Imu Pre-Integration Strategies

Luc Oth

July 4, 2016

1 Introduction

The present document describes different integration methods and evaluates their performance with respect to:

- robustness to integration drift under noisy data
- robustness of noise propagation under noisy data
- computational efficiency of the implementation

To evaluate the above, we generated random B-spline trajectories on which the accelerometer and gyroscope measurements were sampled. The samples are corrupted with white Gaussian noise.

In the following sections, we will be integrating noisy measurements from an IMU:

$$ {}_B\tilde{\boldsymbol{\omega}}_{WB}(t) = {}_B\,\boldsymbol{\omega}_{WB}(t) + \mathbf{b}^g(t) + \boldsymbol{\eta}^g(t), \tag{1.1} $$

$$ {}_B\tilde{\boldsymbol{a}}_{WB}(t) = \mathbf{R}^T_{WB}(t)({}_W\boldsymbol{a}(t) - {}_W\,\mathbf{g}) + \mathbf{b}^a(t) + \boldsymbol{\eta}^a(t), \tag{1.2} $$

where ${}_B\tilde{\boldsymbol{\omega}}_{WB}(t)$ is the rotational velocity observed in the frame of the IMU, ${}_B\mathbf{a}_{WB}$ is the acceleration observed in the body frame, $\mathbf{b}$, η are the bias and noise corrupting the measurements. ${}_W\mathbf{g}$ is the gravity vector. For readability reasons, we will rely on the sloppy notation $\boldsymbol{\omega}(t) \leftarrow_B \tilde{\boldsymbol{\omega}}_{WB}(t)$ and $\mathbf{a}(t) \leftarrow_B \tilde{\boldsymbol{a}}_{WB}(t)$. Please take note, that a proper integration method operates on the bias and gravity corrected measurements. We will be sampling the measurements at a fixed or dynamic sampling time referred to as Δt.

As the standard theory of $\mathcal{R}_3$ applies for the integration of the position and velocity given the measured acceleration, we will focus on the integration theory of the rotational components which has to preserve the structure of the manifold $\mathfrak{so}3$.

In order to propagate the noise, we rely on the covariance of the measurements which is typically given as a covariance matrix of a continuous-time process e.g. $\boldsymbol{\Sigma}^g$ and the associated $\boldsymbol{\eta}^g(t)$. In order to apply a noise propagation in discrete time, the continuous-time specification has to be transferred to its discrete-time equivalent:

$$ \boldsymbol{\Sigma}^{gd} = \frac{1}{\Delta t}\boldsymbol{\Sigma}^g, \tag{1.3} $$

with the associated discrete time random process: $\boldsymbol{\eta}^{gd}_i$.

2 On-Manifold

2.1 Preliminaries

We will use an intuitive derivation of the integration of rotational velocities on $\mathfrak{so}3$.

Given a rotational velocity ${}_B\boldsymbol{\omega}_{WB}$, the associated rotation can be expressed (approximately for small values of Δt) as:

$$\delta\mathbf{R}(t,\Delta t) = Exp\left(\int_t^{t+\Delta t} \boldsymbol{\omega}(\tau)d\tau\right). \tag{2.1}$$

It is well known, that composing two rotations parametrized as rotation matrices, $\mathbf{R}_{AB}$, $\mathbf{R}_{BC}$, is simply:

$$\mathbf{R}_{AC} = \mathbf{R}_{AB}\mathbf{R}_{BC}, \tag{2.2}$$

where the multiplication is a plain matrix product. An integration method that preserves the properties of $\mathfrak{so}3$ can thus be defined as infinitesimal compositions of:

$$\mathbf{R}_{WB}(t+\Delta t) = \mathbf{R}_{WB}(t)\delta\mathbf{R}(t,\Delta t). \tag{2.3}$$

2.2 Angular Velocity Integration

Forward Integration We take the concepts of equation (5.2) applied to (2.3). The increment $\delta\mathbf{R}$ at time t_i becomes:

$$\delta\mathbf{R}(t_i) = Exp\left(\Delta t\boldsymbol{\omega}_i\right). \tag{2.4}$$

Midward Integration With the concepts of equation (5.4) instead of (5.2) we obtain the midward integration on the $\mathfrak{so}3$:

$$\delta\mathbf{R}(t_i) = Exp\left(\frac{1}{2}\Delta t(\boldsymbol{\omega}_i + \boldsymbol{\omega}_{i+1})\right). \tag{2.5}$$

2.3 Noise propagation

The propagation of the noise $\delta\boldsymbol{\phi}_{ij}$ (from measurement j to measurement i) for the forward on-manifold integration is derived in [3]. It is only an approximation as a series of linearization steps are taken in the derivations:

$$\delta\boldsymbol{\phi}_{ij} \approx \Delta\tilde{\mathbf{R}}_{j-1j}^T\delta\boldsymbol{\phi}_{ij-1} + \mathbf{J}_r^{j-1}\boldsymbol{\eta}_{j-1}^{gd}\Delta t. \tag{2.6}$$

$\Delta\tilde{\mathbf{R}}_{j-1j}$ is the change in rotation since the previous integration step (e.g. $\delta\mathbf{R}$ in (2.4)), $\mathbf{J}_r^{j-1}$ is the right-Jacobian of the exponential map from $\mathfrak{so}3$ onto $SO3$.

Although the derivations in [3] base on a forward integration principle, it is straightforward to show that the results remain valid for a midward integration method using (2.5).

Please note, that the Jacobian $\mathbf{J}_r^{j-1} \approx \mathbf{I}_3$ for $\delta\mathbf{R} \approx \mathbf{I}_3$ resp. $\Delta t\boldsymbol{\omega} \approx \mathbf{0}_3$. As we assume an integration at a rather high frequency, this computationally expensive step can easily be left out. In the evaluations we did not see any difference between the two approaches.

3 Quaternion

The derivations below are for Hamiltonian Quaternions (right handed with $\mathbf{q} = [q_w, \mathbf{q}]$, passive). Local perturbations appear on the right hand side: $\tilde{\mathbf{q}} = \mathbf{q} \oplus \Delta\mathbf{q}$. The same convention is used in [1] where most of the below derivations are available in detail.

