also employed in our original analysis [48]. Thus, we evaluate visual SLAM using entropy by cost in terms of bits per seconds (bps): $\frac{E}{c}$. E is the amount of entropy reduction as defined in Eq. (23) and c is the average computational cost in seconds of the whole SLAM pipeline. Corresponding plots are shown in Fig. 10. First one can see that BA seems to be in general more efficient than filtering. Furthermore, there is a pattern that BA is especially efficient for small M, while filtering is only efficient for low accuracy (small M and small N).

Finally, we contrast error with cost in common plots in Fig. 11. Each curve shows the error and cost for a constant number of frames M and varying number of observations N. For the lowest number of frames (bold curves), we also show results for $N=480$. In these plots the bottom left corner is the desired area, where we find the highest accuracy and lowest computational cost. For all four settings, we can observe that BA is clearly inferior to filtering. Furthermore, we see that for setting (i) it is always preferable to choose the lowest number of frames. This is still the case for sideways motion BA with partial scene overlap (settings (ii–iii)) — except for the monocular, low-robustness cases ($M=2$, and $N\in\{15, 30\}$) which are not shown in the plots. However, for filtering (settings (ii–iii)), there is actually a cross-over. In order to reach high accuracy, it seems desirable to increase the number of keyframes M. The monocular setting (iv), low parallax and low scene-overlap, is the most challenging one. The inaccurate case $M=4$ results in a RMSE greater than 0.02 m and is therefore not shown. Here, BA outperforms filtering by magnitudes, but increasing M helps the filter. To summarise, it is usually a good strategy to increase the number of points N. Increasing the number of keyframes M seems only to be sensible if both following requirements are fulfilled: first, we use filtering instead of BA, and thus there is significantly higher cost with respect to N compared to M. Second, there is a varying scene overlap which can be maximised with increasing M.

7. Discussion

We have shown that filter-SLAM can indeed reach the accuracy of BA for moderately difficult motion patterns and scene structures (settings (i–iii)), even if we only filter sparse keyframes. In general, increasing the number of points N leads to a significant increase in accuracy, while increasing the number of frames M primarily establish robustness. Once a level of robustness is reached, a further increase of M has only a minor effect. This shows that the greater efficiency of BA compared to filtering for local SLAM is primarily a cost argument: the cost of BA is linear in N, whereas the cost of filtering is cubic in N. For the sharp forward turn (setting (iv)) using monocular vision, our analysis is slightly different. It illustrates the known problem of Gaussian filters. Since measurement Jacobians are not relinearised, the accuracy can significantly decrease compared to BA. Note, however, that the amount of insight we can gain from setting (iv) is limited. It might be possible to find a better filter parameterisation/implementation which can deal significantly better with this low parallax case. Setting (iv) is merely added as an illustrative example that accuracy of filtering can be inferior to BA, even for very short trajectories. Here, the dominance of BA compared to filtering is primarily an accuracy argument.

The greater cost of a filtering wrt. BA is mainly due to the fact that we represent uncertainties explicitly. In this work, we focused on the SLAM-backend and we did not analyse the accuracy and cost of feature tracking. Instead, we assumed that a perfect data association is given. On the one hand, the availability of the covariance can facilitate features tracking [38,10,4]. On the other hand, modern tracking techniques such as variational optical flow [52] do not require covariances, and are very effective. For the SLAM-backend, it does not seem beneficial to propagate uncertainties explicitly. Thus, one should only calculate covariances if one needs them elsewhere.

[4] In order to assume the best case for filtering, we do not consider covariance estimation and pose estimation given a known map. Thus, we compare the cost of joint BA updates against the cost of the joint filtering steps for the whole trajectory.

