A Geometric Introduction to Lie Theory

Alvin Zhang, alvn.zng@gmail.com

November 2022

1 Meta

This is an intuitive, visual introduction to Lie theory. Readers are expected to have familiarity with high-school calculus and trigonometry.

2 Introduction

What is the "natural" path for an animal or robot to take when moving from one pose to another? This question has applications in robotics, computer vision, and computer graphics. In this post, I use this problem as a case study to introduce the basic concepts behind Lie theory. Rather than starting with an abstract mathematical definition and then presenting applications, I instead analyze a concrete example and use its underlying geometry to motivate and develop the general concepts and assumptions behind Lie theory.

As a bonus, applying this geometric approach to studying Lie groups allows us to derive a new method for computing the logarithmic map from $SE(3) -> se(3)$, which is both faster and numerically stable than existing methods in the literature.

3 The Problem

The problem is as follows: Given the starting and ending "pose"s of a robot or animal, can we estimate what path the animal took in moving between these poses? That is, can we estimate the intermediate poses of that animal's path?

Let's start by clarifying the question, and outlining some assumptions.

First things first. What do I mean by an object's "pose"?

Intuitively, the "pose" of an animal encapsulates both its location in space and its local arrangement of parts: where the head is in relation to the feet, which way the arm is pointing, if the paw is twisted or upside-down. But, this description of "pose" is really complicated and difficult to study.

[Illustration of complicated pose]

In this post, I am going to assume that a robot's "pose" in 3-D space can be fully described by a position (which has three degrees of freedom) and an

orientation (which also has three degrees of freedom). This has the effect of locking all of the joints of the animal in place, so that the local arrangement of parts is fixed: like a plastic toy, the animal has become a "rigid body" which can only be rotated or translated as one unit.

[Insert picture of duck illustrating 6 DOF].

If we think about how most robots or animals move, this is really not too poor of an assumption: the parts which propel the robot or animal forward tend to come back to roughly the same configuration in a cyclical pattern. And the parts which aren't involved in propulsion don't really affect the path that the animal or robot takes.

[Cat vs. amoeba]

So, then, what is a "natural" path between two 3-D poses?

Well, there are of course many paths which will get something from one pose to another. One possibility, which minimizes the distance traveled, is to first rotate to face the ending pose, travel to the destination, and then re-orient to the desired ending pose.

[Illustration (Definitely not walle)]

This is definitely a viable strategy! It is certainly the "natural" path in many contexts. However, the rotational and translational motions happen in separate phases. Maybe we could get from the start to the end faster if we rotated and translated simultaneously. What path would we get if we tried this?

[Illustration (spinning walle)]

This is very direct! However, we usually do not see animals, cars, planes, or boats moving in this way - in most contexts, an animator is probably not going to choose this kind of path.

[Illustration (cat, car, bird/plane)]

Why is this? There are two reasons.

Firstly, animals and robots tend to have a preferred local direction for linear motion. The cat finds it more "natural" to walk forwards (towards the end of its path) than backward (at the beginning) or sideways (in the middle). It is not even possible for a car to move sideways, unless you know how to drift!

Secondly, even if it is possible for a robot or animal to move in all directions, maintaining a [globally] constant linear velocity requires constantly monitoring and changing one's [local] linear velocity. This is a lot of work! (I encourage you to try it out: try to walk 10 meters forward while rotating over 135 degrees. It's surprisingly difficult!) Instead, what if we instead start by choosing a path which maintains constant [local] linear and angular velocities?

[Illustration (stepping, cat, car, boat, bird/plane)]

Maintaining constant linear and angular velocities results in paths that are smooth arcs! These are the "natural" paths which I will be working with for the rest of this post: those traced out by objects moving at constant local linear and angular velocities. Of course, there are other criteria for "natural" paths, such as those discussed previously, which may be more appropriate depending on the situation. However, studying these paths with constant local velocities leads us to some interesting questions: Given two poses, how can I compute the local linear and angular velocities that I need to move between them? And,

what are the intermediate poses along the path? Do I need to simulate applying the local angular and linear velocities over many small time-steps, or is there a closed-form formula that I can apply to get the global pose?

The questions are not easy to answer; luckily, the powerful mathematical tool known as Lie theory can provide answers. However, in its full glory, it is perhaps [too] powerful: because it is so general and abstract (it works in any number of dimensions!), its proper definition uses abstract mathematical language which I find difficult to intuit. Instead, in this post, I will start by analyzing 2-D rotations, which are simple to understand and illustrate. The study of this specific case will then provide the framework for introducing the basics of Lie theory. I then demonstrate the equivalence of the geometric and Lie-theoretic approaches in more complex cases, including that of 3-D rigid body motions, allowing us to answer the questions posed above.

4 2-D Rotation

The two-dimensional analogue of 3-D orientation is simply "heading", the direction that an object in the plane is facing. It is a seemingly simple case with only one degree of freedom, but analyzing this will provide us with the tools that we need to analyze the more complex cases of combined rotation and translation in both 2 and 3 dimensions.

