Monocular Visual-Inertial Odometry in Low-Textured Environments with Smooth Gradients: A Fully Dense Direct Filtering Approach

Alexander Hardt-Stremayr and Stephan Weiss

***Abstract*— State of the art visual-inertial odometry approaches suffer from the requirement of high gradients and sufficient visual texture. Even direct photometric approaches select a subset of the image with high-gradient areas and ignore smooth gradients or generally low-textured areas. In this work, we show that taking all image information (i.e. every single pixel) enables visual-inertial odometry even on areas with very low texture and smooth gradients, inherently interpolating and estimating the scene with no texture based on its informative surrounding. This information propagation is only possible as we estimate all states *and their uncertainties* (robot pose, extrinsic sensor calibration, and scene depth) *jointly* in a fully dense filter framework. Our complexity reduction approach enables real-time execution despite the large size of the state vector. Compared to our previous basic feasibility study on this topic, this work includes higher order covariance propagation and improved state handling for a significant performance gain, thorough comparisons to state-of-the-art algorithms, larger mapping components with uncertainty, self-calibration capability, and real-data tests.**

I. Introduction

In this paper, we further extend our novel fully dense tightly coupled direct extended Kalman filter (EKF) based visual-inertial odometry approach, first presented in [1]. Predicting the intensities (known as direct approach) and depths of *all* the pixels in the image, we enable motion estimation even in low-textured environments with smooth gradients. Having all pixels in the state, we do not need to select high-gradient areas (semi-dense) or single pixels (sparse) with heuristically chosen parameters, but take the raw, full information of the camera. For improved prediction, we represent the scene depth as inverse depth per pixel in the state vector. This enables us to update all state elements (i.e. robot motion state, sensor calibration states, *and* current scene state) in a single step in contrast to calculating state and scene information sequentially and separately (as performed by current state-of-the-art algorithms decoupling the uncertainties of estimated state and environment). With the joint estimation, our approach provides metric depth uncertainty per pixel as a further element to the dense mapping in a statistically consistent fashion together with the robot state. Both the map and the robot state are tightly coupled in the same state vector and estimation process. In low-contrast environment (e.g. as in Figure 1), we show in comparisons that all of the selected state-of-the-art algorithms (ROVIO, OKVIS and VINS-Mono) fail. Also, in environmental conditions favorable to the state-of-the-art algorithms with both high contrast and large variations in scene depth (incl. depth discontinuities), we show that our approach has similar performance (about 0.5% final position drift) making it usable in a large variety of scenarios.

The authors are with the Control of Networked Systems Group, Universität Klagenfurt, Austria (alexander.hardt-stremayr, stephan.weiss)@ieee.org

The research leading to these results has received funding from the ARL within the BAA W911NF-12-R-0011 under grant agreement W911NF-16-2-0112 and from the Austrian Ministry for Transport, Innovation and Technology (BMVIT) under the grant agreement 855468 (Forest-IMATE).

Accepted Jan/2020 for ICRA 2020, DOI follows ASAP ©IEEE.

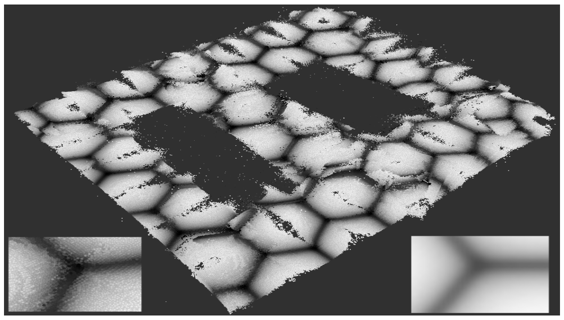

Fig. 1. Visual-inertial odometry (VIO) over low-textured area with smooth gradients. Zoom of the 3D reconstruction (left) and original image used in the VIO algorithm (right). Gray-scale values are surface intensities.

Utilizing matrix properties and complexity reduction methods, the entire estimation process is performed in linear time dependent only on the number of pixels rendering it possible to run in real-time on a standard laptop as a pure CPU implementation. For our experiments, to process a single image (downscaled to 94×60 pixels for updates but using VGA resolution for propagation) requires 250 to 330 milliseconds, yielding a rate of 3 to 4 updates per second.

A. Related Work

Besides our feasibility study very recently shown in [1], to our knowledge, no one-step fully dense direct CPU-based visual-inertial odometry algorithm exists as of now.

DTAM [2] popularized GPU-based trajectory and depth estimation in a fully dense fashion aided by feature based initialization steps [3]. More recent work by Shen et al. added inertial measurements [4], [5] but shifted the focus to the mapping part only. The authors use a sparse feature based non-linear optimization method for visual-inertial odometry (VIO) and post-reconstruct (in real-time) the environment in a dense fashion based on the estimated camera poses and on motion-stereo with semi-global matching. Uncertainty is then attributed based on this reconstruction post-processing. The train of thought to use VIO for subsequent dense map reconstruction is further developed in [6] and [7] also by the group of Shen where baseline heuristics for the subsequent map reconstruction or post-allocation of uncertainty are discussed. [8] and [9] show (motion-)stereo approaches with known baseline for depth estimation on a CPU. Known extrinsics simplify the proces to a line search.

