Article

The International Journal of Robotics Research
2015, Vol. 34(13) 1573–1589

DOI: 10.1177/0278364915602544
ijr.sagepub.com

Asynchronous adaptive conditioning for visual–inertial SLAM

Nima Keivan and Gabe Sibley

Abstract

This paper is concerned with real-time monocular visual–inertial simultaneous localization and mapping (SLAM). In particular a tightly coupled nonlinear-optimization-based solution that can match the global optimal result in real time is proposed. The methodology is motivated by the requirement to produce a scale-correct visual map, in an optimization framework that is able to incorporate relocalization and loop closure constraints. Special attention is paid to achieve robustness to many real world difficulties, including degenerate motions and unobservablity. A variety of helpful techniques are used, including: a relative manifold representation, a minimal-state inverse depth parameterization, and robust non-metric initialization and tracking. Importantly, to enable real-time operation and robustness, a novel numerical dog-leg solver is presented that employs multi-threaded, asynchronous, adaptive conditioning. In this approach, the conditioning edges of the SLAM graph are adaptively identified and solved for both synchronously and asynchronously. In this way one thread focuses on a small number of temporally immediate parameters and hence constitute a natural "front-end"; the other thread adaptively focuses on larger portions of the SLAM problem, and hence is able to re-estimate past parameters in the presence of new information: an ability that is useful for self-calibration, during degenerate motions, or when bias and the direction of gravity are poorly observed. Experiments with real and simulated data for both indoor and outdoor scenarios demonstrate that asynchronous adaptive conditioning is accurate, and able to closely track the batch SLAM maximum likelihood solution in real time.

Keywords

Simultaneous localization and mapping (SLAM), visual–inertial odometry, robotics, inertial measurement unit (IMU), bundle adjustment, mapping

1. Introduction

It is well known that the batch bundle adjustment solution to monocular simultaneous localization and mapping (SLAM) is the gold standard, in that its form defines the Cramer–Rao lower bound and that it takes advantage of all measurements over all time to compute the maximum-likelihood parameter estimate (MLE) (Engels et al., 2006; Triggs et al., 2000). Visual–inertial bundle adjustment (BA) is significantly more challenging than vision-only BA (Leutenegger et al., 2013). Vision-only monocular systems suffer from a well-studied scale ambiguity. Adding an inertial measurement unit (IMU) can make scale observable, however inertial measurements complicate matters when it comes to computing the global MLE solution incrementally in real-time.

For BA to be real-time for use on robots, a *local* approach is typically employed (Mouragnon et al., 2006). With an IMU this is difficult since the local adjustment region may need to be very large in order to ensure observability of certain parameters. Indeed, under certain degenerate motions such as constant velocity forward motion, some parameters may never be observable (though this rarely if ever happens in practice) (Jones et al., 2007; Kelly and Sukhatme, 2010).

An alternative to local BA is to only keep a sliding window of the most recent poses and landmarks active, and marginalize the rest into a prior distribution (Mourikis and Roumeliotis, 2007; Sibley, 2006; Sibley et al., 2010). This is equivalent to a fixed-lag Kalman smoother (Gelb, 1974; Maybeck, 1979) and recently such systems have shown remarkable results (Hesch et al., 2013; Li and Mourikis, 2013b; Li et al., 2014).

Marginalization into a prior distribution like this is predominantly employed for computational efficiency – if it were possible to compute the batch MLE solution

Department of Computer Science, University of Colorado, Boulder, USA

Corresponding author:
Nima Keivan, Department of Computer Science, University of Colorado, Boulder, CO 80309, USA.
Email: nima.keivan@colorado.edu

in real-time it would be preferable. Marginalization is also costly because it introduces conditional dependencies between the remaining parameters causing "fill-in". Fill-in can be addressed by cutting feature tracks and carefully marginalizing poses and landmarks simultaneously (Nerurkar et al., 2013). Marginalization is also potentially dangerous because it bakes in linearization errors, which can lead to over-confident estimates or divergence unless one is careful to maintain consistency (Hesch et al., 2013; Li et al., 2014). Carrying prior distributions induced from marginalization also necessitates an expensive global optimization at loop-closure to obtain the correct marginal. This paper attempts to remedy these issues by avoiding marginalization altogether.

Instead of relying on marginalization, conditioning is used, which has shown surprisingly robust and accurate results in the computer vision community (Engels et al., 2006; Klein and Murray, 2007) and avoids locking in incorrect parameter estimates when used adaptively (Sibley et al., 2009). Using a relative manifold is also helpful because optimal relative transformation estimates in $\mathbf{SE}(3)$ are by definition near zero. This fact allows multiple threads to asynchronously optimize and update different overlapping subsets of the full problem without detriment.

Adaptive asynchronous conditioning has other benefits: it can (a) perform robust initialization even under degenerate motions, (b) allow constant-time loop closure without expensive loop-long re-linearization, (c) operate even during poor observability conditions, (e) avoid inconsistency associated with early marginalization and re-linearization, and (f) track the relative-space maximum likelihood solution in constant time. We find that adaptive asynchronous conditioning is accurate, and closely tracks the global batch optimal solution at a fraction of the computational cost, which enables real-time operation.

2. Methodology

With sparse visual SLAM, the usual concern is with the estimation of keyframe (Klein and Murray, 2007) and landmark poses, based on the image measurements of tracked 3D features (Triggs et al., 2000). The addition of gyroscope and accelerometer measurements however, necessitates the estimation of the body velocity and the sensor biases. With these added parameters, the state vector for the batch visual–inertial SLAM problem is defined as

$$\mathbf{X} = \left[\begin{array}{cccc} \left\{ \begin{array}{cccc} \mathbf{x}_{wp_n}^{\mathrm{T}} & \mathbf{v}_{w_n}^{\mathrm{T}} & \mathbf{b_{g_n}}^{\mathrm{T}} & \mathbf{b_{a_n}}^{\mathrm{T}} \end{array} \right\} & \{\rho_k\} \end{array} \right]^{\mathrm{T}} \quad (1)$$

where $\left\{ \begin{array}{cccc} \mathbf{x}_{wp_n}^{\mathrm{T}} & \mathbf{v}_{w_n}^{\mathrm{T}} & \mathbf{b_{g_n}}^{\mathrm{T}} & \mathbf{b_{a_n}}^{\mathrm{T}} \end{array} \right\}$ is the set of keyframe parameters, defined as follows: $\mathbf{x}_{wp_n} \in \mathbf{SE}(3)$ is the transformation from the coordinates of the nth keyframe to world coordinates, $\mathbf{v}_{w_n} \in \mathbb{R}^3$ is the velocity vector of the nth keyframe in world coordinates, and $\mathbf{b_{g_n}} \in \mathbb{R}^3$ and $\mathbf{b_{a_n}} \in \mathbb{R}^3$ are the gyroscope and accelerometer bias parameters for the nth keyframe respectively. Similarly $\{\rho_k\}$ is the 1-d inverse-depth (Pietzsch, 2008) parameter for the kth landmark.

Note that in this case, the world frame denotes a "lifted" local frame, where a breadth-first search is used to obtain a local coordinate system from the relative map representation (Mei et al., 2011; Sibley et al., 2009). This local coordinate frame simplifies the visual and inertial constraints, which are minimized in the optimization, in comparison to their relative formulations. Once the optimization is complete, the relative representation of the optimized parameters is written back into the map. Note that the relative map representation is not a prerequisite of the proposed method, and is used as it facilitates updates to a single map structure from multiple asynchronous optimizations. Section 2.5 further expands on the specifics of using a relative map representation with visual–inertial SLAM.

The parameterization of $\mathbf{x}_{wp_n}$ deserves special notice. The transformation has six degrees of freedom (DOF): three for translation, and three for rotation. However, 6DOF representations of transformations suffer from singularities, due to what is known as gimbal lock in the rotation DOF. To avoid this problem, the rotation component of the transformation is represented as a quaternion:

$$\mathbf{x}_{wp} = \left[\mathbf{p}_{wp}^{\mathrm{T}} \ \mathbf{q}_{wp}^{\mathrm{T}} \right]^{\mathrm{T}} \quad (2)$$

where $\mathbf{q}_{wp} \in \mathbb{R}^4$ is a quaternion representing the rotation from keyframe to world coordinates, and $\mathbf{p}_{wp} \in \mathbb{R}^3$ is the translation vector from keyframe to world coordinates. This representation has 7DOF and as such, is over-parameterized for the underlying 6DOF. To avoid the null-spaces caused by over-parameterizing the space in an optimization, a *local parameterization* of the space is utilized as follows:

$$\begin{aligned} \mathbf{x}_{wp'} &= \mathbf{x}_{wp} \oplus \mathbf{x}_{pp'} \left(\Delta\mathbf{x}_q, \Delta\mathbf{x}_p \right) \\ &= \left[\left(\mathbf{q}_{wp} \otimes \exp_{\mathbf{q}} \left(\Delta\mathbf{x}_q \right) \right)^{\mathrm{T}} \left(\mathbf{p}_{wp} + \Delta\mathbf{x}_p \right)^{\mathrm{T}} \right]^{\mathrm{T}} \end{aligned} \quad (3)$$

where $\mathbf{x}_{pp'} \left(\Delta\mathbf{x}_q, \Delta\mathbf{x}_p \right)$ represents an *update* applied to the transformation $\mathbf{x}_{wp}$, composed of a 3DOF translation delta $\Delta\mathbf{x}_p$ and a 3DOF rotation delta $\Delta\mathbf{x}_q$, and $\otimes$ is the quaternion multiplication operator. The translation delta is additive and is simply added to the translation vector of $\mathbf{x}_{wp}$. The rotation delta is in the $\mathfrak{so}(3)$ *tangent space* (Strasdat, 2012) and is transformed into the $\mathbf{SO}(3)$ manifold using the exponential operator. The resulting quaternion can then be multiplied by $\mathbf{q}_{wp}$ to apply the update to the rotation. Here $\oplus$ is the update operator, equivalent to matrix multiplication if $\mathbf{x}_{wp}$ and $\mathbf{x}_{pp'}$ were represented as 4×4 homogeneous transformation matrices. $\Delta\mathbf{x}_q$ and $\Delta\mathbf{x}_p$ now represent a 6DOF manifold, which can be used to update the keyframe pose without over-parameterization.

2.1. Probabilistic derivation

Given the state vector in equation (1), a probabilistic method will be derived to obtain the optimal estimates for

the parameters given visual and inertial measurements. The joint probability distribution of the state vector $\mathbf{X}$ and the visual–inertial measurement set $\mathbf{Z}$ can be factored using Bayes' rule as follows:

$$P(\mathbf{X}, \mathbf{Z}) = P(\mathbf{Z}|\mathbf{X})\, P(\mathbf{X}) \tag{4}$$

where $P(\mathbf{Z}|\mathbf{X})$ is the measurement *likelihood* and $P(\mathbf{X})$ is the *prior* term. Assuming conditional independence between measurements, the likelihood term can be factored:

$$P(\mathbf{X}, \mathbf{Z}) = \prod_{i=1}^{n} P(\mathbf{z}_i|\mathbf{X})\, P(\mathbf{X}) \tag{5}$$

where $P(\mathbf{z}_i|\mathbf{X})$ is the likelihood of the ith measurement given the state vector $\mathbf{X}$. The optimal estimate for the state vector parameters is then one that maximizes the joint probability of the state and measurements in equation (4), which is also achieved by maximizing the measurement likelihood and prior probabilities:

$$\widehat{\mathbf{X}} = \arg\max_{\mathbf{X}} P(\mathbf{X}, \mathbf{Z}) = \arg\max_{\mathbf{X}} \left(\prod_{i=1}^{n} P(\mathbf{z}_i|\mathbf{X})\, P(\mathbf{X}) \right) \tag{6}$$

With the assumption that the measurement terms are normally distributed, the likelihood term for the ith measurement term (equation (5)) can be written as a multivariate normal distribution with mean $h(\mathbf{X})$ and covariance $\Sigma_{\mathbf{z}}$ as follows:

$$\begin{aligned} P(\mathbf{z}_i|\mathbf{X}) = & \frac{1}{\sqrt{(2\pi)^k\, |\Sigma_{\mathbf{z}}|}} \\ & \exp\left(-\frac{1}{2} (\mathbf{z}_i - h(\mathbf{X}))^{\mathrm{T}}\, \Sigma_{\mathbf{z}}^{-1}\, (\mathbf{z}_i - h(\mathbf{X})) \right) \propto \\ & \exp\left(-\frac{1}{2} \|\mathbf{z}_i - h(\mathbf{X})\|^2_{\Sigma_{\mathbf{z}}} \right) \end{aligned} \tag{7}$$

where the proportional relation ($\propto$) is used to omit the normalization term, and $\|\mathbf{z} - h(\mathbf{X})\|^2_{\Sigma_{\mathbf{z}}} = (\mathbf{z} - h(\mathbf{X}))^{\mathrm{T}}\, \Sigma_{\mathbf{z}}^{-1}\, (\mathbf{z} - h(\mathbf{X}))$ denotes the squared Mahalanobis distance. $h(\mathbf{X})$ is the measurement function, which predicts the measurement $\mathbf{z}$ given the state vector.

