LIE GROUPS AND SMOOTH MANIFOLDS FOR ROBOTIC STATE ESTIMATION

RAHUL AGGARWAL

Abstract. State estimation is a important capability to enable precise localization for robots with multiple noisy sensors. In this work, we motivate and investigate the fundamental structure that lies behind the idea of rotations and rigid transformations as both manifolds and groups, critically defining a notion of differentiability on these spaces. We then build the machinery for optimization over this latent structure, rooting our study in common robotics problems In particular, we utilize differentiability on state spaces to minimize error functions between observed and predicted data, allowing joint estimation of the system state.

Contents

1. Smooth Manifolds

Before we talk about manifolds, let alone differentiating on manifolds, we need to lock down both of these concepts in a formal sense. A smooth or differentiable manifold is one that looks locally Euclidean. Intuitively, this should mean that every point on said manifold has some open neighborhood around it that can be "flattened" to look like a plane.

Definition 1.1. Let X be a topological space. X is **locally Euclidean** if there is exists some $n \in N$ such that every point $p \in X$ has an open neighborhood $U \subseteq X$ which is homeomorphic to an open subset of $\mathbb{R}^n$.

Manifolds are distinguished from topological spaces by the fact that they are locally Euclidean, albeit with the additional properties of being Hausdorff and second countable (which are out of scope of this paper).

However, as of now, we have not attributed any structure, or *differentiability* to the topological manifold. We need to add to our notion of being locally Euclidean

Date: November 18, 2022.

the idea of smoothness. To do so, we need to be able to think about a differentiable homeomorphism.

Definition 1.2. A smooth map $f : M \to N$ is a **diffeomorphism** if it is a bijection and its inverse $f^{-1} : N \to M$ is smooth as well.

It is useful to think of a topological manifold as "consisting" of a countable collection of open subsets U_i corresponding to maps ϕ_i that carry points in U_i onto an open subset of $\mathbb{R}^n$. Formally each tuple $\{U_i, \phi_i\}$ is called a **coordinate chart**, and the collection of these charts is called an **atlas**. Using this, we can now define a smooth manifold.

Definition 1.3. Let X be an n-dimensional topological manifold. X is smooth if for every coordinate chart $\{U_i, \phi_i\}$ in its atlas, ϕ_i is smooth and if U_i and U_j overlap, then the transition map $\phi_{ij} = \phi_i \circ \phi_j^{-1}$ is a diffeomorphism.

The easiest way to think about smooth manifolds is as a collection of "patches" of Euclidean space.

Remark 1.4. We refer to $\phi_i^{-1} : \mathbb{R}^n \to U_i$ as a **local parameterization** of U_i of the manifold. This is useful when thinking about functions acting on the manifold.

2. Calculus on Manifolds

Derivatives of a function at a point can be thought of the best linear approximation near this point of the function with respect to its arguments. In other words, they are the best linear approximations in Euclidean space. Thus, if we have some smooth map $\psi : \mathbb{R}^n \to \mathbb{R}^m$, the directional derivative of ψ can be obtained by taking the limit

$$d\psi_x(v) = \lim_{t \to 0} \frac{\psi(x + tv) - \psi(x)}{t}$$

When we fix x, we obtain the derivative $d\psi_x : \mathbb{R}^n \to \mathbb{R}^m$, which is a linear map. The **Jacobian** of ψ at x is the matrix representation of $d\psi_x$ with respect to the standard bases.

We can use the local parameterizations we defined earlier to generalize the above definition for manifolds. Specifically, the derivative *identifies* the linear space that best approximates a manifold X around a point x.

Definition 2.1. if X is an n-dimensional smooth manifold, the **tangent space** at a point x in X, $T_x(X)$ is the image of $d\phi_x^{-1}$, where $\phi_x^{-1} : \mathbb{R}^n \to U$ is a local parameterization of the manifold.

Remark 2.2. We can also consider the tangent space of a manifold as a manifold itself. We refer to vectors in this tangent space through the **local coordinates** defined by a local coordinate chart up to the tangent space.

3. Lie Groups

When we think of concepts that groups can represent, one clear example comes from the symmetries they observe on an object. For example the symmetry group of the square D_4 identifies rotations and reflections that change the canonical order of the vertices of a square. Notably, these symmetries are in some way discrete.

However, one might want to consider what sort of group identifies *continuous* symmetries; for example, the rotation of a circle can be thought of to be continually

symmetric. Importantly, it so happens that these rotations are, in some notion, *smooth*.

