Deterministic Visual Mapping

Pierre Bénet, *SBG Systems, Carrière sur Seine, France*
Alexis Guinamard, *SBG Systems, Carrière sur Seine, France*

BIOGRAPHY

Pierre Bénet is a research engineer at SBG Systems since 2019. His research interests include Visual-Inertial Navigation for high precision localization and mapping.

Alexis Guinamard is Chief Technology Officer at SBG Systems. He co-founded SBG Systems in 2007. His research interests include GNSS, Inertial sensors and inertial filtering and more lately vision.

ABSTRACT

This paper presents our development of monocular visual Simultaneous Localization And Mapping (SLAM), and its extensions to inertial aiding and stereo. Following our development of stereo visual odometry RADVO, we were able to apply on our monocular visual SLAM system the same visual feature extraction and matching, and the same outlier removal principles. Hence the feature matching strategy is done with sparse Normalized-Sum of Squared Difference matching and the outlier removal is based on a quadratic loss function. Which makes our system one of the few to be deterministic.

Since our visual features are rather simple and are localized at a pixel, we applied successfully several concepts from direct visual methods such as Direct Sparse Odometry (DSO) or Direct Sparse Mapping (DSM). Hence we model a feature using its inverse distance to the camera and we use about ten active frames and a thousand active features to perform optimization (bundle adjustment) for short term SLAM. Unlike many SLAM systems, our loop closing strategy relies only on prediction rather than place recognition. Like DSM, this is made possible by our short-term SLAM that is precise enough to predict when an old frame will be visible again. We pushed further our development to be able to perform global bundle adjustment of all the features in all the keyframes of a log, optimizing millions of features in a few seconds for post-processing.

We then achieve state of the art precision on the EuRoC dataset and TUM-VI dataset in all sensor configuration. In particular our stereo-inertial SLAM achieves an average accuracy of 2.9 cm on the EuRoC drone in real time. An accuracy that drops to 2.6 cm after a few seconds of post processing using block-sparse conjugate gradient.

I. INTRODUCTION

Visual Simultaneous Localization And Mapping has drown a lot of attention from researchers in the past two decades. Several applications await this technology such as autonomous navigation or survey. However state of the art methods struggle with robustness, real time and versatility. Still today there are big gaps between the algorithms developed from one year to the next [1].

Whereas visual SLAM literature is more concentrated on low resolution camera (0.25 Mp) at a frequency of about 20 hz, photogrammetry literature deals with quite big images (50 Mp) with several seconds between images. Yet significant piece of algorithms between visual SLAM and photogrammetry are similar [2]. In particular the global bundle adjustement that solves the hole system using a sparse matrix solver and the Schur complement trick, that allows to cut the size of the system to solve, not to mention the image preprocessing.

With the emerging of direct SLAM methods, producing more and more precise point cloud [3] on classic visual SLAM dataset [4], we can think that such methods could help getting better 3d reconstruction for photogrammetry. In the other hand photogrammetry algorithms usually estimate the intrinsic and extrinsic camera calibration parameters and every poses of the camera with the help of GNSS position [5]. Such capability [6] is scarce in SLAM systems. We found also semantic capability [7] and meshing reconstruction in photogrammetry algorithms [8] which could be a plus on SLAM algorithms [9].

Whereas photogrammetry output is a 3D point cloud (a map), Simultaneous Localization And Mapping is more interested in Localization, and the map that SLAM builds is meant to be used for further localization only [10]. In addition SLAM must run in real time, using temporal coherency to recognize interests points between images whereas photogrammetry must proceed to extensive search between images to bring the coherency between them.

https://doi.org/10.33012/2021.18084

Today Simultaneous Localization And mapping Algorithms are strongly relying on a carefully designed visual odometry system that will track the images using temporal coherency (ie. a general motion profile). To this end inertial aiding has been particularly investigated because it is well suited for tracking short term movement to bring robustness and precision to visual odometry. Today inertial aiding is a must have feature of visual odometry and Visual SLAM system. We mostly find **tight coupling** approaches [11] for fusing inertial data with visual odometry. However the size and complexity of the visual inertial filter varies greatly depending on the visual odometry algorithm.

To improve the performance of our algorithm we explore a lot of possibilities including imu misalignment and camera calibration. This extensive research brings us to our main contribution:

- A system with unequaled localization accuracy on EuRoC dataset up to our knowledge.
- Which demonstrated the power of latest principles such as loop closing based on prediction only [3] and defered imu initialization [1], while using low level feature [12] and being deterministic [13].
- A highly versatile system able to perform Monocular or stereo with or without imu coupling. Using perspective or fish eye lens. Able to perform global bundle adjustment post-processing. With online or post-processed intrinsic calibration (camera center and camera radial distortion) and extrinsic calibration (imu misalignment and lever arm)

As a fundamental aspect of our work, we will detail our monocular odometry system based on the Variable State Dimension Filter [14], which follows our research on stereo odometry RADVO [13]. We will then presents its numerous capabilities.

II. RELATED WORK

1. Visual Odometry

The odometry algorithm is the core of any visual SLAM framework. From this algorithm, the performance of loop-closing and inertial coupling will depend. Among odometry algorithms we found:

- **Frame to frame odometry**, where the information between two images is reduced to a delta pose between two frames. For inertial coupling this delta pose becomes an additive state [13] of a classic inertial filter [15]. One can also couple visual odometry with a delta rotation and the gyroscope only [10, 16].
- **EKF SLAM**, one of the first attempt to SLAM was to use a small number of features that are part the Kalman filter state [17] [18]. This approach is very limited by the computation time that grows too quickly with the number of features.
- **MSCKF**, the Multi-States Constraint Kalman Filter [19] observes simultaneously features in successive frames (a dozen) and reduce the observation to the poses of the dozen frames. This algorithm was specially developed for imu aiding and the dozen poses are added as states of the filter. Here poses of images are added and deleted on the fly such that a fixed number of states remain. Some limitation about this approach is the necessity of a good initial prediction to obtain good jacobians for update. In addition, the MSCKF has to wait for the features to disappear to observe it. This again reduces the precision of jacobians.
- **Sliding Window Filters (SWF)** [20] are nowadays the most powerful methods for visual odometry. They include not only a dozen poses of the images but also thousands of map points as states. A lot of freedom is taken in state management to improve computational speed and accuracy. Like in MSCKF several poses are kept in the state for further observations but these are carefully chosen to avoid redundancy. The filter behaves at the same time like a nonlinear optimization and an extended Kalman filter. This allows to use all information available at a time, having always the best linearization point possible. Moreover the explicit consideration of features as states avoids statistics approximation of a reduced observation model.

As our system can be classified as a SWF, we will look at this algorithm more closely. Thanks to its formulation using the information matrix rather than the covariance, a significant part of the system to solve is diagonal. Then the problem is solved quickly using the Schur complement trick. The overall process also known as the Variable State Dimension Filter [14] was first used in photogrammetry. [14] gives all the tools to solve the SWF, among them the marginalization procedure to remove the states while staying in an optimal formulation. However according to [14] the VSDF was incompatible with imu prediction because it would have filled the information matrix, loosing the sparsity pattern. However looking deeper to the question, the SWF had the same limitations, and the solution was to use IMU error [21] also referred as IMU preintegration [22] [1].

The mathematical formulation is carrefully chosen to keep a good convergence behavior, for example inverse depth parametrization is used in [23] to avoid convergence issues during bundle adjustement. In this context to keep the system as linear as possible, a dedicated state variable is used to estimate the **gravity direction** for imu initialization, instead of estimating the initial roll

and pitch [24] [22]. Likewise, instead of estimating the scale for monocular inertial directly on the position states, a dedicated **scale state** is introduced [24] [22]. To push further the accuracy improvements, [1] is doing an additive non linear optimization after 2 seconds of visual SLAM to estimate properly a good initialization for imu biases, scale and gravity direction, before to continuously estimate these variables during navigation.