3.1 Preliminaries

3.1.1 Time-Derivative of Quaternions (Local)

$$\dot{\mathbf{q}} = \frac{1}{2}\mathbf{q} \otimes \begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix} \tag{3.1}$$

with $\boldsymbol{\omega}$ an infinitesimal perturbation (e.g. angular rates). Defining an $\boldsymbol{\Omega}(\boldsymbol{\omega})$, specific to the Hamiltonian quaternion representation:

$$\boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\boldsymbol{\omega}^T \\ \boldsymbol{\omega} & -[\boldsymbol{\omega}]_\times \end{bmatrix}, \tag{3.2}$$

we get:

$$\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega})\mathbf{q} = \frac{1}{2}\mathbf{q} \otimes \boldsymbol{\omega}, \tag{3.3}$$

and it follows that

$$\dot{\mathbf{R}} = \mathbf{R}\,[\boldsymbol{\omega}]_\times\,. \tag{3.4}$$

In the following, $\mathbf{q}\{\cdot\}$ is used to refer to a function that converts a rotation vector representation ($\mathbf{v} = \phi\mathbf{u}$) to a quaternion. One way to achieve this is to use the exponential map $\mathbf{q} = \exp\left(\frac{\mathbf{v}}{2}\right)$.

3.2 Angular Velocity Integration

An robust and easy way to derive an integration method for quaternions is to use the time-derivative in (3.3) and develop the Taylor-series expansion for $\mathbf{q}_{i+1}$:

$$\mathbf{q}_{i+1} = \mathbf{q}_i + \dot{\mathbf{q}}_i\Delta t + \frac{1}{2}\ddot{\mathbf{q}}_i\Delta t^2 + \mathcal{O}(\dddot{\boldsymbol{q}}). \tag{3.5}$$

3.2.1 Zeroth Order

Assume that $\boldsymbol{\omega} = \dot{\mathbf{q}}$ is piece-wise constant (all higher order derivatives are $\mathbf{0}$), a well trained eye will be able to identify the Taylor series expansion of the exponential map in $\boldsymbol{\omega}\Delta t$. Thus, follow the below zeroth order integration methods.

Forward

$$\mathbf{q}_{i+1} = \mathbf{q}_i \oplus \mathbf{q}\{\boldsymbol{\omega}_i\Delta t\} \tag{3.6}$$

Backward

$$\mathbf{q}_{i+1} = \mathbf{q}_i \oplus \mathbf{q}\{\boldsymbol{\omega}_{i+1}\Delta t\} \tag{3.7}$$

Midward

$$\mathbf{q}_{i+1} = \mathbf{q}_i \oplus \mathbf{q}\{\frac{\boldsymbol{\omega}_{i+1} + \boldsymbol{\omega}_i}{2}\Delta t\} \tag{3.8}$$

All zeroth order quaternion integration methods respect the structure of the quaternion manifold as it is a simple composition of a quaternion with another, infinitesimal, quaternion.

3.2.2 First Order

We assume $\boldsymbol{\omega} = \dot{\mathbf{q}}$ to be piece-wise linear, thus $\ddot{\mathbf{q}}$ constant. Furthermore we can easily calculate the derivative of $\boldsymbol{\omega}$: $\dot{\boldsymbol{\omega}} = \ddot{\mathbf{q}} = \dfrac{\boldsymbol{\omega}_{i+1} - \boldsymbol{\omega}_i}{\Delta t}$. With some math and again using the Taylor series expansion (3.5):

$$\mathbf{q}_{n+1} \approx \mathbf{q}_n \oplus \mathbf{q}\{\bar{\omega}\Delta t\} + \frac{\Delta t^2}{24}\mathbf{q}_n \oplus \begin{bmatrix} 0 \\ \boldsymbol{\omega}_n \times \boldsymbol{\omega}_{n+1} \end{bmatrix} \tag{3.9}$$

where the first term is equivalent to the zeroth order midwards integration.

According to [1] the order of magnitude of the second term is around $10^{-6}||\boldsymbol{\omega}||$ and often smaller. While the zeroth order integration methods preserve the structure of the quaternion manifold, the sum in the first order integration does not. The norm of the resulting quaternion should be checked regularly and re-normalized if required.

3.2.3 Runge-Kutta

Section 5.2 shows the Runge-Kutta method to integrate a generic function $\dot{\mathbf{x}} = f(\mathbf{x}, t)$, which in the case of quaternions becomes:

$$\dot{\mathbf{q}} = f(\mathbf{q}(t), \boldsymbol{\omega}(t)) = \frac{1}{2}\mathbf{q} \oplus \boldsymbol{\omega} = \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega})\mathbf{q}. \tag{3.10}$$

In the case of a Hamiltonian parametrization of the quaternion,

$$\boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -w_0 & -w_1 & -w_2 \\ w_0 & 0 & w_2 & -w1 \\ w_1 & -w_2 & 0 & w_0 \\ w_2 & w_1 & -w_0 & 0 \end{bmatrix} \tag{3.11}$$

It is expected that both measurements $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_{i+1}$ are available. A $\tilde{}$ is used to refer to temporary values of $\mathbf{q}$.

Third Order Applying the Runge-Kutta algorithm to our measurement model with quaternions results in

$$\begin{aligned}
&\mathbf{k}_1 = \boldsymbol{\omega}_i && \dot{\mathbf{q}}_1 = f(\mathbf{q}_i, \mathbf{k}_1) && \tilde{\mathbf{q}}_1 = \mathbf{q}_i + \frac{1}{2}\Delta t\dot{\mathbf{q}}_1 \\
&\mathbf{k}_2 = \boldsymbol{\omega}_i + \frac{1}{2}(\boldsymbol{\omega}_{i+1} - \boldsymbol{\omega}_i) && \dot{\mathbf{q}}_2 = f(\tilde{\mathbf{q}}_1, \mathbf{k}_2) && \tilde{\mathbf{q}}_2 = \mathbf{q}_i + \Delta t\,(-\dot{\mathbf{q}}_1 + 2\dot{\mathbf{q}}_2) \\
&\mathbf{k}_3 = \boldsymbol{\omega}_{i+1} && \dot{\mathbf{q}}_3 = f(\tilde{\mathbf{q}}_2, \mathbf{k}_3). &&
\end{aligned}$$

The above notation is not entirely consistent with section 5.2, where $\mathbf{k}$ is what is here called $\dot{\mathbf{q}}$, and $\mathbf{k}$ represents an intermediate step to approximate the rotational velocity in between $\boldsymbol{w}_i$ and $\boldsymbol{w}_{i+1}$.