Likewise, with the assumption that the prior term is normally distributed with mean Π and covariance Σ_{Π}, it can be written as

$$P(\mathbf{X}) \propto \exp\left(-\frac{1}{2} \|\mathbf{X} - \Pi\|^2_{\Sigma_{\Pi}} \right) \tag{8}$$

Given equations (7) and (8), equation (6) can be written as

$$\begin{aligned} \hat{\mathbf{X}} = \arg\max_{\mathbf{X}} P(\mathbf{X}, \mathbf{Z}) = \arg\max_{\mathbf{X}} \\ \left[\prod_{i=1}^{n} \exp\left(-\frac{1}{2} \|\mathbf{z}_i - h(\mathbf{X})\|^2_{\Sigma_{\mathbf{z}}} \right) \exp\left(-\frac{1}{2} \|\mathbf{X} - \Pi\|^2_{\Sigma_{\Pi}} \right) \right] \end{aligned} \tag{9}$$

Taking the negative log of equation (9), a cost function can be obtained, in order to obtain the maximum *a posteriori* estimate for the state vector $\mathbf{X}$:

$$\hat{\mathbf{X}} = \arg\min_{\mathbf{X}} \left(\sum_{i=1}^{n} \|\mathbf{z}_i - h(\mathbf{X})\|^2_{\Sigma_{\mathbf{z}}} + \|\mathbf{X} - \Pi\|^2_{\Sigma_{\Pi}} \right) \tag{10}$$

where the first term represents a sum over all measurement residuals, and the second term represents the prior residual. Note that since both the state vector $\mathbf{X}$ and prior mean Π contain rotations, the subtraction operator is not sufficient to obtain a residual between the two. Instead, due to the existence of discontinuities in the space of rotations, the difference between two quaternions is used as a measure of distance:

$$\Delta\mathbf{q} = \log_{\mathbf{q}} \left(\mathbf{q}_{\mathbf{X}} \otimes \mathbf{q}_{\Pi}^{-1} \right) \tag{11}$$

where the $\log_{\mathbf{q}}$ operator transforms the 4DOF difference quaternion from the $\mathbf{SO}(3)$ manifold to the 3DOF $\mathfrak{so}(3)$ tangent space. In Section 2.3, it will be shown that, in the case of inertial measurements, a similar approach is required in order to compute the residual $\mathbf{z}_i - h(\mathbf{X})$ as rotation terms are involved. This detail does not, however, invalidate the aforementioned derivation.

Note that although the aforementioned maximum *a posteriori* formulation includes a prior distribution on the state parameters, the proposed method does not make use of a prior, and as such uses maximum likelihood estimation. This is further explained in Section 2.4.

2.2. Visual measurements

Visual measurements are formed by tracking the 2D image projection location of 3D landmarks in the scene. A residual is then computed from the difference in the predicted 2D image position of the landmark and the actual measured 2D position. Figure 1 shows the parameters involved in a single visual residual. The measurement function $h(\mathbf{X})$ for visual residuals is defined as follows. (Note that the transformation $\mathbf{T}_{wp}$ is the equivalent 4×4 matrix representation of $\mathbf{x}_{wp}$, which is used here for brevity. The underlying implementation uses the quaternion and translation components of $\mathbf{x}_{wp}$):

$$h(\mathbf{X}) = \mathcal{P}(\mathbf{p}_r, \mathbf{X}) = \mathcal{P}\left(\mathbf{T}_{pc}^{-1} \mathbf{T}_{wp_m}^{-1} \mathbf{T}_{wp_r} \mathbf{T}_{pc} \mathcal{P}^{-1}(\mathbf{p}_r, \rho) \right) \tag{12}$$

where ρ is the inverse depth of the landmark, $\mathbf{T}_{wp_r}$ is the transformation from the coordinates of the reference keyframe (in which the landmark was first seen and initialized) to world coordinates, $\mathbf{T}_{wp_m}$ is the transformation from the measurement keyframe to world coordinates, $\mathbf{p}_r$ is the 2D image location where the original feature was initialized in the reference keyframe, $\mathbf{p}_m$ is the measured 2D image location in the measurement keyframe, $\mathbf{T}_{pc}$ is the transformation from the camera to the keyframe coordinates, $\mathcal{P}^{-1}$ is

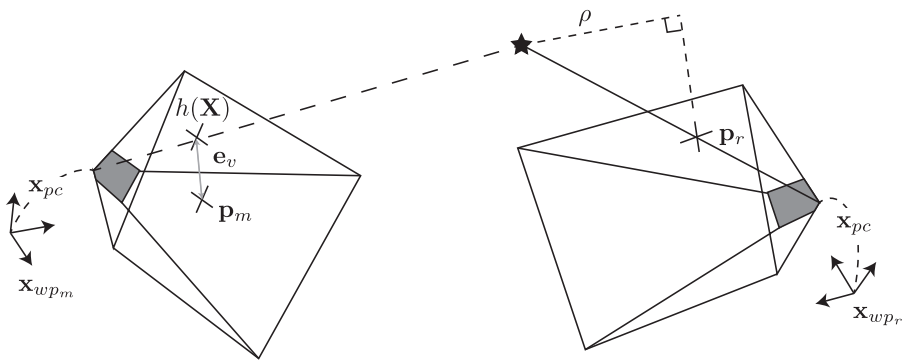

Fig. 1. The formation of a visual residual given a landmark and the reference and measurement keyframes. The landmark is first formed by corner detection, where $\mathbf{p}_r$ denotes the image location of the detected corner in the reference frame $\mathbf{x}_{wp_r}$. The landmark inverse depth parameter ρ specifies the perpendicular distance between the landmark and the image plane. The measurement keyframe reference frame $\mathbf{x}_{wp_m}$ specifies the transformation from the measurement keyframe to world coordinates, and $\mathbf{p}_m$ is the detected image location of the landmark in the measurement frame. As the keyframe pose corresponds to the IMU frame (Section 2.3), $\mathbf{x}_{pc}$ denotes the transformation from the camera to the IMU frame. Given the aforementioned values, the landmark can be projected into the measurement frame, where an error vector ($\mathbf{e}_\mathcal{V}$) is formed with the difference between the predicted and measured landmark location in the image.

a 2D to 3D back-projection function that returns the homogeneous landmark position given the reference 2D image location $\mathbf{p}_r$ and the inverse depth ρ, and $\mathcal{P}$ is a 3D to 2D camera projection function that returns the predicted 2D image coordinates. This operation forms a transform function from the image plane of the reference camera to that of the measurement camera, given their respective poses and the inverse depth of the landmark. The camera to keyframe transformation $\mathbf{T}_{pc}$ is non-zero as the keyframe is collocated on the inertial frame (the frame in which inertial measurements are made), to simplify the inertial integration equations. As shown in Figure 1, the visual measurement residual of the kth landmark in the mth frame is calculated from the error vector between the predicted and measured 2D location, $\mathbf{e}_v$ as follows:

$$\begin{aligned} r_{\mathcal{V}_{m,k}} &= \|\mathbf{e}_{\mathcal{V}_{m,k}}\|^2_{\Sigma_{\mathbf{p}_{m,k}}} \\ &= \|\mathbf{p}_{m,k} - \mathcal{P}\left(\mathbf{T}_{pc}^{-1}\mathbf{T}_{wp_m}^{-1}\mathbf{T}_{wp_{r,k}}\mathbf{T}_{pc}\mathcal{P}^{-1}\left(\mathbf{p}_r, \rho_k\right)\right)\|^2_{\Sigma_{\mathbf{p}_{m,k}}} \end{aligned} \tag{13}$$

where $\mathbf{p}_{m,k}$ is the measured 2D image location of the kth landmark in the mth keyframe with covariance $\Sigma_{\mathbf{p}_{m,k}}$, and $\mathbf{T}_{wp_{r,k}}$ is the transform from coordinates of the reference keyframe of the kth landmark to world coordinates.

Since the camera intrinsics are assumed constant, the back-projection of the reference feature location onto the $z = 1$ plane can be computed once and then stored for all reprojections involving that landmark. This pre-computation step provides significant benefits where non-invertible camera models are used, as the inverse projection function for these camera models can be computationally expensive. The back-projection is undertaken as follows:

$$[\mathbf{x}_k\,;\,1.0] = \mathcal{P}^{-1}\left(\mathbf{p}_r, 1.0\right) \tag{14}$$

where $\mathbf{x}_k \in \mathbb{R}^3$ is the 3D vector representing the feature on the $z = 1$ plane of the reference camera. Given this precalculated vector, the residual can be re-written by simply replacing the $z = 1$ inverse depth value with the parameter ρ, obviating the inverse projection function in

$$\begin{aligned} r_{\mathcal{V}_{m,k}} &= \|\mathbf{e}_{\mathcal{V}_{m,k}}\|^2_{\Sigma_{\mathbf{e}_{m,k}}} \\ &= \|\mathbf{p}_{m,k} - \mathcal{P}\left(\mathbf{T}_{pc}^{-1}\mathbf{T}_{wp_m}^{-1}\mathbf{T}_{wp_{r,k}}\mathbf{T}_{pc}\left[\mathbf{x}_k\,;\,\rho\right]\right)\|^2_{\Sigma_{\mathbf{p}_{m,k}}} \end{aligned} \tag{15}$$

Here the covariance of the error vector $\mathbf{e}_{\mathcal{V}_{m,k}}$ is equal to the covariance of the 2D pixel location detected in the image $\Sigma_{\mathbf{p}_{m,k}}$, due to the fact that the measurement and error vector are in the same space. More formally, if the image measurement is defined as

$$\mathbf{p}_{m,k} \sim \mathcal{N}\left(\bar{\mathbf{p}}_{m,k}, \Sigma_{\mathbf{p}_{m,k}}\right) \tag{16}$$

where $\bar{\mathbf{p}}_{m,k}$ is the true measurement, then the distribution over the error vector can be written as

$$P\left(\mathbf{e}_{\mathcal{V}_{m,k}}|\mathbf{X}\right) = P\left(\mathbf{p}_{m,k} - h\left(\mathbf{X}\right)\right) \sim \mathcal{N}\left(h\left(\mathbf{X}\right), \Sigma_{\mathbf{p}_{m,k}}\right) \tag{17}$$

Therefore, the measurement uncertainty is used directly in the Mahalanobis distance used for the residual. This distinction is made to motivate the covariance derivation in Section 2.3, as the inertial measurement and error vector are not in the same space. Linear error propagation must then be used to obtain the covariance of the error vector. The covariance $\Sigma_{\mathbf{p}_{m,k}}$ can be obtained from the autocorrelation matrix used in extracting Harris corners (Harris and Stephens, 1988). However, due to the fact that the salient corners used for tracking are generally thresholded to have high autocorrelation, a standard value of 1 pixel is used for the x and y dimension covariance. In other words, $\Sigma_{\mathbf{p}_{m,k}} = \mathbf{I}_{2\times 2}$.

2.3. Inertial measurements

Similar to Section 2.2, inertial measurements obtained from the IMU are used in order to form constraints over state variables. The values obtained from the IMU are 3DOF acceleration and angular rate measurements. IMU measurements are taken in the IMU coordinate frame. This fact motivates the use of the IMU coordinate frame as the privileged frame, as the equations governing the rigid body dynamics are greatly simplified. Both accelerometer and gyroscope measurements are assumed to have additive Gaussian noise, and are therefore defined as follows:

$$\omega_m \sim \mathcal{N}\left(\mathbf{q}_{wp}^{-1} \otimes (\omega_w) + \mathbf{b}_g, \Sigma_\omega\right) \tag{18}$$

$$\mathbf{a}_m \sim \mathcal{N}\left(\mathbf{q}_{wp}^{-1} \otimes (\mathbf{a}_w - \mathbf{g}_w) + \mathbf{b}_a, \Sigma_\mathbf{a}\right) \tag{19}$$

where ω_m refers to the angular rate measurements in the IMU frame obtained from the sensor, $\mathbf{q}_{wp}$ is the quaternion defining the rotation from IMU to world coordinates, ω_w is the true angular rate of the IMU in the world frame, $\mathbf{b}_g$ is the gyroscope bias vector, and $\Sigma_\omega \in \mathbb{R}^{3\times3}$ is the covariance matrix specifying the uncertainty in gyroscope measurements. Similarly for acceleration measurements, ω_m is the acceleration measurements in the IMU frame obtained from the sensor, $\mathbf{a}_w$ refers to the true acceleration of the IMU in the world frame, $\mathbf{g}_w$ is the gravity vector in the world frame, $\mathbf{b}_a$ is the accelerometer bias vector, and $\Sigma_\mathbf{a} \in \mathbb{R}^{3\times3}$ is the covariance matrix specifying the uncertainty in the accelerometer measurements. The effects of the Earth's rotation have been ignored in these equations.

