Semi-Direct EKF-based Monocular Visual-Inertial Odometry

Petri Tanskanen[1], Tobias Naegeli[2], Marc Pollefeys[1] and Otmar Hilliges[2]

Abstract— **We propose a novel monocular visual inertial odometry algorithm that combines the advantages of EKF-based approaches with those of direct photometric error minimization methods. The method is based on sparse, very small patches and incorporates the minimization of photometric error directly into the EKF measurement model so that inertial data and vision-based surface measurements are used simultaneously during camera pose estimation. We fuse vision-based and inertial measurements almost at the raw-sensor level, allowing the estimated system state to constrain and guide image-space measurements. Our formulation allows for an efficient implementation that runs in real-time on a standard CPU and has several appealing and unique characteristics such as being robust to fast camera motion, in particular rotation, and not depending on the presence of corner-like features in the scene. We experimentally demonstrate robust and accurate performance compared to ground truth and show that our method works on scenes containing only non-intersecting lines.**

I. Introduction

The problem of estimating the motion of a camera relative to a known 3D scene from a set of images or RGB-D (RGB and depth) frames is one of the fundamental problems in computer vision and robotics. Estimating camera motion enables applications such as vehicle or robot localization [18], 3D reconstruction [23] and augmented reality [22]. Recently a number of approaches have leveraged dense RGB-D data, available in real-time from depth sensing cameras such as the Kinect [26], in combination with ICP-like algorithms for pose estimation [6], [8], [16].

Similar in philosophy but using monocular images only, methods for camera pose estimation using dense surface measurements have been demonstrated [12], [17], [25]. These methods use all data available for pose estimation and hence promise high tracking accuracy and robustness. However, they are computationally expensive and typically require powerful GPUs for real-time performance, prohibiting use in mobile and compute restricted setups.

Direct methods, minimizing photometric error for pose estimation, have recently been adapted to sparse formulations [3], [4] with great success. These methods offer higher precision and robustness than traditional feature extraction and tracking based methods [9] and, in the sparse variant, have comparable or better runtime performance.

Being purely vision based, these methods struggle under fast motion, in particular rotation [3], when the camera is moving along it's focal axis, and in scenes with few corner-like features [4]. Most direct photometric approaches are formulated as energy minimization problem and leverage Gauss-Newton like methods to solve for camera pose. Therefore, tightly coupling IMU and vision data is non-trivial in these frameworks.

On the other hand filter-based approaches to VIO [2], [10], [5] tightly couple inertial measurements with visual data and have demonstrated robustness to fast rotation, partial loss of visual tracking and relatively little drift over time. However, we are not aware of exisiting methods to incorporate direct methods (i.e., photometric error minimization) directly in the measurement model of the EKF framework.

In this paper we propose, to our best knowledge, for the first time an algorithm that combines the use of direct photometric error minimization in an extended kalman filter (EKF) framework. Allowing us to fuse vision and inertial data tightly, almost at the raw sensor level. Both signal sources measure the same motion but have different, complementary sensor characteristics which can provide additional constraints during the optimization camera pose. Fusing the complementary data sources at the lowest possible level allows the estimated system state to constrain and guide the image-space measurements, enforcing consistency between image-space feature positions and 6DOF camera motion. Our approach works with very few (10-20) and very small (as small as 3×3) image-patches. This sparsity allows for an efficient and fast implementation. Furthermore, the method can handle scenes that do not have any corner-like features and hence is suitable for scenarios in which other methods fail.

A. Related Work

1) Dense methods: Dense direct methods operate on surface measurements directly, either depth estimates of a stereo camera or a RGB-D sensor [8], [16], and do not extract sets of features from this data. These approaches require heavy GPU parallelization due to computational cost and tend to have restricted working ranges, due to sensor working principles. Dense monocular methods do not have special sensor requirements but have similar computational costs because they require the build-up of an explicit cost volume[17] or on computing constrained scene flow [15].