2. Feature matching

Feature matching is the data association part between images. A good outsight of feature matching used in visual odometry is available in [1].

a). direct methods

One can categorize visual algorithm into **direct methods** and **feature-based** methods [11]. A direct visual odometry method minimizes the pixel intensity error and jointly estimates illumination changes between images, feature position and image poses. Whereas feature based methods only minimize the feature position error to estimate the image poses. The approaches can be mixed:

- **Direct descent methods** based on the pyramidal lucas Kanade method are used in several "feature based" algorithm such as [25] [26].
- **A semi-direct method** introduced by [27] uses a direct method for state initialization and continues with corners features and edge features optimization.
- **Direct photometric feedback** is used in [18] to optimize the position of a multi-level patches in an IEKF framework.
- **Direct Sparse Odometry** [23] aggregate 8 pixels to form a direct feature in order to make the position of the feature recognizable.

b). Features

Special care is taken for features with EKF based methods. Since the number of feature is limited, features with higher complexity are needed to avoid mismatches or unused features. In addition to use direct photometric optimization, [17] and [18] takes into account the change in scale and the flattening of a patch caused by the viewing.

With SWF higher number of feature can be used and a lower complexity of feature is needed. Thousands of features are used in [23], among them features that can only be situated in 1 dimension because of their weak gradient. Edgelets are also dedicated features to treat points that can only be situated in 1 dimension [27].

Among the simplest features for visual odometry and SLAM we can aggregate a set of a few sparse pixels gradients [8] [16] [13], and associate two features using their sum of absolute differences. This is the Sum of Absolute Difference Matching. It is widely used in dense stereo reconstruction. We use this method in this paper.

Then comes a more versatile feature: ORB, Oriented FAST and rotated BRIEF. The FAST compares pixels in a circle around a corner to check if the corner is valid. The BRIEF is a binary descriptor that is the concatenation of intensity tests between pixels around the corner. The binary nature of the BRIEF makes this feature very efficient for registration in a database. This allows fast place recognition [28] for loop closing based on place recognition [29] [1].

3. Loop closing

Since [30], relocalization based on place recognition is widespread in SLAM algorithms [1, 10, 26, 29, 31]. The method used for localization has mostly been the same database used in ORBSLAM which is DBoW2 [28]. Lately, an alternative database called iobow-lcd [32] which does not need to be trained was used in [29].

Visual odometry plays a very big part in the accuracy of loop closing algorithms. The prediction of visual odometry allows to avoid wrong loop closure. [1] explained that the main accuracy difference between their algorithm and other SLAM systems with loop detection comes from their mid-term data association which is based on the prediction only to close loop, not place recognition.

We can then understand why relocalization based only on predition [3] has proven to be successfull and competitive. This is the relocalization method we use in this paper. At a higher computation price [3] use a pyramidal descent method in order to increase the admissible uncertainty of visual odometry before that relocalization based on prediction becomes useless.

The global map that is necessary to close loops can also jointly be optimized in a parallel thread to improve the real time accuracy [1]. Alternatively to get the best accuracy possible one can also proceed to a global bundle adjustement in post-processing [30]. In this article we present a global optimization working on post-processing only.

III. THE VSDF ALGORITHM

The VSDF is a combination of the EKF and nonlinear optimization [2]. A first part of the VSDF is a filter able to iteratively optimize its states while taking into account a probabilistic prior. The second part of the VSDF allows to add and delete states to keep only the usefull states depending on the variables that need to be optimized. These two parts are tightly dependant because once observations are cumulated to the prior, all variables relative to this observation do not need to be kept in the filter state and are therefore deleted. This makes the VSDF suitable for real time SLAM. As the VSDF was specifically developped for photogrammetry [14], it was in its design also aware of the sparsity pattern of SLAM systems. And during optimization, the linear system was smartly inverted using block inversion also known as the Schur complement method.

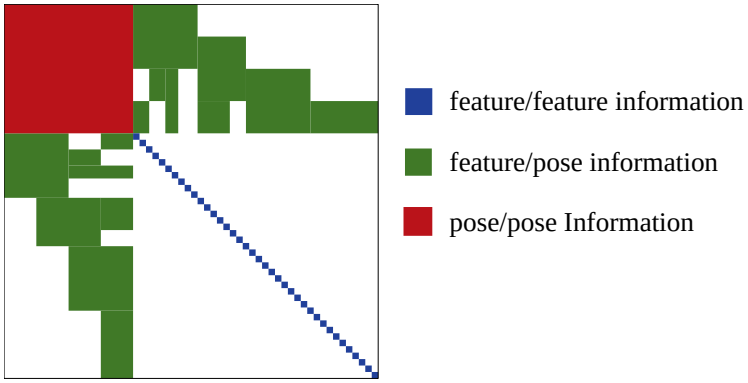

Figure 1: Information matrix for a general sliding window system that is solved using the Schur complement.

1. optimization and observation

Let us begin with the general optimization formula. Given a set of observations $h_k(x)$, which represents reprojection errors with covariance R_k, we define the global optimization problem as:

$$\min_x \sum_k h_k(x)^T R_k^{-1} h_k(x) \tag{1}$$

To solve this mean square problem, one needs to use the Gauss-Newton algorithm to find the place where the gradient of this function cancels. This is an iterative algorithm where at each iteration a correction of x is calculated:

$$dx = \left(\sum_k H_k^T R_k^{-1} H_k \right)^{-1} \left(\sum_k H_k^T R_k^{-1} h_k(x) \right) \tag{2}$$

With $H_k = \frac{\partial h_k}{\partial x}$ the gradient of the observation function. With the new x value the jacobians and observations function must be recomputed and this process repeated until convergence.

This process is well suited for global bundle adjustment, for optimizing the full map. But if the algorithm must run in real time, and images are coming one by one, the observation of old frames can be observed within the meaning of the EKF. For example we want to observe h_1 so that it does not appears anymore in the optimization equation 2. We will approximate the terms involving h_1 by its first order expansion in x. Any linear expansion in x can be obtained using a fixed term x_0 and a coupling term A_0. Then observing h_1 transforms the equation 2 to:

$$dx = \left(A_0 + \sum_{k=2}^{n} H_k^T R_k^{-1} H_k \right)^{-1} \left(A_0(x_0 - x) + \sum_{k=2}^{n} H_k^T R_k^{-1} h_k(x) \right) \tag{3}$$

To be equivalent to the first order expansion x_0 and A_0 are set to:

$$x_0 = x + (H_1^T R_1^{-1} H_1)^{-1} (H_1^T R_1^{-1} h_1(x)) \tag{4}$$

$$A_0 = H_1^T R_1^{-1} H_1 \tag{5}$$

We then understand the meaning of x_0 which is the prior state and A_0 the prior information which is also the inverse of the prior covariance $P_0 = A_0^{-1}$. To observe more variables such as h_2 while keeping the same optimum as equation 3, we set the prior to:

$$x_0^+ = x + \left(A_0^- + H_2^T R_2^{-1} H_2\right)^{-1} \left(A_0^- (x_0^- - x) + H_2^T R_2^{-1} h_2(x)\right) \tag{6}$$

$$A_0^+ = A_0^- + H_2^T R_2^{-1} H_2 \tag{7}$$

Equations 6 and 7 define the general VSDF observation process, and equation 3 defines the VSDF batch procedure for computing the best estimate x. All equations are always computed at the best estimate x. To be effective, the observation process must be done using many observations at the same time. In particular to make A_0 and x_0 well defined at start-up since A_0 needs to be inverted in equation 4. What differs from the classic derivation of marginalization or VSDF is the computation of x_0. This involves inverting the system one more time to perform observation, but it removes the need of a dedicated variable to store $a_0 = A_0 x_0$. And it is more compatible we believe with non linear states such as rotation where in equation 3 the difference $(x_0 - x)$ can be replaced with a dedicated difference metric for the rotation states.