Note that the angular velocity vector $\omega_w \in \mathbb{R}^{3\times1}$ is the minimal representation of the $\mathfrak{so}(3)$ tangent space. As the measurement is in the IMU frame, the tangent space needs to be transformed between two different $\mathbf{SO}(3)$ reference frames. This operation is the Adjoint Map (Strasdat, 2012), which for $\mathbf{SO}(3)$ is simply multiplication by the rotation matrix transforming from the start to end reference frame. Therefore multiplication by the equivalent quaternion $\mathbf{q}_{wp}^{-1}$ can be used to transform the angular velocities in the world tangent space to the IMU frame tangent space.

In order to constrain the state variables, the equations of rigid body motion are used to obtain analytical relationships between the IMU measurements and the state parameters. The integration state for the kth step ($\mathbf{x}_k \in \mathbb{R}^{16}$) is arranged as follows:

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{p}_{wp_k}^\mathrm{T} & \mathbf{q}_{wp_k}^\mathrm{T} & \mathbf{v}_{w_k}^\mathrm{T} & \mathbf{b}_{g_k}^\mathrm{T} & \mathbf{b}_{a_k}^\mathrm{T} \end{bmatrix}^\mathrm{T} \tag{20}$$

The relationship between the state parameters and IMU measurements is derived as follows:

$$\begin{aligned} \dot{\mathbf{p}}_{wp} &= \mathbf{v}_w \\ \dot{\mathbf{v}}_w &= \mathbf{a}_w = \mathbf{q}_{wp} \otimes (\mathbf{a}_m - \mathbf{b}_a) + \mathbf{g}_w \\ \dot{\mathbf{b}}_g &= \mathbf{w}_g \\ \dot{\mathbf{b}}_a &= \mathbf{w}_a \end{aligned} \tag{21}$$

The time-derivatives of translation ($\dot{p}_{wp}$) and velocity ($\dot{\mathbf{v}}_w$) are straightforward. The accelerometer and gyroscope biases are modeled as Gaussian random walk processes. Consequently, their discrete time derivatives are given by the white Gaussian noise vectors $\mathbf{w}_g \in \mathbb{R}^3$, and $\mathbf{w}_a \in \mathbb{R}^3$, where elements of each vector are independently drawn from a zero-mean Gaussian distribution with variance of σ_{b_g}, and σ_{a_g} respectively. The case of rotation derivatives deserves special attention. Given that the skew-symmetric angular velocity tensor $\widehat{\omega}$, denoted by the hat operator (Strasdat, 2012) is in the $\mathfrak{so}(3)$ tangent space, Euler integration of the angular velocities to form an incremental rotation in $\mathbf{SO}(3)$ can be performed using the exponential operator: $\Delta\mathbf{q}_{wp} = \exp_\mathbf{q}(\widehat{\omega}_w\,\mathrm{d}t)$, where $\exp_\mathbf{q}$ returns the $\mathbf{SO}(3)$ rotation in the form of a quaternion.

This incremental rotation can then be compounded with the original rotation matrix. Using a quaternion to represent elements of $\mathbf{SO}(3)$, the discrete integration of the angular velocities for a single time step can then be written as

$$\begin{aligned} \mathbf{q}_{wp_{k+1}} &= \exp(\hat{\omega}_{w_k}\,\mathrm{d}t) \otimes \mathbf{q}_{wp_k} \\ &= \exp_\mathbf{q}\left(\mathbf{q}_{wp} \otimes \widehat{(\omega_{m_k} - \mathbf{b}_{g_k})}\,\mathrm{d}t\right) \otimes \mathbf{q}_{wp_k} \end{aligned} \tag{22}$$

where the IMU frame angular rate measurement ω_{w_k} is transformed into the world frame by the $\mathbf{SO}(3)$ Adjoint. The other parameters are similarly discretely integrated for a single time step:

$$\begin{aligned} \mathbf{p}_{wp_{k+1}} &= \mathbf{p}_{wp_k} + \mathbf{v}_{w_k}\,\mathrm{d}t \\ \mathbf{v}_{w_{k+1}} &= \mathbf{v}_{w_k} + \mathbf{a}_w\,\mathrm{d}t = \mathbf{v}_{w_k} + \left(\mathbf{q}_{wp} \otimes \left(\mathbf{a}_{m_k} - \mathbf{b}_{a_k}\right) + \mathbf{g}_w\right)\mathrm{d}t \\ \mathbf{b}_{g_{k+1}} &= \mathbf{b}_{g_k} \\ \mathbf{b}_{a_{k+1}} &= \mathbf{b}_{a_k} \end{aligned} \tag{23}$$

The gyroscope and accelerometer biases are modeled as random walk processes, and as such are not modified in the integration. However, the uncertainty over the bias parameters grows with the integration. Note that although Euler integration is used here for brevity, in the actual implementation, Runge–Kutta 45 integration is used for higher accuracy.

Given the aforementioned integration approach, the inertial measurement set $\mathbf{Z} = \{\mathbf{z}_{t_0}, \ldots, \mathbf{z}_{t_n}\}$ where $\mathbf{z} = \begin{bmatrix} \omega & \mathbf{a} \end{bmatrix}$ can be used to integrate the $\mathbf{SE}(3)$ pose, velocity and biases from time t_0 to t_n. A residual can then be formed between the integrated state ($\mathbf{x}'$) and the parameters in the next keyframe as shown in Figure 2:

$$r_{\mathcal{I}_j} = \|\mathbf{e}_{\mathcal{I}_j}\|^2_{\Sigma_{\mathbf{e}_{ij}}} = \left\| \begin{bmatrix} \mathbf{p}_{wp_{j+1}} - \mathbf{p}' \\ \log_\mathbf{q}\left(\mathbf{q}_{wp_{j+1}}^{-1} \otimes \mathbf{q}'\right) \\ \mathbf{v}_{w_{j+1}} - \mathbf{v}' \\ \mathbf{b}_{g_{j+1}} - \mathbf{b}_{g_j} \\ \mathbf{b}_{a_{j+1}} - \mathbf{b}_{a_j} \end{bmatrix} \right\|^2_{\Sigma_{\mathbf{e}_{\mathcal{I}_j}}} \tag{24}$$

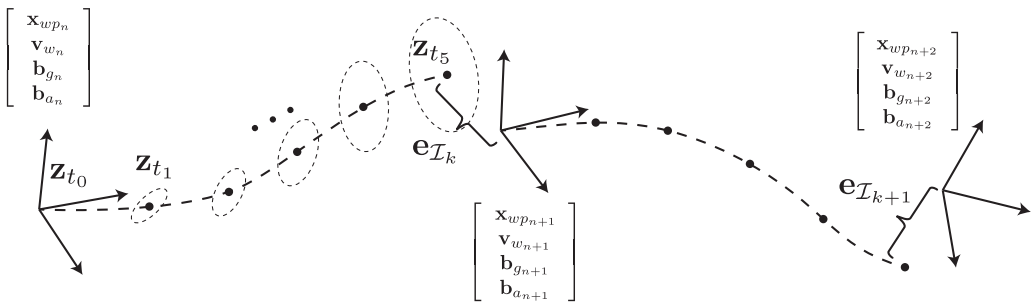

Fig. 2. Inertial residuals formed by integrating inertial measurements between subsequent keyframes. The residual vector $\mathbf{e}_{\mathcal{I}_{k+1}}$ is then formed using the error between the integrated inertial measurements from keyframe n and the state parameters of keyframe $n+1$. The ellipsoids represent the (exaggerated) uncertainty in the integrated state, which grows as more noisy measurements are used in the integration. The final state uncertainty is then used in computing the weight of the residual.

where $\log_{\mathbf{q}}\left(\mathbf{q}_{wp_{j+1}}^{-1} \otimes \mathbf{q}'\right) \in \mathbb{R}^3$ is used to calculate the $\mathfrak{so}(3)$ difference between the integrated orientation $\mathbf{q}'$ and the orientation of the $(j+1)$th keyframe, $\mathbf{q}_{wp_{j+1}}$. This is required, as the quaternion parameterization is redundant, and the underlying space has only 3DOF.

2.3.1. Residual covariance. Unlike the visual residual covariance in equation (17), the covariance of the inertial residual $\Sigma_{\mathbf{e}_{ij}}$ is not due to the uncertainty of a single measurement. Rather, the uncertainty of multiple measurements inflate the covariance of the integrated state as shown in Figure 2. In order to obtain this covariance, the uncertainty of the integrated state must be calculated at each stage of the integration. This involves the propagation of the current uncertainty for the time-step k to time-step $k+1$, as well as adding the uncertainty introduced from the measurements integrated at time k as follows:

$$\begin{aligned}&\Sigma_{\mathbf{x}_{k+1}}\left(\mathbf{x}_k, \Sigma_{\mathbf{x}_k}, \mathbf{z}_k, \Sigma_{\mathbf{z}_k}\right)\\ &\quad= \left(\frac{\partial \mathbf{x}_{k+1}}{\partial \mathbf{x}_k}\right) \Sigma_{\mathbf{x}_k} \left(\frac{\partial \mathbf{x}_{k+1}}{\partial \mathbf{x}_k}\right)^{\mathrm{T}} + \left(\frac{\partial \mathbf{x}_{k+1}}{\partial \mathbf{z}_k}\right) \Sigma_{\mathbf{z}_k} \left(\frac{\partial \mathbf{x}_{k+1}}{\partial \mathbf{z}_k}\right)^{\mathrm{T}}\end{aligned} \tag{25}$$

where $\Sigma_{\mathbf{x}_k} \in \mathbb{R}^{16\times16}$ is the singular covariance matrix of the most recent integration state, and $\Sigma_{\mathbf{z}_k} \in \mathbb{R}^{12\times12}$ is the covariance matrix of the IMU measurements and the bias drift model. It should be noted that $\Sigma_{\mathbf{x}_k}$ is singular due to the redundant 4D representation of rotation, during the integration on the $\mathbf{SE}(3)$ manifold. However, this covariance is propagated into the minimal $\mathfrak{se}(3)$ tangent space before being used in the optimization, and will no longer be singular. (See equation (28).) Here $\mathbf{x}_k \in \mathbb{R}^{16}$ is the integration state matrix arranged as

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{p}_{wp_k}^{\mathrm{T}} & \mathbf{q}_{wp_k}^{\mathrm{T}} & \mathbf{v}_{w_k}^{\mathrm{T}} & \mathbf{b}_{g_k}^{\mathrm{T}} & \mathbf{b}_{a_k}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{26}$$

and $\mathbf{z} = \begin{bmatrix} \omega & \mathbf{a} \end{bmatrix}$ is the measurement vector at time-step k. The IMU measurement and bias covariance matrix is usually diagonal, as the measurement uncertainties of the separate axes of the gyroscope and accelerometer and their biases are not correlated. It is formulated as

$$\Sigma_{\mathbf{z}_k} = \operatorname{diag}\left(\sigma_g^2 \cdot \mathbf{I}_{3\times3}, \sigma_a^2 \cdot \mathbf{I}_{3\times3}, \sigma_{b_g}^2 \cdot \mathbf{I}_{3\times3}, \sigma_{b_a}^2 \cdot \mathbf{I}_{3\times3}\right) \tag{27}$$

where σ_g, σ_a, σ_{b_g}, and σ_{b_a} are the uncertainties for the gyroscope, accelerometer, gyroscope bias and accelerometer bias respectively. The derivatives $\partial \mathbf{x}_{k+1}/\partial \mathbf{x}_k$ and $\partial \mathbf{x}_{k+1}/\partial \mathbf{z}_k$ are straightforward and are obtained by differentiating the single-step integrations in equation (23).