2) Semi-dense direct methods: Recently [3] proposed to estimate depth only for pixels in textured image areas and introduce an efficient epipolar search, enabling real-time visual odometry and semi-dense point cloud reconstruction on a standard CPU and even on mobile platforms [22]. Photometric alignment on sparse, known 3D points has been

[1] Petri Tanskanen and Marc Pollefeys are with the Computer Vision and Geometry Group, Department of Computer Science, ETH Zurich

[2] Tobias Naegeli and Otmar Hilliges are with the Advanced Interactive Technologies Lab, Department of Computer Science, ETH Zurich

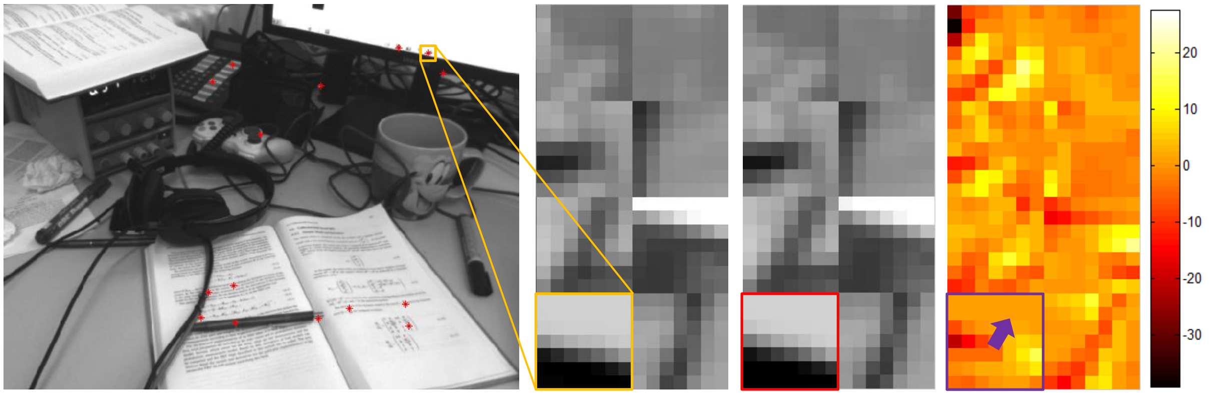

Fig. 1. The left image shows the pixel patches selected for odometry computation on the current camera image. The middle two images show a selection of the pixel patches in the current image and the respective reference patches. The algorithm is optimizing the camera pose and the patch depth by minimizing the intensity residual. The rightmost image shows the intensity residuals with an arrow illustrating the patch motion that is needed to align both patches resulting from the image gradient of the current image.

used by [4] to improve accuracy and robustness of the standard SLAM pipeline of [9]. Most of these approaches either do not use inertial data or treat both data sources mostly independently and only fuse the two at the camera pose level. For example, to estimate metric scale on top of vision based camera pose [24].

3) Visual Inertial Odometry: The EKF framework has been used for vision only camera tracking and structure from motion [2]. It allows for straight forward sensor fusion and hence it is very popular for algorithms designed with mobile platforms in mind, which predominantly are shipped with cameras and IMUs [5], [10]. However, to make the problem computationally tractable typically EKF approaches operate on sets of image-space features. As outlined above this comes with certain issues. In the filtering context a further issue is that they are uncoupled from the estimated system. It is only possible to use predicted locations to support the feature correlation or matching but the correlation itself is completely unconstrained by the overall system state. This requires costly outlier rejection (e.g., RANSAC) to detect features that where not matched or tracked correctly.

To improve feature correlation results, early SLAM approaches have used photometric error and patch-wise normal estimation [13] to improve feature correlation but this was done separately from the standard EKF-SLAM steps. Instead of externally optimizing the homography between filter updates, [7] estimates the patch normal inside the EKF framework. The drawback with these methods is that the local patches have to be reasonably large (25×25 pixel or larger) for the normal to be estimated robustly. This increases computational cost and introduce problems with patches near depth discontinuities, where the texture in a patch would not change consistently with camera motion.

B. Contribution Statement

In this paper we propose a method that based on sparse, very small patches and incorporates the minimization of photometric error directly into the EKF measurement model so that inertial data and vision-based surface measurements are used simultaneously during camera pose estimation. Our formulation allows for an efficient implementation that runs in real-time on a standard CPU and could be implemented on mobile platforms as well. The tight integration of direct surface measurements and inertial data allows to track image regions that are difficult to tackle with approaches that rely on feature trackers like KLT for example line-like structures in images.