Definition 3.1. A **Lie group** G is a group that is also a smooth manifold, thereby possessing a smooth group operation ($\cdot : G \mapsto G$) and smooth inverse ($^{-1}\colon G \mapsto G$)

These lie groups possess the internal structure we need to characterize these continuous, differentiable geometric transformations.

Each lie group is associated with a **Lie Algebra** $\mathfrak{g}$ which can be mapped back to G using the **exponential map**.

$$\exp : \mathfrak{g} \to G$$

which is usually a many-to-one mapping. The corresponding inverse is defined locally around the origin, and fore a "logarithm"

$$\log : G \to \mathfrak{g}$$

An important family of Lie groups are the matrix Lie groups $GL(n, \mathbb{R})$ which is an open subset of $\mathbb{R}^{n\times n}$, and whose group operation is matrix multiplication.

In terms of robotics, two Lie groups we particularly care about are the Special Orthogonal group $SO(n)$, and the Special Euclidean group $SE(n)$, which encode rotations and rigid transformations, respectively. To motivate these groups, consider a robot moving through the plane with some velocity vector

$$(v_x, v_y, \omega)$$

where v represents the robot's linear velocity through space, and ω represents the angular velocity. If we were to just increment the robot's position using this vector, i.e

$$(x_t, y_t, \theta_t) = (x_0 + v_x t, y_0 + v_y t, \omega t)$$

we quickly see that this update does not encode the fact that our orientation is changing. Specifically, every time we move the robot a tiny bit according to the velocity vector, our next movement will move in a *different direction.* (a circular path). In other words, the velocity vector has to be rotated before it can be applied. In practice, we embed the 2D poses $T = (x, y, \theta)$ into the space of 3×3 matrices, such that

$$T_1 T_2 = \begin{bmatrix} R_1 & t_1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R_2 & t_2 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R_1 R_2 & R_1 t_2 + t_1 \\ 0 & 1 \end{bmatrix}$$

where the matrices R are 2D rotation matrices

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

Now our tiny motion of the robot can be written as

$$T(\delta) = \begin{bmatrix} \cos\omega\delta & -\sin\omega\delta & v_x\delta \\ \sin\omega\delta & \cos\omega\delta & v_y\delta \\ 0 & 0 & 1 \end{bmatrix} \approx I + \delta \begin{bmatrix} 0 & -\omega & v_x \\ \omega & 0 & v_y \\ 0 & 0 & 0 \end{bmatrix}$$

We call a vector $\xi = (v, w)$ a 2D twist in robotics, and the matrix with the skew-symmetric element in the top left block as

$$\hat{\xi} \triangleq \begin{bmatrix} 0 & -\omega & v_x \\ \omega & 0 & v_y \\ 0 & 0 & 0 \end{bmatrix}$$

Therefore, a pose at time t is obtained in the limit:

$$T(t) = \lim_{n\to\infty}(I + \frac{t}{n}\hat{\xi})^n$$

Notice, that for real numbers, this series is just the exponential function.

$$\exp(x) = \lim_{n\to\infty}(1 + \frac{x}{n})^n = \sum_{k=0}^{\infty}\frac{x^k}{k!}$$

We similarly define this series for square matrices, letting us write the motion of a robot resulting from a constant twist $\xi = (v, \omega)$ as the *matrix exponential* of $\hat{\xi}$.

$$T(t) = \exp(t\hat{\xi}) \triangleq \lim_{n\to\infty}(I + \frac{t}{n}\hat{\xi})^n = \sum_{k=0}^{\infty}\frac{t^k}{k!}\hat{\xi}^k$$

Note that this is suspiciously similar to the exponential map we defined above and in fact, it is. Taking the space of 2D twists together with the Lie bracket operator for matrix groups ($[A, B] \triangleq AB - BA$), we obtain the Lie algebra $\mathfrak{se}(2)$, which can be mapped using the exponential map onto the space of 2D rigid transformations together with a matrix multiplication operation, otherwise known as the Lie group $SE(2)$.

We can additionally make formal this notion of the relation between the two, and define the hat operator

$$\hat{} : x \in \mathbb{R}^n \to \hat{x} \in \mathfrak{g}$$

When we are dealing with matrix Lie groups, the elements $\hat{x}$ of $\mathfrak{g}$ are $n \times n$ matrices, and the map can be given by

$$\hat{x} = \sum_{i=1}^{n} x_i\, G^i \tag{3.2}$$

where $G^i \in \mathbb{R}^{n\times n}$ are the **Lie group generators**.

Our final observation is the equivalence between the tangent space T_eG and the Lie algebra $\mathfrak{g}$.

