Multi-Camera Visual-Inertial Navigation with Online Intrinsic and Extrinsic Calibration

Kevin Eckenhoff[†], Patrick Geneva[*], Jesse Bloecker[†], and Guoquan Huang[†]

Abstract— **This paper presents a general multi-camera visual-inertial navigation system (mc-VINS) with *online* intrinsic and extrinsic calibration, which is able to utilize all the information from an *arbitrary* number of *asynchronous* cameras. In particular, within the standard multi-state constraint Kalman Filter (MSCKF) framework, we only clone the IMU poses related to a single "base camera" (rather than all cameras) in the state vector, while the IMU poses corresponding to all other camera images are represented via an interpolation of the poses bounding the measuring time. By doing so, we can fuse all observations from all cameras with inertial measurements while allowing for efficient, tightly-coupled state estimation through parallelization and asynchrony. Moreover, we perform online sensor calibration of each camera's intrinsics as well as the spatial (transformation) and temporal (time offset) extrinsic parameters between all involved sensors (cameras and IMU), thus enabling high-fidelity localization. We validate the proposed mc-VINS algorithm in various real-world experiments with different sensor configurations, showing the ability to offer real-time high-precision localization and calibration results.**

I. Introduction

Over the last decade, visual-inertial navigation systems (VINS) have witnessed a great increase in popularity largely because of the increasing ubiquity of MEMS inertial measurement units (IMUs) and cameras (e.g., see [1, 2]). Due to being low-cost and light-weight while still providing rich sensory information, such sensor platforms have become the go-to sensor deployment in various fields such as unmanned aerial vehicles (UAVs) [3, 4] and mobile devices [5].

Within the VINS literature, most efforts have focused on the minimal sensing case of a *single* camera and IMU [1, 6–9]. This configuration fully constrains the estimation problem as the camera provides information to limit the drift inherent to integrating noisy inertial measurements, while the IMU (accelerometer) provides scale information and improved robustness to dynamic motion. Among the current monocular VINS algorithms, one of the most popular lightweight VINS estimators is the multi-state constraint Kalman filter (MSCKF) [6], which has been extended and improved in different directions [7–12]. The key idea of the MSCKF and its variants is to perform propagation using the inertial measurements while efficiently processing visual measurements through a novel update step that does not explicitly require estimating the corresponding 3D features, thereby bounding the computational burden. While 3D motion tracking with minimal sensing capability is of interest, in practice, it is highly desirable to optimally and efficiently fuse *all* information from *multiple* cameras to improve estimation robustness and accuracy [13].

In this paper, we design a general multi-camera VINS (mc-VINS) algorithm that is capable of tightly fusing the visual information from an *arbitrary* number of *non-overlapping*, *asynchronous* cameras and IMU measurements within the MSCKF framework, while limiting the increased computational burden on the estimator. In comparison to the state-of-the-art VINS [1, 8, 9, 14], we not only online calibrate *all* spatial and temporal calibration parameters between the used sensors but also jointly estimate the *intrinsics* for each camera, allowing for online refinement of these parameters. In particular, the main contributions of this work are:

- By leveraging the computationally-efficient MSCKF framework, we develop the tightly-coupled multi-camera VINS (mc-VINS) with online intrinsic and extrinsic calibration. The proposed mc-VINS is able to utilize all information from any number of cameras without constraints on sensor configurations since both spatial/temporal calibration parameters and intrinsics of each camera are simultaneously estimated online.
- Instead of maintaining stochastic clones for each camera, we only perform cloning of the IMU poses at imaging times of a freely chosen "base" camera, and use interpolation on the $SO(3) \times \mathbb{R}^3$ manifold to represent the pose at an arbitrary intermediate time for all other cameras. By representing this interpolation as a function of the unknown time offset between cameras, we can perform both spatial and temporal calibration between all sensors (including the IMU), and additionally refine the *intrinsic* parameters for each camera to allow for high-fidelity estimation.
- We validate the proposed mc-VINS on real multi-camera visual-inertial sensor platforms using different camera configurations and in various environments.

II. Related Work

While monocular-VINS has been widely studied (e.g., see [1, 7–10, 12, 15] and references therein), one straightforward extended configuration over the monocular setting is to use a *stereo* camera, wherein the *two* cameras are mounted such that they observe the same spatial volume from offset camera centers at the same image time. Stereo vision

This work was partially supported by the University of Delaware College of Engineering, the NSF (IIS-1566129), the DTRA (HDTRA1-16-1-0039), and Google Daydream.

[†]The authors are with the Department of Mechanical Engineering, University of Delaware, Newark, DE 19716, USA. Email: {keck,jesseb,ghuang}@udel.edu

[*]The author is with the Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716, USA. Email: pgeneva@udel.edu

enables 3D triangulation of features seen in the overlapping view without requiring motion of the sensor platform, thus allowing for the direct recovery of scale if the spatial transforms between cameras are known. Motivated by this increase in robustness, Sun et al. [3] developed the MSCKF-based stereo-VINS with the particular application to high-speed aerial vehicles. Paul et al. [13] extended the inverse square-root version of the MSCKF (namely SR-ISWF) [5] to provide real-time VINS on mobile devices while allowing for a configuration of both stereo and binocular (non-overlapping) cameras, and showed that the inclusion of more visual information improves the estimation accuracy.

While stereo cameras provide robustness due to their ability to perform feature triangulation and scale recovery even without the IMU, they remain vulnerable to dynamic environmental motion and textureless regions in its given viewing direction. More importantly, the requirement of an overlapping field of view and synchronous camera triggering may not easily extend to an *arbitrary* number of *plug-and-play* cameras – which is a highly-desirable characteristic and could greatly promote the widespread deployment of VINS in practice. Additionally, due to the enforcement of cross-image matching (for example matching features from the left to right stereo image) the process of visual tracking is coupled and cannot be directly parallelized. For these reasons, we propose a general multi-camera VINS in this work, which can tightly fuse the visual information from an *arbitrary* number of *non-overlapping*, *asynchronous* and heterogeneous cameras and the IMU measurements, so that the proposed approach is robust to environmental conditions and single-camera failure. We shall stress that in the proposed mc-VINS, we do *not* perform any cross-image matching, since we have non-overlapping images and instead allow each camera feed to be processed independently and in parallel.