C. System Overview

Our technique is a visual-inertial odometry (VIO) approach, this means that camera pose is estimated only from currently visible regions of the observed 3D scene and we do not maintain a global map of previously extracted feature points, we remove all features from the state space as soon as they leave the field of view of the camera. Note that the proposed approach could easily be extended with standard mapping back-end as for example in [9]. Following the approach in [14] We reformulate the EKF framework which has been used successfully for structure from motion [2] into an *Error State Extended Kalman Filter* ErKF.

Fig. 1 illustrates our approach. A small number of small patches were extracted in previous frames and the corner locations of the patches are projected into a predicted camera pose based on IMU data. An affine warp for the whole patch is computed (cf. Fig. 3). The algorithm then jointly optimizes the camera pose and the patch depth by minimizing the intensity residual. One advantage of this approach is that we do not rely on the extraction of features of a specific type (e.g., corners) but can use any patches with sufficient gradient. In particular, patches which lie on lines (see highlighted region in Fig. 1) or they can be placed in image areas with good texture, similar to the pixel selection in (semi-)dense approaches [3]. Furthermore, we use an inverse depth parametrization for the patch depth which allows us to start tracking without a special initialization sequence as it is necessary with other approaches [4], [9].

II. Error State Filter Design

A. Statespace structure

The camera state $x_c = [p, q_{wc}, v, o_a, o_\omega, t_d]^T \in \mathbb{R}^{16}$ contains the current camera position p, orientation quaternion q_{wc}, linear velocity v, the accelerometer and gyroscope offsets o_a and o_ω and t_d is the time delay between IMU and camera measurements [11]. The point state vector x_m contains the states for the tracked patches. The whole state is then $x = [x_c, x_m]$. We use a error state formulation $\tilde{x} = x - \hat{x}$ which is defined as the difference between the true state x and the estimated state $\hat{x}$. The error state vector is defined as $x_c = [\tilde{p}, \tilde{\theta}, \tilde{v}, \tilde{o}_a, \tilde{o}_\omega, \tilde{t}_d]^T \in \mathbb{R}^{15}$, see [14] for more details. We used a static calibration for the transformation between camera and the IMU, we want to point out that it is possible to include online camera-IMU calibration by following [10].

B. Point Parametrization

The estimated points are parametrized as anchored inverse depth bundles [19]. For every time step where new patches are initialized, the point state vector x_m is augmented with $x_{new} = [p_k, q_k, \rho_{init}, ..., \rho_{init}]^T$ where p_k and q_k are the current camera pose and ρ_{init} the inverse depths which are set to an arbitrary value. In addition to the point state vector the location of each patch in normalized image coordinates in the anchor frame is stored statically in a vector m. The 3D position of a point can be computed as follows:

$$p_i = p_f + \frac{m_i}{\rho_i} R(q_f) \in \mathbb{R}^3 \tag{1}$$

where p_f is the position and $R(q_f)$ the orientation of the according anchor frame and ρ_i the inverse depth of the point.

C. Continuous Time Model

The nonlinear process model follows the standard formulation of [14].

$$\underbrace{\begin{bmatrix} \dot{\tilde{p}} \\ \dot{\tilde{q}} \\ \dot{\tilde{v}} \\ \dot{\tilde{o}}_\omega \\ \dot{\tilde{o}}_a \end{bmatrix}}_{x_{c_{k+1}}} = \underbrace{\begin{bmatrix} v_k \\ q_k \times q(z_\omega - o_{\omega k} + q_\omega) \\ R_{wc}(q_k)(z_a - o_a + q_a) \\ q_{o_\omega} \\ q_{o_a} \end{bmatrix}}_{f(x_k, q_k, u_k)} \tag{2}$$

with $q = [q_a, q_\omega, q_{o_\omega}, q_{o_a}]$ the process noise.

The jacobians of the process model used in the EKF are given as

$$\mathbf{F} = \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_{k|k}, z_\omega, z_a}, \quad \mathbf{G} = \left.\frac{\partial f}{\partial q}\right|_{\hat{x}_{k|k}, z_\omega, z_a}. \tag{3}$$

The 3D points are modelled as static scene points assuming that they do not move in the 3D space. Therefore, the feature space dynamics are given as $\dot{\tilde{p_{f_i}}} = 0$, $\dot{\tilde{q_{f_i}}} = 0$ and $[\dot{\tilde{\rho}}_1 \ldots \dot{\tilde{\rho}}_N] = 0$.