Theorem 3.3. *If $\gamma(s)$ is a curve in $G \subset GL(n;\mathbb{R})$ such that $\gamma(0) = I$, then $\frac{d\gamma}{ds}(0) \in \mathfrak{g}$.*

Proof. Suppose we have a curve γ that satisfies the above properties. For sufficiently small s, We can say that $\log(\gamma(s)) \in \mathfrak{g}$. Therefore,

$$\frac{d\log(\gamma(s))}{ds}(0) = \lim_{\epsilon\to 0}\frac{\log(\gamma(\epsilon))}{\epsilon} \in \mathfrak{g}$$

since we are under a limit and $\mathfrak{g}$ is closed. Furthermore, we know that the logarithm map is just the inverse of the exponential map.

$$\log(\gamma(s)) = (\gamma(s) - I) - \frac{1}{2}(\gamma(s) - I)^2 + \frac{1}{3}(\gamma(s) - I)^3 - \cdots$$

Differentiating and taking $s \to 0$, we observe that all terms except the first approach zero (Remember that $\gamma(0) = I$). Thus, we have

$$\frac{d\gamma}{ds}(0) = \frac{d\log(\gamma(s))}{ds}(0) \in \mathfrak{g}$$

□

In other words, our local coordinates $\xi \in \mathbb{R}^n$ at some point a on our manifold G can be mapped to tangent vectors $a\hat{\xi} \in T_a G$.

4. Optimization for Robotics

Now that we have built our fundamentals, we can start thinking about how we can use these concepts in robotics.

Suppose we had a robotic platform for which we wanted an accurate estimate of the orientation $R \in SO(3)$, which we know to have three degrees of freedom. One common and often used way to refer to the orientation are the Euler angles: roll, pitch, and yaw. However, this representation of orientation suffers from a condition called gimbal lock, which rises from the fact that they do not form a smooth chart on all of $SO(3)$. Instead, we want to optimize on the manifold itself.

Suppose we have some non-linear measurement function:

$$h : SO(3) \to \mathbb{R}^n : R \mapsto z$$

To try and estimate the unknown orientation R, we want to minimize a non-linear least squares error criterion

$$R^* = \arg\min_R \|h(R) - z\|_\Sigma^2 \tag{4.1}$$

where $\|e\|_\Sigma^2 \triangleq e^T \Sigma^{-1} e$ is the squared Mahalanobis distance with covariance Σ.

Remark 4.2. The Mahalanobis distance is the multi-dimensional generalization of the measure of how many standard deviations a point p is from the mean of a distribution D.

To minimize (4.1), we need to know how the non-linear function h behaves in the neighborhood of a linearization point a. Roughly speaking, we want to define a Jacobian matrix H_a such that

$$h(a \oplus \xi) \approx h(a) + H_a \xi \tag{4.3}$$

with ξ being defined in the local coordinates of $SO(3)$. Using this approximation, we can now minimize (4.1) with respect to ξ:

$$\xi^* = \arg\min_\xi \|h(a) + H_a \xi - z\|_\Sigma^2 \tag{4.4}$$

We solve this by setting the derivative (4.4) to zero, yielding

$$H_a^T H_a \xi^* = H_a^T (z - h(a))$$

To make this process concrete for Lie groups, we can define our missing operator $\oplus$ as

$$a \oplus \xi \triangleq a \exp(\hat{\xi}) \tag{4.5}$$

which can be interpreted as mapping from local coordinates to a neighborhood in our Lie group G around a. We can therefore define a notion of differentiability for functions on Lie groups.

Definition 4.6. A function $f : G \to \mathbb{R}^m$ is differentiable at $a \in G$ if there exists a matrix $F_a \in \mathbb{R}^{m \times n}$ such that

$$\lim_{\xi \to 0} \frac{|f(a) + F_a \xi - f(a \exp(\hat{\xi}))|}{|\xi|} = 0$$

The linear map $df_a : \xi \mapsto F_a\xi$ is the derivative of f at a, and the matrix representation of said linear map F_a is the Jacobian.

Thus, starting with an initial estimate R_0, we are able to iteratively find better estimates for R, thereby improving our estimate of the robot's orientation. In particular, numerical methods like the Levenberg-Marquardt and Gauss Newton algorithms can utilize notions of differentiability on state spaces like $SO(3)$ to obtain the optimal step for minimizing the error between predicted and observed measurements.