Houben et al. [16] extended the ORB-SLAM [17] to a system of multiple cameras with varying viewing directions and an IMU for UAVs within a graph-SLAM framework but assumed known sensor calibration and simultaneous triggering of all involved cameras. Recently, Paul and Roumeliotis [18] addressed the problem of increased computational burden in stereo-VINS and proposed an alternating stereo-VINS algorithm. In their system, the *two* cameras in a stereo pair were triggered in an alternating manner, preventing the need to process both images at the same time while still taking advantage of the offset camera centers provided by a stereo configuration. In addition, they further reduced computation by explicitly estimating the historical IMU poses corresponding to only *one* of the camera's imaging times, while using pose interpolation to represent the state at intermediate times corresponding to the other camera. While in this work we use a similar interpolation scheme to reduce computation, we simultaneously perform time offset and spatial calibration between $n \geq 2$ cameras.

An integral part of any multi-sensor fusion system is the spatial (relative transformation), temporal (time offset), and intrinsic (e.g., focal length, camera center, and distortion parameters) calibration parameters for each sensor, as errors in the values of these parameters can greatly degrade localization performance – if not catastrophically. Calibration can be broadly divided into two main categories. Offline methods perform a computationally expensive solution process in exchange for providing highly accurate calibration estimates. In particular, Furgale, Rehder, and Siegwart [19] have developed a multi-sensor calibration system that can perform spatial, temporal, and intrinsic calibration of an arbitrary number of cameras along with an IMU. However, performing offline calibration could be a tedious process that limits deployment time and requires the calibration to be repeated if the sensor configuration changes. In addition, treating the calibration parameters provided by these methods as "known" (zero uncertainty) may lead to unmodelled errors, thereby introducing estimation inconsistency [20]. By contrast, online methods treat the calibration parameters as unknown random variables, simultaneously estimating them along with the navigation states. While many VINS algorithms perform online calibration of the spatial extrinsic transform between the camera and IMU, relatively few also estimate the time offset between them [3, 21]. Systems that *do* perform online temporal calibration [22–24], however, are typically limited to a *single* IMU-camera pair. One of the most notably complete systems in this category is by Li et. al. [20], who performed online calibration of both the extrinsic parameters between the IMU and camera as well as the *intrinsics* of both sensors.

III. The Proposed mc-VINS Algorithm

In this section, within the standard MSCKF framework [6], we present in detail the proposed multi-camera (mc)-VINS with online instrinsic and extrinsic sensor calibration.

A. State Vector

As standard, the inertial navigation state is given by:

$$\mathbf{x}_I = \begin{bmatrix} {}^I_G\bar{q}^\top & \mathbf{b}_\omega^\top & {}^G\mathbf{v}_I^\top & \mathbf{b}_a^\top & {}^G\mathbf{p}_I^\top \end{bmatrix}^\top \qquad (1)$$

where ${}^I_G\bar{q}$ is the JPL unit quaternion [25] associated with the rotation matrix, ${}^I_G\mathbf{R}$, which rotates vectors from the global frame of reference $\{G\}$ into the local frame $\{I\}$ of the IMU, $\mathbf{b}_\omega$ and $\mathbf{b}_a$ are the gyroscope and accelerometer biases which corrupt the IMU measurements, ${}^G\mathbf{p}_I$ is the position of the IMU expressed in the global frame, and ${}^G\mathbf{v}_I$ is the corresponding velocity. Associated with this navigation state we also define the corresponding *error* state as:

$$\delta\mathbf{x}_I = \begin{bmatrix} {}^I\delta\boldsymbol{\theta}_G^\top & \delta\mathbf{b}_\omega^\top & {}^G\delta\mathbf{v}_I^\top & \delta\mathbf{b}_a^\top & {}^G\delta\mathbf{p}_I^\top \end{bmatrix}^\top \qquad (2)$$

The true value of the state, $\mathbf{x}_I$, estimated value $\hat{\mathbf{x}}_I$, and error state, $\delta\mathbf{x}_I$, are related by the generalized update operation [26]:

$$\mathbf{x}_I = \hat{\mathbf{x}}_I \boxplus \delta\mathbf{x}_I \qquad (3)$$

where for vector quantities, $\mathbf{v}$, this operation is simply addition, i.e., $\mathbf{v} = \hat{\mathbf{v}} + \delta\mathbf{v}$, while for quaternions we have:

$$\bar{q} \approx \begin{bmatrix} \frac{1}{2}\delta\boldsymbol{\theta} \\ 1 \end{bmatrix} \otimes \hat{\bar{q}} \qquad (4)$$

where $\otimes$ denotes quaternion multiplication. In addition, we can also write the inverse operation $\delta\mathbf{x} = \mathbf{x} \boxminus \hat{\mathbf{x}}$.

Secondly, adhering to the standard MSCKF, we maintain a window of IMU clones at the past M imaging times t_j. Since we have an asynchronous multi-camera system, we maintain only those imaging times associated with one of the cameras which we denote as the "base" camera, as this greatly decreases the number of variables in our state, and thus the computational burden:

$$\mathbf{x}_{cl} = \begin{bmatrix} {}^{I(t_k)}_{G}\bar{q}^\top & {}^{G}\mathbf{p}_{I(t_k)}^\top & \cdots & {}^{I(t_{k-M+1})}_{G}\bar{q}^\top & {}^{G}\mathbf{p}_{I(t_{k-M+1})}^\top \end{bmatrix}^\top \tag{5}$$

Lastly, with the multi-camera setting under consideration, we also estimate *extrinsics* including both the spatial (relative pose) and temporal (time offset) calibration parameters, as well as the *intrinsics* of each camera. Specifically, each camera state contains the following parameters:

$$\mathbf{x}_{ci} = \begin{bmatrix} {}^{C_i}_{I}\bar{q}^\top & {}^{C_i}\mathbf{p}_I^\top & {}^{i}t_b & \boldsymbol{\zeta}_i^\top \end{bmatrix}^\top \tag{6}$$

$$\boldsymbol{\zeta}_i = \begin{bmatrix} f_{xi} & f_{yi} & p_{xi} & p_{yi} & \mathbf{d}_i^\top \end{bmatrix}^\top \tag{7}$$

where ${}^{i}t_b$ is the time offset between camera i and the base camera, f_{xi}, f_{yi} represent the focal lengths, p_{xi}, p_{yi} denote the location of the principal point, and $\mathbf{d}_i$ refers to the vector of distortion parameters whose length/definition depends on the camera model being used (see [27]). For the base camera, however, we store its temporal misalignment with respect to the IMU, ${}^{b}t_I$ as the relative time offset to itself is zero.