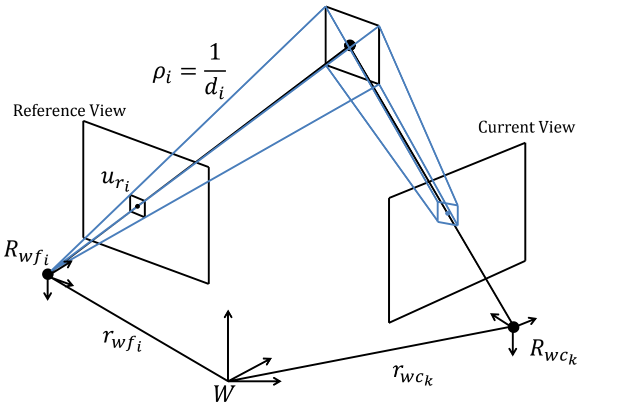

Fig. 3. Estimating per-pixel intensity differences. Given a reference camera pose $R_{wf_i}|r_{wf_i}$ at time i and predicted camera pose $R_{wc_k}|r_{wc_k}$ the center location u_{r_i} of a patch in the reference view is projected into 3D world coordinates and re-projected into the current view. We compute an affine warp to transform the pixel coordinates of all pixels in the small patch around the point location in the current camera view into the reference view. The per-pixel intensity value differences form the residual to minimize.

D. Prediction

Following the continuous-discrete hybrid approach suggested in [21] we perform a 4^{th} order Runga-Kutta integration of the continuous motion equations given in II-C. The error covariance $\mathbf{P} = \left[\begin{smallmatrix} \mathbf{P}_{CC} & \mathbf{P}_{CM} \\ \mathbf{P}_{MC} & \mathbf{P}_{MM} \end{smallmatrix}\right]$ is propagated by:

$$\mathbf{P}_{k+1|k} = \begin{bmatrix} \mathbf{P}_{CC_{k+1|k}} & \mathbf{\Phi}(t_{k+1}, t_k)\mathbf{P}_{CM_{k|k}} \\ \mathbf{P}_{MC_{k|k}}\mathbf{\Phi}(t_{k+1}, t_k)^\top & \mathbf{P}_{MM_{k|k}} \end{bmatrix} \tag{4}$$

The camera error covariance is numerically integrated by

$$\dot{\mathbf{P}}_{CC} = \mathbf{F}\mathbf{P}_{CC} + \mathbf{P}_{CC}\mathbf{F}^\top + \mathbf{G}\mathbf{Q}\mathbf{G}^\top \tag{5}$$

where $\mathbf{Q}$ represents the process noise and $\mathbf{\Phi}(t_{k+1}, t_k)$ is integrated by

$$\dot{\mathbf{\Phi}}(t_k + \tau, t_k) = \mathbf{F}\mathbf{\Phi}(t_k + \tau, t_k), \tau \in [0, T]. \tag{6}$$

III. Photometric Update

The photometric update is different from standard visual odometry approaches that use 2D image positions from an external feature tracker or matcher. In our case the measurement model $h(x)$ is used to directly predict the appearance of a pixel patch (the 1 dimensional intensity values of the pixels) of a reference view given the pixel values in the current camera view (see Figure 3).

More specifically, for every pixel patch the current estimate of the 3D location of the center pixel is transformed into the current camera frame:

$$h_c(x) = R_{ic_k} R_{cw_k}(\rho_i(r_{wf_i} - r_{wc_k}) + R_{wf_i}\pi^{-1}(u_{r_i})) + r_{ci}\ , \tag{7}$$

where h_c is a vector from the predicted current camera center towards the 3D location of the patch center, R_{wf_i}, r_{wf_i} are the rotation and position of the reference view, R_{wc_k}, r_{wc_k} the predicted rotation and position of the current camera,