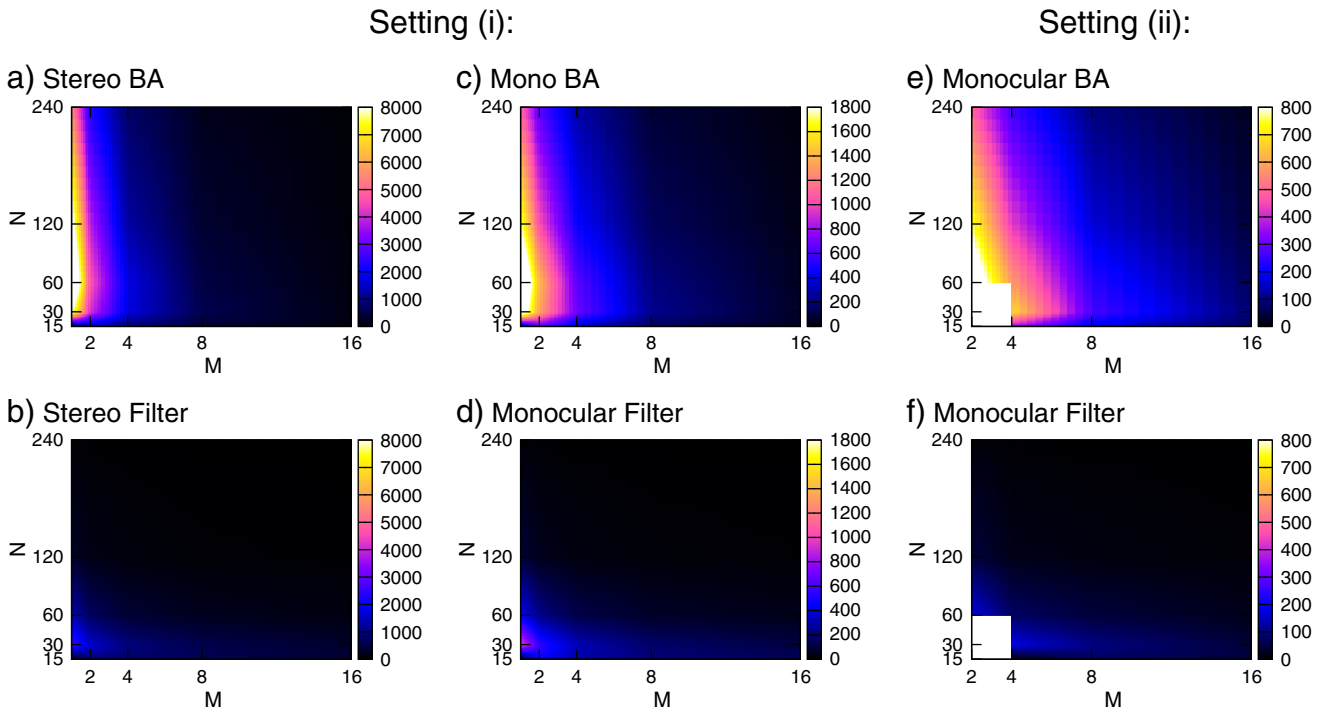

Fig. 10. Accuracy/cost measure in bits per second (bps).

In addition, we did not focus on all aspects of SLAM in our analysis. We intentionally did not consider large-scale SLAM and loop-closing since these issues have been intensively studied in the past. A SLAM framework which works reliably locally, whether it is BA or filtering, can easily be applied to a large scale problems using methods such as sub-mapping or graph-based global optimisation. Furthermore, it was shown recently that loop-closing can be solved efficiently using appearance-based methods [41,8] which can be formulated independently from metric SLAM systems. Thus, we assume in our analysis that the choice between BA and filtering is not relevant at this global level.

7.1. Rao-Blackwellised particle filters for visual SLAM

In our analysis, we concentrated on Gaussian filters and BA. Other approaches for SLAM are based on Rao-Blackwellised particle filters (RBPF) [35] such as Sim et al.'s stereo framework [47] and Eade and Drummond's monocular framework [16]. Eade and Drummond superseded the RBPF framework with their filter-based sub-mapping approach [17]. We believe that particle filters are a wasteful way of representing distributions which are unimodal and approximately Gaussian. Still, an elaborate comparison of RBPFs to BA would be an interesting topic for future work.