4.1 Coordinate Systems

Before we can do anything with headings, we have to answer the most basic question: how can we describe an object's "heading" with numbers? Just as we can describe an object's 2-D position in terms of an x-coordinate and a y-coordinate, can we uniquely specify an object's 2-D orientation?

One important thing to note is that, even in the case of 2-D position, coordinates are relative to a base frame. If I tell you that a object is at coordinates $(x, y) = (3, 4)$, that gives no information about where the object actually is until I tell you where $(0, 0)$ is, which direction the x-axis is pointing, and (in real life) the units.

[Illustration of different coordinate systems]

Since headings have only one degree of freedom, we should only need to use one number to describe an object's 2-D orientation. This can be achieved as follows: pick a reference heading to be the 0 heading; then, for any other heading h, one can uniquely identify h by the angle between it and the 0 heading. Let's call this coordinate "axis" $\measuredangle$ (analogously to x or y for 2-D position). Then, it is convention to take the direction of the positive x-axis to be the 0 heading, and to measure angles in radians, with $+\measuredangle$ "pointing" in the counter-clockwise direction. I will use this convention in the rest of this post.

[Illustration showing $\measuredangle$]

4.2 Group Property

This brings us to an interesting observation about headings: the difference between two headings is in fact another heading.

[Illustration]

This fact has deep consequences: it means that the set of headings and the 2-D rotations that [transform] between headings [can be thought of equivalently]. In fact, in the rest of this post, I will use "heading", 2-D orientation, and 2-D rotation interchangeably: a heading [is] a 2-D rotation, and can act on another heading to produce a third heading. I will use $a(b)$ to denote applying a rotation a to a heading b.

The more mathematically inclined may note that this is the first characteristic of a mathematical "group": the others being that: (2) there is an identity rotation which does not change other headings, (3) for every heading, there is an inverse heading which brings it back to the identity heading, and (4) for any three headings, $a(b(c)) = (a(b))(c)$.

[Illustration of group properties]

4.3 Representation

In the section about Coordinate Systems[link], I introduced a simple coordinate system to an object's heading. It is minimal in the sense that we have one number $[\theta]_{\measuredangle}$ to describe a system with one degree of freedom, and elegant in that sense that the group operations are simple:

1. $a(b) := [a + b]_{\measuredangle}$,
2. $I := [0]_{\measuredangle}$, and
3. $a^{-1} := [-a]_{\measuredangle}$.

However, this representation does not accurately capture the geometry of the group, in the sense that there are infinitely many representations of the same object: for any integer k, $[0]_{\measuredangle} \equiv [k \cdot 2\pi]_{\measuredangle}$.

[Illustration of equivalent points]

What if we instead take a geometric approach, one that captures the underlying structure of the group of 2-D rotations? Since an object's heading is just the direction that it is facing, then there is a one-to-one mapping between headings and points on the unit circle: in fact, the spaces of 2-D rotation is mathematically isomorphic to the unit circle. Therefore, let us define:

$$[\cos(\theta), \sin(\theta)]_{\odot} :\equiv [\theta]_{\measuredangle}$$

[Illustration]

This removes the redundancy of the $\measuredangle$ representation and reflects the geometry of 2-D rotations. However, the implementation of the group operations is not clean: it seems that one would have to convert from $\odot$ back to the $\measuredangle$ representation, perform the group operation there, and then re-covert from $\measuredangle$ to $\odot$.

However, there is another option, which actually makes combining "heading"s very easy! Instead of going back to one number, why don't we add two more?

$$[\cos(\theta), \sin(\theta)]_{\odot} :\equiv \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}_M$$

This is clearly equivalent to the $\odot$ representation: we just took the $\odot$ vector and made it into the first column, while the second column has no free parameters once the first column is determined.

In this M-representation, the group operations are actually just matrix operations! We can verify that they are equivalent to the $\measuredangle$ group operations:

1.

$$\begin{aligned} a(b) &= [a]_{\measuredangle}([b]_{\measuredangle}) \\ &:= ([a]_{\measuredangle})_M([b]_{\measuredangle})_M \\ &= \begin{bmatrix} \cos([a]_{\measuredangle}) & -\sin([a]_{\measuredangle}) \\ \sin([a]_{\measuredangle}) & \cos([a]_{\measuredangle}) \end{bmatrix}_M \begin{bmatrix} \cos([b]_{\measuredangle}) & -\sin([b]_{\measuredangle}) \\ \sin([b]_{\measuredangle}) & \cos([b]_{\measuredangle}) \end{bmatrix}_M \\ &= \begin{bmatrix} \cos([a]_{\measuredangle})\cos([a]_{\measuredangle}) - \sin([b]_{\measuredangle})\sin([b]_{\measuredangle}) & -\cos([a]_{\measuredangle})\sin([b]_{\measuredangle}) - \cos([b]_{\measuredangle})\sin([a]_{\measuredangle}) \\ \cos([a]_{\measuredangle})\sin([b]_{\measuredangle}) + \cos([b]_{\measuredangle})\sin([a]_{\measuredangle}) & \cos([a]_{\measuredangle})\cos([a]_{\measuredangle}) - \sin([b]_{\measuredangle})\sin([b]_{\measuredangle}) \end{bmatrix}_M \\ &= \begin{bmatrix} \cos([a+b]_{\measuredangle}) & -\sin([a+b]_{\measuredangle}) \\ \sin([a+b]_{\measuredangle}) & \cos([a+b]_{\measuredangle}) \end{bmatrix}_M \\ &= [a+b]_{\measuredangle} \end{aligned}$$