B. IMU Propagation

As the sensor platform navigates in the space, the IMU measures both the angular velocity, $\boldsymbol{\omega}_m$, and local linear acceleration, $\mathbf{a}_m$, which are utilized for propagation:

$$\boldsymbol{\omega}_m = \boldsymbol{\omega} + \mathbf{b}_\omega + \mathbf{n}_\omega \tag{8}$$

$$\mathbf{a}_m = {}^{I}\mathbf{a} + {}^{I}_{G}\mathbf{R}\,{}^{G}\mathbf{g} + \mathbf{b}_a + \mathbf{n}_a \tag{9}$$

where $\boldsymbol{\omega}$ and ${}^{I}\mathbf{a}$ represent the true local angular velocity and local linear acceleration of the IMU, while $\mathbf{n}_\omega$ and $\mathbf{n}_a$ are continuous-time Gaussian white noises, and ${}^{G}\mathbf{g} \approx [0\ 0\ 9.81]^\top$ is the known global gravity.

It is important to note that due to hardware latency of on-board processing, the time reported by the base camera will *differ* from the same time expressed in the IMU's clock. We consider a time ${}^{b}t$ as expressed in the base camera's clock, which is related to the same instant represented in the IMU clock, ${}^{I}t$, by a time offset ${}^{b}t_I$, i.e.,

$${}^{I}t = {}^{b}t + {}^{b}t_I \tag{10}$$

As this time offset is unknown, we include it as a parameter in our state vector to be estimated.

With the estimate of this time offset ${}^{b}\hat{t}_I$, whenever we receive the $(k+1)$-th image with reported time ${}^{b}t_{k+1}$, we perform propagation of our IMU up to the *estimated* time of the image as expressed in the IMU clock [see (10)]: ${}^{I}\hat{t}_{k+1} = {}^{b}t_{k+1} + {}^{b}\hat{t}_I$. Specifically, we propagate the IMU from its current time ${}^{I}\hat{t}_k$ (actually the estimate of the IMU time for k-th image at the time of the previous propagation, *before* update) up to this new time by processing all IMU measurements $\mathcal{I}$ collected over the time interval $[{}^{I}\hat{t}_k, {}^{I}\hat{t}_{k+1}]$, based on the conventional IMU dynamics whose integration yields the prediction function $\mathbf{f}(\cdot)$ [28]:

$$\hat{\mathbf{x}}_{I({}^{I}\hat{t}_{k+1})|k} = \mathbb{E}\left[\mathbf{f}\left(\mathbf{x}_{I({}^{I}\hat{t}_k)}, \mathcal{I}, \mathbf{n}_I\right)\right] = \mathbf{f}\left(\hat{\mathbf{x}}_{I({}^{I}\hat{t}_k)|k}, \mathcal{I}, \mathbf{0}\right)$$

where the $\hat{\mathbf{x}}_{|k}$ notation denotes the estimate for variable $\mathbf{x}$ given all measurements up to the k-th image, while $\mathbf{n}_I$ is the stacked IMU noises. Based on this prediction function, we compute the state-transition matrix across the interval $\boldsymbol{\Phi}_k$ as well as the corresponding noise covariance $\mathbf{Q}_k$ as in [7, 8]. Once computed, the covariance after propagation $\mathbf{P}_{k+1|k}$ can be calculated as:

$$\mathbf{P}_{k+1|k} = \begin{bmatrix} \boldsymbol{\Phi}_k \mathbf{P}_{II,k|k} \boldsymbol{\Phi}_k^\top + \mathbf{Q}_k & \boldsymbol{\Phi}_k \mathbf{P}_{IS,k|k} \\ \mathbf{P}_{SI,k|k} \boldsymbol{\Phi}_k^\top & \mathbf{P}_{SS,k|k} \end{bmatrix} \tag{11}$$

where the subscripts I and S denote the partition of the covariance respectively with respect to the IMU navigation states (1) and the rest of the time-invariant states including cloned states (5) and calibration parameters (6)-(7).

C. State Augmentation

After propagating to time step $k+1$, we only have a state estimate of the IMU at the *estimated* time ${}^{I}\hat{t}_{k+1}$, while we actually need to express all camera measurements as a function of the IMU pose at the *true* time ${}^{I}t_{k+1}$. To accomplish this, we utilize stochastic cloning [29]. Specifically, consider our current state $\mathbf{x}$ with covariance $\mathbf{P}$, which we wish to augment with another state that can be written as a function $\mathbf{g}(\mathbf{x})$. The augmented state mean and covariance of the corresponding Gaussian can be computed as: $\mathcal{N}\left(\begin{bmatrix} \hat{\mathbf{x}} \\ \mathbf{g}(\hat{\mathbf{x}}) \end{bmatrix}, \ \boldsymbol{\Psi}\mathbf{P}\boldsymbol{\Psi}^\top\right)$, where $\boldsymbol{\Psi} = \begin{bmatrix} \mathbf{I} \\ \frac{\partial \mathbf{g}(\hat{\mathbf{x}} \boxplus \delta\mathbf{x}) \boxminus \mathbf{g}(\hat{\mathbf{x}})}{\partial \delta\mathbf{x}} \end{bmatrix}$. Using this methodology, the following linearized stochastic cloning is performed to create an estimate of the IMU at this true time in a manner analogous to [22]:

$${}^{G}\mathbf{p}_{I({}^{I}t_{k+1})} \approx {}^{G}\mathbf{p}_{I({}^{I}\hat{t}_{k+1})} + {}^{G}\mathbf{v}_{I({}^{I}\hat{t}_{k+1})} \delta^{b}t_I \tag{12}$$

$${}^{I({}^{I}t_{k+1})}_{G}\mathbf{R} \approx \text{Exp}\left(-\boldsymbol{\omega}\delta^{b}t_I\right) {}^{I({}^{I}\hat{t}_{k+1})}_{G}\mathbf{R} \tag{13}$$