Using the uncertainty propagation in equation (25), the uncertainty of the final state $\Sigma_{\mathbf{x}'}$ can be obtained. Note that at the first step of the integration, the uncertainty of the state is set to 0. More formally: $\Sigma_{\mathbf{x}_0} = 0_{16\times16}$. The uncertainty of the error vector $\mathbf{e}_{\mathcal{I}}$ can then be obtained from $\Sigma_{\mathbf{x}'}$:

$$\Sigma_{\mathbf{e}_{\mathcal{I}}} \in \mathbb{R}^{15\times15} = \left(\frac{\partial \mathbf{e}_i}{\partial \mathbf{x}'}\right) \Sigma_{\mathbf{x}'} \left(\frac{\partial \mathbf{e}_i}{\partial \mathbf{x}'}\right)^{\mathrm{T}} \tag{28}$$

where $\partial \mathbf{e}_{\mathcal{I}}/\partial \mathbf{x}' \in \mathbb{R}^{15\times16}$ is the Jacobian of the function that forms the error residual from the integrated state $\mathbf{x}'$, and the subsequent keyframe:

$$\frac{\partial \mathbf{e}_{\mathcal{I}}}{\partial \mathbf{x}'} = \begin{bmatrix} -\mathbf{I}_{3\times3} & 0_{3\times4} & 0_{3\times9} \\ 0_{3\times3} & \partial \log_{\mathbf{q}}\left(\mathbf{q}_{wp_{j+1}}^{-1} \otimes \mathbf{q}'\right)/\partial \mathbf{q}' & 0_{3\times9} \\ 0_{9\times3} & 0_{9\times4} & -\mathbf{I}_{9\times9} \end{bmatrix} \tag{29}$$

Due to the simple subtractions used to form the errors, this Jacobian is mostly comprised of the negative identity matrix $-\mathbf{I}$. The only exception is the orientation term. As a redundant 4D quaternion parameterization is used during the integration, the covariance term for it will be singular. The orientation block in $\partial \mathbf{e}_{\mathcal{I}}/\partial \mathbf{x}'$ is then used to propagate this uncertainty into the $\mathfrak{se}(3)$ tangent-space where the 3D error will be formed.

Note that depending on the uncertainty and number of IMU measurements used in the integration, $\Sigma_{\mathbf{e}_{\mathcal{I}}}$ could become singular or badly conditioned, in which case a pseudo-inverse should be used to compute the Mahalanobis distance in equation (7). Unlike the visual residual covariance, the inertial residual covariance depends on the state,

via propagation through the state integration. However, this dependence is assumed to be negligible, as the residual covariances are recomputed before every optimization iteration.

2.3.2. Jacobian calculation. In order to optimize the residual in equation (24) in a maximum-likelihood estimator, partial derivatives of the error vector $\mathbf{e}_{\mathcal{I}_j}$ with respect to the involved state parameters are required. The error vector $\mathbf{e}_{\mathcal{I}_j}$ is formed by integrating IMU measurements starting at the jth keyframe, and comparing the integrated state with that of the $(j+1)$th keyframe. Therefore the involved state parameters are the position, orientation, velocity and biases of these two keyframes. Given the inertial residual in equation (24), it can be observed that the derivatives with respect to the $(j+1)$th keyframe are easily obtained, as they do not involve the integration of the inertial measurements. However, the same cannot be said for the jth keyframe, as it is the starting state for the integration. The partial derivatives would then need to be computed step-by-step through the integration using the chain rule. In order to simplify the required partial derivatives, some of the starting keyframe state parameters can be factored in the integration, as shown in Appendix B. The error vector and residual can then be re-written given this factorization, by replacing the components of the final integration state $\mathbf{x}'$ with their factorized forms:

$$
\begin{aligned}
r_{\mathcal{I}_j} &= \|\mathbf{e}_{\mathcal{I}_j}\|^2_{\Sigma_{\mathbf{e}_{ij}}} \\
&= \left\| \begin{bmatrix} \mathbf{p}_{wp_{j+1}} - \left(\mathbf{p}_{wp_j} + \mathbf{v}_{w_j}\Delta t + \frac{1}{2}\mathbf{g}_w\Delta t^2 + \mathbf{q}_{wp_j} \otimes \Delta\mathbf{p}\right) \\ \log_{\mathbf{q}}\left(\mathbf{q}^{-1}_{wp_{j+1}} \otimes \left(\mathbf{q}_{wp_j} \otimes \Delta\mathbf{q}\right)\right) \\ \mathbf{v}_{w_{j+1}} - \left(\mathbf{v}_{w_j} + \mathbf{g}_w\Delta t + \mathbf{q}_{wp_j} \otimes \Delta\mathbf{v}\right) \\ \mathbf{b}_{g_{j+1}} - \mathbf{b}_{g_j} \\ \mathbf{b}_{a_{j+1}} - \mathbf{b}_{a_j} \end{bmatrix} \right\|^2_{\Sigma_{\mathbf{e}_{\mathcal{I}_j}}}
\end{aligned}
\tag{30}
$$

where Δt is the total time of the integration, and $\Delta\mathbf{p}$, $\Delta\mathbf{q}$, and $\Delta\mathbf{v}$ are specially integrated deltas of position, orientation and velocity, which do not depend on any state parameters, as described in Appendix B. Given this factorization, partial derivatives with respect to velocity, position and orientation of the jth keyframe need not be computed through the integration, and can be directly evaluated. Unfortunately the same cannot be said of the partial derivatives with respect to the bias: $\partial\mathbf{e}_{\mathcal{I}_j}/\partial\mathbf{b}_{g_j}$ and $\partial\mathbf{e}_{\mathcal{I}_j}/\partial\mathbf{b}_{a_j}$, which have to be computed using the chain rule as follows

$$
\frac{\partial\mathbf{e}_{\mathcal{I}_j}}{\partial\mathbf{b}_j} = \frac{\partial\mathbf{e}_{\mathcal{I}_j}}{\partial\mathbf{x}'} \cdot \frac{\partial\mathbf{x}_n}{\partial\mathbf{b}_j} = \frac{\partial\mathbf{e}_{\mathcal{I}_j}}{\partial\mathbf{x}'} \cdot \left(\frac{\partial\mathbf{x}_n}{\partial\mathbf{x}_{n-1}} \cdot \frac{\partial\mathbf{x}_{n-1}}{\partial\mathbf{x}_{n-2}} \cdot \ldots \cdot \frac{\partial\mathbf{x}_1}{\partial\mathbf{b}_j}\right) \tag{31}
$$

It is worth noting that due to the additive noise model used for IMU measurements (equations (18) and (19)), the partial derivative of the integration step with respect to the biases ($\partial\mathbf{x}_n/\partial\mathbf{b}_j$) is in fact equal to the partial derivative with respect to the measurement noise ($\partial\mathbf{x}_{k+1}/\partial\mathbf{z}_k$), which is used in the propagation of the measurement uncertainty in equation (25). This partial derivative can be easily computed from the single-step integration in equations (22) and (23) and used for both the partial derivatives of the error vector, and to propagate the measurement uncertainties.

2.4. Optimization formulation

The previous sections outline the formulation of inertial and visual residuals. In a batch setting, all measurements collected would be used in order to optimize the state parameters. However, this is clearly not an acceptable solution for online systems, as the computational complexity of the problem is unbounded as more measurements are collected. One way to deal with this problem is to recursively marginalize landmarks and keyframes that fall outside a fixed-size window of active parameters. These older keyframes then form a prior distribution, which is used alongside the visual and inertial residuals (Leutenegger et al., 2013; Li and Mourikis, 2013a; Li et al., 2014; Mourikis and Roumeliotis, 2007). This approach however generally results in inconsistencies in the estimator, due to the multiple linearization points used in the marginalization process. One approach to fixing the inconsistencies is to use *first estimate* Jacobians (Li et al., 2014), however this issue will greatly reduce the tolerance of the estimator towards nonlinearities. Another issue with carrying a marginal distribution is difficulties when closing loops, as the marginal distribution will no longer be valid after the loop closure.

An alternate approach is to condition on parameters that fall outside the fixed-size window instead. By conditioning, an assumption is made that the estimates for these past parameters have converged and are correct. Figure 3 shows the graphical model representing the conditioning approach, where the conditioning edges are composed of measurements that involve both the active and inactive parameters. Conversely, active edges are composed of measurements that involve only active parameters. Equation (6) is then modified to remove the prior and to split the measurement likelihood term in two: active and conditioning measurements:

$$
\begin{aligned}
\widehat{\mathbf{X}}_a &= \arg\max_{\mathbf{X}_a} P\left(\mathbf{X}_a, \mathbf{Z}\right) = \arg\max_{\mathbf{X}_a} \\
&\left(\prod_{i=1}^{n_a} P\left(\mathbf{z}_i|\mathbf{X}_a\right) \prod_{j=1}^{n_c} P\left(\mathbf{z}_j|\mathbf{X}_a, \mathbf{X}_i\right)\right)
\end{aligned}
\tag{32}
$$

where n_a and n_c are the number of active and conditioning residuals respectively. Note that the likelihood term for the conditioning residuals is conditioned on both the active ($\mathbf{X}_a$) and inactive ($\mathbf{X}_i$) state parameters, but only the active parameters are optimized. Given the inertial and visual residuals discussed in the previous section, and the aforementioned conditioning approach, the total cost optimized is formulated as

$$
e = \sum_{m=1}^{n_k}\sum_{k=1}^{n_l} \|\mathbf{e}_{\mathcal{V}_{m,k}}\|^2_{\Sigma_{\mathbf{e}_{\mathcal{V}_{m,k}}}} + \sum_{j=1}^{n_k} \|\mathbf{e}_{\mathcal{I}_j}\|^2_{\Sigma_{\mathbf{e}_{\mathcal{I}_j}}} \tag{33}
$$

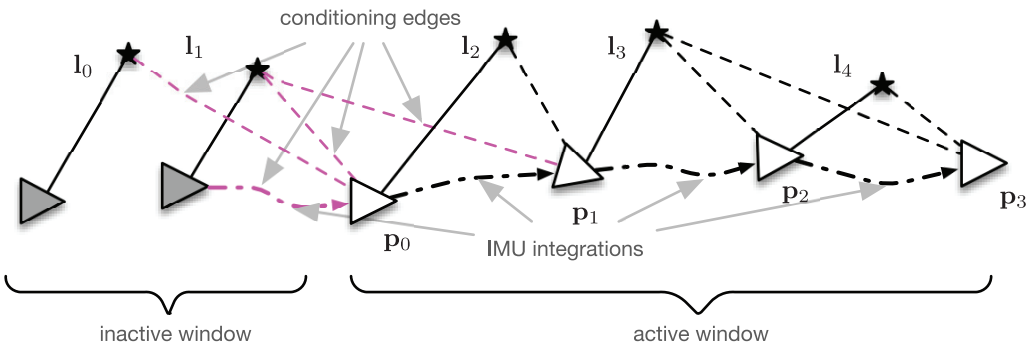

Fig. 3. The graphical model representing the conditioning sliding window optimization over a number of active keyframes. Edges in the graph are formed by either visual or inertial residuals and are shown as dashed lines. The solid lines represent implicit edges modeled by the inverse depth parameterization between landmarks and their reference keyframes. The conditioning edges are formed by residuals, which involve both active and inactive parameters.

where n_k and n_l are the total number of keyframes and landmarks respectively. Residuals that involve inactive parameters in $\mathbf{X}_i$ are evaluated normally. However, the partial derivatives of these residuals with respect to the inactive parameters are omitted. This method is used extensively with a fixed-window optimization in visual SLAM (Klein and Murray, 2007; Mei et al., 2011).

In order to minimize the given cost and obtain the maximum likelihood state estimates, the normal equations are utilized to iteratively update the active parameters $\mathbf{X}_a$:

$$\mathbf{J}^{\mathrm{T}}\mathbf{W}\mathbf{J}\Delta\mathbf{X}_a = \mathbf{J}^{\mathrm{T}}\mathbf{W}\mathbf{e} \tag{34}$$

where $\mathbf{e} = \begin{bmatrix} \mathbf{e}_{\mathcal{V}_0} & \dots & \mathbf{e}_{\mathcal{V}_n}, & \mathbf{e}_{\mathcal{I}_0} & \dots & \mathbf{e}_{\mathcal{I}_m} \end{bmatrix}^{\mathrm{T}}$ is the residual vector, $\mathbf{J} = {}^{\partial \mathbf{e}}/_{\partial \mathbf{x}_a}$ is the Jacobian of the residual vector with respect to the active state parameters, and the block diagonal weight matrix is composed of the inverse covariance matrix for every residual: $\mathbf{W} = \mathrm{diag}\left(\Sigma^{-1}_{\mathbf{e}_{\mathcal{V}_0}}, \dots, \Sigma^{-1}_{\mathbf{e}_{\mathcal{V}_n}}, \Sigma^{-1}_{\mathbf{e}_{\mathcal{I}_0}}, \dots, \Sigma^{-1}_{\mathbf{e}_{\mathcal{I}_m}}\right)$. The dog-leg trust region method (Powell, 1970) is used to solve equation (34). Note that the update to the state vector $\Delta\mathbf{X}_a$ is additive for all parameters except the orientation parameter $\mathbf{q}_{wp}$, where the update is applied through the exponential operator as per equation (3).