7.2. Middle ground between BA and filtering

While we focussed on the two extreme cases, there is a broad middle ground between filtering and BA.[5] Let us reconsider the three properties of our filter concept defined in Section 3. While all three properties are inherently coupled for the EKF, information filters can deal with them independently. Let us lift property 2: indeed, if we never marginalise out past poses and invisible features, we keep the corresponding information matrix relatively sparse, thus leading to the class of exactly sparse information filters [19]. Imagine the corresponding Markov random field: in general, all point observations are connected to several poses: the anchor pose and the observer poses. However, no point is directly connected to a point and no pose is connected to a pose. This leads to a similar, but slightly different sparseness structure than standard BA. In BA the corresponding Jacobian has one frame block and one point block per row (=observation), while the Jacobian of our sparse filter has several frame blocks and one point block per row. Still, the point block of the information matrix remains block-diagonal, the Schur-complement would be applicable, and the algorithmic complexity would decrease to the level of BA. However, there are two caveats. First, if we compute the covariance $\Sigma = \Lambda^{-1}$, the performance benefit would vanish. Thus, we do not have cheap access to the covariance (=forfeit property 3), and therefore lose the main advantage of Gaussian filters. Second, the Jacobians are only linearised once and the update of the information matrix remains additive. Thus, this exactly sparse filter remains inferior to BA.

Another option is to follow the approach of Sibley et al. [45] and partially lift property 1. We represent some variables using a Gaussian, while others are represented as in BA. In particular, it is sensible to deal with a sliding window of the last current poses using batch processing. All corresponding observations are saved, no uncertainties are maintained and the Jacobians are constantly re-linearised. We represent variables outside this sliding window using a Gaussian distribution, assuming they are well estimated so no further re-linearisation is necessary. This sliding window approach basically performs BA for local motion estimates, and is therefore covered by our analysis.

Typically in BA-SLAM and opposed to filtering, the SLAM problem is solved from scratch each time a new note is added to the graph. Kaess et al. [25] introduced a framework for incremental BA using variable reordering and just-in-time relinearisation. For large scale mapping, this framework can have a lower computational cost than batch BA. However, it remains unclear whether there is a significant performance benefit for local SLAM.

[5] Strictly speaking, the Gauss–Newton filter, which we used in the comparison, is already one step towards BA. Some poses – the anchor poses $A_1,\ldots,A_k$ – are not marginalised out; so their means get constantly re-estimated. In addition, the current pose T_i does not have a motion prior and is therefore 'bundle adjusted'.