The final result is obtained by combining all intermediate results:

$$\mathbf{q}_{i+1} = \mathbf{q}_i + \frac{1}{6}\Delta t \left(\dot{\mathbf{q}}_1 + 4\dot{\mathbf{q}}_2 + \dot{\mathbf{q}}_3\right). \tag{3.12}$$

Fourth Order Similar to the third order case, we apply the Runge-Kutta algorithm up to fourth order:

$$\begin{aligned}
&\mathbf{k}_1 = \boldsymbol{\omega}_i && \dot{\mathbf{q}}_1 = f(\mathbf{q}_i, \mathbf{k}_1) && \tilde{\mathbf{q}}_1 = \mathbf{q}_i + \frac{1}{2}\Delta t \dot{\mathbf{q}}_1 \\
&\mathbf{k}_2 = \boldsymbol{\omega}_i + \frac{1}{2}\left(\boldsymbol{\omega}_{i+1} - \boldsymbol{\omega}_i\right) && \dot{\mathbf{q}}_2 = f(\tilde{\mathbf{q}}_1, \mathbf{k}_2) && \tilde{\mathbf{q}}_2 = \mathbf{q}_i + \frac{1}{2}\Delta t \dot{\mathbf{q}}_2 \\
&\mathbf{k}_3 = \boldsymbol{\omega}_i + \frac{1}{2}\left(\boldsymbol{\omega}_{i+1} - \boldsymbol{\omega}_i\right) && \dot{\mathbf{q}}_3 = f(\tilde{\mathbf{q}}_2, \mathbf{k}_3) && \tilde{\mathbf{q}}_2 = \mathbf{q}_i + \Delta t \dot{\mathbf{q}}_3 \\
&\mathbf{k}_4 = \boldsymbol{\omega}_{i+1} && \dot{\mathbf{q}}_4 = f(\tilde{\mathbf{q}}_3, \mathbf{k}_4), &&
\end{aligned}$$

and finally

$$\mathbf{q}_{i+1} = \mathbf{q}_i + \frac{1}{6}\Delta t \left(\dot{\mathbf{q}}_1 + 2\dot{\mathbf{q}}_2 + 2\dot{\mathbf{q}}_3 + \dot{\mathbf{q}}_4\right). \tag{3.13}$$

Neither the linear interpolation of the quaternion derivatives, nor the summations preserve the quaternion manifold - although they are valid approximations for small values. To obtain valid results, $\mathbf{q}_{i+1}$ has to be re-normalized in every step.

3.2.4 Crouch-Grossman

A major drawback of a Runge-Kutta integration is that it is designed for values that evolve in $\mathcal{R}_4$. Quaternions, however, only evolve on a subspace of $\mathcal{R}_4$, namely the unit sphere. The Crouch-Grossman Lie Group method defines the Runge-Kutta method directly on the $\mathfrak{so}3$ manifold. A basic introduction is available in [2] and the thorough derivations in [4]. The generalized formulas for arbitrary orders are also available in the literature, we restrict ourselves to the third and fourth order.

Third Order

$$\begin{aligned}
\mathbf{k}_1 &= \boldsymbol{\omega}_i \\
\mathbf{k}_2 &= \boldsymbol{\omega}_i + \frac{3}{4}\left(\boldsymbol{\omega}_{i+1} - \boldsymbol{\omega}_i\right) \\
\mathbf{k}_2 &= \boldsymbol{\omega}_i + \frac{17}{24}\left(\boldsymbol{\omega}_{i+1} - \boldsymbol{\omega}_i\right)
\end{aligned}$$

$$\mathbf{q}_{i+1} = \mathbf{q}_i \exp\left(\frac{13}{51}\Delta t \mathbf{k}_1\right) \exp\left(-\frac{2}{3}\Delta t \mathbf{k}_2\right) \exp\left(\frac{24}{17}\Delta t \mathbf{k}_3\right). \tag{3.14}$$

Fourth Order The fourth order method requires a total of five steps. The determination of the coefficients it not trivial and only numerical values are available.

$$
\begin{aligned}
\mathbf{k}_1 &= \boldsymbol{\omega}_i \\
\mathbf{k}_2 &= \boldsymbol{\omega}_i + c_2 \left(\boldsymbol{\omega}_{i+1} - \boldsymbol{\omega}_i\right) \\
\mathbf{k}_2 &= \boldsymbol{\omega}_i + c_3 \left(\boldsymbol{\omega}_{i+1} - \boldsymbol{\omega}_i\right) \\
\mathbf{k}_2 &= \boldsymbol{\omega}_i + c_4 \left(\boldsymbol{\omega}_{i+1} - \boldsymbol{\omega}_i\right) \\
\mathbf{k}_2 &= \boldsymbol{\omega}_i + c_5 \left(\boldsymbol{\omega}_{i+1} - \boldsymbol{\omega}_i\right)
\end{aligned}
$$

$$
\mathbf{q}_{i+1} = \mathbf{q}_i \exp\left(b_1 \Delta t \mathbf{k}_1\right) \exp\left(b_2 \Delta t \mathbf{k}_2\right) \exp\left(b_3 \Delta t \mathbf{k}_3\right) \exp\left(b_4 \Delta t \mathbf{k}_4\right) \exp\left(b_5 \Delta t \mathbf{k}_5\right). \tag{3.15}
$$

With the coefficients b_i and c_i as below:

$$
\begin{aligned}
b_1 &= 0.1370831520630755 & c_1 &= 0 \\
b_2 &= -0.0183698531564020 & c_2 &= 0.8177227988124852 \\
b_3 &= 0.7397813985370780 & c_3 &= 0.3859740639032449 \\
b_4 &= -0.1907142565505889 & c_4 &= 0.3242290522866937 \\
b_5 &= 0.3322195591068374 & c_5 &= 0.8768903263420429.
\end{aligned}
$$

3.3 Noise Propagation

As we did not observe any significant differences between our implementations of noise propagation, we restrict our evaluations to the zeroth order, and third and fourth order Runge-Kutta methods.