Due to the connectivity of the visual–inertial SLAM graph, new residuals may significantly alter the estimates for parameters that are no longer in the active window. Due to this same connectivity, active parameters that are conditioned on badly estimated past parameters are themselves poorly estimated, and can result in total divergence of the solution. In the case of visual–inertial SLAM, parameters such as velocity, and implicitly estimated parameters such as the direction of gravity are particularly sensitive to mis-estimation, and can derail the solution if not updated in the presence of new measurements. As such, a fixed-window optimization will not approximate the batch maximum likelihood solution. This motivates the introduction of the adaptive window optimization.

2.5. Adaptive window implementation

The adaptive local BA dynamically sets the window size, in order to appropriately fold in parameters as needed. Especially when using an IMU, future measurements can significantly affect estimates of past and inactive parameters for a fixed window size. Dynamically adjusting the window serves to allow the optimization to include parameters when new measurements are available that change their estimates, and when these new estimates affect the current active set of parameters Examples of these parameters are velocities, accelerometer and gyroscope biases, and the direction of gravity, which is implicitly parameterized.

Similarly to the adaptive method previously presented in Sibley et al. (2009), the condition used to determine the size of the optimization window is based on the residuals. In Sibley et al. (2009), parameters were included in the optimization if changes in the visual residuals were larger than a specific threshold. For the case of visual–inertial SLAM, since multiple sensor modalities are used, the Mahalanobis distance for the residuals is computed instead, and thresholded in a χ^2 test, in order to probabilistically determine when residuals are outside their expected intervals. In particular, the condition used to assess whether the size of the window needs to be increased between two optimization iterations is based on the residuals observed in the conditioning edges shown in Figure 3, after the kth optimization has converged. The measurement covariances can then be used to assess whether the conditioning residuals are within expected bounds using a χ^2 test and to adjust the size for the $(k+1)$th optimization. The conditioning Mahalanobis distance for projection residuals is

$$e_{c_{\mathcal{V}}} = \sum_{i \in C} \|\mathbf{r}_{\mathcal{V}_i}\|^2_{\Sigma_{V_i}} \tag{35}$$

where the summation is over the set C comprising all conditioning visual residuals, as shown in Figure 3. Likewise, the conditioning Mahalanobis distance for the single inertial

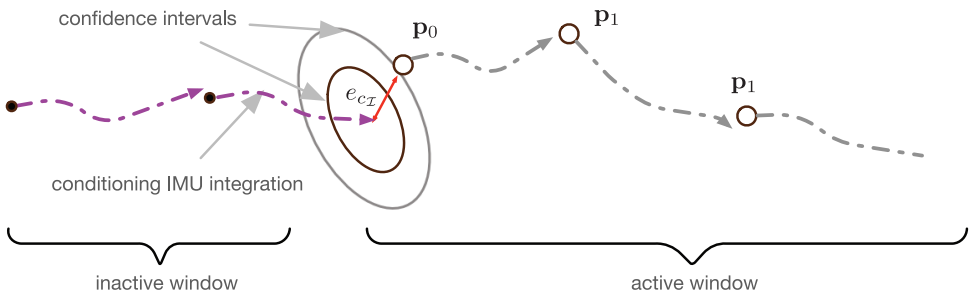

Fig. 4. Graphical model of the adaptive condition shown for inertial residuals. The ellipsoids show the confidence intervals of the Mahalanobis distance between the final integration pose and the first active keyframe as obtained from a χ^2 test. The size of the adaptive window is then expanded if the Mahalanobis distance is larger than a threshold confidence interval. Once expanded, the optimization is run over the larger window and the process is repeated by checking the same edge (which will no longer be a conditioning edge, as it will involve only active parameters).

residual connecting the active and inactive sets is

$$e_{c_\mathcal{I}} = \|\mathbf{r}_{\mathcal{I}_c}\|^2_{\Sigma_{\mathcal{I}_c}} \tag{36}$$

Given either $e_{c_\mathcal{I}}$ or $e_{c_\mathcal{V}}$, their corresponding adaptive condition variables $\alpha_{k_\mathcal{V}}$ and $\alpha_{k_\mathcal{I}}$ can be defined as

$$\begin{aligned} \alpha_{k_\mathcal{V}} &= \frac{e_{c_\mathcal{V}}}{\text{Inv}\,\chi^2\,(\beta, d)} \\ \alpha_{k_\mathcal{I}} &= \frac{e_{c_\mathcal{I}}}{\text{Inv}\,\chi^2\,(\beta, d)} \end{aligned} \tag{37}$$

where $d = 2|C| + 15$, and $\alpha_{k_\mathcal{V}}$ and $\alpha_{k_\mathcal{I}}$ represent the adaptive condition variables for visual and inertial residuals respectively, $\text{Inv}\,\chi^2\,(\beta, d)$ is the inverse cumulative χ^2 distribution for d dimensions evaluated for the confidence interval β, where d is the total dimension of the conditioning edge vector (15 for the conditioning inertial residual, and $2 \times |C|$ for the conditioning visual residuals). Since the multivariate Mahalanobis distance has a d-dimensional χ^2 distribution, the cumulative form of the χ^2 can be used to estimated if the given Mahalanobis distance lies in a certain confidence interval. Initially if $\alpha_{k_\mathcal{V}} > 1$ or $\alpha_{k_\mathcal{I}} > 1$, the conditioning residuals for either visual or inertial measurements lie outside the βth percentile probability as expected from the residual covariance (as shown in Figure 4), so the window size is increased, and the optimization is run to convergence with the now larger window. This increase in the window size is continued while the following conditions hold:

$$\begin{aligned} &\left(\alpha_{k+1_\mathcal{V}} > 1 \vee \alpha_{k+1_\mathcal{I}} > 1\right) \wedge \left(\left(\alpha_{k+1_\mathcal{V}} + \alpha_{k+1_\mathcal{I}}\right)\right. \\ &\quad \left. \leq \gamma \cdot \left(\alpha_{k_\mathcal{V}} + \alpha_{k_\mathcal{I}}\right)\right) \end{aligned} \tag{38}$$

where the $\wedge$ and $\vee$ signify the AND and OR boolean algebra operators, and $\gamma = (1 - 1e^{-5})$ is a tuning parameter to ensure that the condition variables decrease more than a specific threshold with regards to the previous optimization. If the condition in equation (38) is not met, the window is resized to its default minimum length and new frames are added to the window. Note that during the expansion, even though the conditioning edges of iteration k are replaced with earlier edges in the graph, the condition variables $\alpha_{k_\mathcal{V}}$ and $\alpha_{k_\mathcal{I}}$ are still computed at the same edge as the first optimization instance where the condition in equation (38) was met.

In the case of the inertial residual Mahalanobis distance (equation (36)), the residual covariance actually depends on the state, as it is propagated through the inertial integration as per equation (25). Consequently, the true Mahalanobis distance after the optimization must be computed by propagating the IMU measurement uncertainties given the latest estimate of the states. However, since the residual covariances are recomputed before each iteration of the optimization, and changes to the parameters are expected to be small at convergence, the change in the covariance of the inertial residual is assumed to be negligible in the final optimization iteration. Note that in the case of visual residuals, the residual covariance does not depend on the state.

The intuition behind the adaptive criterion on equation (38) is that if new residuals would affect the estimates of past parameters, and those parameters are not part of the active window, tension will be introduced in the conditioning edges that connect the active to the inactive parameters in the form of errors that are not explained by the measurement uncertainty. If the errors in the conditioning edges are indeed caused by mis-estimation of inactive parameters outside the window, including these parameters in the window should reduce these errors until they are within expected intervals based on measurement uncertainties. In the case that the conditioning error is not decreasing but is still outside expected bounds, the window size is returned to its default minimal value and the expansion is stopped, as the error is more likely explained by outlier measurements.

In order to use a relative map representation with visual–inertial measurements, the architecture presented in Sibley et al. (2009) has to be extended to accommodate the new parameters that are estimated: keyframe velocities, gravity direction and IMU biases. As the biases are estimated in the IMU frame, they are already relative and so need no further attention. The velocities are estimated in the "lifted" local frame, and are therefore relative to the orientation of

the first lifted pose. The same can be said about the direction of gravity in the lifted frame, $\mathbf{g}_w$. In order to store the velocity estimates and gravity direction in the map, they are converted to a relative parameterization, which for the kth keyframe is as follows:

$$\begin{aligned}\mathbf{v}_{p_k} &= \mathbf{q}_{wp_k}^{-1} \otimes \mathbf{v}_{w_k} \\ \mathbf{g}_{p_k} &= \mathbf{q}_{wp_k}^{-1} \otimes \mathbf{g}_w \end{aligned} \tag{39}$$

where $\mathbf{v}_{p_k}$ and $\mathbf{g}_{p_k}$ are the velocity and gravity direction vectors relative to the keyframe orientation. These values are stored in the map along with the relative transform between keyframes. Given this map representation, a local window can be "lifted" by performing a breadth first search starting from a given root node. The local gravity vector $\mathbf{g}_w$ can then be computed by inverting equation (39). The local velocity for each keyframe $\mathbf{v}_{w_k}$ is computed similarly. These values, alongside the local position and orientation of the keyframe obtained as per Sibley et al. (2009), are then used as the optimization state parameters.

The use of a relative map representation ensures that updates to the map remain small. While the optimization is performed in a "lifted" local frame (referred to as the world frame), the resulting parameters are pushed back into the map in their relative form, allowing multiple BAs to update it asynchronously without clashing.

Note that if the window size expands to the point where a batch optimization is performed, the relevant null-spaces are regularized to prevent the Hessian becoming singular. For the visual–inertial case, the null-spaces for the batch maximum likelihood estimation are the 3 unobservable translations of the root pose, and the unobservable yaw degree of freedom around gravity. In order to remove the yaw null-space, the rotation axis most parallel to that of the gravity vector is regularized in the first pose.

Since the time needed to run the optimization to convergence scales with the size of the window, the aforementioned solution could potentially be too slow for real-time operation, if the window size expands too much. To obtain a real-time solution, two optimization windows are run simultaneously in a multi-threaded environment. One thread runs a constant-width window optimization that is guaranteed to run at framerate, while the other runs the adaptive window optimization, which could potentially run slower than framerate. Special care is taken to ensure that these two optimizations, which run on different subsets of parameters, do not destructively interfere with one another. Fortunately, the relative map representation is quite conducive to this approach. Since only relative values are stored in the map, updates to the parameters are small. This would not be the case for a global map representation, where changes to past keyframes could potentially have significant downstream effects on more recent keyframes. Given the relative map representation, a greedy update approach is chosen, where synchronization between the threads is only performed during the *lift* and *write* operations, but otherwise updates are written to the map as soon as either optimization thread has run to convergence.

3. Results

In order to evaluate the proposed method, experiments were run on two sensor platforms: A hand-held mobile device (hereby referred to as rig A) with a global shutter wide-field-of-view camera with 640×480 resolution and a commercial MEMS IMU sampled at 120Hz, and a custom hand-held rig (referred to as rig B) with a wide-field-of-view camera at 1280×960 resolution downsampled to 640×480 with a commercial grade IMU sampled at 200 Hz. The sensor specifications are listed in Table 1. Both cameras capture images at 30 frames per second. A comparison between the images captured from the rigs is shown in Figure 5. In all experiments, the AAC system comprises a fixed-window estimator (as per Section 2.4) with a 10 keyframe window width and an asynchronous adaptive estimator (as per Section 2.5) with a minimum window size of 20 keyframes.

3.1. Tracking and keyframing

Visual residuals are established as per Section 2.2 by tracking salient image points between frames, to form a *feature track*. This is accomplished via a method inspired by the tracking component of Forster et al. (2014), where the photometric error of a re-projected feature patch is directly minimized to obtain the new location of the feature. In order to initialize new feature tracks, Harris corners (Harris and Stephens, 1988) are used in the areas of the image where not enough active tracks are present. In the direct approach employed, epipolar geometry is respected when minimizing the photometric error to localize the tracked features in new images. As such, the RANSAC (Fischler and Bolles, 1981) step generally employed when descriptor-based matching is used is no longer required. There is slight tolerance to violations of epipolar geometry to enable tracking if the landmarks or keyframe poses are not well estimated, and as such non-static objects in the scene may still violate epipolar geometry over the length of a feature track. These outliers are rejected in the fixed and adaptive window optimizations based on the ratio of outlier visual residuals to the total number of residuals in the feature track. A normalized cross-correlation score is computed between the current and original feature patches and is thresholded (at 0.875) to reject feature patches that have changed too much in appearance. In all examples, the tracker was configured to attempt to track 128 landmarks across the image. The value of the tuning parameter β from equation (37) is set to 0.1, meaning that $\alpha_k > 1$ if the conditioning Mahalanobis distance for either visual or inertial residuals is larger than the 10% χ^2 confidence interval for the given dimension d. The value of the tuning parameter β was experimentally chosen to ensure that the window size was expanded when required

Table 1. Specifications for the sensors used in each rig.