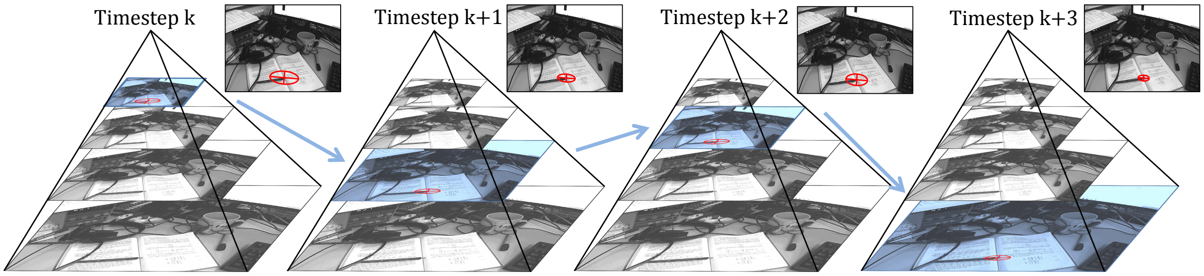

Fig. 2. The optimal pyramid level for pixel patch alignment is selected based on the estimated variance of the projected point location in pixel space. If the camera motion is fast, higher levels are used, if there is low variance on the camera pose lower levels are used for higher precision.

R_{ic_k}, r_{ci_k} the camera-IMU transformation, u_{r_i} is the stored center pixel of the tracked patch in the reference frame, ρ_i the inverse depth and π^{-1} is the camera back-projection function. Then the point is projected into image space with the precalibrated camera parameters. The measurement function $h(x)$ is then used to compute the appearance of the reference pixels given the current state estimates and the current camera image:

$$h(x) = I_k(\pi(h_c)) \ . \tag{8}$$

Here, the measurement model equation $h(x)$ is given for a single pixel. In order to increase robustness we extend the single measurement to a patch around this point. In doing so we make the assumption that the scene around this point is planar because only the depth of the single point is modeled. However, since we are using small patches (3×3pixels), the assumption of a locally flat scene can be made similar to [4]. To further reduce the degrees of freedom, we assume the patch normal to be orthogonal to the image plane in the anchor frame. This assumptions allows us to model the appearance of the pixels surrounding the center point via an affine warp A, encoding in-plane rotation of the patch, the depth dependent size of the patch and some shear caused by a camera observing the patch from a different angle.

The residual r_i, recursively minimized during camera pose estimation by the Kalman filter, is the photometric error between all the pixels in the reference patch and the pixels in the warped patch, extracted from the current camera view:

$$r_i = I_r(u_{r_i}) - I_k(A(R_{wf_i} r_{wf_i}, R_{wc_k} r_{wc_k}, u_{r_i}, \rho_i)) \ , \tag{9}$$

with I_k the current image, I_r the reference image, $R_{wf_i} r_{wf_i}$ and $R_{wc_k} r_{wc_k}$ the $[R|t]$ rotation and translations of the reference and the current view, u_{r_i} the location of the center pixel and ρ_i the inverse depth of the point. This residual is computed for all points that are currently in the state space.

Finally, the Kalman filter update step requires the linearization of the measurement function H, computed as the derivative of $h(x)$ with respect to the states x:

$$H = \frac{\partial h(x)}{\partial x} = \nabla I_k \frac{\partial \pi(h_c)}{\partial h_c(x)} \frac{\partial h_c(x)}{\partial x} \ , \tag{10}$$

with ∇I_k being the image gradient of the warped patch extracted from the current frame, $\frac{\partial \pi(h_c)}{\partial h_c(x)}$ is the 2×3 camera projection derivate matrix.

Assuming familiarity with the EKF framework, the equations given here and in the previous sections should be sufficient to implement the proposed algorithm. However, there are a number of details that can be taken into consideration in order to increase robustness in real-world settings and improve performance. We briefly discuss these in the following subsections.

A. *Patch extraction*

The patches can be selected using many different methods and in particular there is no requirement for patches to be centered on corners. In the experimental section we demonstrate the performance of our technique using only patches that are centered on single, non-intersecting lines. A simple implementation could just extract FAST keypoints [20] on an uniform grid. However, we noticed that selecting image areas based on the Shi-Tomasi score that are stable over the whole scale space of the image pyramid leads to better and more stable results.