2. state management

Once observations are processed, there are parts of the state that will no longer influence other states. For example one can observe all features that appear in a frame within the meaning of the VSDF, then remove all the position of the features that have been observed, and remove the states that are relative to this frame (its pose).

States deletion while remaining optimal is called marginalization [33]. Whereas removing states of a Kalman filter is straight forward, removing states of an information filter involves using the Schur complement.

Table 1: Marginalization on a Gaussian distribution expressed in covariance and information

	Kalman Filter	Information Filter
Distribution	$\mathcal{N}\left(\begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix} \begin{bmatrix} P_{\alpha\alpha} & P_{\alpha\beta} \\ P_{\beta\alpha} & P_{\beta\beta} \end{bmatrix}\right)$	$\mathcal{N}^{-1}\left(\begin{bmatrix} a_\alpha \\ a_\beta \end{bmatrix} \begin{bmatrix} A_{\alpha\alpha} & A_{\alpha\beta} \\ A_{\beta\alpha} & A_{\beta\beta} \end{bmatrix}\right)$
Marginalization	$x = x_\alpha$ $P = P_{\alpha\alpha}$	$a = a_\alpha - A_{\alpha\beta} A_{\beta\beta}^{-1} a_\beta$ $A = A_{\alpha\alpha} - A_{\alpha\beta} A_{\beta\beta}^{-1} A_{\beta\alpha}$

To sum up, the Schur complement method is used both for VSDF optimization and for the state suppression.

3. prediction extension

Adding states to the VSDF filter is straightforward, one only needs to initialize the line and column of the prior information matrix to zero ($A_{i*} = A_{*i}^T = 0$). For example a new frame on the system can be added. But in case there is a prior on the frame position coming from an imu, we cannot write easily the link there is with this frame inside the information matrix. What we want is state prediction. This can only be performed on the covariance. By doing so the sparsity of the information matrix is lost [14]. [2] managed to reduces the complexity of such procedure using Cholesky decomposition. Today imu preintegration treats this problem more successfully, imu prediction is replaced by imu observation. And the imu related states (speed, biases...) can be iteratively refined during a short period before to be marginalized.

Still there remains room for a method that uses optimization and imu prediction: The idea is to observe all observations before prediction. The success of such a method lies mostly in the following assumptions:

- All frames are located accurately after a few Gauss newton iterations.

To validate this hypothesis, a high number of new features is needed at each new image to provide enough observation. To do so we risk to use features that have already been taken, but we neglect this problem thanks to another assumption:

- A same feature with different points of view can be considered as a different features which will provide quite independant observations. This assumption is successfully used in DSO [23] which allows to use always a high number of features on each new image to instantly locate the new frame closely to the optimum.

4. dropping observations

This prediction method is only useful with imu coupling. In case of pure visual SLAM, the observation of features can be delayed. However this increases the number of feature to optimize at the same time. As the improvement due to the reuse of features seems minimal compared to the increase in computation time, we do not reuse features for visual only SLAM.

However what is done in visual only SLAM and not done in visual inertial SLAM is dropping completely the measures of frames that are not keyframes. In inertial SLAM regular observations are necessary to avoid imu biases to diverges, But in visual only SLAM, frames that are not keyframes are just redundant. They are only used to pre-position the next keyframe. The features anchored in a frame that is not a keyframe disappear when the next frame comes without being observed or marginalized.

5. loop closing approximation

Our loop closing strategy lies in the reobservation of fixed old keyframes that are located close to the current image. Neither the VSDF nor the SWF allow in their optimal formulation to reobserve such frames. To do so we must use the approximation that the older keyframes are precisely enough located so that we can consider them as fixed. Many algorithms such as [30] [3] use fixed keyframes to anchor the young frames that are optimized in the sliding window. These algorithms do not use marginalization nor a variable state dimension filter. However [34] developed a hybrid sliding window filter able to observe fixed frame while marginalizing others.

We apply the same principle: we use 10 frames in the sliding window made of the current moving frame, 5 moving keyframes and 4 fixed keyframes. Table 2 shows the final architecture. The improvement is twofold:

- Since the keyframes are fixed, there is no need to consider their covariance, so the system to solve is smaller.
- This opens the way to loop closing by linking the current frame to any frames that cross its way and look roughly to the same direction. Since we neglect the covariance of fixed frames we can embed any old frame as a fixed frame without bothering with its lost covariance.

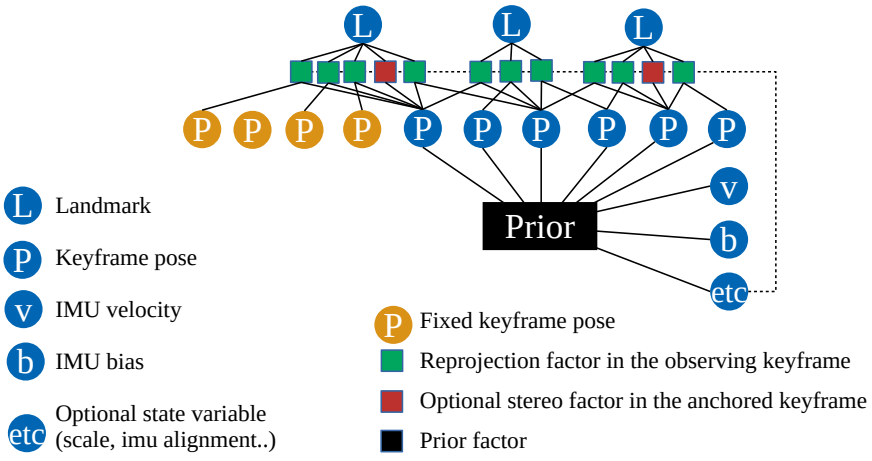

Figure 2: Structure of our Variable State Dimension Filter with 10 keyframes among them 4 are fixed.

We pick a set of 4 keyframes at each keyframe creation. These keyframes are chosen by sorting all previous keyframes according to a custom metric depending on the distance of a frame and the difference of the looking direction with the current frame. Contrary to [3] we do not take in to account the average feature depth to pick our keyframes. Whereas with good prediction

situation this could increase the loop closing efficiency, in case of different profile of feature depth caused by occlusion or blur, loop closure could be missed.

IV. MONOCULAR SLAM

1. model formulation

To model the feature position we use the representation employed by direct approaches [23]. 3D features are represented as inverse depth in a reference frame.

$$X_k = \begin{pmatrix} x_k \\ y_k \\ z_k \end{pmatrix} = \pi_k^{-1}(d_k) = \begin{pmatrix} u_k/(fd_k) \\ v_k/(fd_k) \\ 1/d_k \end{pmatrix} \tag{8}$$

This reference frame is also called the anchored frame [34] for this feature. The 3 degrees of freedom of a feature anchored in a frame comes from the ray defined by the pixel coordinate of the feature in its anchored frame and the inverse depth that places the features on the ray. During optimization, the pixel coordinate of the feature does not move, but the attitude of the anchored frame can move, so the ray of the feature will move. Its position on the ray moves also as its inverse depth is being optimized.