Rig	IMU	Camera
A	Project Tango Peanut IMU @ 120 Hz	Project Tango Peanut Wide-Angle Camera @ 30 fps, 640 × 480
B	Microstrain 3DM-GX3-25 @ 200 Hz	Point Grey FL3-U3-13Y3M-C @ 30 fps, 640 × 480

(a) (b)

Fig. 5. Comparison of images obtained from rig A (a) and rig B (b) as described in Section 3. Rig A is a mobile hand-held device with a wide-angle lens and a global shutter camera capturing at 30 fps and with 640 × 480 resolution. Rig B is a custom hand-held device with a global shutter camera capturing at 30 fps and with 1280 × 960 resolution downsampled to 640 × 480. The picture quality and contrast of rig B is generally much better. Both rigs use a commercial grade MEMS IMUs: rig A sampled at 120 Hz and rig B at 200 Hz. Rig B features a much larger lens and higher quality camera.

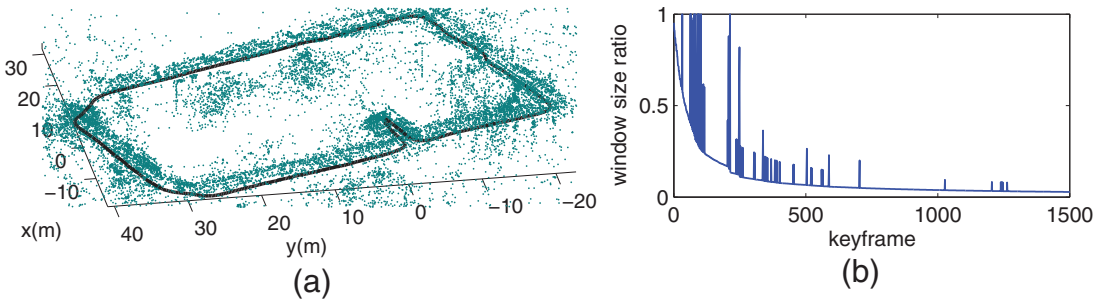

Fig. 6. (a) Trajectory and structure estimate resulting from running the proposed solution on a ∼200 m outdoor dataset captured using rig B. The error between the start and end keyframe poses is 0.33% of the distance traveled. (b) The window size ratio (computed by dividing the adaptive window size by the total number of keyframes) for the trajectory, the initialization region is at the beginning of the trajectory with the ratio reaching 1, signifying a batch solution.

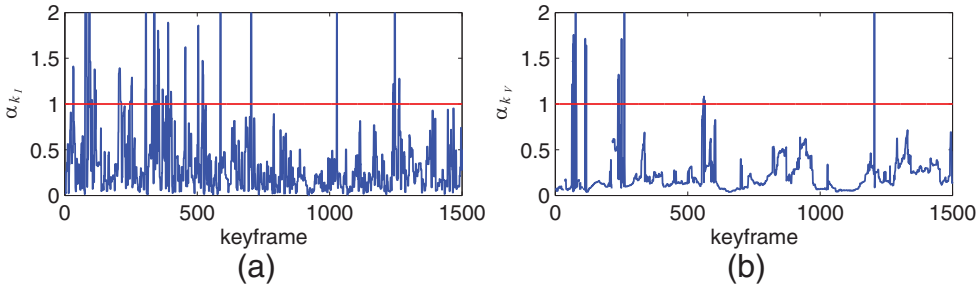

Fig. 7. Inertial (a) and visual (b) adaptive condition variable plots (see Section 2.5) for the trajectory in Figure 6(a). The red line indicates the threshold over which the condition in equation (38) will be true, and therefore the window size will be expanded. The expansion spikes in Figure 6(b) are correlated to the keyframes at which either the visual or inertial condition variable is > 1 in (a) or (b)

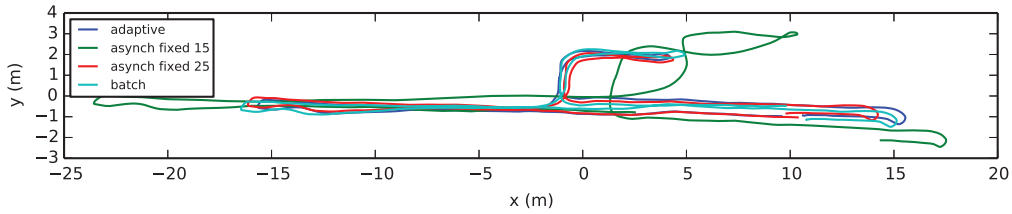

Fig. 8. Comparison of trajectories estimated by different BA configurations. It can be seen that the 25 keyframe asynchronous fixed-size window BA and the adaptive window BA both produce trajectories close to the batch solution, however the 15 keyframe fixed-size asynchronous fixed window BA diverges substantially from the batch solution.

to estimate past and present parameters on all datasets. Statistically, a threshold on the 10th percentile is rather strict, as only a small percentage of errors should reside in this range, and most errors should be higher. The experimentally obtained value of 10% could potentially be due to measurement covariances that are set too low, resulting in under-estimated Mahalanobis distances.

Keyframing is used as a means to both increase performance and to alleviate problems arising from a stationary camera. The keyframing approach is similar to that of Klein and Murray (2007). Heuristics are used over the total distance and rotation since the last keyframe, as well as the percentage of feature tracks still active. If any of the heuristics are met, a new keyframe is placed and the feature track measurements are inserted into the map. Images that are not keyframes are simply used for localization against the map. Note that IMU measurements are recorded irrespective of keyframing. The system runs at 30 frames per second on a 2.5 GHz Core i7 laptop with two threads. One thread running the synchronous estimator as described in Section 2.4 and the other running the AAC (Section 2.5).

3.2. Experiments

The first experiment consists of a ∼200 m outdoor loop captured with rig B with identical start and end positions. The trajectory and reconstruction are shown in Figure 6(a), where the final error between the start and end keyframe poses as a percentage of traveled distance is 0.33%. Extension 1 shows the captured images and estimated trajectory for this experiment. Figure 6(b) shows the asynchronous adaptive window size ratio, calculated by dividing the adaptive window size by the total number of keyframes. It can be observed that, near the beginning of the trajectory, the asynchronous window ratio does reach one, which signifies that the asynchronous estimator is solving the batch solution with all keyframes and measurements. However, this period is short lived, after which the size of the adaptive window settles to an approximately constant number of keyframes. The spikes in Figure 6(b) correspond to increases in the adaptive window size due to the condition in equation (38) being met. These spikes can be matched against Figure 7(a) and (b), which show the plots for $\alpha_{k_\mathcal{V}}$, and $\alpha_{k_\mathcal{I}}$ (as explained in Section 2.5), where the red line indicates the threshold after which the condition is met and the adaptive window size is expanded.

Figure 8 shows the trajectory reconstruction of a short indoor sequence obtained using rig A showing the comparison between the batch solution, which uses all keyframes and measurements, and several options for the asynchronous estimator. It is observed that the adaptive window size is the one that most closely matches the batch. While the fixed-size asynchronous window with 25-keyframes performs fairly well compared with the batch solution, the 15-keyframe fixed-size window clearly diverges. Rig A was also used to capture two outdoor datasets shown in Figures 9 and 10. Figure 9(a) shows the trajectory reconstruction superimposed over an aerial photo for a ∼200 m outdoor loop. It can be seen that the adaptive result closely matches the batch result, while a fixed-window optimization has significantly higher error. The batch translation error between the start and end keyframes for this trajectory is 0.71% of the traveled distance and 0.72% when using the AAC estimator. The window size ratio (similar to Figure 6(b)) for this dataset is shown in Figure 9(b), where it is observed that after an initialization period where the size ratio is frequently 1 (indicating a batch solution), the window size decreases to an approximately constant number.

Figure 10(a) shows the reconstructed trajectory for a ∼400m outdoor trajectory obtained using rig A. The 20-keyframe fixed-window optimization clearly diverges and is unable to estimate the trajectory, while the batch and AAC results match closely, with error of 1.33% and 1.42% respectively. Figure 10(b) shows the window size ratio for this trajectory, where, once again, after a short initialization period, the AAC window size remains approximately constant.

While the approach is observed to be accurate, a large discrepancy in accuracy is observed between datasets captured with the two different rigs in Section 3. Given that both rigs were calibrated with the same offline calibration tool, the discrepancy can be attributed to both the higher image quality of rig B due to the higher quality camera and much larger lens, and to the higher sampling rate for the IMU of rig B. These effects are also visible in the window size ratio plots between rigs A and B (Figure 6(b) versus Figure 9(b) or Figure 10(b)), where it can be seen that adaptive window expansion is much more prevalent in

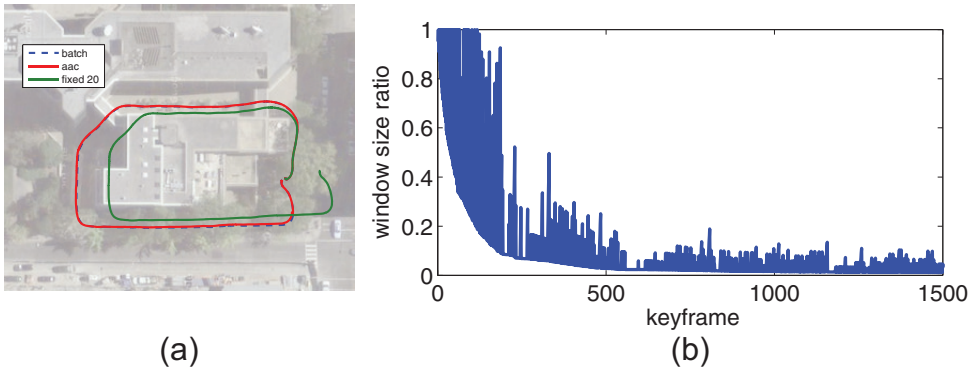

Fig. 9. A loop consisting of an outdoor ∼200 m dataset taken on foot with rig A, superimposed over satellite imagery. The batch translation error between the start and end keyframes as a percentage of traveled distance is 0.71% while the AAC error is 0.72%. The resulting poses obtained by running AAC, batch and a fixed-window optimization over the data are shown in (a), and the ratio of the AAC window to the total number of keyframes is shown in (b). A ratio of 1.0 indicates a batch solve. It can be seen that a fixed window optimization is unable to accurately match the batch solution.

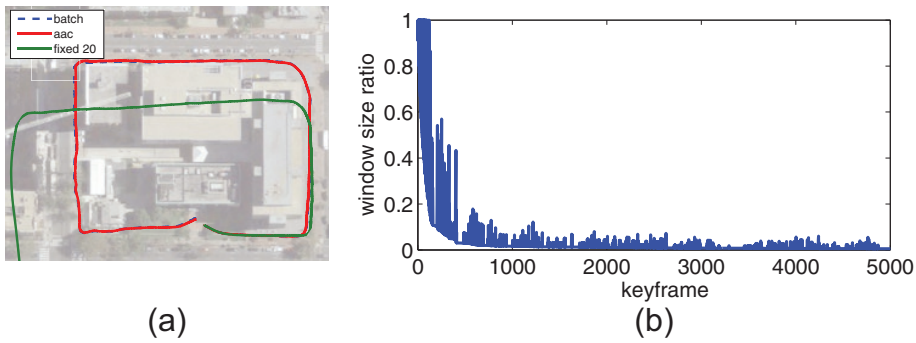

Fig. 10. A loop consisting of an outdoor ∼400 m dataset taken on foot with rig A, superimposed over satellite imagery. The batch translation error between the start and end keyframes as a percentage of traveled distance is 1.33% while the AAC error is 1.42. The resulting poses obtained by running AAC, batch and a fixed-window optimization over the data are shown in (a), and the ratio of the AAC window to the total number of keyframes is shown in (b). A ratio of 1.0 indicates a batch solve. It can be seen that the solution obtained using a fixed-window batch solver diverges completely.

Figure 9(b) and Figure 10(b), which were captured with rig A. This signifies that good parameter estimates were less likely to be computed given the minimum asynchronous window size, and frequent expansions were necessary. This is in contrast to the higher quality data obtained with rig B, and the resulting window size ratio shown in Figure 6(b). Another factor that has a large impact on accuracy is the value of the IMU measurement covariance matrix, $\Sigma_{\mathbf{z}_k}$. The weight placed on the IMU residual is derived from the propagation of this covariance matrix, and an underestimation of the uncertainty can lead to significant reduction of the system accuracy due to over-reliance on noisy IMU data.