Where $\text{Exp}(\cdot)$ is the $SO(3)$ matrix exponential which maps a vector in $\mathbb{R}^3$ to a rotation matrix [30]. That is, because we can write the pose at the true time as a function of the pose at the estimated time, as well as the time offset error (both of which are contained in our state), we can augment our state to contain this new pose. The Jacobians for this cloning are given by:

$$\frac{\partial^{G}\delta\mathbf{p}_{I({}^{I}t_{k+1})}}{\partial^{G}\delta\mathbf{p}_{I({}^{I}\hat{t}_{k+1})}} = \mathbf{I}_3, \ \frac{\partial^{G}\delta\mathbf{p}_{I({}^{I}t_{k+1})}}{\partial\delta^{b}t_I} = {}^{G}\hat{\mathbf{v}}_{I({}^{I}\hat{t}_{k+1})}$$

$$\frac{\partial^{I({}^{I}t_{k+1})}\delta\boldsymbol{\theta}_G}{\partial^{I({}^{I}\hat{t}_{k+1})}\delta\boldsymbol{\theta}_G} = \mathbf{I}_3, \ \frac{\partial^{I({}^{I}t_{k+1})}\delta\boldsymbol{\theta}_G}{\partial\delta^{b}t_I} = \boldsymbol{\omega}_m({}^{I}\hat{t}_{k+1}) - \hat{\mathbf{b}}_w({}^{I}\hat{t}_{k+1})$$

With these Jacobians, we obtain the covariance of the augmented state as explained above [29].

D. Multi-Camera Measurements Update

Consider a 3D feature, ${}^G\mathbf{p}_f$, which is captured by the i-th camera at imaging time ${}^bt_k$ with respect to the *base* camera. In this work we track features extracted uniformly using FAST [31] and tracked independently for each camera's image stream using KLT [32], with outliers rejected via 8-point RANSAC. The perspective projection measurement function for this feature is given by:

$$\mathbf{z}_k = \mathbf{w}_i\left(\boldsymbol{\Pi}\left({}^{C_i({}^bt_k)}\mathbf{p}_f\right),\ \boldsymbol{\zeta}_i\right) + \mathbf{n}_k,\ \boldsymbol{\Pi}\left(\begin{bmatrix} x \\ y \\ z \end{bmatrix}\right) = \begin{bmatrix} \frac{x}{z} \\ \frac{y}{z} \end{bmatrix}$$

where $\mathbf{w}_i(\cdot)$ is the function mapping the normalized image coordinates onto the image plane based on the camera intrinsics $\boldsymbol{\zeta}_i$ and the camera model used (e.g., radial-tangential or fisheye [33]), and ${}^{C_i({}^bt_k)}\mathbf{p}_f = [x\ y\ z]^\top$ is the position of the feature expressed in camera i's frame at true time ${}^bt_k$:

$${}^{C_i({}^bt_k)}\mathbf{p}_f = {}^{C_i}_I\mathbf{R}\,{}^{I({}^bt_k)}_G\mathbf{R}\left({}^G\mathbf{p}_f - {}^G\mathbf{p}_{I({}^bt_k)}\right) + {}^{C_i}\mathbf{p}_I \quad (14)$$

Letting $\mathbf{a} = \boldsymbol{\Pi}\left({}^{C_i({}^bt_k)}\mathbf{p}_f\right)$ denote the normalized image coordinates, we note that when computing Jacobians for this function, dependence on the intrinsics comes from the image plane mapping function $\mathbf{w}_i$:

$$\frac{\partial \mathbf{z}_k}{\partial \delta \boldsymbol{\zeta}_i} = \frac{\partial \mathbf{w}_i\left(\mathbf{a},\ \boldsymbol{\zeta}_i\right)}{\partial \delta \boldsymbol{\zeta}_i} \quad (15)$$

while for other variables $\mathbf{v}$, the dependency is from ${}^{C_i({}^bt_k)}\mathbf{p}_f$:

$$\frac{\partial \mathbf{z}_k}{\partial \delta \mathbf{v}} = \frac{\partial \mathbf{w}_i\left(\mathbf{a},\ \boldsymbol{\zeta}_i\right)}{\partial \mathbf{a}} \frac{\partial \mathbf{a}}{\partial {}^{C_i({}^bt_k)}\delta\mathbf{p}_f} \frac{\partial {}^{C_i({}^bt_k)}\delta\mathbf{p}_f}{\partial \delta \mathbf{v}} \quad (16)$$

In summary, rather than using the undistorted, normalized pixel coordinates as measurements (as is typically done), we utilize the *raw* image coordinates, allowing us to calibrate the intrinsics of each camera along with the rest of the state.

As features are tracked over the sliding window of camera clones, if the feature reaches its maximum track length or is lost, we perform the MSCKF update with its measurements. Specifically, let $\mathbf{r}$ denote the stacked vector of residuals associated with this feature, $\mathbf{r}_k = \mathbf{z}_k - \mathbf{w}_i(\hat{\mathbf{a}}, \hat{\boldsymbol{\zeta}}_i)$. The linearized system for this feature can be written as:

$$\mathbf{r} = \mathbf{H}_x\delta\mathbf{x} + \mathbf{H}_f\,{}^G\delta\mathbf{p}_f + \mathbf{n} \quad (17)$$

where $\mathbf{H}_x$ and $\mathbf{H}_f$ refer to the stacked Jacobians of the residuals with respect to the state and feature variables respectively, while $\mathbf{n}$ is the stacked noise vector with covariance $\mathbf{R}$. To bound our problem size by not explicitly storing the feature position into our state vector, we perform linear marginalization of the feature position by the projecting this system onto the nullspace of the feature Jacobian, which is spanned by the columns of matrix $\mathbf{N}$.

$$\mathbf{N}^\top\mathbf{r} = \mathbf{N}^\top\mathbf{H}_x\delta\mathbf{x} + \mathbf{N}^\top\mathbf{H}_f\,{}^G\delta\mathbf{p}_f + \mathbf{N}^\top\mathbf{n} \quad (18)$$

$$\Rightarrow \mathbf{r}' = \mathbf{H}'_x\delta\mathbf{x} + \mathbf{n}' \quad (19)$$

where the transformed noise, $\mathbf{n}'$, has covariance $\mathbf{N}^\top\mathbf{R}\mathbf{N}$. As this transformed measurement is only a function of variables in our state, it can be utilized in the EKF update.