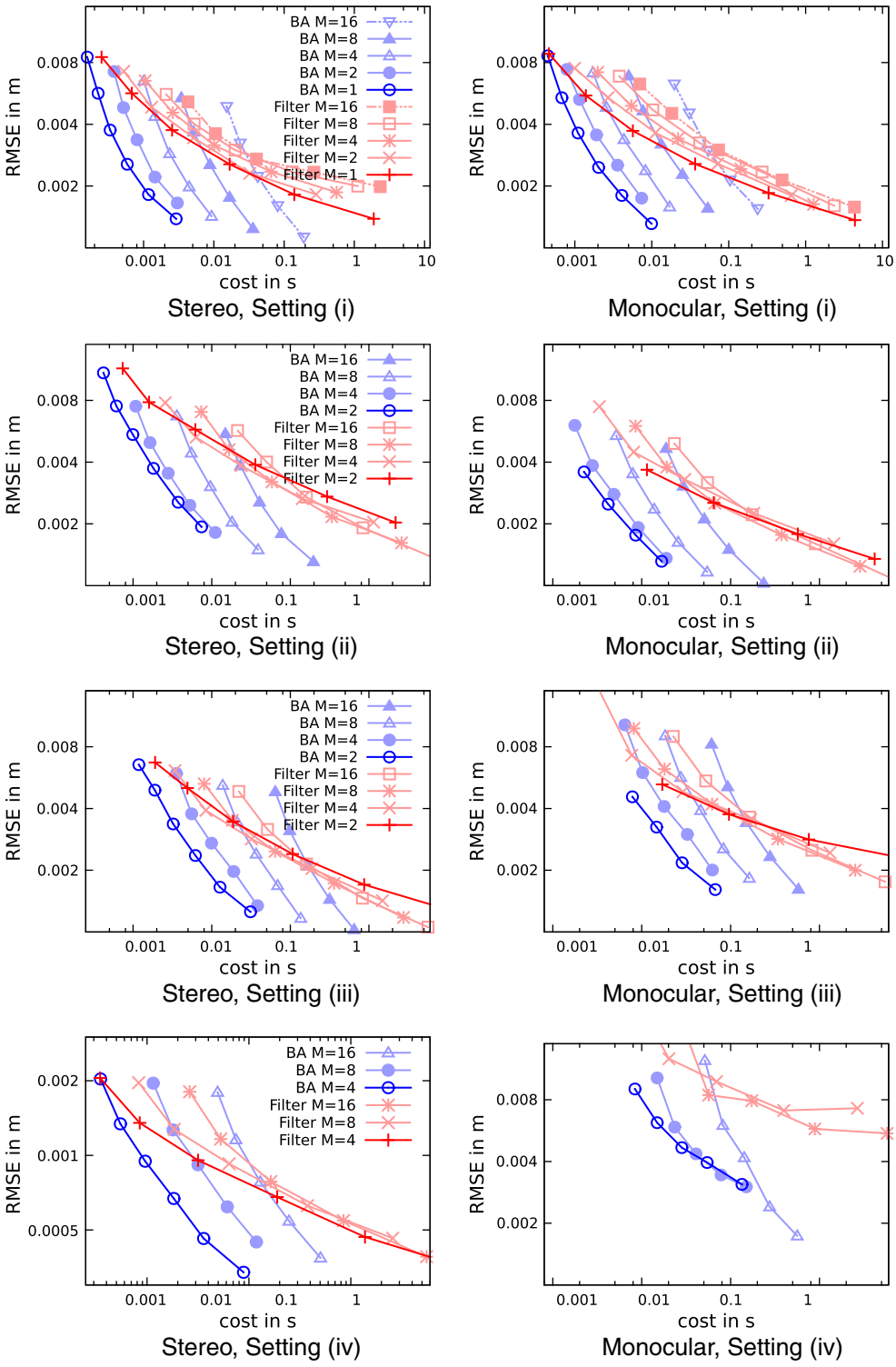

Fig. 11. Error versus cost on a logarithmic scale.

8. Conclusion

In this paper, we have presented a detailed analysis of the relative merits of filtering and bundle adjustment for real-time visual SLAM in terms of accuracy and computational cost. We performed a series of experiments using Monte Carlo simulations for motion in local scenes. Compared to our previous work [48], we lifted several assumptions by considering partial scene overlap, full SLAM pipelines including monocular bootstrapping and feature initialisation, and stereo SLAM. Nevertheless, our conclusion remains: in order to increase the accuracy of visual SLAM it is usually more profitable to increase the number of features than the number of frames. This is the key reason why BA is more efficient than filtering for visual SLAM. Although this analysis delivers valuable insight into real-time visual SLAM, there is space for further work. In this analysis we assumed known data association. However, the accuracy of a SLAM backend such as BA is highly coupled with the performance of the visual front-end — the feature tracker. A detailed analysis of this coupling would be worthwhile.

Acknowledgements

This research was supported by the European Research Council Starting Grant 210346, the Spanish MEC Grant DPI2009-07130 and EU FP7-ICT-248942 RoboEarth. We are grateful to our close colleagues at Imperial College London and the Universidad de Zaragoza for many discussions.

Appendix A. Entropy reduction

The differential entropy of a Gaussian $X = \langle \mu_X, \Sigma_X \rangle$ is defined as:

$$H(X) = \frac{1}{2} log_2\Big((2\pi e)^N det(\Sigma_X)\Big). \tag{A.1}$$

Now, the difference between two Gaussians $X = \langle \mu_X, \Sigma_X \rangle$ and $Y = \langle \mu_Y, \Sigma_Y \rangle$ can be described using the difference of entropy:

$$E(X,Y) := H(X) - H(Y). \tag{A.2}$$

If $H(X) > H(Y)$ this can be seen as an entropy reduction measure: How much more accuracy do we gain, if we do Y instead of X. It holds that:

$$E(X,Y) = H(X) - H(Y) \tag{A.3}$$

$$= \frac{1}{2} log_2\Big((2\pi e)^N det(\Sigma_X)\Big) - \frac{1}{2} log_2\Big((2\pi e)^N det(\Sigma_Y)\Big) \tag{A.4}$$

$$= \frac{1}{2} log_2\left(\frac{det(\Sigma_X)}{det(\Sigma_Y)}\right). \tag{A.5}$$

For numerical stability, the natural logarithms of the absolute values the determinants of Σ_X and Σ_Y are calculated directly, subtracted and normalised afterwards:

$$E(X,Y) = \frac{1}{2ln(2)} (ln|det(\Sigma_X)| - ln|det(\Sigma_Y)|). \tag{A.6}$$

Here, we use that the determinant of a covariance matrix is always positive.