Our approach could best be compared to state of the art in direct semi-dense [10] and sparse [11] visual odometry methods. However, they feature a so-called two-step approach estimating motion and depth information (i.e. map) separately leading to a disjoint uncertainty for both elements. Furthermore, to select the sparse or semi-dense area, a pre-processing selecting high-gradient image areas is done. Low-gradient environment inherently lead to failures of such algorithms as the selection process will not return suitable image areas. Similarly, pseudo-dense (semi-dense among gradients [12], super-pixels for monotonous regions [13]) VIO algorithms [14] were presented by Concha et al. with the same limitations as [10] and [11].

Tangential to our approach is work using depth sensors (e.g. RGB-D cameras) directly providing depth measurements simplifying the process with limitation of sensitivity to sunlight and sensor range. The authors of [15]–[17] presented CPU based implementations, whereas [18] with different extensions as e.g.in [19], [20], and integration of IMU information in [21] present GPU implementations.

Apart of the pure motion estimation, several works include self-calibration aspects ([22]–[27]) in visual-inertial odometry frameworks. None of them tackles this aspect in a fully dense framework as we do here. Other recent work based on deep-learning methods for depth estimation ([28]–[30]) do not yet make use of inertial readings.

B. Contributions

It is important to note that (particularly compared to the one-step semi-dense direct approaches in [23], [24], [31]) the key to our approach is to include every single pixel in the estimation. Only this truly dense approach allows the tight linkage of the uncertainties between motion and scene depth and, thus, the usage in over even very low-textured, low-gradient areas. We demonstrate this capability in emulated and real data in Section III. We showed an initial feasibility study of this capability in our recent contribution [1], proving that it is of use to cover short-term low-gradient areas. The current work extended our initial study in scope, duration and robustness: the higher order covariance propagation led to improved consistency enabling long trajectories with significantly reduced drift; the improved handling of depth propagation enabled the motion over larger cluttered scenes; and the more accurate image down-sampling while propagating at high resolution enabled the precision for larger scale motion estimation and inherent mapping with uncertainty.

Those adaptions now extend our approach to a fully dense direct filter-based monocular visual-inertial odometry algorithm, capable of motion estimation in arbitrary areas yielding uncertainty attributed dense maps in a one-step process. Namely, and in particular view to our previous work [1], the contributions of this paper are:

- Performance: higher order covariance propagation and pixel propagation (depth and intensity) to enable more precise motion estimation in both high-contrast and low-contrast environments without parameter tuning.
- Evaluation: performance comparison to the frameworks VINS-Mono, OKVIS, and ROVIO in different scenarios
- Mapping: probabilistic map generation of a medium sized environment demonstrating simultaneous dense mapping and motion estimation with joint uncertainty
- Self-Calibration: addition of IMU intrinsics and camera-IMU extrinsics (incl. performance evaluation) rendering the framework self-calibrating
- Real-world tests: Evaluation with real-world data and real-world parameters (lens distortion, illumination changes) using the EuRoC dataset.

II. Dense Direct Visual-Inertial Odometry

The main difference to other (semi-)dense direct approaches comparing predicted pixel intensities with the measured one is our single-step update, correcting both pose and depth estimation at the same time for all pixels in the image. This requires to add the depth per pixel to the state, yielding a large state vector and providing a metric depth uncertainty estimation for each pixel, connected to the pose estimation in a single covariance matrix.

A. System Description

The state of our system is composed of a core state $\mathbf{x}_c$ and is extended with inverse depth per pixel $\boldsymbol{\rho}$ and intensity value $\boldsymbol{i}$. $\mathbf{x}_c$ contains position ${}_W\boldsymbol{p}_{WI}$, orientation $\bar{\mathbf{q}}_{WI}$ and velocity ${}_W\mathbf{v}_{WI}$ of the robot frame in the world frame, position ${}_I\boldsymbol{p}_{IC}$ and orientation $\bar{\mathbf{q}}_{IC}$ of the camera frame in the robot frame and the IMU intrinsics (biases $\mathbf{b}_{\boldsymbol{\omega}}$ and $\mathbf{b}_{\mathbf{a}}$ for angular velocity and linear acceleration respectively).

$$\mathbf{x} = \begin{bmatrix}\mathbf{x}_c & \boldsymbol{\rho} & \boldsymbol{i}\end{bmatrix}^T \tag{1}$$

$$\mathbf{x}_c = \begin{bmatrix}{}_W\boldsymbol{p}_{WI} & {}_W\mathbf{v}_{WI} & \bar{\mathbf{q}}_{WI} & \mathbf{b}_{\boldsymbol{\omega}} & \mathbf{b}_{\mathbf{a}} & {}_I\boldsymbol{p}_{IC} & \bar{\mathbf{q}}_{IC}\end{bmatrix} \tag{2}$$