B. *Image Pyramid Level Selection*

In our method we attain predictions and associated uncertainties for all state variables and their covariances. This can be used to compute the variance on the point location in image space and consequently allows to select the optimal level in the image pyramid such that convergence is guaranteed (see Fig. 2). Compared to the standard approach of iterating through the whole image pyramid starting from the highest level, this approach saves computation time while still offering the advantage of a larger convergence radius of the higher pyramid levels and the precision of the lower levels. In addition as the method selects the lowest possible level for convergence, it also reduces the risk of converging towards a wrong local minimum if the optimization on higher levels would not converge towards the correct image location.

The error covariance can be computed by omitting the image gradient when taking the derivative of the measurement function $\frac{\partial h}{\partial x}$:

$$H_\pi = \frac{\partial \pi}{\partial h_c} \frac{\partial h_c}{\partial x} \ , \tag{11}$$

$$S_\pi = H_\pi P_{k-1|k-1} H_\pi^\top \ . \tag{12}$$

The major axis of the error ellipsoid in image space is then the largest Eigenvalue of the 2×2 matrix S_π. To guarantee convergence the length of this axis should be smaller than 1 pixel at the respective pyramid level. These calculations can be done while computing the derivate H during the update step before the image gradient of the pixel patch is computed. The only overhead is the computation of the 2×2 matrix S and its Eigenvalues.

C. Iterated Sequential Update

Inherently our formulation requires the processing of many measurements for each update step (every pixel is a measurement). Unfortunately this impacts runtime performance. The size of the Jacobian $\frac{\partial h}{\partial x}$ and as consequence, the size of the innovation covariance matrix S will be $ns \times ns$, where n is the number of patches and s the patch size in pixels. Because S needs to be inverted during every EKF update step, the size of S directly impacts the runtime.

In the case of the linear Kalman filter, sequential updates [1] can be utilized to alleviate this situation. We observed that if one iteratively re-linearizes the measurement matrix $H = \frac{\partial h}{\partial x_{seq}}$ around each updated estimated state sequentially, the algorithm produces very good estimates in practice (see Alg. 1). The sequential update reduces the computations to n inversions of a $s \times s$ matrix which drastically enhances runtime performance.

Algorithm 1 Iterated Sequential EKF Update

```
Require: Proc. and meas. noise covariances: $Q$, $R$
  Prediction Step:
  $\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_{k-1})$
  $P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k$

  Update Step:
  $\hat{x}_{k|k,0} = \hat{x}_{k|k-1}, \; \hat{\tilde{x}}_{k|k,0} = \mathbf{0}$ and $P_{k|k,0} = P_{k|k-1}$
  for i do
      $S_{k,i} = \left( H_{k|k,i-1} P_{k|k,i-1} H_{k|k,i-1}^T + R \right)$
      $K_{k,i} = P_{k|k,i-1} H_{k|k,i-1}^T S_{k,i}^{-1}$
      $\hat{\tilde{x}}_{k|k,i} = \hat{\tilde{x}}_{k|k,i-1} + K_{k,i} \left( z_r - h(\hat{x}_{k|k,i-1}) \right)$
      $\hat{x}_{k|k,i} = \hat{x}_{k|k,i-1} \oplus \hat{\tilde{x}}_{k|k,i}$
      $P_{k|k,i} = \left( I - K_{k,i} H_{k|k,i-1} \right) P_{k|k,i-1}$
  end for
  $\hat{x}_{k|k} = \hat{x}_{k|k,n}$ and $P_{k|k} = P_{k|k,n}$
```

IV. Experimental Results

We performed a comparison against ground truth acquired from a Vicon system and two of recently published methods that use a photometric approach in a semi-dense manner [3] and patch-based on corner locations [4]. We moved the camera around a regular office space, see an example image from the dataset in Figure 4 on the left. The top plot shows the 3D view of the final positions of the tracked points during the sequence and the trajectory of our method together with ground truth. The third plot shows the position in all axes, as can be seen, our method tracks the camera pose in typical scenes with equal quality than the compared methods. The initialization for [3] was difficult in this particular scene and its performance did not match the expected level.