To observe this feature k on another frame i, its 3D position in the anchored frame a must be transformed to global position, and then re-transformed to local coordinates of frame i

$$Y_{k,i}(X_k) = R_i^T (R_a X_k + t_a - t_i) \tag{9}$$

Then this 3D position is re-projected in i

$$\begin{pmatrix} \widehat{u_{k,i}} \\ \widehat{v_{k,i}} \end{pmatrix} = \pi(X_{k,i}) = \begin{pmatrix} fx_{k,i}/z_{k,i} \\ fy_{k,i}/z_{k,i} \end{pmatrix} \tag{10}$$

The observation of feature k in frame i is then:

$$h_{k,i}(\Theta_a, \Theta_i, d_i) = \pi \circ Y(\Theta_a, \Theta_i) \circ \pi^{-1}(d_k) - \begin{pmatrix} u_{k,i} \\ v_{k,i} \end{pmatrix} \tag{11}$$

This observation involves the position of the feature in the anchored and target frame in pixel $(u_k, v_k)^T$ and $(u_{k,i}, v_{k,i})^T$ and also the state variable relative to its depth d_k and the states of the anchored frame Θ_a and target frame Θ_i.

The observations are weighted using the Cauchy loss, a robust weighting function close to the one used in [3] which is stricter than the more commonly used Huber loss used in [23] and [1]. The noise covariance of the observation depends of the localization precision of the feature in the frame. Since the feature is located at a pixel precision, we assumed the pixel coordinate has a standard deviation of half a pixel. The Cauchy loss is computed using the average residual of all the observations of the same feature. The number of observation per feature varies from 1 to the maximum number of keyframes on the window $max(N_{obs}) = 10$.

2. feature matching

a). descriptor

Interest points are matched using Normalized Sum of Squared Difference [13] inspired from [12]: To measure similarity between pixels for matching, a descriptor vector is defined. The descriptor vector is composed of 32 pixels of the spatial derivative of the image situated in a sparse manner around the interest point. Then to measure the distance between two descriptors, we compute their Normalized Sum of Squared Differences (NSSD). We keep the best match among the feature tested.

To reduce the tests between features and avoid wrong associations, prior poses of the frames are needed along with a prior inverse depth of the feature. To yield a good prior depth for the features in monocular case, an inverse depth map is tracked the same way [23] does: The points from the last keyframe is reprojected on the next frame, and slightly dilated, creating a semi dense inverse depth map. In case no inverse depth prior is available at some feature point, the median inverse depth of the last

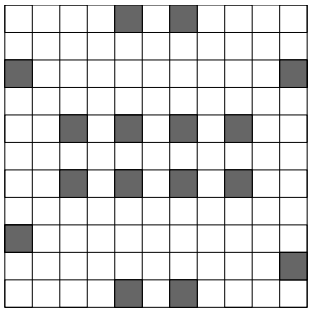

Figure 3: Position of the pixels used for the descriptor. There are 16 pixels distributed in a sparse manner in a 11x11 window around the feature center. The descriptor is the concatenation of the intensity of the smoothed x gradient and y gradient (ie. sobel transform) at these pixel positions.

keyframe is used. Figure 4 shows a thousand features matched between the last keyframe and the current frame thanks to this technique.

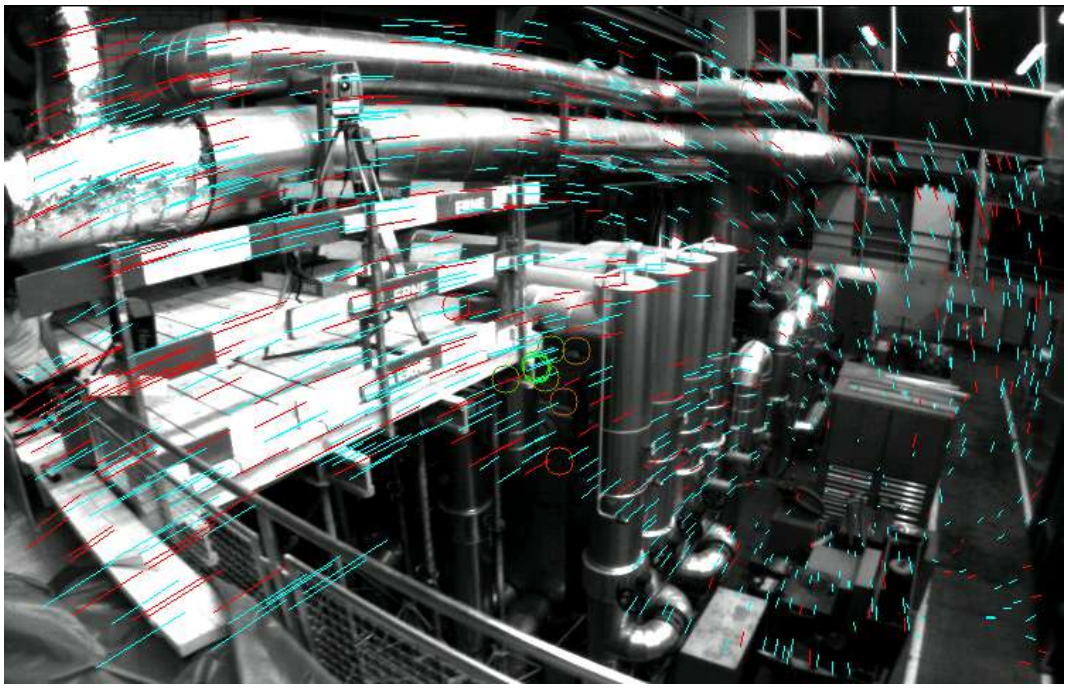

Figure 4: Flow of the features matched between the last keyframe and the current frame in the sequence MH_03_medium. In blue are the 70% best features according to the chi2 test and the red are the 30% worst. The viewing direction of the 10 keyframes with respect to the last one is displayed with circles. Green circles represent the latest keyframe and red circles represent the oldest.

b). stereo extension

Stereo observation can be added to the system in case a stereo rig is available. Our approach is similar to [35]. The algorithm mostly work the same way as for monocular, the stereo observation for a feature is added on the anchored keyframe only. To obtain robust startup in case of stereo, only stereo features are used during the first second. Then stereo and pure monocular features are allowed. Monocular only observations in stereo mode allows to treat generate cases for stereo. For example:

- When the camera is too close from an object (for example when the drone is on the ground), the stereo disparity is too big and stereo matching can fail, but this is not an issue for monocular which continue the tracking.
- When the viewing distance is too far (if the drone gets altitude), the disparity of a feature becomes too small, and can bias the results. Stereo must be deactivated.

3. keyframe management

a). keyframe creation

All frames are tracked the same way, but some frames will be considered as keyframes and will be reused in the window system. Two tests are done for the keyframe decision:

- If the mean translationnal flow [23] of the features between the current frame and the last keyframe exceeds a value then the frame becomes a keyframe. The translationnal flow can be computed for a feature using the observation function by setting its inverse depth to 0. $f_t = \overline{||h_{k,i}(\Theta_a, \Theta_i, 0)||}$.
- Or if the rotation between the current frame and the last keyframe exceeds a value then the frame becomes a keyframe.

b). keyframe distribution

Once a frame becomes a keyframe, another keyframe of the sliding window must be removed. We want in the end to have more keyframes close to the recent ones, and keep a few old frames at the end of the window to be able to close little loops and slow down drift.