We refer to [1] for extended state/variable definition and [25] for both continuous and discretized equations for the core states $f(\mathbf{x}_c)$ as well as the description of the corresponding process noise matrix $Q_{\mathbf{x}_c}$. The derivation of the dynamics of the inverse pixel depth $\dot{\boldsymbol{\rho}} = f_{\boldsymbol{\rho}}(\mathbf{x})$ and the pixel intensity $\dot{\boldsymbol{i}} = f_{\boldsymbol{i}}(\mathbf{x})$, conceptually introduced in [1], are as follows (with p_I as the pixel ray on the image plane and $e_z = [0\ 0\ 1]$):

$$\dot{\rho} = -e_z \rho^2 {}_C\dot{\boldsymbol{p}}_{CP} \tag{3}$$

$$\dot{i} = \boldsymbol{g} K (I_{d_{3\times3}} - K^{-1} p_I e_z) \rho_C \dot{\boldsymbol{p}}_{CP} \tag{4}$$

Both inverse depth and image intensity dynamics depend on the time derivative of the projected point coordinate in the camera frame ${}_C\dot{\boldsymbol{p}}_{CP}$. $\dot{\rho}$ corresponds to the change of the camera z coordinate, while $\dot{i}$ transforms the coordinate change into the pixel frame, using the reprojection on the image plane $(I_{d_{3\times3}} - K^{-1} p_I e_z)$, the intrinsic camera matrix

K and the gradient at the pixel position $\boldsymbol{g}$. ${}_C\dot{\boldsymbol{p}}_{CP}$ depends on elements of $\mathbf{x}_c$ and IMU readings as input.

$$
{}_C\dot{\boldsymbol{p}}_{CP} = -\mathbf{C}_{IC}{}^T\mathbf{C}_{WI}{}^T({}_W\mathbf{v}_{WI} + \dot{\mathbf{C}}_{WI}\,{}_I\boldsymbol{p}_{IC} + \dot{\mathbf{C}}_{WI}\mathbf{C}_{IC}\,{}_C\boldsymbol{p}_{CP}) \tag{5}
$$

B. State Propagation

While both intensity and inverse depth vectors are stored in the state vector, only the core state $\mathbf{x}_c$ is propagated upon IMU readings for performance reasons. On each camera reading, the state change introduced by IMU readings is accumulated (i.e. integrated) and the pixel propagation is calculated before comparing the intensity with the camera measurement. By integrating camera rotation to $\mathbf{C}_{21}$ and translation to ${}_2\boldsymbol{p}_{21}$, we can transform pixel positions between the previous frame 1 and the current frame 2, using their inverse depths ρ_1, ρ_2.

$$
p_{I2} = K\frac{\mathbf{C}_{21}K^{-1}p_{I1}\rho_1^{-1} + {}_2\boldsymbol{p}_{21}}{\rho_2^{-1}} \tag{6}
$$

However, as pixel coordinates are integer numbers, care has to be taken to not introduce a bias to the estimator through rounding errors. Simply performing a linear extrapolation of the rounding-induced intensity shift using the image gradient multiplied with the position difference introduces errors for large depth differences. A more exact solution is to first calculate the target rounded pixel coordinates and then to select the source coordinates.

As those will not be rounded, it is possible to perform a sub-pixel intensity lookup for propagation. For this, a knowledge of the scene depth at the position of the target rounded pixel coordinates in the target frame is needed. This scene depth may differ from the estimated depth at the rounded source pixel coordinates, resulting in a different pixel position depending on the depth difference.

Figure 2 illustrates the problem and depth interpolation as our solution. While on the left side, the depth estimation is available for every pixel center, the coordinates in the target frame are off-center. Shifting those coordinates needs depth interpolation to be able to do the backwards propagation.

$$
g_{\rho_x} = \frac{1}{\rho_{lower}} - \frac{1}{\rho_{upper}}, g_{\rho_y} = \frac{1}{\rho_{right}} - \frac{1}{\rho_{left}} \tag{7}
$$

This depth gradient $\boldsymbol{g}_\rho = [g_{\rho_x}, g_{\rho_y}, 0]$ spans a plane around the source pixel p_{I1} as calculated in Equation (7) with $\boldsymbol{\rho}_{[location]}$ being the inverse depth of the pixel relative to the currently considered pixel. By transforming the rounded target pixel $\lfloor p_{I2} \rceil$ back into the first frame with a now unknown depth, the intersection between this ray and the depth gradient plane is calculated. This intersection point is used to both generate the propagated depth in the new frame and, after calculating the intersection with the image plane, to do a sub-pixel based lookup of the image intensity propagated to the target pixel.

$$
\rho_2^{-1}\lfloor p_{I2} \rceil = K\mathbf{C}_{21}K^{-1}(p_{I1}+\Delta p_{I1})(\rho_1^{-1}+\boldsymbol{g}_\rho\Delta p_{I1})+K{}_2\boldsymbol{p}_{21} \tag{8}
$$

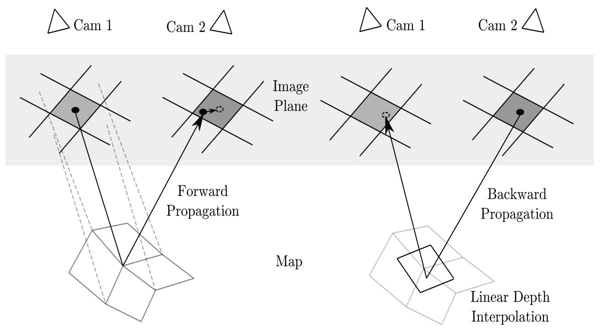

Fig. 2. Left: forward propagation of a pixel center (black dot) using estimated camera pose-change (from Cam1 to Cam2) and estimated depth. The propagated center (black dot) will not exactly result in the center of a new pixel (circle). Right: backward propagation of a pixel center (black dot) from the target frame (Cam2) into the source frame (circle in the image plane of Cam1) using estimated camera pose change and interpolated depth based on depth gradients around the source pixel. The map is only depicted for better understanding. Our approach does not maintain a map in the global frame, instead it estimates inverse depths per pixel per camera frame.