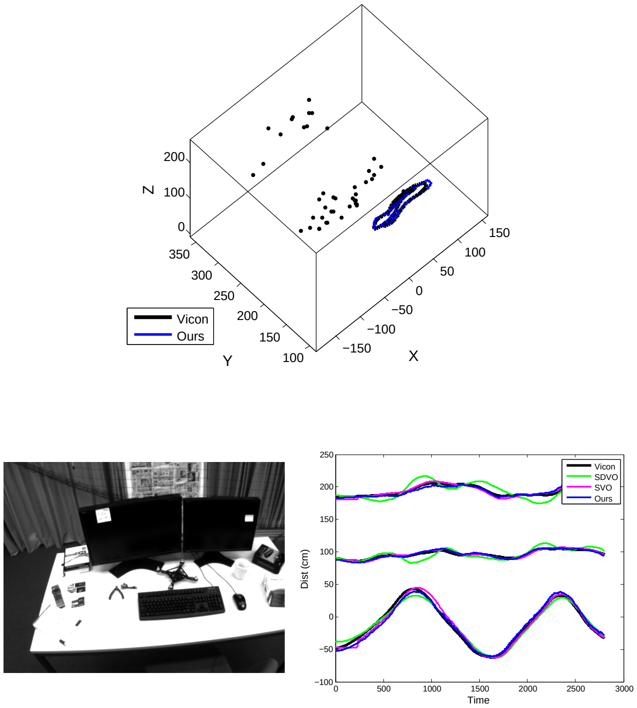

Fig. 4. Blue: Trajectory from the presented algorithm, Magenta: Semi Direct Visual Odometry (SVO) [4], Green: Semi-Dense Visual Odometry (SDVO) [3], Black: Ground Truth (VICON data). The initialization of SDVO had issues in the selected scene, due to the suboptimal initial map the performance is not as good as can be expected.

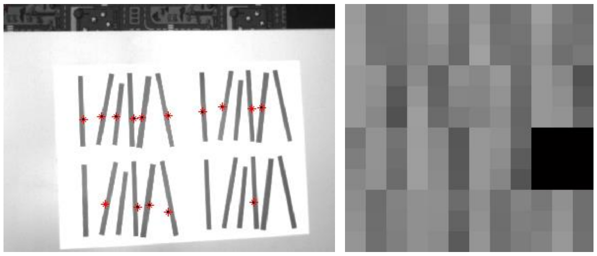

Fig. 5. Thanks to the constraints on the pixel patches, the algorithm is able to initialize even on this difficult scene consisting only of almost vertical lines. On the right side the used 3x3 pixel patches are visible.

The most compelling advantage of our constrained direct method is that it do not rely on the presence of corner like features. In particular, our implementation works on scenes that *only* contain (non-intersecting) lines. Figure 5 shows a demonstration of such a scene. It is clear that methods that rely on external trackers like KLT will fail in this scenario since the tracker is not able to fix the tracked points at a position and thus the point will start to randomly slide along the edge. Since in our implementation the location of the patches are constrained by the model in the filter, the algorithm is able to fully initialize with patches that lie on these kinds of edges even with a patch size of only 3x3 pixels.

Figure 6 shows the results of a challenging dataset with a camera moving in front of a curtain having almost only line-like structure in view. Only few patches where placed on corner-like areas, this was enough to fix the camera pose from drifting in vertical direction. This demonstrates that the proposed method can be used to track scenes that are rather hard for methods that rely on unconstrained feature correspondences as is the case in many indoor scenes.

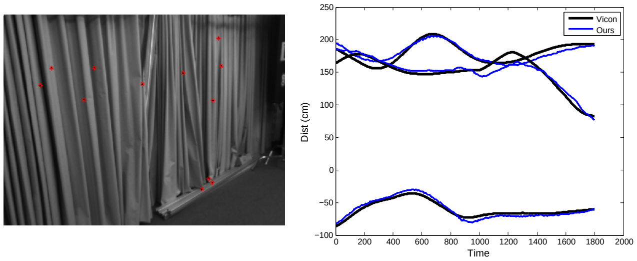

Fig. 6. Visual-Inertial odometry on a scene with lines. Left: patches on lines. Right: comparison with ground truth (VICON data).

The runtimes for the photometric update in unoptimized C code from MATLAB on a Core i5 desktop computer is 12 ms, thus already allowing for real-time use. We plan to implement a fully optimized version for mobile ARM CPUs.

