The new IMU Factor

Frank Dellaert

September 6, 2019

Navigation States

Let us assume a setup where frames with image and/or laser measurements are processed at some fairly low rate, e.g., 10 Hz.

We define the state of the vehicle at those times as attitude, position, and velocity. These three quantities are jointly referred to as a NavState $X_b^n \triangleq \{R_b^n, P_b^n, V_b^n\}$, where the superscript n denotes the *navigation frame*, and b the *body frame*. For simplicity, we drop these indices below where clear from context.

Vector Fields and Differential Equations

We need a way to describe the evolution of a NavState over time. The NavState lives in a 9-dimensional manifold M, defined by the orthonormality constraints on $\mathbb{R}$. For a NavState X evolving over time we can write down a differential equation

$$\dot{X}(t) = F(t, X) \tag{1}$$

where F is a time-varying **vector field** on M, defined as a mapping from $\mathbb{R} \times M$ to tangent vectors at X. A **tangent vector** at X is defined as the derivative of a trajectory at X, and for the NavState manifold this will be a triplet

$$\left[\dot{R}(t,X), \dot{P}(t,X), \dot{V}(t,X)\right] \in \mathfrak{so}(3) \times \mathbb{R}^3 \times \mathbb{R}^3$$

where we use square brackets to indicate a tangent vector. The space of all tangent vectors at X is denoted by $T_X M$, and hence $F(t,X) \in T_X M$. For example, if the state evolves along a constant velocity trajectory

$$X(t) = \{R_0, P_0 + V_0 t, V_0\}$$

then the differential equation describing the trajectory is

$$\dot{X}(t) = [0_{3x3}, V_0, 0_{3x1}], \quad X(0) = \{R_0, P_0, V_0\}$$

Valid vector fields on a NavState manifold are special, in that the attitude and velocity derivatives can be arbitrary functions of X and t, but the derivative of position is constrained to be equal to the current velocity $V(t)$:

$$\dot{X}(t) = \left[\dot{R}(X,t), V(t), \dot{V}(X,t)\right] \tag{2}$$

Suppose we are given the **body angular velocity** $\omega^b(t)$ and non-gravity **acceleration** $a^b(t)$ in the body frame. We know (from Murray84book) that the derivative of R can be written as

$$\dot{R}(X,t) = R(t)[\omega^b(t)]_\times$$

where $[\theta]_\times \in so(3)$ is the skew-symmetric matrix corresponding to θ, and hence the resulting exact vector field is

$$\dot{X}(t) = \left[\dot{R}(X,t), V(t), \dot{V}(X,t)\right] = \left[R(t)[\omega^b(t)]_\times, V(t), g + R(t)a^b(t)\right] \tag{3}$$

Local Coordinates

Optimization on manifolds relies crucially on the concept of **local coordinates**. For example, when optimizing over the rotations $SO(3)$ starting from an initial estimate R_0, we define a local map Φ_{R_0} from $\theta \in \mathbb{R}^3$ to a neighborhood of $SO(3)$ centered around R_0,

$$\Phi_{R_0}(\theta) = R_0 \exp\left([\theta]_\times\right)$$

where exp is the matrix exponential, given by

$$\exp\left([\theta]_\times\right) = \sum_{k=0}^{\infty} \frac{1}{k!}[\theta]_\times^k \tag{4}$$

which for $SO(3)$ can be efficiently computed in closed form.

The local coordinates θ are isomorphic to tangent vectors at R_0. To see this, define $\theta = \omega t$ and note that

$$\left.\frac{d\Phi_{R_0}\left(\omega t\right)}{dt}\right|_{t=0} = \left.\frac{dR_0 \exp\left([\omega t]_\times\right)}{dt}\right|_{t=0} = R_0[\omega t]_\times$$

Hence, the 3-vector ω defines a direction of travel on the $SO(3)$ manifold, but does so in the local coordinate frame define by R_0.

A similar story holds in $SE(3)$: we define local coordinates $\xi = [\omega t, vt] \in \mathbb{R}^6$ and a mapping

$$\Phi_{T_0}(\xi) = T_0 \exp \hat{\xi}$$

where $\hat{\xi} \in \mathfrak{se}(3)$ is defined as

$$\hat{\xi} = \begin{bmatrix} [\omega]_\times & v \\ 0 & 0 \end{bmatrix} t$$

and the 6-vectors ξ are mapped to tangent vectors $T_0\hat{\xi}$ at T_0.