Equation (8) shows a reformulation of Equation (6) with the rounded target pixel $\lfloor p_{I2} \rceil$ depending on an unknown pixel offset in the source frame Δp_{I1}. Further reformulation of Equation (8) will bring it into an $Ax = b$ representation as shown in Equation (9).

$$
\begin{aligned}
A =&(I_{d_{3\times3}} - \lfloor p_{I2} \rceil * e_z)K\mathbf{C}_{21}K^{-1}(\rho_1^{-1} + (p_{I1} + \Delta p_{I1})\boldsymbol{g}_\rho) \\
x =&\Delta p_{I1} \\
b =&-(I_{d_{3\times3}} - \lfloor p_{I2} \rceil * e_z)K(\mathbf{C}_{21}K^{-1}p_{I1}\rho_1^{-1} + {}_2\boldsymbol{p}_{21})
\end{aligned} \tag{9}
$$

Solving Equation (9) iteratively with $A^{-1}b$ will yield the pixel offset of the intersection point in the source frame, assuming the depth plane given by $\boldsymbol{g}_\rho$. With this, the combined mean is calculated in Equation 10 by performing a weighted sum with each weight w_i depending on the overlap of the neighboring source pixel and the interpolated pixel.

$$
\rho_{interpolated} = \sum_{i=1}^{4} w_i\rho_i, \sum_{i=1}^{4} w_i = 1 \tag{10}
$$

C. Covariance Propagation

In Equations (3) and (4), the time derivatives of the states are expressed as products of random variables. Although the product Z of Gaussian random variables X and Y (with means $\mu_{X,Y}$ and variances $\sigma^2_{X,Y}$) is not Gaussian distributed and therefore a source for inconsistencies, it can be approximated as shown in Equation (11), assuming the covariance is negligible.

$$
\mu_Z = \mu_X\mu_Y\sigma_Z^2 \approx \mu_X\sigma_Y^2\mu_X^T + \mu_Y\sigma_X^2\mu_Y^T + \sigma_X^2\sigma_Y^2 \tag{11}
$$

The linearized uncertainty propagation of JPJ^T, with J being a Jacobian and P being a covariance matrix, contains the parts $\mu_X\sigma_Y^2\mu_X^T + \mu_Y\sigma_X^2\mu_Y^T$, leaving $\sigma_X^2\sigma_Y^2$ as remaining not yet included part of the approximated propagated uncertainty.

For Equation (3), the inverse depth time derivative is the product of the inverse depth squared $\boldsymbol{\rho}^2$ and the time derivative of the coordinate in the three-dimensional metric space. The 3×3 uncertainty matrix $P_{\Delta_C\boldsymbol{p}_{CP}}$ can be calculated using the corresponding Jacobian and the core state covariance P_{xx}.

$$\sigma^2_{\rho_{mult}} = 2^2 \rho^2 e_z P_{\Delta_C \boldsymbol{p}_{CP}} e_z^T \sigma^2_\rho \tag{12}$$

The inverse depth variance $\sigma^2_{\rho_{mult}}$ is added to the entry for this pixel in the inverse depth process noise matrix Q_{dd}. For Equation (4), the multiplicative covariance approximation of three components is calculated. Besides $P_{\Delta_C \boldsymbol{p}_{CP}}$ and σ^2_ρ, the image gradient covariance matrix P_g is used. The resulting multiplicative uncertainty propagation $\sigma^2_{i_{mult}}$ is added to the intensity process noise matrix Q_{ii}. The calculation can be derived similarly to Equation (12).

D. Image Downscaling

The algorithm operates on a downscaled image to speed up calculations and to make operations on a CPU feasible. As an example, an image resolution of 640×480 pixel may be downscaled to 80×60 pixels, with each downscaled pixel corresponding to 64 pixels in the original image. To calculate both downscaled intensity values and corresponding gradients, first the intensity gradient between each neighboring pixels is stored in the original image, yielding intensity i and gradients $\boldsymbol{g}_x$ and $\boldsymbol{g}_y$. For each downscaled pixel, these values are used to generate both mean and covariance values.