References

[1] A. Azarbayejani, A.P. Pentland, Recursive estimation of motion, structure, and focal length, IEEE Trans. Pattern Anal. Mach. Intell. 17 (6) (1995) 562–575.
[2] B.M. Bell, F.W. Cathey, The iterated Kalman filter update as a Gauss–Newton method, IEEE Trans. Autom. Control. 38 (2) (1993) 294–297.
[3] A. Chiuso, P. Favaro, H. Jin, S. Soatto, Structure from motion causally integrated over time, IEEE Trans. Pattern Anal. Mach. Intell. 24 (4) (2002) 523–535.
[4] M. Chli, A.J. Davison, Active matching for visual tracking, Robot. Auton. Syst. 57 (12) (2009) 1173–1187 (Special Issue 'Inside Data Association').
[5] J. Civera, A.J. Davison, J.M.M. Montiel, Inverse depth parametrization for monocular SLAM, IEEE Trans. Rob. 24 (5) (2008) 932–945.
[6] J. Civera, O. Grasa, A.J. Davison, J.M.M. Montiel, 1-point RANSAC for EKF filtering. Application to real-time structure from motion and visual odometry, J. Field Rob. 27 (5) (2010) 609–631.
[7] L.A. Clemente, A.J. Davison, I. Reid, J. Neira, J.D. Tardós, Mapping large loops with a single hand-held camera, Proceedings of Robotics: Science and Systems (RSS), 2007.
[8] M. Cummins, P. Newman, Highly scalable appearance-only SLAM — FAB-MAP 2.0, Proceedings of Robotics: Science and Systems (RSS), 2009.
[9] A.J. Davison, Real-time simultaneous localisation and mapping with a single camera, Proceedings of the International Conference on Computer Vision (ICCV), 2003.
[10] A.J. Davison, Active search for real-time vision, Proceedings of the International Conference on Computer Vision (ICCV), 2005.
[11] A.J. Davison, N.D. Molton, I. Reid, O. Stasse, MonoSLAM: real-time single camera SLAM, IEEE Trans. Pattern Anal. Mach. Intell. 29 (6) (2007) 1052–1067.
[12] M. Deans, M. Herbert, Experimental comparison of techniques for localization and mapping using a bearing-only senser, Experimental Robotics VII, 2001, pp. 395–404.
[13] F. Dellaert, Square root SAM, Proceedings of Robotics: Science and Systems (RSS), 2005.
[14] P. Dyer, S. McReynolds, Extension of square-root filtering to include process noise, J. Optim. Theory Appl. 3 (6) (1969) 444–458.
[15] E. Eade. Monocular Simultaneous Localisation and Mapping. PhD thesis, University of Cambridge, 2008.
[16] E. Eade, T. Drummond, Scalable monocular SLAM, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2006.
[17] E. Eade, T. Drummond, Monocular SLAM as a graph of coalesced observations, Proceedings of the International Conference on Computer Vision (ICCV), 2007.
[18] C. Engels, H. Stewénius, D. Nistér, Bundle adjustment rules, Proceedings of Photogrammetric Computer Vision, 2006.
[19] R.M. Eustice, H. Singh, J.J. Leonard, Exactly sparse delayed state filters, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2005.
[20] J. Gallier, Geometric Methods and Applications for Computer Science and Engineering, Springer-Verlag, 2001.
[21] C.G. Harris, J.M. Pike, 3D positional integration from image sequences, Proceedings of the Alvey Vision Conference, 1987, pp. 233–236.
[22] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, second edition Cambridge University Press, 2004.
[23] Y. Jeong, D. Nister, D. Steedly, R. Szeliski, I.S. Kweon, Pushing the envelope of modern methods for bundle adjustment, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010, pp. 1474–1481.
[24] S.J. Julier, K. Jeffrey Uhlmann, A counter example to the theory of simultaneous localization and map building, IEEE International Conference on Robotics and Automation, 2001.
[25] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert. iSAM2: Incremental smoothing and mapping using the Bayes tree. Int. J. Rob. Res. (IJRR), 2012. To appear.
[26] M. Kaess, A. Ranganathan, F. Dellaert, iSAM: incremental smoothing and mapping, IEEE Trans. Rob. 24 (6) (2008) 1365–1378.
[27] G. Klein, D.W. Murray, Parallel tracking and mapping for small AR workspaces, Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR), 2007.