3.3. Initialization

In Figures 6(b), 9(b) and 10(b) it can be observed that for a short time after the start of the sequence, the window size ratio tends to be equal to 1, indicating a batch solution. This behavior arises naturally due to the adaptive window formulation (Section 2.5) and serves to initialize values which may not be immediately observed at the beginning of the trajectory. Among the values that tend to fluctuate the most in this initialization phase are the accelerometer and gyroscope biases, the 2DOF of orientation with respect to gravity, and the velocity vectors for the keyframes. As these quantities are not immediately available, new information about them will *stress* the conditioning edges and force the AAC estimator to expand the window in order to update all affected poses. However, once the initial estimates for these parameters have converged, the AAC window seldom expands to the batch solution.

4. Discussion

It was observed that in real-life situations, parameters such as velocity, gravity and bias are observable with adaptive conditioning. This is of course contingent upon sufficient excitation of the sensors. In the hand-held datasets there

is an ever present oscillatory acceleration due to the walking motion, which quickly renders the unknown parameters observable. Given this, a short required window size is observed in order to closely estimate the batch MLE solution. As expected, window growth is also seen in situations where scale and consequently velocity are ambiguous. An example of this is at sharp turns that introduce a slew of uninitialized new landmarks while simultaneously cutting tracks from established landmarks. The net result is a scale ambiguity that requires a larger window size to resolve, which is automatically discovered.

It must be noted that in the case of prolonged degenerate motion (such as constant velocity), the size of the window required to properly estimate parameters can grow without bound to the batch solution. However, in the real datasets that were used to evaluate the system, such a scenario was never encountered, as enough excitation of the sensors was generally observed, and the window size was bounded.

The system is observed to be accurate, however consistent discrepancies in accuracy are observed between reconstructions captured with different visual–inertial rigs. Given equal offline calibration of the two rigs, the effect of sensor quality and IMU sampling frequency on the accuracy of the reconstruction is observed to be significant, and is a candidate for further study.

When using asynchronous BA, care must be taken so as to ensure sufficient update frequency of the asynchronous solution in order to ensure overlap with the synchronous BA. This is required to keep the synchronous BA in the overall solution basin as solved by the asynchronous BA. As expected from the relative framework, the updates to the edges and inverse depth parameters for landmarks are small and no interference was observed between the two threads, even when the asynchronous BA is solving a batch solution during initialization (Section 3.3), in which case its updates will be markedly slower than the synchronous BA.

The use of conditioning in lieu of marginalization presents desirable properties. However, it must be noted that marginal covariances are no longer computed for state estimates, and parameters that are conditioned upon are assumed to be correctly estimated. Since these marginal covariances are ignored, a loss of accuracy can generally be expected. In this case of adaptive conditioning, a concerted effort is to re-estimate past parameters in the presence of new measurements, constantly reducing their error where possible. As a result, it is shown that the adaptive conditioning method is still able to obtain very high accuracy, especially when using quality sensors.

5. Conclusions

Adaptive asynchronous conditioning (AAC) is a novel solution to real-time visual–inertial SLAM. The approach automatically scales and focuses computation to capture an accurate approximation to the batch MLE solution in both the keyframe poses and the map structure, and avoids the downsides associated with marginalization, such as incorrect linearization and inconsistency. Further, AAC avoids the computational difficulties associated with carrying prior distributions, such as the need to compute global optimizations at loop closure.

The proposed method offers a natural "front-end" while simultaneously allowing larger portions of the problem to influence the solution. It is thus able to produce estimates in real-time, and also re-estimate past parameters in the presence of new information: an ability that is useful for self-calibration, during degenerate motions, or when bias and gravity are poorly observed.

Funding

This work was supported by Google, The MITRE Corporation, and The Toyota Motor Corporation.

References

Engels C, Stewenius H and Nister D (2006) Bundle adjustment rules. In: *Photogrammetric computer vision, volume 2*, pp. 124–131. Amsterdam: Elsevier.

Fischler MA and Bolles RC (1981) Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* 24(6): 381–395.

Forster C, Pizzoli M and Scaramuzza D (2014) SVO: Fast semi-direct monocular visual odometry. In: *IEEE international conference on robotics and automation*, pp. 15–22. IEEE.

Gelb A (1974) *Applied Optimal Estimation*. Cambridge, MA: MIT Press.

Harris C and Stephens M (1988) A combined corner and edge detector. In: *Alvey vision conference, volume 15*, Manchester, UK, p. 50. Alvey Vision Club.

Hesch JA, Kottas DG, Bowman SL and Roumeliotis SI (2013) Camera–IMU-based localization: Observability analysis and consistency improvement. *The International Journal of Robotics Research* 33: 182–201.

Jones E, Vedaldi A and Soatto S (2007) Inertial structure from motion with autocalibration. In: *ICCV workshop on dynamical vision*. IEEE.

Kelly J and Sukhatme GS (2010) Visual–inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration. *The International Journal of Robotics Research* 30(1): 56–79.

Klein G and Murray D (2007) Parallel tracking and mapping for small AR workspaces. In: *6th IEEE and ACM international symposium on mixed and augmented reality (ISMAR 2007)*, pp. 1–10. Washington, DC: IEEE Computer Society Press.

Leutenegger S, Furgale PT, Rabaud V, Chli M, Konolige K and Siegwart R (2013) Keyframe-based visual–inertial SLAM using nonlinear optimization. In: *Robotics: Science and Systems*. Available at: http://www. roboticsproceedings.org/rss09/p37.pdf

Li M and Mourikis AI (2013a) 3-D motion estimation and online temporal calibration for camera–IMU systems. In: *Proceedings IEEE international conference on robotics and automation (ICRA)*, 5709–5716. IEEE.

Li M and Mourikis AI (2013b) High-precision, consistent EKF-based visual–inertial odometry. *The International Journal of Robotics Research* 32(6): 690–711.
Li M, Yu H, Zheng X and Mourikis AI (2014) High-fidelity sensor modeling and calibration in vision-aided inertial navigation. In: *IEEE international conference on robotics and automation*, Hong Kong, pp. 409–416. IEEE.
Lupton T and Sukkarieh S (2012) Visual–inertial-aided navigation for high-dynamic motion in built environments without initial conditions. *IEEE Transactions on Robotics* 28(1): 61–76.
Maybeck PS (1979) *Stochastic Models, Estimation, and Control* (*Mathematics in Science and Engineering*, Vol. 141). Boston, MA: Academic Press, Inc.
Mei C, Sibley G, Cummins M, Newman PM and Reid ID (2011) RSLAM: A system for large-scale mapping in constant-time using stereo. *International Journal of Computer Vision* 94(2): 198–214.
Mouragnon E, Lhuillier M, Dhome M, Dekeyse F and Sayd P (2006) Real time localization and 3D reconstruction. In: *Proceedings of computer vision and pattern recognition*. New York, USA, pp. 363–370. IEEE.
Mourikis AI and Roumeliotis SI (2007) A multi-state constraint Kalman filter for vision-aided inertial navigation. In: *IEEE international conference on robotics and automation*, pp. 3565–3572. IEEE.
Nerurkar ED, Wu KJ and Roumeliotis SI (2013) C-KLAM: Constrained keyframe localization and mapping for long-term navigation. In: *IEEE international conference on robotics and automation workshop on long-term autonomy*, pp. 3638–3643. IEEE.
Pietzsch T (2008) Efficient feature parameterisation for visual SLAM using inverse depth bundles. In: *British machine vision conference*, pp. 5.1–5.10. BMVA Press.
Powell M (1970) *Numerical Methods for Nonlinear Algebraic Equations*. New York: Gordon and Breach Science.
Sibley G (2006) *Sliding window filters for SLAM*. Report no. CRES-06-004, University of Southern California, Center for Robotics and Embedded Systems.
Sibley G, Matthies L and Sukhatme G (2010) Sliding window filter with applications to planetary landing. *Journal of Field Robotics* 27(5): 587–608.
Sibley G, Mei C, Ried I and Newman P (2009) Adaptive relative bundle adjustment. In: *Robotics: Science and Systems*, pp. 1–8. Cambridge MA: MIT Press.
Strasdat H (2012) *Local Accuracy and Global Consistency for Efficient Visual SLAM*. PhD Thesis, Imperial College London, UK.
Triggs B, McLauchlan P, Hartley R and Fitzgibbon A (2000) Bundle adjustment – a modern synthesis. In: *Vision Algorithms: Theory and Practice* (*Lecture Notes in Computer Science*, Vol. 1883), pp. 298–375. Berlin: Springer-Verlag.

Appendix A: Index to Multimedia Extension

Archives of IJRR multimedia extensions published prior to 2014 can be found at http://www.ijrr.org, after 2014 all videos are available on the IJRR YouTube channel at http://www.youtube.com/user/ijrrmultimedia

Table of Multimedia Extension

Extension	Media type	Description
1	Video	The approach outlined in the paper is applied to a 200 m outdoor dataset in order to estimate the trajectory of the camera as well as the sparse map. Data was captured using rig A at 1280x960@30 fps downsampled to 640x480@30 fps. The dataset starts and ends at the same location. Based on this, the estimated trajectory achieves an error of 0.33% as a percentage of distance traveled.

Appendix B

In this section, the IMU integration equations are expanded to factorize the starting parameter values where possible, similarly to the factorizations presented in Li and Mourikis (2013b) and Lupton and Sukkarieh (2012). The motivation behind this is to obviate the need to compute the derivatives with respect to the starting parameters by traversing the integration via the chain rule. A successful factorization allows the derivatives to be calculated after the integration is performed, both increasing the accuracy of the derivatives as well as increasing performance. It is shown that this factorization is possible for all starting parameters except for the IMU biases.

B.1. Rotation integration factorization

In the interest of brevity, the derivation of the factorized rotation equations is first performed using rotation matrices (denoted as $\mathbf{R}$). The factorized form can then easily be written using quaternions (equation (45)), which is the final form used in the implementation. Consequently, the exponential and adjunct operators performed on rotation matrices are denoted as $\exp_{\mathbf{R}}$ and $\mathrm{Ad}_{\mathbf{R}}$ to disambiguate them from their quaternion counterparts. The integration of angular velocities from time 0 to n can be written as

$$\mathbf{R}_{wp_n} = \exp_{\mathbf{R}}\left(\widehat{\omega}_{w_{n-1}}\,\mathrm{d}t_{n-1}\right) \times \cdots \times \exp_{\mathbf{R}}\left(\widehat{\omega}_{w_0}\,\mathrm{d}t_0\right)\mathbf{R}_{wp_0} \tag{40}$$

where $\mathbf{R}_{wp_0}$ is the rotation matrix describing the starting rotation. Since this rotation is in fact a parameter in the optimization, derivatives with respect to it will be required in order to construct the problem Jacobian. Since the measurement in world coordinates depends on the orientation of the IMU frame, its formulation must be taken into account to obtain the derivative. Fortunately, the starting orientation can be factored from the integration. To do this, a slightly different form of the adjunct is used where: $\widehat{\mathrm{Ad}_{\mathbf{R}}\cdot\omega} = \mathrm{Ad}_{\mathbf{R}}\cdot\widehat{\omega}\cdot\mathrm{Ad}_{\mathbf{R}}^{-1}$ (Strasdat, 2012). Since for SO(3), $\mathrm{Ad}_{\mathbf{R}} = \mathbf{R}$,

equation (22) can be rewritten as

$$\begin{aligned}\mathbf{R}_{wp_{k+1}} &= \exp_{\mathbf{R}}(\hat{\omega}_{w_k}\,\mathrm{d}t)\,\mathbf{R}_{wp_k}\\ &= \exp_{\mathbf{R}}\left(\mathbf{R}_{wp}\widehat{(\omega_{m_k}-\mathbf{b}_{g_k})}\,\mathrm{d}t\mathbf{R}_{wp}^{-1}\right)\mathbf{R}_{wp_k}\end{aligned} \tag{41}$$

which gives the slightly modified form of equation (40):

$$\begin{aligned}\mathbf{R}_{wp_n} = \exp_{\mathbf{R}}&\left(\mathbf{R}_{wp_{n-1}}\widehat{(\omega_{m_{n-1}}-\mathbf{b}_{g_{n-1}})}\,\mathrm{d}t_{n-1}\mathbf{R}_{wp_{n-1}}^{T}\right)\\ &\times\cdots\times\exp_{\mathbf{R}}\left(\mathbf{R}_{wp_0}\widehat{(\omega_{m_0}-\mathbf{b}_{g_0})}\,\mathrm{d}t_0\mathbf{R}_{wp_0}^{-1}\right)\mathbf{R}_{wp_0}\end{aligned} \tag{42}$$