The gradient covariance matrix P_g is used in propagation noise calculations. For an area with large arbitrary intensity changes, the gradients can be near zero as positive and negative gradients in the original image are compensating each other in the superpixel while the entries in P_g will be large as those intensity changes sum up. For an area with a continuous gradient, P_g will be small as gradients in the original image do not deviate from the gradient mean.

$$\begin{aligned} y_{small} &= \sum_{i=1}^{n} y_i = \sum_{i=1}^{n} (i_{measured_i} - i_{propagated_i}) \\ R &= \sigma^2_{y_{small}} = \frac{1}{n} \sum_{i=1}^{n} (y_i - y_{small})^2 \end{aligned} \tag{13}$$

To further improve propagation quality, the pixel intensity is first propagated on the original image and then downscaled as shown in Equation (13), giving both mean y_{small} and variance $\sigma^2_{y_{small}}$ of the residual for a superpixel. Contrary to image-pyramid approaches, this assumes that areas with arbitrary intensity changes and therefore varying residuals will yield less information than monotonous areas and residuals.

Similarly to other direct approaches with roots in the Lucas-Kanade method, the state update is calculated around a linearization point using the image gradients as partial derivatives of the pixel intensities. Especially in areas with strongly changing gradients, discontinuities in the gradient direction occur and result in erroneous depth calculations.

Instead of a blurred downsampling, our approach uses the gradient uncertainty P_g to apply a weightening factor to each correction in the covariance propagation. We extended our original approach to use both source and target pixel covariances to improve the algorithm quality in areas with strongly changing gradients as this covers both the propagation from a monotonous area into an area with arbitrarily changing intensity values and vice versa.

III. Evaluation

A. Comparisons

We compared our approach to three existing VIO algorithms using synthetically generated datasets. VINS-Mono [32] and OKVIS [26] are both feature-based bundle-adjustment approaches. ROVIO [24] is a direct filter based approach, selecting a set of image patches and using intensity residuals to perform the update.

All datasets have been played at a slower rate of $0.1\times$ the original speed, providing enough time for each algorithm to completely process every frame. All camera images for the subsequent tests with realistic data have been generated in the Unity3D photo-realistic game engine at a resolution of 640×480 at 20 frames and pinhole camera without distortion unless otherwise noted. IMU messages were generated at a rate of 200 samples per second with time alignment between IMU messages and camera images and with the following continuous noise values: $\sigma^2_a = 1.86\text{e}{-3} m/s^2$ for linear acceleration noise, $\sigma^2_{b_a} = 1.87\text{e}{-4} m/s^3$ for linear acceleration bias noise, $\sigma^2_\omega = 4.33\text{e}{-4} rad/s$ for angular velocity noise and $\sigma^2_{b_\omega} = 2.66\text{e}{-5} rad/s^2$ for angular velocity bias noise. Those values correspond to those of typical MEMS sensor as measured on our real AscTec Hummingbird platforms. Camera extrinsics have been initialized with the correct state to allow fair comparisons between different algorithms.

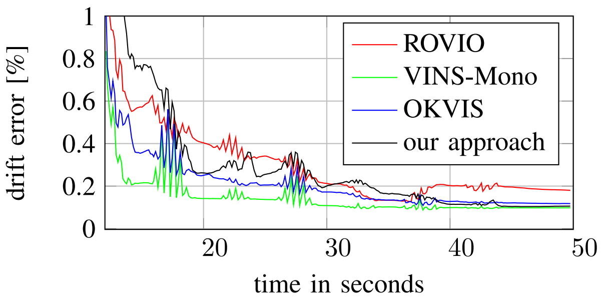

Fig. 3. Averaged drift over time expressed as global position error divided by distance travelled. Red: ROVIO, blue: OKVIS, green: VINS-Mono, black: ours. After initialization, all algorithms show a similar drift performance.

In Figure 3, each algorithm was applied on ten simulations of a cluttered scenario (see Figure 5) always navigating the same trajectory of 40 meters length, but each time with different ground texture and re-generated random IMU noise. The position drift in percent compared to distance travelled was averaged over all ten runs. After drift accumulation during initialization phase, all algorithms show a similar behavior. All trajectories were aligned to ground truth in the same fashion. For our approach, the mean drift at the end was 0.035, -0.001 and -0.026 m for x, y and z respectively, with a standard deviation of 0.024, 0.021 and 0.016 m.

In Figure 4, a comparison between the same algorithms is performed on a longer, 200m, trajectory (100m forth and back again). Deviations from ground truth mostly originate

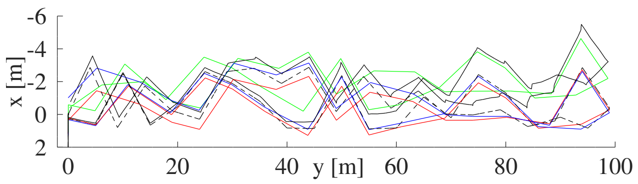

Fig. 4. Trajectories over a 200m distance: Red: ROVIO, blue: OKVIS, green: VINS-Mono, black: ours, black dashed: ground truth. Except for some yaw drift, all algorithms show similar behaviour.

from yaw drift as all algorithms have less than 0.5% position drift when returning to the starting position. Both Figures 3 and 4 show that our approach gives results comparable to current state of the art algorithms under their favorable scene conditions (i.e. well textured).