[28] G. Klein, D.W. Murray, Parallel tracking and mapping on a camera phone, Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR), 2009.
[29] K. Konolige, M. Agrawal, FrameSLAM: from bundle adjustment to real-time visual mapping, IEEE Trans. Rob. 24 (2008) 1066–1077.
[30] R. Kuemmerle, G. Grisetti, H. Strasdat, K. Konolige, W. Burgard, g2o: a general framework for graph optimization, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2011.
[31] Jongwoo Lim, Marc Pollefeys, Jan-Michael Frahm, Online environment mapping, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011.
[32] R. Mahony, J.H. Manton, The geometry of the Newton method on non-compact Lie groups, J. Glob. Optim. 23 (3) (2002) 309–327.
[33] P. McLauchlan, I. Reid, D. Murray, Recursive affine structure and motion from image sequences, Proceedings of the European Conference on Computer Vision (ECCV), 1994.
[34] C. Mei, G. Sibley, M. Cummins, P. Newman, I. Reid, RSLAM: a system for large-scale mapping in constant-time using stereo, Int. J. Comput. Vision 94 (2011) 198–214.
[35] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, FastSLAM: a factored solution to the simultaneous localization and mapping problem, Proceedings of the AAAI National Conference on Artificial Intelligence, 2002.
[36] J.M.M. Montiel, J. Civera, A.J. Davison, Unified inverse depth parametrization for monocular SLAM, Proceedings of Robotics: Science and Systems (RSS), 2006.
[37] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, P. Sayd, Real-time localization and 3D reconstruction, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2006.
[38] J. Neira, J.D. Tardós, Data association in stochastic mapping using the joint compatibility test, IEEE Trans. Robot. Autom. 17 (6) (2001) 890–897.
[39] D. Nistér, An efficient solution to the five-point relative pose problem, IEEE Trans. Pattern Anal. Mach. Intell. 26 (6) (2004) 756–777.
[40] D. Nistér, O. Naroditsky, J. Bergen, Visual odometry, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2004.
[41] D. Nister, H. Stewenius, Scalable recognition with a vocabulary tree, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2006.
[42] T. Pietzsch, Efficient feature parameterisation for visual SLAM using inverse depth bundles, Proceedings of the British Machine Vision Conference (BMVC), 2008.
[43] P. Pinies, J.D. Tardós, Large scale SLAM building conditionally independent local maps: application to monocular vision, IEEE Trans. Rob. 24 (5) (2008) 1094–1106.
[44] G. Sibley, L. Matthies, G. Sukhatme, Bias reduction filter convergence for long range stereo, 12th International Symposium of Robotics Research, 2005.
[45] G. Sibley, L. Matthies, G. Sukhatme, A sliding window filter for incremental SLAM, Unifying Perspectives in Computational and Robot Vision, 2008, pp. 103–112.
[46] G. Sibley, C. Mei, I. Reid, P. Newman, Adaptive relative bundle adjustment, Proceedings of Robotics: Science and Systems (RSS), 2009.
[47] R. Sim, P. Elinas, M. Griffin, J.J. Little, Vision-based SLAM using the Rao-Blackwellised particle filter, Proceedings of the IJCAI Workshop on Reasoning with Uncertainty in Robotics, 2005.
[48] H. Strasdat, J.M.M. Montiel, A.J. Davison, Real-time monocular SLAM: why filter? Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2010.
[49] H. Strasdat, J.M.M. Montiel, A.J. Davison, Scale drift-aware large scale monocular SLAM, Proceedings of Robotics: Science and Systems (RSS), 2010.
[50] A.N. Tikhonov, V.I.A. Arsenin, Solutions of Ill-posed Problems, Winston, Washington, DC, 1977.
[51] B. Triggs, P. McLauchlan, R. Hartley, A. Fitzgibbon, Bundle adjustment — a modern synthesis, Proceedings of the International Workshop on Vision Algorithms, in Association with ICCV, 1999.
[52] M. Werlberger, T. Pock, H. Bischof, Motion estimation with non-local total variation regularization, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010.