Since $\mathbf{R}_{wp_{n-1}}$ is the orientation as integrated up to step $n-1$, it can be factored into the starting orientation $\mathbf{R}_{wp_0}$, and an integrated orientation delta up to time-step $n-1$, $\mathbf{R}_{p_0p_{n-1}}$. More formally: $\mathbf{R}_{wp_n} = \mathbf{R}_{wp_0}\mathbf{R}_{p_0p_n}$. Using this factorization, equation (42) becomes

$$\begin{aligned}\mathbf{R}_{wp_n} = \exp_{\mathbf{R}}&\left(\mathbf{R}_{wp_0}\mathbf{R}_{p_0p_{n-1}}\widehat{(\omega_{m_{n-1}}-\mathbf{b}_{g_{n-1}})}\,\mathrm{d}t_{n-1}\mathbf{R}_{p_0p_{n-1}}^{-1}\mathbf{R}_{wp_0}^{-1}\right)\\ &\times\cdots\times\exp_{\mathbf{R}}\left(\mathbf{R}_{wp_0}\widehat{(\omega_{m_k}-\mathbf{b}_{g_k})}\,\mathrm{d}t_0\mathbf{R}_{wp_0}^{-1}\right)\mathbf{R}_{wp_0}\end{aligned} \tag{43}$$

Using the Lie matrix exponential identity $\exp_{\mathbf{R}}(\mathbf{AXA}^{-1}) = \mathbf{A}\exp_{\mathbf{R}}(\mathbf{X})\,\mathbf{A}^{-1}$, equation (42) becomes

$$\begin{aligned}\mathbf{R}_{wp_n} = \mathbf{R}_{wp_0}&\mathbf{R}_{p_0p_{n-1}}\exp_{\mathbf{R}}\left(\widehat{(\omega_{m_{n-1}}-\mathbf{b}_{g_{n-1}})}\,\mathrm{d}t_{n-1}\right)\\ &\mathbf{R}_{p_0p_{n-1}}^{-1}\mathbf{R}_{wp_0}^{-1}\times\cdots\times\mathbf{R}_{wp_0}\\ &\exp_{\mathbf{R}}\left(\widehat{(\omega_{m_k}-\mathbf{b}_{g_k})}\,\mathrm{d}t_0\right)\mathbf{R}_{wp_0}^{-1}\mathbf{R}_{wp_0}\end{aligned} \tag{44}$$

Canceling out terms, and using quaternions to represent the rotation matrix $\mathbf{R}$, the final form of the integration is obtained:

$$\begin{aligned}\mathbf{q}_{wp_n} = \mathbf{q}_{wp_0}\otimes\Bigg[&\mathbf{q}_{p_0p_{n-1}}\otimes\exp_{\mathbf{q}}\left(\widehat{(\omega_{m_{n-1}}-\mathbf{b}_{g_{n-1}})}\,\mathrm{d}t_{n-1}\right)\otimes\\ &\mathbf{q}^{-1}{}_{p_0p_{n-1}}\otimes\ldots\otimes\exp\left(\widehat{(\omega_{m_k}-\mathbf{b}_{g_k})}\,\mathrm{d}t_0\right)\Bigg]\\ &= \mathbf{q}_{wp_0}\otimes\Delta\mathbf{q}\end{aligned} \tag{45}$$

where $\Delta\mathbf{q}$ represents the integration of bias-corrected angular velocities starting from an identity orientation, and does not depend on $\mathbf{q}_{wp_0}$. The derivative of $\partial\mathbf{q}_{wp_n}/\partial\mathbf{q}_{wp_0}$ can therefore be computed without the need to use the chain rule through the integration.

B.2. Velocity integration factorization

Similarly to Appendix B.1, the final integrated velocity is obtained by integrating the accelerometer measurements in the world frame from time 0 to n:

$$\begin{aligned}\mathbf{v}_{w_n} &= \mathbf{v}_{w_0} + \mathbf{a}_{w_0}\,\mathrm{d}t_0 + \cdots + \mathbf{a}_{w_{n-1}}\,\mathrm{d}t_{n-1}\\ &= \mathbf{v}_{w_0} + \left(\mathbf{q}_{wp_0}\otimes\left(\mathbf{a}_{m_0}-\mathbf{b}_{a_0}\right)+\mathbf{g}_w\right)\,\mathrm{d}t_0\\ &\quad+\cdots+\left(\mathbf{q}_{wp_{n-1}}\otimes\left(\mathbf{a}_{m_{n-1}}-\mathbf{b}_{a_{n-1}}\right)+\mathbf{g}_w\right)\,\mathrm{d}t_{n-1}\end{aligned} \tag{46}$$

where accelerometer measurements in the IMU frame are rotated into the world frame as per equation (21). It is immediately obvious that the effect of the initial velocity $\mathbf{v}_{w_0}$ is already factorized. Further inspection reveals that the gravity term is always added in the world frame, and so it can be factorized out as well:

$$\begin{aligned}\mathbf{v}_{w_n} = \mathbf{v}_{w_0} + \mathbf{g}_w\Delta t + \mathbf{q}_{wp_0}\otimes\left(\mathbf{a}_{m_0}-\mathbf{b}_{a_0}\right)\,\mathrm{d}t_0+\cdots\\ +\mathbf{q}_{wp_{n-1}}\otimes\left(\mathbf{a}_{m_{n-1}}-\mathbf{b}_{a_{n-1}}\right)\,\mathrm{d}t_{n-1}\end{aligned} \tag{47}$$

Similarly to the operation applied to equation (42), the rotation matrix taking the accelerometer measurements from the IMU to world coordinates can be decomposed as $\mathbf{R}_{wp_n} = \mathbf{R}_{wp_0}\mathbf{R}_{p_0p_n}$, allowing the factorization of the initial rotation, $\mathbf{R}_{wp_0}$:

$$\begin{aligned}\mathbf{v}_{w_n} &= \mathbf{v}_{w_0} + \mathbf{g}_w\Delta t + \mathbf{q}_{wp_0}\otimes\left(\mathbf{a}_{m_0}-\mathbf{b}_{a_0}\right)\,\mathrm{d}t_0+\cdots+\\ &\quad\mathbf{q}_{wp_0}\otimes\mathbf{q}_{p_0p_{n-1}}\otimes\left(\mathbf{a}_{m_{n-1}}-\mathbf{b}_{a_{n-1}}\right)\,\mathrm{d}t_{n-1}\\ &= \mathbf{v}_{w_0} + \mathbf{g}_w\Delta t + \mathbf{q}_{wp_0}\otimes\Bigg[\left(\mathbf{a}_{m_0}-\mathbf{b}_{a_0}\right)\,\mathrm{d}t_0+\cdots+\\ &\quad\mathbf{q}_{p_0p_{n-1}}\otimes\left(\mathbf{a}_{m_{n-1}}-\mathbf{b}_{a_{n-1}}\right)\,\mathrm{d}t_{n-1}\Bigg]\end{aligned} \tag{48}$$

where the term in square brackets represents the integration of the bias-corrected accelerometer measurements rotated into the identity reference frame and not corrected for gravity. Referring to this term as $\Delta\mathbf{v}$, the final integrated velocity is formulated as follows:

$$\mathbf{v}_{w_n} = \mathbf{v}_{w_0} + \mathbf{g}_w\Delta t + \mathbf{q}_{wp_0}\otimes\Delta\mathbf{v}$$

B.3. Translation integration factorization

As with Appendices B.1 and B.2, the final position is obtained by integrating velocities obtained from the integration in Appendix B.2:

$$\mathbf{p}_{wp_n} = \mathbf{p}_{wp_0} + \mathbf{v}_{w_0}\,\mathrm{d}t_0 + \cdots + \mathbf{v}_{w_{n-1}}\,\mathrm{d}t_{n-1} \tag{49}$$

The velocity terms can be substituted with the integrated accelerometer measurements as per Appendix B.2:

$$\begin{aligned}\mathbf{p}_{wp_n} &= \mathbf{p}_{wp_0} + \mathbf{v}_{w_0}\,\mathrm{d}t_0 + \left(\mathbf{v}_{w_0}+\mathbf{a}_{w_0}\,\mathrm{d}t_0\right)\,\mathrm{d}t_1+\cdots+\\ &\quad\left(\mathbf{v}_{w_0}+\mathbf{a}_{w_0}\,\mathrm{d}t_0+\cdots+\mathbf{a}_{w_{n-2}}\,\mathrm{d}t_{n-2}\right)\,\mathrm{d}t_{n-1}\\ &= \mathbf{p}_{wp_0} + \mathbf{v}_{w_0}\Delta t + \left(\mathbf{a}_{w_0}\,\mathrm{d}t_0\right)\,\mathrm{d}t_1+\cdots+\\ &\quad\left(\mathbf{a}_{w_0}\,\mathrm{d}t_0+\cdots+\mathbf{a}_{w_{n-2}}\,\mathrm{d}t_{n-2}\right)\,\mathrm{d}t_{n-1}\end{aligned} \tag{50}$$

where it is immediately observed that the initial velocity ($\mathbf{v}_{w_0}$) can be factored out since $\mathbf{v}_{w_0}\,\mathrm{d}t_0 + \mathbf{v}_{w_0}\,\mathrm{d}t_1 + \cdots + \mathbf{v}_{w_0}\,\mathrm{d}t_n = \mathbf{v}_{w_0}\Delta t$. The accelerometer measurements in the world frame ($\mathbf{a}_w$) can be replaced by the IMU frame measurements as per Appendix B.2:

$$\begin{aligned}\mathbf{p}_{wp_n} = \mathbf{p}_{wp_0} + \mathbf{v}_{w_0}\Delta t + \left(\mathbf{q}_{wp_0} \otimes \left(\mathbf{a}_{m_0} - \mathbf{b}_{a_0}\right) + \mathbf{g}_w\right)\,\mathrm{d}t_0\,\mathrm{d}t_1 \\ + \cdots + \left(\left(\mathbf{q}_{wp_0} \otimes \left(\mathbf{a}_{m_0} - \mathbf{b}_{a_0}\right) + \mathbf{g}_w\right)\,\mathrm{d}t_0 \\ + \cdots + \left(\mathbf{q}_{wp_{n-1}} \otimes \left(\mathbf{a}_{m_{n-2}} - \mathbf{b}_{a_{n-2}}\right) + \mathbf{g}_w\right)\mathrm{d}t_{n-2}\right)\,\mathrm{d}t_{n-1}\end{aligned} \tag{51}$$

Since the gravity term is constant, and is double integrated in the global coordinate system, it can be replaced by its closed form:

$$\begin{aligned}\mathbf{p}_{wp_n} = \mathbf{p}_{wp_0} + \mathbf{v}_{w_0}\Delta t + \frac{1}{2}\mathbf{g}_w\Delta t^2 + \left(\mathbf{q}_{wp_0} \otimes \left(\mathbf{a}_{m_0} - \mathbf{b}_{a_0}\right)\right) \\ \mathrm{d}t_0\,\mathrm{d}t_1 + \cdots + \left(\left(\mathbf{q}_{wp_0} \otimes \left(\mathbf{a}_{m_0} - \mathbf{b}_{a_0}\right)\right)\,\mathrm{d}t_0 \\ + \cdots + \left(\mathbf{q}_{wp_{n-1}} \otimes \left(\mathbf{a}_{m_{n-2}} - \mathbf{b}_{a_{n-2}}\right)\right)\,\mathrm{d}t_{n-2}\right)\,\mathrm{d}t_{n-1}\end{aligned} \tag{52}$$

As per Appendix B.1, the rotation term $\mathbf{R}_{wp_n}$ which is used to rotate the accelerometer measurements from the IMU to the world frame can be decomposed as follows: $\mathbf{R}_{wp_n} = \mathbf{R}_{wp_0}\mathbf{R}_{p_0p_n}$. Using this equation (52) can be re-written as follows:

$$\begin{aligned}\mathbf{p}_{wp_n} = \mathbf{p}_{wp_0} + \mathbf{v}_{w_0}\Delta t + \frac{1}{2}\mathbf{g}_w\Delta t^2 \\ + \mathbf{q}_{wp_0} \otimes \Bigg[\left(\mathbf{a}_{m_0} - \mathbf{b}_{a_0}\right)\,\mathrm{d}t_0\,\mathrm{d}t_1 + \cdots + \\ \left(\left(\mathbf{a}_{m_0} - \mathbf{b}_{a_0}\right)\,\mathrm{d}t_0 + \cdots + \\ \left(\mathbf{q}_{p_0p_{n-1}} \otimes \left(\mathbf{a}_{m_{n-2}} - \mathbf{b}_{a_{n-2}}\right)\right)\,\mathrm{d}t_{n-2}\right)\,\mathrm{d}t_{n-1}\Bigg]\end{aligned} \tag{53}$$

where the term in the square brackets represents the double integration of the accelerometer measurements rotated into the identity reference frame and without gravity compensation. Referring to this term as $\Delta\mathbf{p}$, the final integrated velocity is formulated as follows:

$$\mathbf{p}_{wp_n} = \mathbf{p}_{wp_0} + \mathbf{v}_{w_0}\Delta t + \frac{1}{2}\mathbf{g}_w\Delta t^2 + \mathbf{q}_{wp_0} \otimes \Delta\mathbf{p} \tag{54}$$