B. Uncertainty-Aware 3D Mapping

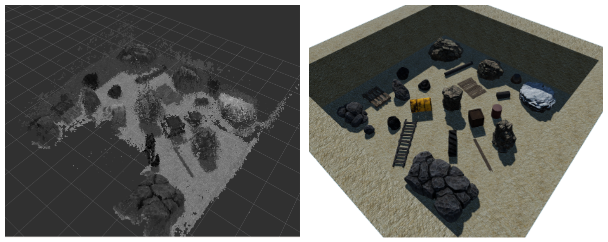

Fig. 5. Left: isometric view of the accumulated point cloud generated by our algorithm. Right: corresponding isometric ground truth view in the simulation environment.

Our dense direct approach performs a depth estimation for every pixel, initialized with a high uncertainty and improved over time. This yields an uncertainty-annotated 3D map of the environment as a byproduct. Figure 5 shows the accumulated point cloud generated by our algorithm in one run for Figure 3. The map shows all accumulated 3D points which have an uncertainty below a user selected threshold. No loop closure or other post-adjustments were made.

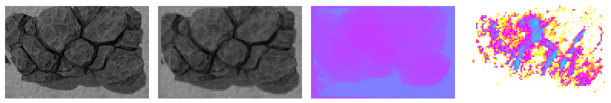

Fig. 6. Left to right: the camera image as generated in the simulation framework, the downscaled *prediction* of the image, the inverse depth map in the current image, and the certainty of the inverse depth estimation (white is less certain)

Figure 6 displays the internal states generating those depth estimations. At high-gradient areas, the depth estimation is more certain than at low-gradient areas. However, as our approach uses all available pixels, the remaining information available enables the algorithm to still converge to the correct depth values in low-gradient areas, albeit at a slower rate.

C. Featureless Areas

By using all available information in the image, visual-inertial odometry is possible even on scenery consisting only of low-gradient areas as shown in Figure 8.

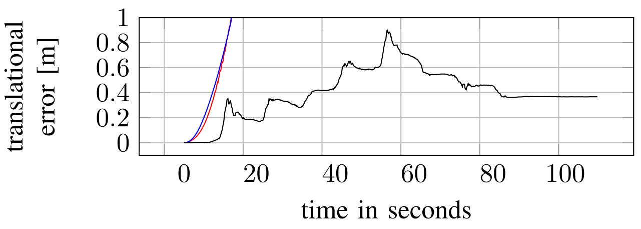

Fig. 7. Pose estimation on low-gradient areas. Red: ROVIO, blue: OKVIS, black: our approach. VINS-Mono did not initialize at all. Both ROVIO and OKVIS diverge, presumably because of pure IMU integration due to lack of visual information for IMU correction.

As Figure 7 shows, our approach is able to estimate the pose with an error similar to the drift seen in Figure 3 over a 40 meter trajectory including still phases to test hover conditions. Conversely, due to the lack of strong gradients and well behaving texture all tested state-of-the-art algorithms fails: Either no initialization takes place (VINS-Mono) or pure IMU integration continues leading to vast position drift without any visual correction due to the lack of contrast (OKVIS, ROVIO). Over time, easily estimated depth values at higher gradients propagate to low-gradient areas, yielding a depth estimation over the entire image. An insight to this aspect is given in Figure 8 where the top row shows only the points of the estimated scene with a minimal uncertainty on the left and all estimated 3D points on the right. While the centers of the hexagons clearly have higher uncertainty (holes in the top left image and white areas in the bottom right image) the algorithm estimates depth for all image pixels. The bottom left image of the Figure shows that, according to the uncertainty, the depth estimate on the hexagon centers (blue ellipse) is less accurate but still well made due to the inherent information propagation of the neighboring, more certain areas (yellow box). This is also reflected in the uncertainty map on the bottom right of the Figure (white is high, blue low uncertainty).

D. Self-Calibration

By adding the pose of the camera in the IMU frame, consisting of translation ${}_{I}\boldsymbol{p}_{IC}$ and rotation $\bar{\mathbf{q}}_{IC}$, to the state $\mathbf{x}_c$, we easen the else rather strict requirement of exact calibration. Instead, after starting with a rough estimate, these calibration states converge towards the correct values. The speed of convergence strongly depends on the quality of observability which in turn depends on the trajectory. For these tests, we used the same type of trajectory as for the other evaluations in this paper without considering observability aspects. Incorporating observability aware motion generation [27], as presented by Hausman et al., would both speed up convergence time and decrease remaining error.

Figure 9 shows the estimation of both camera/IMU translation and rotation over time. Five seconds after start of movement, the estimated translation converges within 10 % of the true values. As this state is difficult to observe, this is an expected behaviour. With more rotation-heavy movements, the state would further converge. The camera/IMU rotation converges to the true values within 20 seconds. As additional

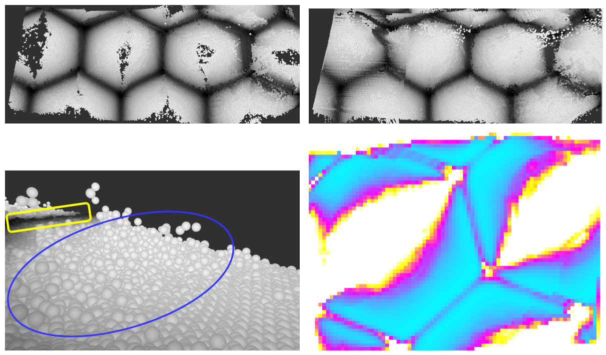
Fig. 8. Depth estimation and uncertainty estimation in smooth and low gradient environments. Top left: estimated scene intensity and depth only displaying points with a low uncertainty. Clearly, the centers of the hexagons yield least information as the gradient is lowest. Hence the hexagon centers appear as holes as the point uncertainty is above the threshold. Top right: estimated scene depth displaying all points. Bottom left: close-up and side view of the hexagon center to appreciate the depth estimation of the uncertain areas. Although the uncertainty is high for the hexagon centers (blue ellipse), the depth is inherently well interpolated using the information of the surrounding, more certain environment (e.g. yellow box). Bottom right: uncertainty map again highlighting the information coming from the (smooth) gradients and less from the hexagon centers (white is uncertain, blue is certain).