Our procedure is almost the same as the one from DSO [23]:

- The latest two keyframes are kept (I_1 and I_2)
- If there are frames with less than 5% of their points visible in I_1 the oldest of them is removed.
- If not the keyframe which maximizes a distance score $s(I_i)$ is removed:

$$s(I_i) = \sqrt{d(i,1)} \sum_{j \neq i} (d(i,j) + \varepsilon)^{-1} \tag{12}$$

where $d(i,j)$ is the Euclidean distance between keyframes I_i and I_j. As this score does not depends on the looking direction of I_1 but only on the position on I_1 the 5% test becomes indispensable to treat cases where the keyframes does not look anymore on the good direction. This avoids the failure which consists in ending up with only frames that cannot be matched with the current frame.

In our case if loop closing is enabled, the 4 last keyframes among the 10 keyframes are replaced by others depending on our loop closing procedure. That means that the latter algorithm for deleting keyframes has no effect in case one of the 4 last keyframe are chosen to be deleted. However as the last 4 keyframes chosen for loop closing enter the window, at the next keyframe decision, they will influence the distance score as well. To avoid any interference between the two keyframe management procedures, we designed the keyframe selection procedure so that it keeps the same spacial distribution as the keyframe removing procedure.

V. INERTIAL COUPLING

1. model

Our visual-inertial filter is made of a classic EKF [15] augmented with carefully chosen old state variables such as keyframe poses. During observation the feature parameters are also added to the state of the system, yielding the following state:

$$x = (v, b_a, b_g, a_i, l_i, s, \Theta_0, ..., \Theta_5, \rho_0, ..., \rho_N) \tag{13}$$

with:

- v the speed
- b_a the accelerometer biases
- b_g the gyrometer biases
- a_i an optional alignment angle between imu and vision
- l_i an optional lever arm between imu and vision
- s the scale correction between vision and imu, only useful for monocular inertial
- Θ_i the pose of a frame i made of its position and attitude.
- ρ_j the inverse depth parameters of the feature j.

We also added several values that can be estimated online such as camera to imu misalignment and lever arm and also a specific variable to refine scale during navigation after the first initialization step for monocular inertial.

We have defined the procedure of feature observation in parts IV.1 and III.1. We will now present the prediction part.

Our Kalman prediction is the same as the EKF SLAM prediction [36]. This prediction can be efficiently done since a big part of the covariance is unmodified:

$$\dot{P} = \begin{pmatrix} F \\ 0 \end{pmatrix} P + P \begin{pmatrix} F \\ 0 \end{pmatrix}^T + \begin{pmatrix} G & 0 \\ 0 & 0 \end{pmatrix} \tag{14}$$

with $F = \frac{f(x_r)}{\partial x_r}$ the jacobian of the prediction function $f(x_r)$ of the "classic EKF", ie. the prediction of the state when it is reduced to $x_r = (v, b_a, b_g, a_i, l_i, s, \Theta_0)$. Likewise only the reduced part of the state is predicted $\dot{x_r} = f(x_r)$.

In our case the information matrix needs to be inverted for prediction. To make the operation tractable, all the feature parameters are removed from the state before prediction (see III.3). This limitation could be removed in further development thanks to imu preintegration [22]. In the meantime this formulation allows us to use advanced states with lever arm and misalignment estimation.

2. monocular initialization

Whereas stereo inertial navigation initializes with any help, monocular inertial navigation has to deal with the scale estimation. [22] lets the filter estimate the scale, but as the problem is highly non-linear it exhibits slow convergence and still needs additive procedure to converge.

Lately, [1] proposed to address the issue by first solving pure monocular odometry for a few seconds, and then using this trajectory to find the imu biases, speed, scale and gravity direction in an optimization procedure during which the monocular trajectory is untouched. This procedure is quite optimized since only a short period of time is necessary for initialization, so the imu biases in this period can be approximated as constant, and the optimization procedure to align imu and vision becomes quite lightweight.

We propose to follow [1] but instead of using pure monocular at start, we use tight coupled monocular with gyrometer. Indeed angles exhibit no convergence issue. This improves the initialization over a pure monocular method and save us the expense of estimating the gyrometer biases later in the optimization procedure. To sum up we solve gyrometer coupled monocular odometry for a few seconds, and then use this trajectory to find the mean accelerometer biases, speed, scale and gravity direction in an optimization procedure.

Once the initial biases, speed, scale and gravity direction is estimated, the tight coupling begins and the scale is continuously refined as in [1].

VI. EXPERIMENTAL RESULTS

Our system is evaluated using 3 datasets:

- The EuRoC [4] dataset using four sensor configurations: Monocular, Monocular-inertial, Stereo and Stereo-Inertial.
- The TUM-VI [37] dataset with fish eye cameras in Monocular-inertial and Stereo-inertial configurations.
- A photogrammetry dataset where the calibration of the camera will be experimented.

We measure the accuracy using the absolute translation error. It is the translation root-mean-square-error (RMSE) after rotation and translation fitting between the ground truth and the result. In case of pure monocular SLAM, the scale is unobservable. So the scale is jointly fitted before to compute the RMSE. We use EVO [38] a third-party evaluation software to compute these metrics.

1. Results on EuRoC

EuRoC dataset [4] is a set of 11 sequences of an indoor flying drone which is equipped with two 0.25Mp cameras and an inertial momentum unit (IMU). It exhibits several rapid motions, blur and brightness change. While beeing difficult, it provides a lot of loop closing to detect and use. We compare the performance of our algorithm with the most relevant systems in the literature.

Table 2: Performance comparisons on the EuRoC dataset (RMSE ATE in m.).

		MH01	MH02	MH03	MH04	MH05	V101	V102	V103	V201	V202	V203	Avg [1]
Monocular	DSO [23]	0.046	0.046	0.172	3.810	0.110	0.089	0.107	0.903	0.044	0.132	1.152	0.601
	DSM [3]	0.039	0.036	0.055	0.057	**0.067**	0.095	0.059	0.076	0.056	0.057	**0.784**	**0.126**
	ORB-SLAM3 [1]	**0.016**	**0.027**	**0.028**	0.138	0.072	0.033	**0.015**	**0.033**	**0.023**	**0.029**	-	-
	DVM (ours)	0.032	0.069	0.067	**0.051**	0.083	**0.028**	0.529	0.366	0.088	0.336	1.348	0.272
Stereo	SVO [27]	0.08	0.07	0.27	2.42	0.54	0.04	0.08	0.36	0.07	0.14	-	-
	RADVO [13][2]	0.097	0.062	0.154	0.209	0.168	0.081	0.186	0.232	0.139	0.151	3.426	0.446
	ORB-SLAM3 [1]	0.029	**0.019**	**0.024**	0.085	**0.052**	**0.035**	**0.025**	**0.061**	**0.041**	**0.028**	**0.521**	**0.084**
	DVM (ours)	**0.020**	0.033	0.029	**0.048**	0.061	0.037	0.071	0.132	0.050	0.093	1.444	0.183
Monocular Inertial	VI-DSO [22]	0.062	0.044	0.117	0.132	0.121	0.059	0.067	0.096	0.040	0.062	0.174	0.089
	HSWO [34]	0.048	0.036	0.063	0.099	0.063	**0.035**	0.036	0.057	**0.027**	0.027	0.072	0.051
	ORB-SLAM3 [1]	0.062	0.037	**0.046**	**0.075**	**0.057**	0.049	**0.015**	**0.037**	0.042	**0.021**	**0.027**	**0.043**
	HybVIO [39]	0.18	0.066	0.12	0.21	0.31	0.069	0.06	0.081	0.055	0.091	0.13	0.13
	DVM (ours)	**0.019**	**0.024**	0.060	0.119	0.089	0.036	0.024	0.041	0.071	0.046	0.059	0.053
Stereo Inertial	SOFT-SLAM [10]	0.028	0.042	0.038	0.096	0.058	0.042	0.034	0.057	0.072	0.069	0.173	0.065
	RADVO [13]	0.074	0.074	0.092	0.086	0.087	0.074	0.089	0.101	0.070	0.048	0.675	0.133
	ORB-SLAM3 [1]	0.036	0.033	0.035	0.051	0.082	0.038	0.014	0.024	**0.032**	**0.014**	0.024	0.035
	HybVIO [39]	0.088	0.081	0.038	0.071	0.11	0.044	0.034	0.040	0.073	0.040	0.051	0.061
	DVM (ours)	**0.019**	**0.017**	**0.031**	**0.038**	**0.050**	**0.037**	**0.013**	**0.023**	0.041	0.025	**0.022**	**0.029**

[1] Average error of the sequences.
[2] Error obtained by our-self using RADVO.