One concrete example for robotic state estimation is estimating a robot's pose using its measurements of predefined landmarks defined in the environment. In this case, we have measurements z_{ij} predicted by

$$z_{ij} = h(T_i, p_j) = \pi(T_i p_j)$$

where T_i is the pose of the i^{th} camera, p_j is the location of the j^{th} landmark, and $\pi : (x, y, z) \mapsto (x/z, y/z)$ is a *camera projection function.* Note that here, our measurement function has two parameters, a rigid pose and a point. Therefore, when jointly optimizing robot state, we have to additionally reason about the "effect" that rotations and rigid transformations have on points and poses, transforming them geometrically from one point in space to another.

5. Actions and their Derivatives

Specifically, we define the **action** of a group on a space as an equivalence (or homomorphism) to transformations of that space. For example, the usual action of an m-dimensional matrix group G is a matrix-vector multiplication of $\mathbb{R}^n$, as in $f : G \times \mathbb{R}^n \to \mathbb{R}^n$

$$f(T, p) = Tp$$

The next logical step is to consider the derivative of this transformation. Because this function is defined on $G \times \mathbb{R}^n$, the derivative is a linear map $Df : \mathbb{R}^{m+n} \to \mathbb{R}^n$

$$df_{(T,p)}(\xi, \delta p) = d_1 f_{(T,p)}(\xi) + d_2 f_{(T,p)}(\delta p)$$

Theorem 5.1. *The Jacobian matrix of the group action $f(T, p) = Tp$ at (T, p) is*

$$F_{(T,p)} = \begin{bmatrix} TH(p) & T \end{bmatrix} = T \begin{bmatrix} H(p) & I_n \end{bmatrix}$$

Proof. We first take the derivative $d_2 f$ with respect to p:

$$f(T, p + \delta p) = T(p + \delta p) = Tp + T\delta p = f(T, p) + d_2 f(\delta p)$$

As for the derivative $d_1 f$, which is in respect to a change in T, we need to find a linear map $d_1 f$ such that

$$Tp + d_1 f(\xi) \approx f(T \exp(\hat{\xi}), p) = T \exp(\hat{\xi}) p$$

We know that the matrix exponential is given by the series $\exp(X) = I + X + \frac{X^2}{2!} + ...$ Therefore, we order

$$T \exp(\hat{\xi}) p \approx T(I + \hat{\xi}) p = Tp + T\hat{\xi}p$$

and so $d_1 f(\xi) = T\hat{\xi}p$. Finally, to complete the proof, we just need to show that

$$\hat{\xi}p = H(p)\xi$$

where $H(p)$ is an $n \times m$ matrix that depends only on p. As discussed in 3.2, we can express the map $\mathbb{R}^n \to \mathfrak{g}, \xi \to \hat{\xi}$ in terms of the Lie algebra generators G^i, and using Einstein summation. Thus,

$$(\hat{\xi}p)^i = \hat{\xi}^i_j p^j = G^i_{jk}\xi^k p^j = (G^i_{jk}p^j)\xi^k = H^i_k(p)\xi^k$$

□

These actions have interesting geometric interpretations.

Example 5.2. For 3D rotations $R \in SO(3)$ of a point, we have the operator $\hat{} : \omega \mapsto [\omega]_\times$ and the Lie algebra generators

$$G_{k=1} : \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} G_{k=2} : \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix} G_{k=3} : \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Thus $H(p)$ is equal to

$$G_{k=1}\, p^1 + G_{k=2}\, p^2 + G_{k=3}\, p^3 = \begin{pmatrix} 0 & p^3 & -p^2 \\ -p^3 & 0 & p^1 \\ p^2 & -p^1 & 0 \end{pmatrix} = [-p]_\times$$

Therefore, the Jacobian of $f(R, p) = Rp$ is

$$F_{(R,p)} = R \begin{bmatrix} [-p]_\times & I_3 \end{bmatrix}$$

Here, $R[-p]_\times$ tells us what how our point p changes (i.e velocity) with respect to changes in the rotation R.

6. Conclusion

If there were such a thing as a perfect sensor, state estimation would not be needed. Unfortunately, in the real world, sensors are always corrupted with some type of noise, creating significant challenge in recovering proper state estimates. Indeed, modern day sensor fusion and localization extensively utilizes the machinery we have built up in this paper and in practice are able to jointly estimate the many different variables contained within a robotic platform to industry-level precision.

References

[1] Victor Guillemin and Alan Pollack. *Differential Topology*, volume. 370, American Mathematical Society, 2010.

[2] Richard M Murray, Zexiang Li, S Shankar Sastry, and S Shankara Sastry. *A mathematical introduction to robotic manipulation*. CRC Press, 1994.

[3] Michael Spivak. *Calculus on Manifolds*, volume 1. WA Benjamin New York, 1965.