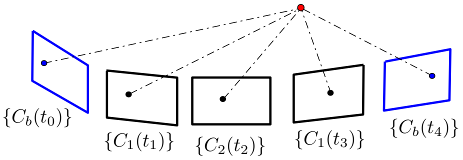

Fig. 1: Illustration of how asynchronous multi-camera measurements are collected. We have cloned at the base camera imaging times: $\{C_b(t_0)\}$, $\{C_b(t_4)\}$ (blue). A series of measurements between these times from other non-base cameras C_1 and C_2 are received, requiring us to interpolate these poses in terms of the base camera so that they can be utilized in the MSCKF update.

1) Asynchronous multi-camera measurements: Note that the cloned state $\mathbf{x}_{cl}$ (5) only contains the IMU poses at the true imaging times of the *base* camera, while the camera measurements (14) require that for all non-base cameras we can express the pose at *their* image times, which may not align with those of the base camera (see Figure 1). Clearly, without solving this issue, the inclusion of all other measurements that are not collected synchronously with the base camera cannot be written as functions of the state and used in the update. Therefore, we employ a $SO(3) \times \mathbb{R}^3$ linear interpolation method between these poses to allow for the incorporation of measurements at arbitrary times. We note that while higher-order interpolation schemes may be utilized for improved accuracy, in this work we leverage a linear scheme for its computational efficiency.

In particular, assuming the measured time reported by the i-th camera as ${}^it_k$, to express this time in the base camera's clock, we must compensate for the time offset, i.e., ${}^bt_k = {}^it_k + {}^it_b$. Let ${}^bt_1$ and ${}^bt_2$ denote the times for the bounding base camera/IMU clones between which ${}^b\hat{t}_k = {}^it_k + {}^i\hat{t}_b$ falls. We linearly interpolate the cloned poses at ${}^bt_1$ and ${}^bt_2$ to find the pose at at the measurement time:

$${}^{I({}^bt_k)}_G\mathbf{R} = \mathrm{Exp}\left(\lambda_k \mathrm{Log}\left({}^{I({}^bt_2)}_G\mathbf{R}\,{}^G_{I({}^bt_1)}\mathbf{R}\right)\right){}^{I({}^bt_1)}_G\mathbf{R} \quad (20)$$

$${}^G\mathbf{p}_{I({}^bt_k)} = (1-\lambda_k)\,{}^G\mathbf{p}_{I({}^bt_1)} + \lambda_k\,{}^G\mathbf{p}_{I({}^bt_2)} \quad (21)$$

$$\lambda_k = \frac{{}^it_k + {}^it_b - {}^bt_1}{{}^bt_2 - {}^bt_1} \quad (22)$$

where $\mathrm{Log}(\cdot)$ is the inverse operation of $\mathrm{Exp}(\cdot)$ which maps a rotation matrix to a vector in $\mathbb{R}^3$. These equations essentially interpolate both the orientation and position under the approximation of constant linear and angular velocity over the interval. As images of the base camera typically arrive at high rate (around 20 Hz in most applications) as compared to the physical camera motion, we argue that this serves as a good approximation. Due to the fact that marginalization is only ever performed on the oldest clone, any required camera pose will typically be within 25 milliseconds of a clone in the sliding window. Note that a similar interpolation scheme was used in our prior work on asynchonous multi-sensor fusion [34] but without either the intrinsic and extrinsic (spatial and temporal) calibration of multiple non-base cameras as focused on in this work.

Let $\mathbf{x}_I({}^bt_1)$, $\mathbf{x}_I({}^bt_2)$ and $\mathbf{x}_I({}^bt_k)$ denote the IMU clones at the neighboring times, and the interpolated value, respectively. By substituting the above expressions into the measurement function, we have the following Jacobians with respect to the bounding IMU clones and the time offset ${}^it_b$:

$$\frac{\partial^{C_i({}^bt_k)}\delta\mathbf{p}_f}{\partial\delta\mathbf{x}_I({}^bt_1)} = \frac{{}^{C_i({}^bt_k)}\delta\mathbf{p}_f}{\partial\delta\mathbf{x}_I({}^bt_k)}\frac{\partial\delta\mathbf{x}_I({}^bt_k)}{\partial\delta\mathbf{x}_I({}^bt_1)} \tag{23}$$

$$\frac{\partial^{C_i({}^bt_k)}\delta\mathbf{p}_f}{\partial\delta\mathbf{x}_I({}^bt_2)} = \frac{{}^{C_i({}^bt_k)}\delta\mathbf{p}_f}{\partial\delta\mathbf{x}_I({}^bt_k)}\frac{\partial\delta\mathbf{x}_I({}^bt_k)}{\partial\delta\mathbf{x}_I({}^bt_2)} \tag{24}$$

$$\frac{\partial^{C_i({}^bt_k)}\delta\mathbf{p}_f}{\partial\delta^it_b} = \frac{{}^{C_i({}^bt_k)}\delta\mathbf{p}_f}{\partial\delta\mathbf{x}_I({}^bt_k)}\frac{\partial\delta\mathbf{x}_I({}^bt_k)}{\partial\delta^it_b} \tag{25}$$