As shown in table 2 we achieve the best result in the most precise configuration (stereo inertial) on the EuRoC dataset. Moreover we are able to compete with the best algorithm available in the literature in all other categories. While competitive performance in inertial aided configuration is more common among odometry algorithms, it is not the case in pure stereo and pure monocular configurations on the EuRoC dataset. Apart from ORB-SLAM [30] and its derivative, only DSM [3] and DVM(ours) manage to have competitive results in monocular situation. However DSM [3] required a significant higher computation cost to achieve their results. Likewise, in pure stereo, this is more surprising to see this phenomenon again where only ORB-SLAM [30] based algorithms and DVM (ours) are far ahead other algorithms.

We also include results of RADVO [13] as it is interesting to see how a frame-to-frame odometry system behave comparing to sliding window systems. Surprisingly it stands a right place in this comparison thanks to its robustness whereas other sliding window algorithm tends to experience failure. We can understand such behavior because of the simplicity of frame-to-frame algorithms that decorelates results from one frame to another, whereas errors on a sliding window can be amplified and lead to system divergence.

2. Results on TUM-VI

The TUM-VI dataset [37] includes 28 sequences in different environments. The dataset have been recorded on foot with a stereo fish eye camera and an IMU. The starting point and ending points are given in a ground truth. The distance travelled during a sequence varies from a few meters to 1km. As the camera is handheld it exhibits dynamic motion that makes pure visual odometry difficult but allows the IMU biases to be estimated.

Starting from easy to difficult logs there are:

- 6 ”rooms” sequences which takes place in the room where the ground truth is available. This makes easier logs than on the EuRoC dataset.
- 5 ”corridor” sequences where the stereo rig goes threw a corridor and comes back. According to [1] there are several loops to close in these dataset.
- 3 ”slides” sequences where the stereo rig goes down two floors in a closed slide. This sequence is interesting as in the slide almost no camera information is usefull, and the imu alone gives the position at the end of the slide. Hopefully as

Table 3: Performance comparisons in the TUM-VI dataset (RMSE ATE in m.).

	Mono-Inertial			Stereo-Inertial					
Seq.	VINS-Mono	ORB-SLAM3	DVM	OKVIS	ROVIO	BASALT	ORB-SLAM3	DVM	Length (m)
corridor1	0.63	**0.04**	1.11	0.33	0.47	0.34	**0.03**	0.27	305
corridor2	0.95	**0.02**	1.10	0.47	0.75	0.42	**0.02**	0.49	322
corridor3	1.56	**0.31**	0.97	0.57	0.85	0.35	**0.02**	0.46	300
corridor4	0.25	0.17	**0.15**	0.26	0.13	0.21	0.21	**0.07**	114
corridor5	0.77	**0.03**	0.92	0.39	2.09	0.37	**0.01**	0.29	270
magistrale1	2.19	**0.56**	7.10	3.49	4.52	1.20	0.24	1.38	918
magistrale2	3.11	**0.52**	7.50	2.73	13.43	1.11	**0.52**	6.12	561
magistrale3	**0.40**	4.89	2.88	1.22	14.80	0.74	1.86	**0.68**	566
magistrale4	5.12	**0.13**	0.88	0.77	39.73	1.58	**0.16**	3.01	688
magistrale5	**0.85**	1.03	1.44	1.62	3.47	**0.60**	1.13	0.90	458
magistrale6	2.29	**1.30**	2.09	3.91	X	3.23	**0.97**	1.02	771
outdoors1	74.96	70.79	**16.90**	X	101.95	255.04	32.23	**14.35**	2656
outdoors2	133.46	**14.98**	16.21	73.86	21.67	64.61	**10.42**	16.63	1601
outdoors3	36.99	39.63	**14.37**	32.38	26.10	38.26	54.77	**12.28**	1531
outdoors4	16.46	25.26	**13.71**	19.51	X	17.53	11.61	**7.84**	928
outdoors5	130.63	14.87	**9.65**	13.12	54.32	7.89	8.95	**3.47**	1168
outdoors6	133.60	**16.84**	21.86	96.51	149.14	65.50	**10.70**	29.7	2045
outdoors7	21.90	7.59	**3.74**	13.61	49.01	4.07	4.58	**3.74**	1748
outdoors8	83.36	27.88	**4.77**	16.31	36.03	13.53	11.02	**2.38**	986
room1	0.07	**0.01**	0.03	0.06	0.16	0.09	**0.01**	**0.01**	146
room2	0.07	**0.02**	**0.02**	0.11	0.33	0.07	**0.01**	**0.01**	142
room3	0.11	0.04	**0.02**	0.07	0.15	0.13	**0.01**	**0.01**	135
room4	0.04	**0.01**	0.03	0.03	0.09	0.05	**0.01**	**0.01**	68
room5	0.20	**0.02**	0.03	0.07	0.12	0.13	**0.01**	**0.01**	131
room6	0.08	**0.01**	0.02	0.04	0.05	0.02	**0.01**	0.02	67
slides1	**0.68**	0.97	3.15	0.86	13.73	**0.32**	0.41	3.26	289
slides2	**0.84**	1.06	2.12	2.15	0.81	**0.32**	0.49	0.60	299
slides3	**0.69**	**0.69**	2.64	2.58	4.68	0.89	**0.47**	1.48	383

presented in [37] most visual inertial system do not struggle too much in this situation.

- 6 ”magistrale” sequences, where the stereo rig goes threw corridors and a high ceilings hall.
- 8 ”outdoors” sequences, the longest sequences. The main difficulties in these sequence are the slowly moving sky [1] that can take up to more than half the field of view of the camera. Whereas [1] treats this problem by discarding all points that are farther than 20m, we discard all point that are ten meters above the camera. Hence this is more versatile solution as it can apply for high flying drone looking downward, for example for photogrammetry.

Our main strength is the the robustness and precision we achieve on outdoors datasets as shows Table 3. Thanks to our accurate sky rejection and high frequency keyframing, we are able to reduce drift on these very long sequences and achieve the best accuracy. As well as in small sequences (the rooms) we reach highest precision right behind ORB-SLAM3, where prediction is sufficient enough for our algorithm to close loops.

In the other sequences we have competitive and robust performances. But we see that an algorithm based on place recognition like [1] can close more loops to achieve better performances. However loop closing based on prediction can still be improved as shows [3] using pyramidal descent for example.