Derivative of The Local Coordinate Mapping

For the local coordinate mapping $\Phi_{R_0}\left(\theta\right)$ in $SO(3)$ we can define a 3×3 Jacobian $H(\theta)$ that models the effect of an incremental change δ to the local coordinates:

$$\Phi_{R_0}\left(\theta + \delta\right) \approx \Phi_{R_0}\left(\theta\right) \exp\left([H(\theta)\delta]_\times\right) = \Phi_{\Phi_{R_0}(\theta)}\left(H(\theta)\delta\right) \tag{5}$$

This Jacobian depends only on θ and, for the case of $SO(3)$, is given by a formula similar to the matrix exponential map,

$$H(\theta) = \sum_{k=0}^{\infty} \frac{(-1)^k}{(k+1)!}[\theta]_\times^k$$

which can also be computed in closed form. In particular, $H(0) = I_{3\times 3}$ at the base R_0.

Numerical Integration in Local Coordinates

Inspired by the paper "Lie Group Methods" by Iserles et al. [1], when we have a differential equation on $SO(3)$,

$$\dot{R}(t) = F(R,t), \quad R(0) = R_0 \tag{6}$$

we can transfer it to a differential equation in the 3-dimensional local coordinate space. To do so, we model the solution to (6) as

$$R(t) = \Phi_{R_0}(\theta(t))$$

To find an expression for $\dot{\theta}(t)$, create a trajectory $\gamma(\delta)$ that passes through $R(t)$ for $\delta = 0$, and moves $\theta(t)$ along the direction $\dot{\theta}(t)$:

$$\gamma(\delta) = R(t+\delta) = \Phi_{R_0}\left(\theta(t) + \dot{\theta}(t)\delta\right) \approx \Phi_{R(t)}\left(H(\theta)\dot{\theta}(t)\delta\right)$$

Taking the derivative for $\delta = 0$ we obtain

$$\dot{R}(t) = \left.\frac{d\gamma(\delta)}{d\delta}\right|_{\delta=0} = \left.\frac{d\Phi_{R(t)}\left(H(\theta)\dot{\theta}(t)\delta\right)}{d\delta}\right|_{\delta=0} = R(t)[H(\theta)\dot{\theta}(t)]_\times$$

Comparing this to (6) we obtain a differential equation for $\theta(t)$:

$$\dot{\theta}(t) = H(\theta)^{-1}\left\{R(t)^T F(R,t)\right\}^{\check{}}, \quad \theta(0) = 0_{3\times 1}$$

In other words, the vector field $F(R,t)$ is rotated to the local frame, the inverse hat operator is applied to get a 3-vector, which is then corrected by $H(\theta)^{-1}$ away from $\theta = 0$.

Retractions

Note that the use of the exponential map in local coordinate mappings is not obligatory, even in the context of Lie groups. Often it is computationally expedient to use mappings that are easier to compute, but yet induce the same tangent vector at T_0. Mappings that satisfy this constraint are collectively known as **retractions**. For example, for $SE(3)$ one could use the retraction $\mathcal{R}_{T_0} : \mathbb{R}^6 \to SE(3)$

$$\mathcal{R}_{T_0}(\xi) = T_0 \left\{\exp\left([\omega t]_\times\right), vt\right\} = \left\{\Phi_{R_0}(\omega t), P_0 + R_0 vt\right\}$$

This trajectory describes a linear path in position while the frame rotates, as opposed to the helical path traced out by the exponential map. The tangent vector at T_0 can be computed as

$$\left.\frac{d\mathcal{R}_{T_0}(\xi)}{dt}\right|_{t=0} = [R_0[\omega]_\times, R_0 v]$$

which is identical to the one induced by $\Phi_{T_0}(\xi) = T_0 \exp\hat{\xi}$.

The NavState manifold is not a Lie group like $SE(3)$, but we can easily define a retraction that behaves similarly to the one for $SE(3)$, while treating velocities the same way as positions:

$$\mathcal{R}_{X_0}(\zeta) = \left\{\Phi_{R_0}(\omega t), P_0 + R_0 vt, V_0 + R_0 at\right\}$$

Here $\zeta = [\omega t, vt, at]$ is a 9-vector, with respectively angular, position, and velocity components. The tangent vector at X_0 is

$$\left.\frac{d\mathcal{R}_{X_0}(\zeta)}{dt}\right|_{t=0} = [R_0[\omega]_\times, R_0 v, R_0 a]$$

and the isomorphism between $\mathbb{R}^9$ and $T_{X_0}M$ is $\zeta \to [R_0[\omega t]_\times, R_0 vt, R_0 at]$.