Define a rotational error state, $\delta\boldsymbol{\theta}$, such that $\delta\mathbf{R} \approx \mathbf{I}_3 + [\delta\boldsymbol{\theta}]_\times$ resp. $\delta\boldsymbol{\theta} \approx \begin{bmatrix} 1 & \delta\boldsymbol{\theta}/2 \end{bmatrix}^T$. We also define a function $\mathbf{R}\{\cdot\}$ that, similar to $\mathbf{q}\{\cdot\}$, converts its arbitrary argument to a rotation matrix.

The noise covariance, $\mathbf{Q}_\eta$, is propagated under a Gaussian noise assumption in a linear system following a basic Kalman Filter principle:

$$
\boldsymbol{\Sigma}_{i+1} = \mathbf{A}\boldsymbol{\Sigma}_i\mathbf{A}^T + \mathbf{Q}_\eta, \tag{3.16}
$$

$\mathbf{A}$ describes how small changes of the state $\mathbf{x}$ are propagated. For a linear system it takes the form

$$
\mathbf{x}_{i+1} = \mathbf{A}\mathbf{x}_i. \tag{3.17}
$$

The noise covariance $\mathbf{Q}_\eta$ is the integrated discrete-time sensor noise during the time interval Δt:

$$
\mathbf{Q}_\eta = \boldsymbol{\Sigma}^d \Delta t^2 = \boldsymbol{\Sigma}\Delta t. \tag{3.18}
$$

3.3.1 Zeroth Order

Following the derivations in [1], one will come up with the (simplified) error state dynamics in the form of (3.17)

$$
\delta\boldsymbol{\theta}_{i+1} = \mathbf{R}^T\{\boldsymbol{\omega}\Delta t\}\delta\boldsymbol{\theta}_i + \boldsymbol{\theta}_I, \tag{3.19}
$$

$\boldsymbol{\theta}_I$ being the initial value (typically $\mathbf{0}$).

The covariance of the rotational error state is finally propagated using:

$$\boldsymbol{\Sigma}_{i+1}^{\delta\boldsymbol{\theta}} = \mathbf{R}^T\{\boldsymbol{\omega}_i\Delta t\}\boldsymbol{\Sigma}_i^{\delta\boldsymbol{\theta}}\mathbf{R}\{\boldsymbol{\omega}_i\Delta t\} + \boldsymbol{\Sigma}_i^{\boldsymbol{\omega}}\Delta t \tag{3.20}$$

Equation (3.20), shown for a forward integration, is independent of the applied zeroth order integration type. The equivalent midward variant is obtained by replacing $\boldsymbol{\omega}_i$ with the corresponding midward value $\frac{1}{2}(\boldsymbol{\omega}_{i+1} + \boldsymbol{\omega}_i)$.

3.3.2 Runge-Kutta

Given a linear continuous-time state system

$$\dot{\mathbf{x}} = \mathbf{A}(t)\mathbf{x}(t), \tag{3.21}$$

one can define a discrete-time state transition matrix $\boldsymbol{\Phi}_{t_{i+1}|t_i}$ that describes the change of the state $\mathbf{x}_i$ from time step t_i to t_{i+1}

$$\mathbf{x}(t_{i+1}) = \boldsymbol{\Phi}_{t_{i+1}|t_i}\mathbf{x}(t_i). \tag{3.22}$$

Please note, that $\boldsymbol{\Phi}_{t_i|t_i} = \mathbf{I}$. The transition matrix is directly used to propagate the noise covariance between discrete time-steps:

$$\boldsymbol{\Sigma}_{i+1}^{\delta\boldsymbol{\theta}} = \boldsymbol{\Phi}_{t_{i+1}|t_i}\boldsymbol{\Sigma}_i^{\delta\boldsymbol{\theta}}\boldsymbol{\Phi}_{t_{i+1}|t_i}^T + \boldsymbol{\Sigma}_i^{\boldsymbol{\omega}}\Delta t. \tag{3.23}$$

With the Runge-Kutta approach directly applied to the dynamic state transition matrix we obtain the corresponding discrete state transition matrix.

Third Order Given the intermediate states $\tilde{\mathbf{q}}_{\{1,2,3\}}$ and $\mathbf{k}_{\{1,2,3\}}$ from (3.12)

$$\begin{aligned}
&\boldsymbol{\Phi}_0 = \mathbf{I}_3 \\
&\dot{\boldsymbol{\Phi}}_1 = [\mathbf{k}_1]_\times \boldsymbol{\Phi}_0 && \boldsymbol{\Phi}_1 = \boldsymbol{\Phi}_0 + \frac{1}{2}\Delta t\dot{\boldsymbol{\Phi}}_1 \\
&\dot{\boldsymbol{\Phi}}_2 = [\mathbf{k}_2]_\times \boldsymbol{\Phi}_1 && \boldsymbol{\Phi}_2 = \boldsymbol{\Phi}_0 + \Delta t\left(-\dot{\boldsymbol{\Phi}}_1 + 2\dot{\boldsymbol{\Phi}}_2\right) \\
&\dot{\boldsymbol{\Phi}}_3 = [\mathbf{k}_3]_\times \boldsymbol{\Phi}_2
\end{aligned}$$

$$\boldsymbol{\Phi}_{i+1} = \mathbf{I}_3 + \frac{1}{6}\Delta t\left(\dot{\boldsymbol{\Phi}}_1 + 4\dot{\boldsymbol{\Phi}}_2 + \dot{\boldsymbol{\Phi}}_3\right). \tag{3.24}$$

Fourth Order Given the intermediate states $\tilde{\mathbf{q}}_{\{1,2,3,4\}}$ and $\mathbf{k}_{\{1,2,3,4\}}$ from (3.13)

$$\begin{aligned}
&\boldsymbol{\Phi}_0 = \mathbf{I}_3 \\
&\dot{\boldsymbol{\Phi}}_1 = [\mathbf{k}_1]_\times \boldsymbol{\Phi}_0 && \boldsymbol{\Phi}_1 = \boldsymbol{\Phi}_0 + \frac{1}{2}\Delta t\dot{\boldsymbol{\Phi}}_1 \\
&\dot{\boldsymbol{\Phi}}_2 = [\mathbf{k}_2]_\times \boldsymbol{\Phi}_1 && \boldsymbol{\Phi}_2 = \boldsymbol{\Phi}_0 + \frac{1}{2}\Delta t\dot{\boldsymbol{\Phi}}_2 \\
&\dot{\boldsymbol{\Phi}}_3 = [\mathbf{k}_3]_\times \boldsymbol{\Phi}_2 && \boldsymbol{\Phi}_3 = \boldsymbol{\Phi}_0 + \Delta t\dot{\boldsymbol{\Phi}}_3 \\
&\dot{\boldsymbol{\Phi}}_4 = [\mathbf{k}_4]_\times \boldsymbol{\Phi}_3
\end{aligned}$$

$$\boldsymbol{\Phi}_{i+1} = \mathbf{I}_3 + \frac{1}{6}\Delta t\left(\dot{\boldsymbol{\Phi}}_1 + 2\dot{\boldsymbol{\Phi}}_2 + 2\dot{\boldsymbol{\Phi}}_3 + \dot{\boldsymbol{\Phi}}_4\right). \tag{3.25}$$