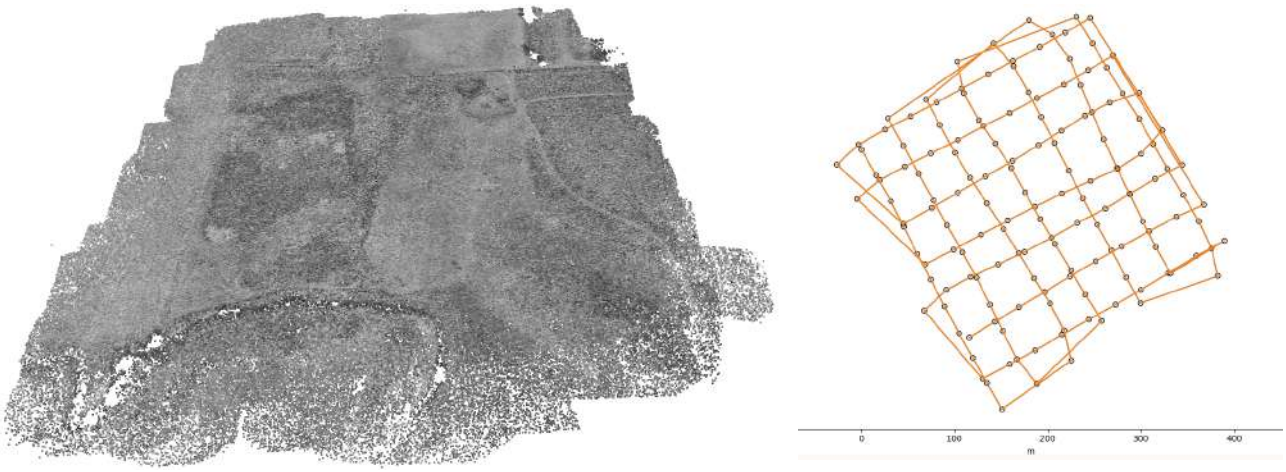

Figure 5: Left: coarse point cloud of a high resolution double grid acquisition. The image have been processed at a quarter of their initial resolution. Right: disposition of the 150 images used in this 3d point cloud.

3. Results on Mapping

Here we process a double grid image acquisition of a flying drone. The acquisition is made of 150 images at 42 Mp. We implemented the coarse part of a photogrammetry processing that is global bundle adjustment at 1/4 of the resolution, thus processing images of 2.6 Mp. This coarse bundle adjustement uses still a higher resolution than the real time dataset such as EuRoC.

The image position are prepositionned using RTK GNSS. About 10000 features are matches per images, which makes about one million features optimized simultaneously as shows figure 5. Then the intrinsic camera calibration is jointly estimated with each image poses. For more precision we also made possible GNSS coupling with joint lever arm estimation.

4. Calibration

Several parameters are needed to successfully couple the different sensors.

- The **intrinsic calibration** refers to the camera distortion parameters made of the radial distortion parameters, the focal length and the image center.
- The **extrinsic calibration** refers to the affine transformation between the different sensors.

These calibration parameters are usually estimated beforehand [4] [37]. However such calibration procedure can be insufficient. The extrinsic calibration parameters can shift over time. The initial calibration can be incorrect. As a matter of fact, such problem happens very often: On the three dataset presented here, all three had to be recalibrated.

- The affine transform between left and right camera on several EuRoC dataset needed correction as shown by [16].
- We experienced better performance on all TUM-VI logs, using the affine transformation between imu and vision estimated online by our-self.
- High resolution mapping always requires joint intrinsic calibration. Whereas little perturbation have negligible impact on low resolution cameras, they must be compensated for high resolution cameras.

In this context it becomes important to have broad calibration capability.

a). extrinsic calibration

We then developed an online estimation of the affine transform between the imu frame and the visual frame. The model that allows us to estimate such lever arm and misalignement is not different from any classic EKF as explained in V.1. We first tested it on EuRoC dataset. We realized that the online calibration precision was equivalent to the one given with the dataset: The SLAM accuracy using the affine transform estimated by ourself was the same as using the affine transform provided in

the dataset. In addition, we tested this online estimation with more difficult start, for example with a zero lever arm, or with five degrees of misalignement between the stereo rig and the imu: The convergence was still a success. We also successfully performed this estimation in monocular inertial, thus stereo is not mandatory.

After having tried and tested this estimation, we applied it on TUM-VI, and at our surprise, our estimated parameters were quite different from the one given in the dataset. And the SLAM accuracy using the newly estimated alignment and lever arm was better.

b). intrinsic calibration

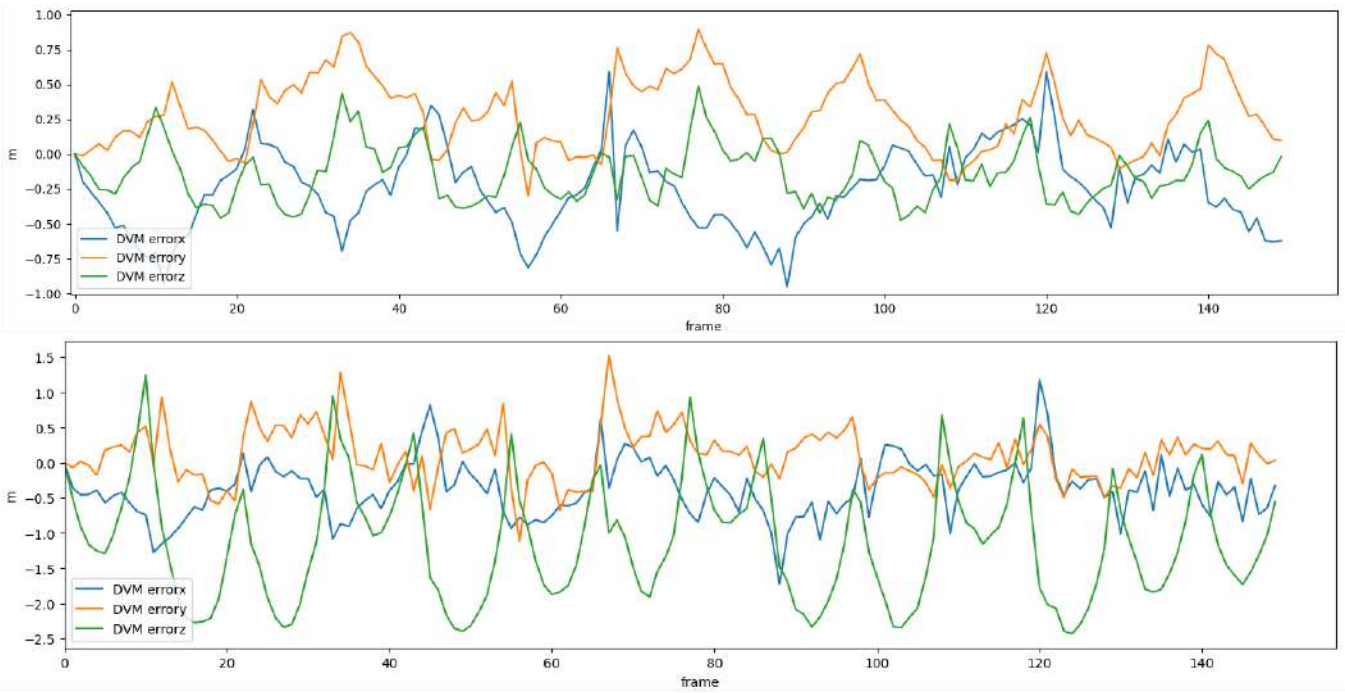

Figure 6: Difference between GNSS and our results after global bundle adjustement. Top: reusing the camera distortion parameter estimated from a GNSS aided run. Bottom: estimating the camera distortion without prior. The unaided calibration run exhibit higher error on x and y and a particular error on z which is characteristic of an erroneous curvature of the global trajectory of the UAV. This particular error can be explained by the ambiguity between the soil curvature and the camera radial distortion.