To compute these Jacobians, we use the following approximations for a small angle $\boldsymbol{\psi}$ [35]:

$$\begin{aligned}\mathrm{Exp}\left(\boldsymbol{\theta}+\boldsymbol{\psi}\right) &\approx \mathrm{Exp}\left(\mathbf{J}_l\left(\boldsymbol{\theta}\right)\boldsymbol{\psi}\right)\mathrm{Exp}\left(\boldsymbol{\theta}\right) & (26)\\ &\approx \mathrm{Exp}\left(\boldsymbol{\theta}\right)\mathrm{Exp}\left(\mathbf{J}_r\left(\boldsymbol{\theta}\right)\boldsymbol{\psi}\right) & (27)\end{aligned}$$

where $\mathbf{J}_l(\cdot)$ and $\mathbf{J}_r(\cdot)$ are the left and right Jacobians of $SO(3)$ [30], respectively. By defining for convenience ${}^2_1\hat{\boldsymbol{\theta}} = \mathrm{Log}\left({}^{I({}^bt_2)}_{I({}^bt_1)}\hat{\mathbf{R}}\right)$, we have the following Jacobians with detailed derivations found in our technical report [36]:

$$\begin{aligned}
\frac{\partial^{I({}^bt_k)}\delta\boldsymbol{\theta}_G}{\partial^{I({}^bt_1)}\delta\boldsymbol{\theta}_G} &= -\left(\hat{\lambda}_k\mathbf{J}_l\left(\hat{\lambda}_k{}^2_1\hat{\boldsymbol{\theta}}\right)\mathbf{J}_r^{-1}\left({}^2_1\hat{\boldsymbol{\theta}}\right) - \mathrm{Exp}\left(\hat{\lambda}_k{}^2_1\hat{\boldsymbol{\theta}}\right)\right)\\
\frac{\partial^{I({}^bt_k)}\delta\boldsymbol{\theta}_G}{\partial^{I({}^bt_2)}\delta\boldsymbol{\theta}_G} &= \hat{\lambda}_k\mathbf{J}_l\left(\hat{\lambda}_k{}^2_1\hat{\boldsymbol{\theta}}\right)\mathbf{J}_l^{-1}\left({}^2_1\hat{\boldsymbol{\theta}}\right)\\
\frac{\partial^{I({}^bt_k)}\delta\boldsymbol{\theta}_G}{\partial\delta^it_b} &= -\frac{1}{{}^bt_2-{}^bt_1}{}^2_1\hat{\boldsymbol{\theta}}\\
\frac{\partial^G\delta\mathbf{p}_{I({}^bt_k)}}{\partial^G\delta\mathbf{p}_{I({}^bt_1)}} &= (1-\hat{\lambda}_k)\mathbf{I}, \quad \frac{\partial^G\delta\mathbf{p}_{I({}^bt_k)}}{\partial^G\delta\mathbf{p}_{I({}^bt_2)}} = \hat{\lambda}_k\mathbf{I}\\
\frac{\partial^G\delta\mathbf{p}_{I({}^bt_k)}}{\partial\delta^it_b} &= \frac{1}{{}^bt_2-{}^bt_1}\left({}^G\hat{\mathbf{p}}_{I({}^bt_2)} - {}^G\hat{\mathbf{p}}_{I({}^bt_1)}\right) \qquad (28)
\end{aligned}$$

As a result, by incorporating this representation, we can estimate both the spatial and temporal calibration parameters for all cameras, while *only* storing those poses related to the base camera's imaging times.

E. Note on Complexity

A key advantage of the proposed approach is the ability to parallelize the visual tracking front-end. Since all cameras can be processed independently, we argue that one can perform "camera-edge" visual tracking allowing for the horizontal scaling of the visual front-ends. As seen in Figure 2, the visual front-ends can be treated as independent and images from all cameras can be processed in parallel. For example, in practice we could let each camera have a local micro-computer or hardware embedded processor ("camera-edge" processing) that performs feature tracking that upon completion can be sent to the centralized estimator for asynchronous fusion. The only introduced increase in computation is the possibly larger magnitude of features used during update, which can be managed by careful selection of a subset of features to be processed.

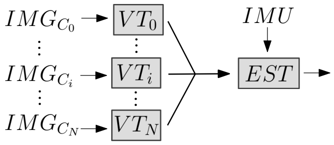

Fig. 2: Illustration showing how the proposed system horizontally scales as more images are added. Simply scaling of the visual tracker (VT) allows for the parallelization of feature tracking, which feeds these tracks to the estimator (EST) for processing.

IV. Experiential Results

A. Duo-Camera Configuration

For our first experiment, a custom 3D printed mount was used to attach two Omnivision OV7251 fisheye global shutter cameras to the Snapdragon Flight.[1] The cameras were mounted in a binocular configuration such that one faced forward and the other faced downward as shown in Figure 4. The overall size of the quadrotor was 10in × 11in, with 5in rotors. The Snapdragon Flight was mounted underneath. The cameras were triggered at 30Hz and the onboard InvenSense MPU-9250 IMU ran at 500Hz. The dataset was recorded at the University of Delaware's Spencer Lab, in which the quadrotor traveled from the first floor, traversed the staircase to the third floor, and returned to the starting location to allow for computation of the start-end error. The approximately 143 meter trajectory traveled can be seen in Figure 3 (top-left). The extrinsic calibration values provided by the offline calibration toolbox Kalibr [19], which are expected to be close to the actual values, were taken as a proxy for the ground-truth. To illustrate calibration, we manually perturbed these ground-truth values, which were then used as the initial estimates in the filter. The intrinsics values provided from Kalibr were not perturbed in this experiment, but were refined online. We bounded the maximum number of features extracted from each image at 250, and utilized an MSCKF window size of 15. To account for the randomness in RANSAC-based vision, the averaged results from ten runs are reported.

As shown in Figure 3 (center), the calibration parameters quickly converge from poor initial guesses to the results reported by Kalibr. This validates the proposed method of online calibration. As no ground-truth position estimates were available, and as our trajectory returned to the starting location, the difference in the reported start and end locations serves as our error metric. The start-end error for this trajectory was approximately 0.45 meters, amounting to 0.33% of the total path. This shows that even with incorrect initial calibration, our system can recover and, by utilizing the information from multiple image streams, can provide accurate trajectory estimates. By contrast, the same system using only the base camera provided an error of 0.65 m (0.48%), of the path, thereby validating the improvement of the multi-camera system. To highlight the effect of intrinsics refinement, we also ran the duo-camera estimator *without*