V. Conclusion

In this paper we presented a novel, Kalman filter-based semi-direct visual inertial odometry approach that combines the advantages of a tightly coupled visual-inertial Kalman filter and the robustness and precision of direct photometric methods. We demonstrated how the photometric update can be built into a standard error-state Kalman filter odometry algorithm. We proposed an efficient implementation that reduces the impact of the larger number of measurements when minimizing the photometric residual error. Finally, we demonstated that our proposed algorithm matches the tracking quality of other state of the art approaches, and in addition, thanks to the rigid scene constraints the proposed algorithm can work with pixel patches lying only on line-like structures and is even able to fully initialize without special procedure in such scenes.

References

[1] B. D. Anderson and J. B. Moore. *Optimal filtering*. Courier Dover Publications, 2012.

[2] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(6):1052–1067, 2007.

[3] J. Engel, J. Sturm, and D. Cremers. Semi-dense visual odometry for a monocular camera. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1449–1456. IEEE, 2013.

[4] C. Forster, M. Pizzoli, and D. Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, 2014.

[5] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis. Towards consistent vision-aided inertial navigation. In *Algorithmic Foundations of Robotics X*. Springer, 2013.

[6] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera. In *Proc. ACM User Interface Software and Technologies*, UIST '11, pages 559–568, 2011.

[7] H. Jin, P. Favaro, and S. Soatto. A semi-direct approach to structure from motion. *The Visual Computer*, pages 377–394, 2003.

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

[9] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pages 225–234. IEEE, 2007.

[10] M. Li and A. I. Mourikis. High-precision, consistent EKF-based visual-inertial odometry. *International Journal of Robotics Research*, 2013.

[11] M. Li and A. I. Mourikis. Online temporal calibration for camera–imu systems: Theory and algorithms. *The International Journal of Robotics Research*, 33(7):947–964, 2014.

[12] L. Matthies, R. Szeliski, and T. Kanade. Incremental estimation of dense depth maps from image sequences. In *Computer Vision and Pattern Recognition, 1988. Proceedings CVPR'88., Computer Society Conference on*, pages 366–374. IEEE, 1988.

[13] N. Molton, A. J. Davison, and I. Reid. Locally planar patch features for real-time structure from motion. In *BMVC*, pages 1–10, 2004.

[14] A. I. Mourikis, N. Trawny, S. I. Roumeliotis, A. E. Johnson, A. Ansar, and L. Matthies. Vision-aided inertial navigation for spacecraft entry, descent, and landing. *Robotics, IEEE Transactions on*, 25(2):264–280, 2009.

[15] R. A. Newcombe and A. J. Davison. Live dense reconstruction with a single moving camera. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1498–1505. IEEE, 2010.

[16] 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 *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011.

[17] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. Dtam: Dense tracking and mapping in real-time. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2320–2327. IEEE, 2011.

[18] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–652. IEEE, 2004.

[19] T. Pietzsch. Efficient feature parameterisation for visual slam using inverse depth bundles. In *Proceedings of the British Machine Vision Conference*, 2008.

[20] E. Rosten and T. Drummond. Fusing points and lines for high performance tracking. In *IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 1508–1515 Vol. 2, Oct 2005.

[21] S. I. Roumeliotis, G. S. Sukhatme, and G. A. Bekey. Circumventing dynamic modeling: Evaluation of the error-state kalman filter applied to mobile robot localization. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 2, pages 1656–1663. IEEE, 1999.

[22] T. Schöps, J. Engel, and D. Cremers. Semi-dense visual odometry for AR on a smartphone. In *ISMAR*, September 2014.

[23] P. Tanskanen, K. Kolev, L. Meier, F. Camposeco, O. Saurer, and M. Pollefeys. Live metric 3d reconstruction on mobile phones. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 65–72. IEEE, 2013.

[24] S. Weiss and R. Siegwart. Real-time metric state estimation for modular vision-inertial systems. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4531–4537. IEEE, 2011.

[25] A. Wendel, M. Maurer, G. Graber, T. Pock, and H. Bischof. Dense reconstruction on-the-fly. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. to appear, 2012.

[26] Z. Zhang. Microsoft kinect sensor and its effect. *IEEE MultiMedia*, 2012.