0	Manifold Forward
1	Manifold Midward
2	Quaternion Forward
3	Quaternion Midward
4	Quaternion Runge-Kutta 3
5	Quaternion Runge-Kutta 4
6	Quaternion Crouch-Grossmann 3
7	Quaternion Crouch-Grossmann 4

Table 1: Implemented integration methods and corresponding number.

4 Results

4.1 Setup

The data used in the experiments in this section was generated using a continuous-time parametrization of a randomly generated trajectory. The trajectory is parametrized as a n'th order, 6-dimensional B-Spline where: i) the first three elements represent a translation; ii) the last three elements a rotation-vector parametrization of a rotation. We are fully aware of the limitations of a rotation vector parametrization but given the known setup, the use is safe.

The order of the B-Spline was chosen to be at least $n = 5$ to ensure non-constant accelerations but varied during the experiments. The number of basis functions was set to at least 50, and the regularization weight used for the initialization of the spline was set to max. 0.0001. The order, number of basis functions and regularization weight was varied during the experiments to obtain trajectories of different complexity levels. Details on modelling 3-dimensional motion as a continous-time B-Spline are best looked up in [5].

The simulated rotational velocity measurements are perturbed with white Gaussian noise with noise density $\sigma = 0.000186$, corresponding to one of our real calibrations.

The parameters of the rotation-vector parametrization of a sample curve used for evaluation is given in Figure 1 and 2.

To save space in the graphics and result tables, the integrations will be referred to by numbers. Table 1 shows the mapping of index to integration method.

4.2 Integration Drift under Noise

4.2.1 Integration Drift

We ran all integration methods introduced in the previous section on randomly generated trajectories. The rotational velocity ω was corrupted with Gaussian noise. To obtain statistically relevant results, despite the noisy signals, all 8 methods were used on 200 different trajectories with independently sampled Gaussian noise. The generated trajectories are 30 seconds long. Figure 3 shows the resulting cumulative drift errors in a box-plot. The numerical values are available in Table 2

Both the manifold and quaternion forward methods are clearly inferior to all

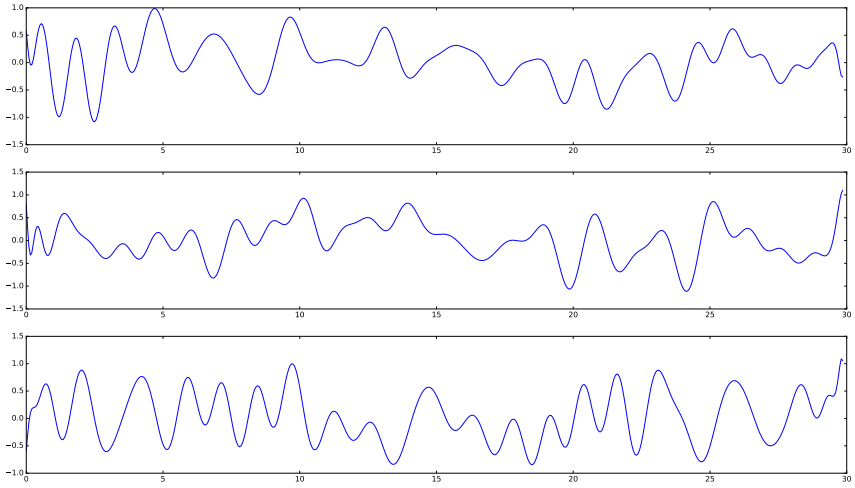

Figure 1: Sample trajectory used to generate the results. The x, y, z components of the rotation-vector parametrization.

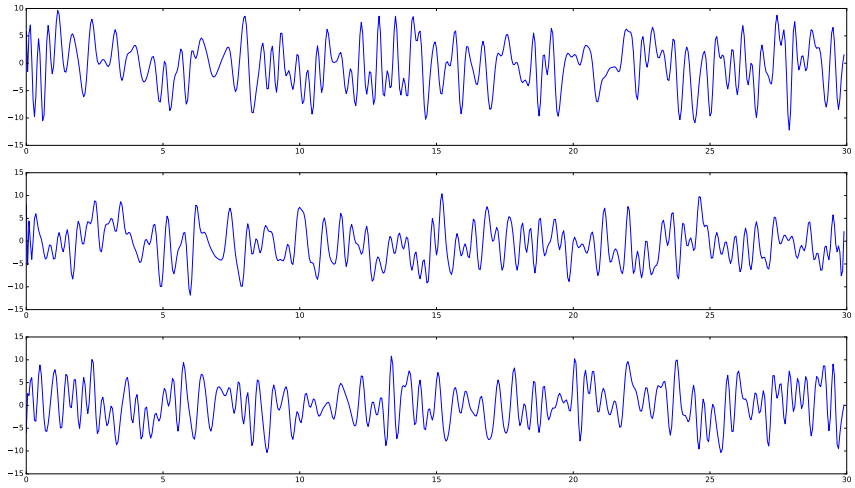

Figure 2: Sample trajectory used to generate the results. The x, y, z components of the rotation-vector parametrization for an aggressive rotational motion ($10\times$ larger).