Integration in Local Coordinates

We now proceed exactly as before to describe the evolution of the NavState in local coordinates. Let us model the solution of the differential equation (1) as a trajectory $\zeta(t) = [\theta(t), p(t), v(t)]$, with $\zeta(0) = 0$, in the local coordinate frame anchored at X_0. Note that this trajectory evolves away from X_0, and we use the symbols θ, p, and v to indicate that these are integrated rather than differential quantities. With that, we have

$$X(t) = \mathcal{R}_{X_0}(\zeta(t)) = \{\Phi_{R_0}(\theta(t)), P_0 + R_0 p(t), V_0 + R_0 v(t)\} \tag{7}$$

We can create a trajectory $\gamma(\delta)$ that passes through $X(t)$ for $\delta = 0$

$$\gamma(\delta) = X(t+\delta) = \left\{\Phi_{R_0}\left(\theta(t) + \dot{\theta}(t)\delta\right), P_0 + R_0\left\{p(t) + \dot{p}(t)\delta\right\}, V_0 + R_0\left\{v(t) + \dot{v}(t)\delta\right\}\right\}$$

and taking the derivative for $\delta = 0$ we obtain

$$\dot{X}(t) = \left.\frac{d\gamma(\delta)}{d\delta}\right|_{\delta=0} = \left[R(t)[H(\theta)\dot{\theta}(t)]_\times, R_0\,\dot{p}(t), R_0\,\dot{v}(t)\right]$$

Comparing that with the vector field (3), we have exact integration iff

$$\left[R(t)[H(\theta)\dot{\theta}(t)]_\times, R_0\,\dot{p}(t), R_0\,\dot{v}(t)\right] = \left[R(t)[\omega^b(t)]_\times, V(t), g + R(t)a^b(t)\right]$$

Or, as another way to state this, if we solve the differential equations for $\theta(t)$, $p(t)$, and $v(t)$ such that

$$\begin{aligned}
\dot{\theta}(t) &= H(\theta)^{-1}\,\omega^b(t) \\
\dot{p}(t) &= R_0^T\,V_0 + v(t) \\
\dot{v}(t) &= R_0^T\,g + R_b^0(t)a^b(t)
\end{aligned}$$

where $R_b^0(t) = R_0^T R(t)$ is the rotation of the body frame with respect to R_0, and we have used $V(t) = V_0 + R_0 v(t)$.

Application: The New IMU Factor

In the IMU factor, we need to predict the NavState X_j from the current NavState X_i and the IMU measurements in-between. The above scheme suffers from a problem, which is that X_i needs to be known in order to compensate properly for the initial velocity and rotated gravity vector. Hence, the idea of Lupton was to split up $v(t)$ into a gravity-induced part and an accelerometer part

$$v(t) = v_g(t) + v_a(t)$$

evolving as

$$\begin{aligned}\dot{v}_g(t) &= R_i^T g \\ \dot{v}_a(t) &= R_b^i(t) a^b(t)\end{aligned}$$

The solution for the first equation is simply $v_g(t) = R_i^T gt$. Similarly, we split the position $p(t)$ up in three parts

$$p(t) = p_i(t) + p_g(t) + p_v(t)$$

evolving as

$$\begin{aligned}\dot{p}_i(t) &= R_i^T V_i \\ \dot{p}_g(t) &= v_g(t) = R_i^T gt \\ \dot{p}_v(t) &= v_a(t)\end{aligned}$$

Here the solutions for the two first equations are simply

$$\begin{aligned}p_i(t) &= R_i^T V_i t \\ p_g(t) &= R_i^T \frac{gt^2}{2}\end{aligned}$$

The recipe for the IMU factor is then, in summary. Solve the ordinary differential equations

$$\begin{aligned}\dot{\theta}(t) &= H(\theta(t))^{-1} \omega^b(t) \\ \dot{p}_v(t) &= v_a(t) \\ \dot{v}_a(t) &= R_b^i(t) a^b(t)\end{aligned}$$

starting from zero, up to time t_{ij}, where $R_b^i(t) = \exp[\theta(t)]_\times$ at all times. Form the local coordinate vector as

$$\zeta(t_{ij}) = [\theta(t_{ij}), p(t_{ij}), v(t_{ij})] = \left[\theta(t_{ij}), R_i^T V_i t_{ij} + R_i^T \frac{g t_{ij}^2}{2} + p_v(t_{ij}), R_i^T g t_{ij} + v_a(t_{ij})\right]$$

Predict the NavState X_j at time t_j from

$$X_j = \mathcal{R}_{X_i}(\zeta(t_{ij})) = \left\{\Phi_{R_0}\left(\theta(t_{ij})\right), P_i + V_i t_{ij} + \frac{g t_{ij}^2}{2} + R_i\, p_v(t_{ij}), V_i + g t_{ij} + R_i\, v_a(t_{ij})\right\}$$

Note that the predicted NavState X_j depends on X_i, but the integrated quantities $\theta(t)$,$p_v(t)$, and $v_a(t)$ do not.