[1] https://developer.qualcomm.com/hardware/qualcomm-flight

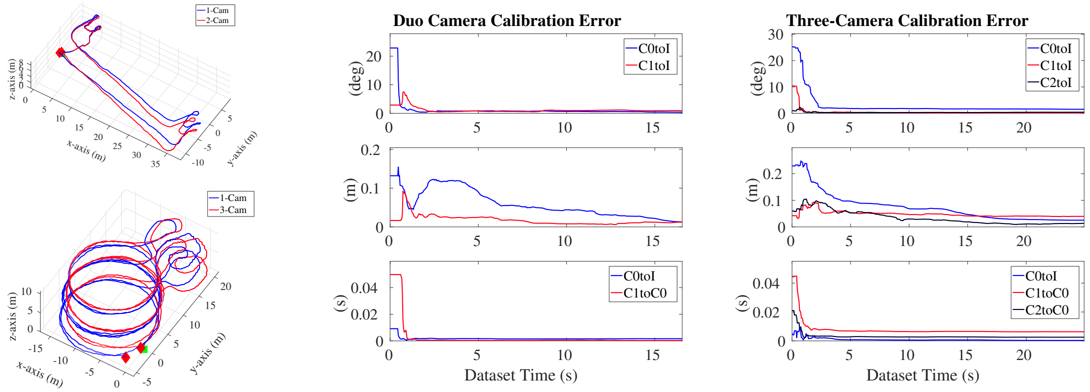

Fig. 3: The trajectories of the proposed mc-VINS and minimal sensing cases are shown for both the 140 meter long duo-camera (top-left) and 440 meter long three-camera (bottom-left) datasets. The calibration error for the duo-camera configuration (center) and the three-camera configuration (right) show only the first few seconds of the total dataset and also plot the camera to IMU time offset in the bottom most time offset error plots.

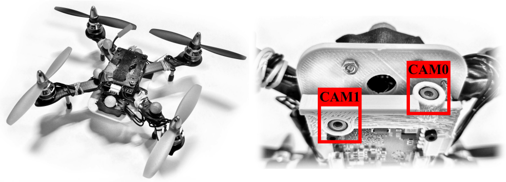

Fig. 4: Overview picture of the MAV (top). The camera configuration with the frontwards camera (CAM0) and downwards facing camera (CAM1) are shown in the bottom.

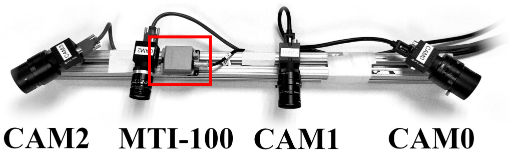

Fig. 5: Overview picture of the three-camera configuration. Note that in this experiment only one of the forward facing cameras was used (CAM1), along the the two side-facing fisheye cameras (CAM0/2).

online intrinsic refinement, and found the error to be 1.37 m (1.00%), a clear decrease in accuracy. Lastly, during processing, the system remained real-time, as the time to perform single image feature tracking (which can be run in parallel for each camera) plus MSCKF update utilizing the information from all cameras took on average 0.023 seconds (44 Hz) compared to the 30 Hz rate of each camera.

B. Three-Camera Configuration

For our second experiment, a custom rig with multiple Pointgrey Blackfly cameras and an MTI-100 IMU was designed and is shown in Figure 5. One Blackfly faced forward, while two others which were given fisheye lenses and tilted in opposite directions. Each camera operated asynchronously at 20 Hz. An approximately 440 meter closed-loop trajectory was performed across multiple floors in Gore Hall at the University of Delaware under poor lighting conditions, with the resulting trajectory shown in Figure 3 (bottom-left). As in the previous experiment, we report the averaged results from ten runs. As we have increased the number of cameras as compared to the previous configuration, we utilized a smaller MSCKF window size of 12 to ensure real-time performance (in this experiment, image tracking plus MSCKF update took on average 0.03 seconds). As shown in Figure 3 (right), our system was able to perform accurate calibration of the involved systems, despite having extremely poor initial guesses (over 20 degree and 0.2 meter error for the base camera to IMU transformation), our system was able to converge to values close to those reported by Kalibr, while the final position error was approximately 0.61 meters (0.14%). By contrast, the same dataset processed with the base camera alone yielded a final ending error of 3.58 meters (0.80%). This large increase in performance clearly validates our desire to include more cameras into the estimation process. For this experiment, we found that online intrinsic calibration did not improve the estimate (the non-calibrating version had 0.49 m error), indicating that the Kalibr parameters were very accurate in this scenario.

V. Conclusions

In this paper we have proposed a real-time multi-camera VINS (mc-VINS) within the MSCKF framework which is capable of fusing the information from an arbitrary number of asynchronous cameras. To limit the increase of computational burden, we only performed stochastic cloning at times corresponding to one of the cameras, and represented the state at intermediate times through linear interpolation. In addition, for robust performance, online calibration was performed for the spatial and temporal calibration parameters between all sensors, as well as for the intrinsics of each camera. The proposed estimator was tested on multiple camera configurations in real-world experiments, where it was shown to be able to provide highly accurate online calibration and trajectory estimation. In the future, we will increase the number of cameras to investigate the scaling properties of the system.

References

[1] T. Qin, P. Li, and S. Shen. "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator". In: *IEEE Transactions on Robotics* 34.4 (2018), pp. 1004–1020.

[2] C. X. Guo, K. Sartipi, R. C. DuToit, G. A. Georgiou, R. Li, J. O'Leary, E. D. Nerurkar, J. A. Hesch, and S. I. Roumeliotis. "Resource-Aware Large-Scale Cooperative Three-Dimensional Mapping Using Multiple Mobile Devices". In: *IEEE Transactions on Robotics* (2018), pp. 1–21.

[3] K. Sun, K. Mohta, B. Pfrommer, M. Watterson, S. Liu, Y. Mulgaonkar, C. J. Taylor, and V. Kumar. "Robust Stereo Visual Inertial Odometry for Fast Autonomous Flight". In: *IEEE Robotics and Automation Letters* 3.2 (Apr. 2018), pp. 965–972. ISSN: 2377-3766.

[4] T. Do, L. C. Carrillo-Arce, and S. I. Roumeliotis. "High-speed autonomous quadrotor navigation through visual and inertial paths". In: *The International Journal of Robotics Research* (2018).

[5] K. J. Wu, A. M. Ahmed, G. A. Georgiou, and S. I. Roumeliotis. "A square root inverse filter for efficient vision-aided inertial navigation on mobile devices". In: *2015 Robotics: Science and Systems Conference, RSS 2015*. MIT Press Journals. 2015.