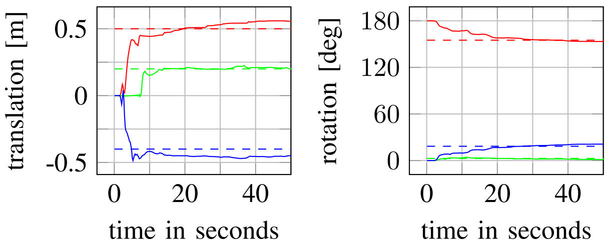

Fig. 9. Estimation of translation (left) and rotation (right) between camera and IMU. Red, green and blue correspond to x, y and z respectively. The dashed lines are the true values while solid lines are estimated.

drift during the initialization phase is a side-effect of filter-based self-calibration, and for fair comparisons, previous estimations use an already converged state.

E. Real-World Data

We extended our real-world tests to the EuRoC MAV dataset [33], which consists of a time-synchronized camera and IMU system with vicon ground truth data. We selected the vicon room set V1_01_easy as it contains both high-frequency and low-frequency areas.

Figure 10 shows the estimated state and a 3D reconstruction of the scene. Although scene-depth estimation is more certain for high-gradient areas, low-gradient depth estimation still yields the correct depth.

In Figure 11, the estimated position is compared to the ground truth value. Rotational trajectory alignment has been performed. With increasing and decreasing position error (e.g. in z dimension at second 60), the corresponding position variance is also changing accordingly. During the 58.6 m trajectory, the absolute translation error (RMSE) is 0.56 m at its maximum and 0.11 m at the end. This end drift is comparable to current state of the art algorithms which show an error between 0.04 m and 0.1 m [34]. We note at this

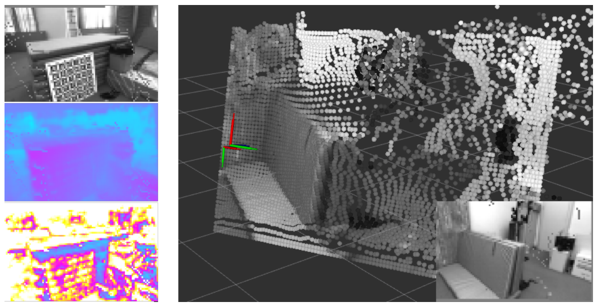
Fig. 10. Internal state and reconstruction during the EuRoC V1_01_easy dataset. Left top to bottom: predicted image intensities, estimated depth, uncertainty of estimated depth (white is less certain) of a scene showing a high-gradient environment. Right: reconstruction of all pixels (large image) and image intensities (bottom right) of a second scene, showing correct estimation in low-gradient environment. Note that low gradient pixels further away (right side) from the current camera (colored tripod) are not yet well reconstructed due to lack of information from both motion and scene.

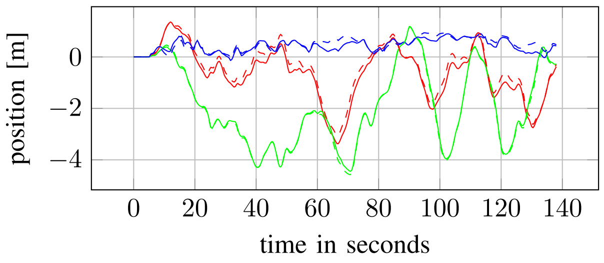

Fig. 11. Estimation and ground truth of the EuRoC V1_01_easy dataset. Red, green and blue correspond to position in x, y and z respectively. The dashed lines are the ground truth while solid lines are estimated.

point that our approach only uses frame-to-frame information without keeping older information.

IV. Conclusion

In this work, we significantly advanced our feasibility study in [1] to a full grown visual-inertial odometry framework with self-calibration properties and, to our knowledge for the first time, capable of performing well in very low-textured environments with very smooth gradients.

The theoretical contributions with respect to the higher order covariance propagation as well as the inclusion of the forward-backward propagation for pixel depth and intensity interpolation during the propagation step were introduced as important novel aspects to noticeably improve performance compared to our previous work.

We also showed that our approach estimates the motion with a precision comparable to current methods (about 0.5% final position drift) in scenes with high gradients and good texture (only there, comparisons were feasible: other approaches failed in low-textured scenes). For this, we compared our approach against three VIO frameworks deemed to be among the currently best performing ones in the community (OKVIS, ROVIO, VINS-Mono) in a photo-realistic simulation scene.