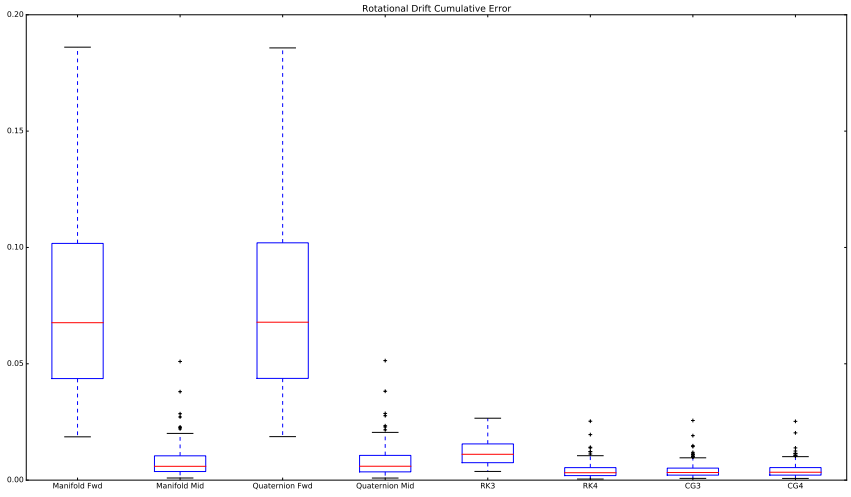

Figure 3:

0	Manifold Forward	0.0749788
1	Manifold Midward	0.00785299
2	Quaternion Forward	0.0750557
3	Quaternion Midward	0.00787657
4	Quaternion Runge-Kutta 3	0.0120488
5	Quaternion Runge-Kutta 4	0.00411811
6	Quaternion Crouch-Grossmann 3	0.00418711
7	Quaternion Crouch-Grossmann 4	0.00419546

Table 2: Average cumulative drift of different integration methods after 200 runs on simulated trajectories. Spline-Order: 5, Noise $\sigma = 0.00186$, rotation multiplier 5, 50 segments.

higher order methods. The midward integration methods perform an implicit low-pass filtering which reduces the drift by reducing the effect of the uncertainty that a single measurement would introduce. The higher order methods - 4'th order Runge-Kutta, and 3rd and 4th order Crouch-Grossman - only slightly improve over the midpoint methods.

The integration methods were tested against a set of trajectories of different complexity, for every set, 200 runs were performed. Table 3 summarizes the parameters used to generate the said trajectories. Figure 4 shows the results graphically and the averages are available in Table 4.

4.2.2 Real Dataset

The rotation integration was performed on a real dataset of the EUROC series (MH_01_easy). The ground-truth bias was used to correct the gyroscope

Run	Hz	Spline Order	Noise ($1e4$)	Segments	Smoothing	Rotation Multiplier
0	200	5	1.86	50	$1e-4$	1
1	200	5	3.72	50	$1e-4$	1
2	200	5	7.44	50	$1e-4$	1
3	20	5	1.86	50	$1e-4$	1
4	200	10	1.86	50	$1e-4$	1
5	200	10	1.86	200	$1e-4$	1
6	200	10	1.86	200	$1e-6$	1
7	200	10	1.86	50	$1e-4$	10
8	200	10	1.86	50	$1e-4$	5

Table 3: Parameters used to generate different trajectories for the drift evaluation.

	0	1	2	3	4	5	6	7	8
0	3.94058	3.93493	3.94597	0.418478	2.04392	3.174	9.72347	0.135967	7.49788
1	0.107683	0.200063	0.401721	4.24869	0.116234	0.31039	1.275	2.90529	0.785299
2	3.94248	3.95241	3.96111	0.418429	2.03719	3.18374	9.72697	0.135978	7.50557
3	0.112839	0.1992	0.410831	4.24528	0.117766	0.313399	1.28355	2.90502	0.787657
4	0.688499	0.7136	0.768424	9.30135	0.345993	0.587262	1.61293	2.82292	1.20488
5	0.109847	0.208381	0.386993	3.26243	0.10692	0.178501	0.860189	1.47977	0.411811
6	0.104299	0.200145	0.403532	3.28245	0.107502	0.174976	0.861394	1.47441	0.418711
7	0.105074	0.202421	0.398457	3.24674	0.108499	0.179947	0.862777	1.46404	0.419546

Table 4: Average values for different trajectories. The rows are the different integration methods, the columns show the different trajectories.

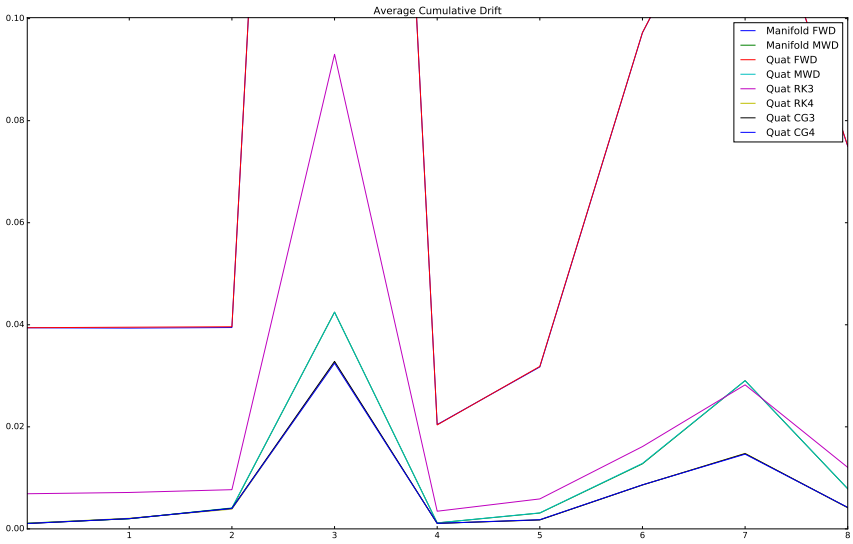

Figure 4: Average cumulative drift for runs with different trajectories

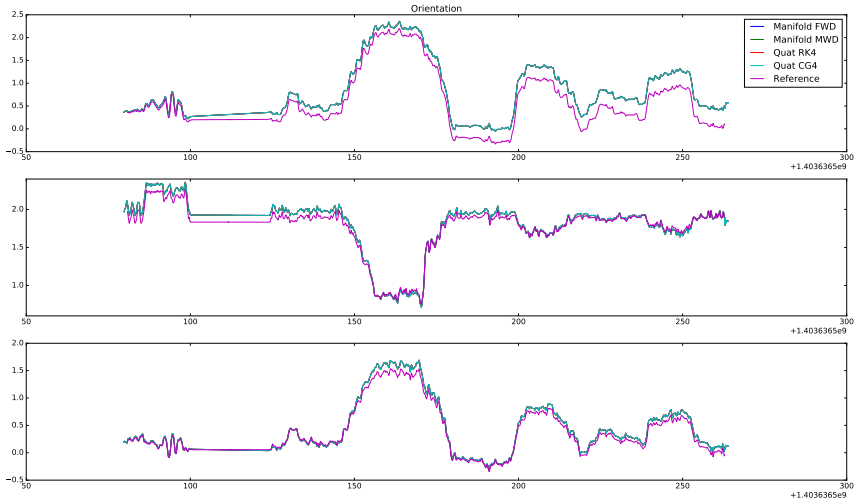

Figure 5: Integration drift evaluated on a real dataset where the ground-truth drift was subtracted prior to integration. The plots show the x, y and z coordinates of a rotation vector parametrization.

measurements prior to integration, Figure 5.