[6] A. I. Mourikis and S. I. Roumeliotis. "A multi-state constraint Kalman filter for vision-aided inertial navigation". In: *Proceedings of the IEEE International Conference on Robotics and Automation*. Rome, Italy, Apr. 2007, pp. 3565–3572.

[7] J. Hesch, D. Kottas, S. Bowman, and S. Roumeliotis. "Consistency Analysis and Improvement of Vision-aided Inertial Navigation". In: *IEEE Transactions on Robotics* PP.99 (2013), pp. 1–19.

[8] J. Hesch, D. Kottas, S. Bowman, and S. Roumeliotis. "Camera-IMU-based localization: Observability analysis and consistency improvement". In: *International Journal of Robotics Research* 33 (2014), pp. 182–201.

[9] M. Li and A. I. Mourikis. "High-Precision, Consistent EKF-based Visual-Inertial Odometry". In: *International Journal of Robotics Research* 32.6 (2013), pp. 690–711.

[10] G. Huang, M. Kaess, and J. Leonard. "Towards Consistent Visual-Inertial Navigation". In: *Proc. of the IEEE International Conference on Robotics and Automation*. Hong Kong, China, May 2014, pp. 4926–4933.

[11] G. Huang, K. Eckenhoff, and J. Leonard. "Optimal-State-Constraint EKF for Visual-Inertial Navigation". In: *Proc. of the International Symposium on Robotics Research*. Sestri Levante, Italy, Sept. 2015.

[12] Z. Huai and G. Huang. "Robocentric Visual-Inertial Odometry". In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. (to appear). Madrid, Spain, Oct. 2018.

[13] M. K. Paul, K. Wu, J. A. Hesch, E. D. Nerurkar, and S. I. Roumeliotis. "A comparative analysis of tightly-coupled monocular, binocular, and stereo VINS". In: *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE. 2017, pp. 165–172.

[14] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale. "Keyframe-based visual-inertial odometry using nonlinear optimization". In: *International Journal of Robotics Research* (Dec. 2014).

[15] A. Mourikis, N. Trawny, S. Roumeliotis, A. Johnson, A. Ansar, and L. Matthies. "Vision-Aided Inertial Navigation for Spacecraft Entry, Descent, and Landing". In: *Robotics, IEEE Transactions on* 25.2 (Apr. 2009), pp. 264–280. ISSN: 1552-3098.

[16] S. Houben, J. Quenzel, N. Krombach, and S. Behnke. "Efficient multi-camera visual-inertial SLAM for micro aerial vehicles". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Oct. 2016, pp. 1616–1622.

[17] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós. "ORB-SLAM: a Versatile and Accurate Monocular SLAM System". In: *IEEE Transactions on Robotics* 15.2 (2015), pp. 1147–1163.

[18] M. K. Paul and S. I. Roumeliotis. "Alternating-Stereo VINS: Observability Analysis and Performance Evaluation". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018.

[19] P. Furgale, J. Rehder, and R. Siegwart. "Unified temporal and spatial calibration for multi-sensor systems". In: *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Nov. 2013, pp. 1280–1286.

[20] M. Li, H. Yu, X. Zheng, and A. I. Mourikis. "High-fidelity sensor modeling and self-calibration in vision-aided inertial navigation". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. May 2014, pp. 409–416.

[21] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale. "Keyframe-based visual–inertial odometry using nonlinear optimization". In: *The International Journal of Robotics Research* 34.3 (2015), pp. 314–334.

[22] M. Li and A. I. Mourikis. "Online Temporal Calibration for Camera-IMU Systems: Theory and Algorithms". In: *International Journal of Robotics Research* 33.7 (June 2014), pp. 947–964.

[23] M. Li and A. I. Mourikis. "Vision-aided inertial navigation with rolling-shutter cameras". In: *International Journal of Robotics Research* 33.11 (Sept. 2014), pp. 1490–1507.

[24] T. Qin and S. Shen. "Online temporal calibration for monocular visual-inertial systems". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 3662–3669.

[25] N. Trawny and S. I. Roumeliotis. *Indirect Kalman Filter for 3D Attitude Estimation*. Tech. rep. University of Minnesota, Dept. of Comp. Sci. & Eng., Mar. 2005.

[26] C. Hertzberg, R. Wagner, U. Frese, and L. Schröder. "Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds". In: *Information Fusion* 14.1 (2013), pp. 57–77.

[27] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[28] A. B. Chatfield. *Fundamentals of High Accuracy Inertial Navigation*. Reston, VA: American Institute of Aeronautics and Astronautics, Inc., 1997.

[29] S. I. Roumeliotis and J. W. Burdick. "Stochastic Cloning: A generalized framework for processing relative state measurements". In: *Proc. of IEEE Conf. on Robotics and Automation*. Washington, DC, May 2002, pp. 1788–1795.

[30] G. Chirikjian. *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*. Vol. 2. Springer Science & Business Media, 2011.

[31] E. Rosten, R. Porter, and T. Drummond. "Faster and Better: A Machine Learning Approach to Corner Detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.1 (Jan. 2010), pp. 105–119. ISSN: 0162-8828.

[32] S. Baker and I. Matthews. "Lucas-Kanade 20 Years On: A Unifying Framework". In: *International Journal of Computer Vision* 56 (Mar. 2004), pp. 221–255.

[33] OpenCV Developers Team. *Open Source Computer Vision (OpenCV) Library*. Available: http://opencv.org.

[34] P. Geneva, K. Eckenhoff, and G. Huang. "Asynchronous Multi-Sensor Fusion for 3D Mapping and Localization". In: *Proc. of the IEEE International Conference on Robotics and Automation*. 2018.

[35] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza. "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation". In: *Robotics: Science and Systems*. Georgia Institute of Technology. 2015.

[36] K. Eckenhoff, P. Geneva, J. Bloecker, and G. Huang. *Multi-Camera Visual-Inertial Navigation with Online Intrinsic and Extrinsic Calibration*. Tech. rep. RPNG-2018-MCVINS. Available: http://udel.edu/~ghuang/papers/tr_mc-vins.pdf. University of Delaware, 2018.