Unique to our approach, we further showed the capability of simultaneous dense mapping and motion estimation including the joint uncertainty of the mapped environment and motion states. We demonstrated how our algorithm works well in a low-textured environment with very smooth gradients. In the comparison with other approaches, all tested algorithms failed to conceive a visual update under such conditions.

Lastly, besides demonstrating our approach using the realistic data in our simulation environment, we also showed the correct functioning of our approach in a public dataset.

References

[1] A. Hardt-Stremayr and S. Weiss, "Towards fully dense direct filter-based monocular visual-inertial odometry," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 4710–4716.

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

[3] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, Nov 2007, pp. 225–234.

[4] K. Qiu and S. Shen, "Model-aided monocular visual-inertial state estimation and dense mapping," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Sept 2017, pp. 1783–1789.

[5] Z. Yang, F. Gao, and S. Shen, "Real-time monocular dense mapping on aerial robots using visual-inertial fusion," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 4552–4559.

[6] K. Wang and S. Shen, "Adaptive baseline monocular dense mapping with inter-frame depth propagation," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct 2018, pp. 3225–3232.

[7] Y. Ling, K. Wang, and S. Shen, "Probabilistic dense reconstruction from a moving camera," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct 2018, pp. 6364–6371.

[8] Y. Ling and S. Shen, "Dense visual-inertial odometry for tracking of aggressive motions," in *International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2015, pp. 576–583.

[9] V. Usenko, J. Engel, J. Stückler, and D. Cremers, "Direct visual-inertial odometry with stereo cameras," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1885–1892.

[10] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *European Conference on Computer Vision*. Springer, 2014, pp. 834–849.

[11] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2018.

[12] A. Concha Belenguer and J. Civera Sancho, "Dpptam: Dense piecewise planar tracking and mapping from a monocular sequence," in *International Conference on Intelligent Robots and Systems (IROS)*, no. ART-2015-92153. IEEE, 2015.

[13] A. Concha and J. Civera, "Using superpixels in monocular slam," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 365–372.

[14] A. Concha, G. Loianno, V. Kumar, and J. Civera, "Visual-inertial direct slam," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1331–1338.

[15] C. Kerl, J. Sturm, and D. Cremers, "Robust odometry estimation for rgb-d cameras," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2013, pp. 3748–3754.

[16] C. Kerl, J. Stuckler, and D. Cremers, "Dense continuous-time tracking and mapping with rolling shutter rgb-d cameras," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2264–2272.

[17] M. Kuse and S. Shen, "Robust camera motion estimation using direct edge alignment and sub-gradient method," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 573–579.

[18] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison *et al.*, "Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera," in *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, 2011, pp. 559–568.

[19] T. Whelan, M. Kaess, H. Johannsson, M. Fallon, J. J. Leonard, and J. McDonald, "Real-time large-scale dense rgb-d slam with volumetric fusion," *The International Journal of Robotics Research*, vol. 34, no. 4-5, pp. 598–626, 2015.

[20] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "Elasticfusion: Real-time dense slam and light source estimation," *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1697–1716, 2016.

[21] T. Laidlow, M. Bloesch, W. Li, and S. Leutenegger, "Dense rgb-d-inertial slam with map deformations," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 6741–6748.

[22] S. Weiss, M. W. Achtelik, M. Chli, and R. Siegwart, "Versatile distributed pose estimation and sensor self-calibration for an autonomous mav," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2012, pp. 31–38.

[23] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct ekf-based approach," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 298–304.

[24] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended kalman filter based visual-inertial odometry using direct photometric feedback," *The International Journal of Robotics Research*, vol. 36, no. 10, pp. 1053–1072, 2017.

[25] S. Weiss, M. W. Achtelik, S. Lynen, M. C. Achtelik, L. Kneip, M. Chli, and R. Siegwart, "Monocular vision for long-term micro aerial vehicle state estimation: a compendium," *Journal of Field Robotics*, vol. 30, no. 5, pp. 803–831, 2013.

[26] S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, K. Konolige, and R. Siegwart, "Keyframe-based visual-inertial slam using nonlinear optimization," *Proceedings of Robotis Science and Systems (RSS) 2013*, 2013.

[27] K. Hausman, J. Preiss, G. S. Sukhatme, and S. Weiss, "Observability-aware trajectory optimization for self-calibration with application to uavs," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1770–1777, 2017.

[28] M. Kuse, S. P. Jaiswal, and S. Shen, "Deep-mapnets : A residual network for 3d environment representation," in *International Conference on Image Processing (ICIP)*. IEEE, Sept 2017, pp. 2652–2656.

[29] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison, "Codeslam - learning a compact, optimisable representation for dense visual slam," *CoRR*, vol. abs/1804.00874, 2018.

[30] H. Luo, Y. Gao, Y. Wu, C. Liao, X. Yang, and K. Cheng, "Real-time dense monocular slam with online adapted depth prediction network," *IEEE Transactions on Multimedia*, 2018.

[31] X. Zheng, Z. Moratto, M. Li, and A. I. Mourikis, "Photometric patch-based visual-inertial odometry," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3264–3271.

[32] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[33] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[34] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots," in *International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 2502–2509.