Intrinsic camera calibration is more difficult. The self estimation of distortion parameters seems to give less precise parameters than the ones obtained using a check-board grid. However the estimation of such parameters using additive information as shows [40] seems to be the key to achieve best accuracy. This additive information comes most generally from **GNSS**. In this case the joint global bundle adjustment and calibration is successful.

we further show in figure 6 that the calibration precision is dependent on the precision of additive sensor data. When the weight of the GNSS position is zero, the camera distortion is underestimated, and a wrong curvature appears on the map. On the other hand, setting the weight of GNSS position to its real RTK fixed uncertainty (5cm), a better distortion calibration is obtained, giving no curvature on the map. A similar behavior can be experienced using the photogrammetry software Pix4D [5].

This phenomenon can be understood by the weak observability of the photogrammetry dataset presented here. in figure 5 the mapped soil is very flat and the pictures have a limited overlap. Hence the curvature of the soil and the radial distortion of the camera are ambiguous.

VII. CONCLUSION

We have presented a highly versatile, fully deterministic VSLAM method. Our approach uses as well photogrammetry background as SLAM background. Instead of being limited by one or the other, or their intersection, we managed to combine all functionnality of both, leading to an unprecedented precision and versatility. Our main contribution is the development of

a Variable State Dimension Filter able to perform IMU tight-coupling and loop closing, and online lever arm and alignment calibration.

Our system runs at real time using only one core, on EuRoC or TUM-VI without accuracy loss. Thanks to its simple feature able to capture little details, it also yields photogrammetry grade point cloud. We show that after a sufficiently high degree of accuracy, the loop closure based only on prediction is efficient without the need of an additive algorithm to handle convergence. Still this work could benefit of further developments such as place recognition, GNSS tight coupling, online global bundle adjustment.

REFERENCES

[1] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, 2021.

[2] M. C. Deans, "Maximally informative statistics for localization and mapping," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 2. IEEE, 2002, pp. 1824–1829.

[3] J. Zubizarreta, I. Aguinaga, and J. M. M. Montiel, "Direct sparse mapping," *IEEE Transactions on Robotics*, vol. 36, no. 4, pp. 1363–1370, 2020.

[4] 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.

[5] J. Vallet, F. Panissod, C. Strecha, and M. Tracol, "Photogrammetric performance of an ultra light weight swinglet uav," Tech. Rep., 2011.

[6] N. Keivan and G. Sibley, "Online slam with any-time self-calibration and automatic change detection," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 5775–5782.

[7] C. Becker, N. Häni, E. Rosinskaya, E. d'Angelo, and C. Strecha, "Classification of aerial photogrammetric 3d point clouds," *arXiv preprint arXiv:1705.08374*, 2017.

[8] E. Tola, C. Strecha, and P. Fua, "Efficient large-scale multi-view stereo for ultra high-resolution image sets," *Machine Vision and Applications*, vol. 23, no. 5, pp. 903–920, 2012.

[9] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1689–1696.

[10] I. Cvišić, J. Ćesić, I. Marković, and I. Petrović, "Soft-slam: Computationally efficient stereo visual simultaneous localization and mapping for autonomous unmanned aerial vehicles," *Journal of field robotics*, vol. 35, no. 4, pp. 578–595, 2018.

[11] M. Servières, V. Renaudin, A. Dupuis, and N. Antigny, "Visual and visual-inertial slam: State of the art, classification, and experimental benchmarking," *Journal of Sensors*, vol. 2021, 2021.

[12] A. Geiger, J. Ziegler, and C. Stiller, "Stereoscan: Dense 3d reconstruction in real-time," in *2011 IEEE intelligent vehicles symposium (IV)*. Ieee, 2011, pp. 963–968.

[13] P. Bénet and A. Guinamard, "Robust and accurate deterministic visual odometry," in *Proceedings of the 33rd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2020)*, 2020, pp. 2260–2271.

[14] P. F. McLauchlan, "The variable state dimension filter applied to surface-based structure from motion," 1999.

[15] P. Bénet, F. Novella, M. Ponchart, P. Bosser, and B. Clement, "State-of-the-art of standalone accurate auv positioning-application to high resolution bathymetric surveys," in *OCEANS 2019-Marseille*. IEEE, 2019, pp. 1–10.

[16] I. Cvišić and I. Petrović, "Stereo odometry based on careful feature selection and tracking," in *2015 European Conference on Mobile Robots (ECMR)*. IEEE, 2015, pp. 1–6.

[17] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct ekf-based approach," in *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2015, pp. 298–304.

[18] 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.

[19] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 3565–3572.

[20] G. Sibley, L. Matthies, and G. Sukhatme, "Sliding window filter with application to planetary landing," *Journal of Field Robotics*, vol. 27, no. 5, pp. 587–608, 2010.

[21] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[22] L. Von Stumberg, V. Usenko, and D. Cremers, "Direct sparse visual-inertial odometry using dynamic marginalization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 2510–2517.

[23] 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, 2017.

[24] R. Mur-Artal and J. D. Tardós, "Visual-inertial monocular slam with map reuse," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 796–803, 2017.

[25] T. Qin, J. Pan, S. Cao, and S. Shen, "A general optimization-based framework for local odometry estimation with multiple sensors," *arXiv preprint arXiv:1901.03638*, 2019.

[26] V. Usenko, N. Demmel, D. Schubert, J. Stückler, and D. Cremers, "Visual-inertial mapping with non-linear factor recovery," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 422–429, 2019.

[27] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "Svo: Semidirect visual odometry for monocular and multicamera systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2016.

[28] D. Gálvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[29] M. Ferrera, A. Eudes, J. Moras, M. Sanfourche, and G. Le Besnerais, "Ov^{2}slam: A fully online and versatile visual slam for real-time applications," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1399–1406, 2021.

[30] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[31] X. Gao, R. Wang, N. Demmel, and D. Cremers, "Ldso: Direct sparse odometry with loop closure," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2198–2204.

[32] E. Garcia-Fidalgo and A. Ortiz, "ibow-lcd: An appearance-based loop-closure detection approach using incremental bags of binary words," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3051–3057, 2018.

[33] M. Walter, R. Eustice, and J. Leonard, "A provably consistent method for imposing sparsity in feature-based slam information filters," in *Robotics Research*. Springer, 2007, pp. 214–234.

[34] J. Jiang, X. Niu, R. Guo, and J. Liu, "A hybrid sliding window optimizer for tightly-coupled vision-aided inertial navigation system," *Sensors*, vol. 19, no. 15, p. 3418, 2019.

[35] R. Wang, M. Schworer, and D. Cremers, "Stereo dso: Large-scale direct sparse visual odometry with stereo cameras," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3903–3911.

[36] S. Thrun, W. Burgard, D. Fox *et al.*, "Probabilistic robotics, vol. 1," 2005.

[37] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stückler, and D. Cremers, "The tum vi benchmark for evaluating visual-inertial odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1680–1687.

[38] M. Grupp, "evo: Python package for the evaluation of odometry and slam." https://github.com/MichaelGrupp/evo, 2017.

[39] O. Seiskari, P. Rantalankila, J. Kannala, J. Ylilammi, E. Rahtu, and A. Solin, "Hybvio: Pushing the limits of real-time visual-inertial odometry," *arXiv preprint arXiv:2106.11857*, 2021.

[40] I. Cvišić, I. Marković, and I. Petrović, "Recalibrating the kitti dataset camera setup for improved odometry accuracy," *arXiv preprint arXiv:2109.03462*, 2021.