A Simple Euler Scheme

To solve the differential equation we can use a simple Euler scheme:

$$\begin{aligned}\theta_{k+1} = \theta_k + \dot{\theta}(t_k)\Delta_t &= \theta_k + H(\theta_k)^{-1} \omega_k^b \Delta_t && (8) \\ p_{k+1} = p_k + \dot{p}_v(t_k)\Delta_t &= p_k + v_k \Delta_t && (9) \\ v_{k+1} = v_k + \dot{v}_a(t_k)\Delta_t &= v_k + \exp\left([\theta_k]_\times\right) a_k^b \Delta_t && (10)\end{aligned}$$

where $\theta_k \triangleq \theta(t_k)$, $p_k \triangleq p_v(t_k)$, and $v_k \triangleq v_a(t_k)$. However, the position propagation can be done more accurately, by using exact integration of the zero-order hold acceleration a_k^b:

$$\theta_{k+1} = \theta_k + H(\theta_k)^{-1}\,\omega_k^b \Delta_t \tag{11}$$

$$p_{k+1} = p_k + v_k \Delta_t + R_k a_k^b \frac{\Delta_t^2}{2} \tag{12}$$

$$v_{k+1} = v_k + R_k a_k^b \Delta_t \tag{13}$$

where we defined the rotation matrix $R_k = \exp\left([\theta_k]_\times\right)$.

Noise Propagation

Even when we assume uncorrelated noise on ω^b and a^b, the noise on the final computed quantities will have a non-trivial covariance structure, because the intermediate quantities θ_k and v_k appear in multiple places. To model the noise propagation, let us define $\zeta_k = [\theta_k, p_k, v_k]$ and rewrite Eqns. (11-13) as the non-linear function f

$$\zeta_{k+1} = f\left(\zeta_k, a_k^b, \omega_k^b\right)$$

Then the noise on ζ_{k+1} propagates as

$$\Sigma_{k+1} = A_k \Sigma_k A_k^T + B_k \Sigma_\eta^{ad} B_k + C_k \Sigma_\eta^{gd} C_k \tag{14}$$

where A_k is the 9×9 partial derivative of f wrpt ζ, and B_k and C_k the respective 9×3 partial derivatives with respect to the measured quantities a^b and ω^b.

We start with the noise propagation on θ, which is independent of the other quantities. Taking the derivative, we have

$$\frac{\partial \theta_{k+1}}{\partial \theta_k} = I_{3x3} + \frac{\partial H(\theta_k)^{-1}\omega_k^b}{\partial \theta_k}\Delta_t$$

It can be shown that for small θ_k we have

$$\frac{\partial H(\theta_k)^{-1}\omega_k^b}{\partial \theta_k} \approx -\frac{1}{2}[\omega_k^b]_\times \text{ and hence } \frac{\partial \theta_{k+1}}{\partial \theta_k} = I_{3x3} - \frac{\Delta t}{2}[\omega_k^b]_\times$$

For the derivatives of p_{k+1} and v_{k+1} we need the derivative

$$\frac{\partial R_k a_k^b}{\partial \theta_k} = R_k[-a_k^b]_\times \frac{\partial R_k}{\partial \theta_k} = R_k[-a_k^b]_\times H(\theta_k)$$

where we used

$$\frac{\partial\left(Ra\right)}{\partial R} \approx R[-a]_\times$$

and the fact that the dependence of the rotation R_k on θ_k is the already computed $H(\theta_k)$.

Putting all this together, we finally obtain

$$A_k \approx \begin{bmatrix} I_{3\times3} - \frac{\Delta_t}{2}[\omega_k^b]_\times & & \\ R_k[-a_k^b]_\times H(\theta_k)\frac{\Delta_t}{2}^2 & I_{3\times3} & I_{3\times3}\Delta_t \\ R_k[-a_k^b]_\times H(\theta_k)\Delta_t & & I_{3\times3} \end{bmatrix}$$

The other partial derivatives are simply

$$B_k = \begin{bmatrix} 0_{3\times3} \\ R_k \frac{\Delta_t}{2}^2 \\ R_k \Delta_t \end{bmatrix}, \quad C_k = \begin{bmatrix} H(\theta_k)^{-1}\Delta_t \\ 0_{3\times3} \\ 0_{3\times3} \end{bmatrix}$$

References

[1] Arieh Iserles, Hans Z Munthe-Kaas, Syvert P Nørsett, and Antonella Zanna. Lie-group methods. *Acta Numerica 2000*, 9:215–365, 2000.