As a matter of fact, the magnitude of the uncertainty on the bias' Brownian motion, typically of magnitude $1e-4$ or $1e-5$, appears to massively outweigh the drift accountable to the integrator choice, producing cumulative errors in the range of $1e-3$ after integrations of 6000 measurements over 30 seconds. For a measurement stream at above 100Hz, the choice of integrator appear to be rather irrelevant given the scale of the influencing uncertainties. However, discarding forward and backward integrators appears to be a good choice performance and complexity-wise.

4.3 Noise Propagation

The noise propagation is evaluated based on a Monte-Carlo simulation of the integration and a subsequent variance estimation of the resulting drift. Typically, 20000 Monte-Carlo runs were performed.

The covariance at every integration step is estimated by calculating the error state of the integrated value with a ground-truth value:

$$\delta\boldsymbol{\theta} = \log\left(\delta\boldsymbol{R}_{ground-truth}^{T}\delta\boldsymbol{R}_{estimate}\right), \tag{4.1}$$

log is the matrix logarithm of a vector on $\mathfrak{se}3$.

The similarity of two covariance matrices is expressed using the matrix norm:

$$\delta\Sigma = ||\boldsymbol{\Sigma}_{ground-truth} - \boldsymbol{\Sigma}_{estimated}|| \tag{4.2}$$

There are no significant differences between the proposed integration methods. Except for the 3rd order Runge-Kutta method which seems to become less stable in the presence of fast rotation changes.

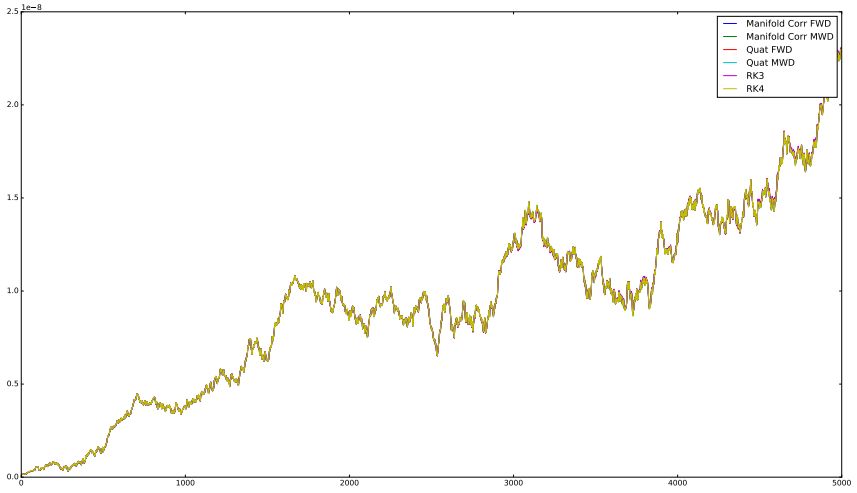

Figure 6: Noise propagation for a noise density of $1.86e-4$, $\Delta t = 0.005$.

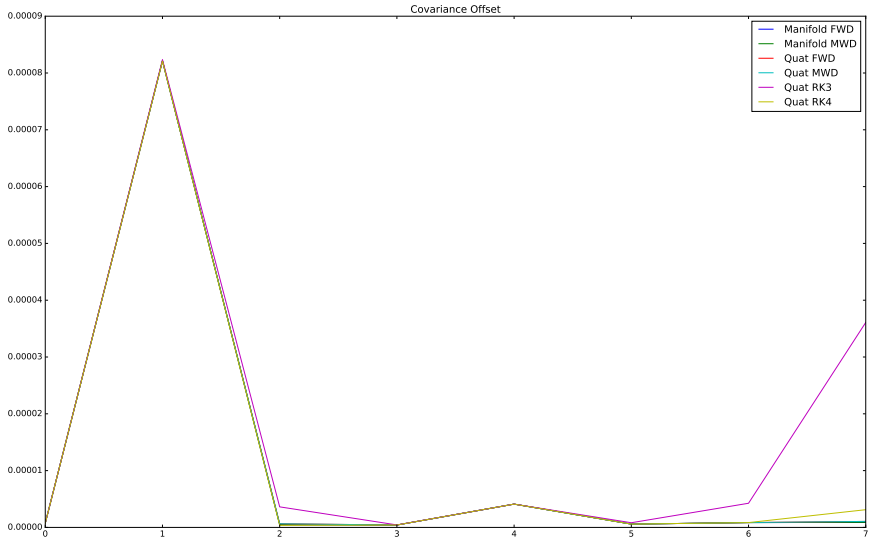

Figure 7: Noise propagation for different trajectories.

Run	Hz	Spline Order	Noise ($1e4$)	Segments	Smoothing	Rotation Multiplier
0	200	5	1.86	50	$1e-6$	1
1	200	5	18.6	50	$1e-6$	1
2	20	5	1.86	50	$1e-6$	1
3	100	5	1.86	50	$1e-6$	1
4	1000	5	1.86	50	$1e-6$	1
5	200	5	1.86	200	$1e-6$	2
6	200	5	1.86	200	$1e-6$	4
7	200	5	1.86	200	$1e-6$	10

Table 5: Parameters used to generate different trajectories for the covariance propagation evaluation.

	0	1	2	3	4	5	6	7
0	7.42013e-7	8.21704e-5	4.68377e-7	4.20485e-7	4.10374e-6	5.86244e-7	8.50302e-7	8.86004e-7
1	7.42001e-7	8.21675e-5	4.95865e-7	4.20391e-7	4.10374e-6	5.86208e-7	8.50439e-7	8.9136e-7
2	7.41973e-7	8.21019e-5	6.472e-7	4.22319e-7	4.10381e-6	5.80872e-7	8.5574e-7	1.0576e-6
3	7.41973e-7	8.21019e-5	6.472e-7	4.22319e-7	4.10381e-6	5.80872e-7	8.5574e-7	1.0576e-6
4	7.42765e-7	8.2346e-5	3.61947e-6	4.38474e-7	4.10378e-6	8.23784e-7	4.25238e-6	3.60825e-5
5	7.41974e-7	8.21021e-5	3.52353e-7	4.22279e-7	4.10381e-6	5.81326e-7	8.50033e-7	3.11836e-6

Table 6: Covariance offset for different integration methods (rows) and trajectories (columns)

4.4 Computational Complexity

The execution times of single integration steps are shown in Figure 8. The times include the integration itself and the the covariance propagation step - except for the two Crouch-Grossman methods which only contain the integration and a quaternion forward covariance propagation. As one might expect, any method that involves higher order computations and / or evaluations of exponential maps tends to be more expensive. Especially the Crouch-Grossman approaches tend to be much more expensive, even without a proper covariance propagation step. The cheapest method in terms of computational complexity and quality of the integration is the Quaternion Midward integration.

The proposed simplification of the on-manifold noise propagation slightly improves the performance and brings the on-manifold integration onto a same level with a zeroth order quaternion integration.

5 General Integration Methods

A thorouh introduction to integration methods with application to error state Kalman filtering is given in [1].

For a differential equation of the form:

$$\dot{\mathbf{x}} = f(t, \mathbf{x}) \tag{5.1}$$

evaluated at discrete time intervals $\mathbf{x}_n$, we sample the associated values of $\dot{\mathbf{x}}_i$.

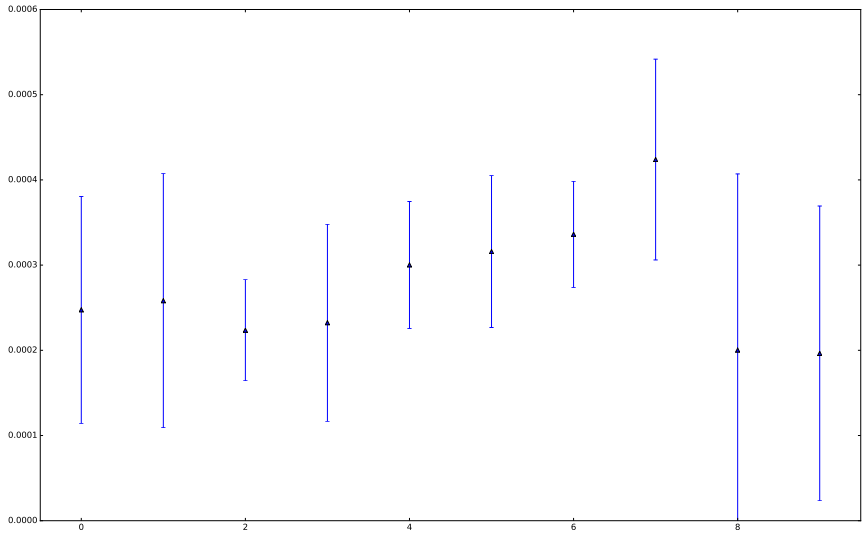

Figure 8: Timings of the integrators - averages after 250′000 steps. The indices 8 and 9 use the simplified On-Manifold covariance propagation.

5.1 Zeroth Order

Assuming that the value of $f(t)$ is piece-wise constant, and given the values $\dot{\mathbf{x}}_i$, $\dot{\mathbf{x}}_{i+1}$, three methods emerge immediately.

We assume that the value of $\dot{\mathbf{x}}$ corresponds to the sample $\dot{\mathbf{x}}_i$:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \Delta t \dot{\mathbf{x}}_i, \tag{5.2}$$

It is also perfectly valid, to assume that the value of $\dot{\mathbf{x}}$ corresponds to the sample $\dot{\mathbf{x}}_{i+1}$:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \Delta t \dot{\mathbf{x}}_{i+1}. \tag{5.3}$$

Both methods perform well in noise-free situations. Applying a midpoint integration is more robust to noisy values of $\dot{\mathbf{x}}_i$ as it implements an implicit averaging filter:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \frac{1}{2}\Delta t \left(\dot{\mathbf{x}}_i + \dot{\mathbf{x}}_{i+1}\right). \tag{5.4}$$

5.2 Runge-Kutta

The generalized Runge-Kutta method of n'th order iterates the following steps:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \Delta t \sum_{i=1}^{n} b_i k^{(i)}, \tag{5.5}$$

$$\mathbf{k}^{(i)} = \mathbf{f}(\mathbf{q}^{(i)}, t_k + c_i \Delta t), \tag{5.6}$$

$$\mathbf{q}^{(i)} = \mathbf{q}_k + \Delta t \sum_{j=1}^{i-1} a_{ij} \mathbf{k}^{(i)}, \tag{5.7}$$

where a_{ij}, b_i and c_i are given parameters typically represented in a Butcher table.

As the values of f are typically not available explicitly but sample at discrete time-steps, t_i, one would linearly interpolate the values at t_i and t_{i+1} to come up with an estimate at $t_i + c_i \Delta t$.

References

[1] Quaternion kinematics for the error-state KF. Joan Solà February 2, 2016, http://www.iri.upc.edu/people/jsola/JoanSola/objectes/notes/kinematics.pdf

[2] Andrle, Michael S., and John L. Crassidis. Geometric integration of quaternions. Journal of Guidance, Control, and Dynamics 36.6 (2013): 1762-1767.

[3] Forster, Christian, et al. On-Manifold Preintegration Theory for Fast and Accurate Visual-Inertial Navigation. arXiv preprint arXiv:1512.02363 (2015).

[4] Crouch, Peter E., and R. Grossman. Numerical integration of ordinary differential equations on manifolds. Journal of Nonlinear Science 3.1 (1993): 1-33.

[5] Furgale, Paul, Timothy D. Barfoot, and Gabe Sibley. Continuous-time batch estimation using temporal basis functions. Robotics and Automation (ICRA), 2012 IEEE International Conference on. IEEE, 2012.