2.

$$I = (I_2)_M = [0]_{\measuredangle}$$

3.

$$a^{-1} = [-a]_{\measuredangle} = \begin{bmatrix} \cos([-a]_{\measuredangle}) & -\sin([-a]_{\measuredangle}) \\ \sin([-a]_{\measuredangle}) & \cos([-a]_{\measuredangle}) \end{bmatrix}_M$$

and applying the group composition operation shows that

$$aa^{-1} = ([a]_{\measuredangle})_M([-a]_{\measuredangle})_M = ([a-a]_{\measuredangle})_M = ([0]_{\measuredangle})_M = I,$$

as desired.

So, we now have 3 different ways to represent headings! However, depending on the situation, it may be more convenient to use one over another. For instance, the $\odot$ representation is great for illustrations, the $\measuredangle$ representation is simple and linear, and the M representation supports the group operations while respecting the underlying unit-circle geometry.

[Table: representation, linear, illustration, isomorphic to unit circle (respects underlying geometry)]

4.4 Velocity

So, now that we have these tools, let's return to our original problem: what is the constant-velocity path between two headings?

Let's first define what it means to move at a constant angular velocity. This is pretty simple, especially in the $\measuredangle$ representation.

If an object moves at a [constant] angular velocity $[\omega]_{\frac{d}{dt}\measuredangle}$, then for any starting time t_0 and ending time t_1, we have:

$$[\omega]_{\frac{d}{dt}\measuredangle} := \frac{[\theta_{t=t_1} - \theta_{t=t_0}]_\measuredangle}{t_1 - t_0}$$

Therefore, if an object starts at $[\theta_{t=t_0}]_\measuredangle$ and moves at a constant angular velocity $[\omega]_{\frac{d}{dt}\measuredangle}$, we can compute its path as follows:

$$[\theta_{t=t_0+T}]_\measuredangle = [\theta_{t=t_0}]_\measuredangle + T[\omega]_{\frac{d}{dt}\measuredangle}$$

But, what about the other representations ($\odot$ and M) for 2-D rotations? After all, the $\measuredangle$ representation has the downside that it doesn't truly capture the geometry of the space.

Note: At this point, you may be asking yourself: why are we putting so much work into studying the $\odot$- and M-representations? It seems a lot simpler to just work with the θ representation by adding or subtracting 2π as needed. The reason is precisely that the $\measuredangle$ representation does not have the same underlying geometry as the group of 2-D rotations. Just one dimension higher, in 3-D, we will see that the equivalent $\measuredangle$-like representation is even more difficult to work with and requires even more special edge-case handling. Just as it is better to use the right screwdriver for the screw, working with a representation that [does] have the same geometry as the underlying space is more elegant and just makes sense. Studying these representations also leads us to the fundamental assumptions underlying Lie theory and reveals specific symmetries in the underlying geometric space which are applicable to higher-dimensional problems.

4.4.1 Velocity in the Isomorphic Representations

If we start at $[\theta]_\measuredangle$ and apply an angular velocity $[\omega]_{\frac{d}{dt}\measuredangle}$ for an infinitesimal time dt, we find that

$$\begin{aligned}\lim_{dt\to 0} \frac{([[\theta + [\omega]_{\frac{d}{dt}\measuredangle} \cdot dt]_\measuredangle)_M - ([\theta]_\measuredangle)_M}{dt} &= \omega \frac{d}{d\theta} \begin{bmatrix}\cos(\theta) & \sin(\theta)\end{bmatrix}_\odot \\ &= \omega \begin{bmatrix}-\sin(\theta) & \cos(\theta)\end{bmatrix}\end{aligned}$$

Plotting this vector on the unit circle $\odot$, we find that it is always tangent to the unit circle at the point $([\theta]_\measuredangle)_\odot$.

[Illustration]

This makes a lot of sense! Since the unit circle is [locally] Euclidean, we expect that any instantaneous motion will lie in the tangent space. And the

unit circle is smooth, so we expect the tangent space to vary smoothly with the angle $([\theta]_{\measuredangle}))_{\odot}$.

In fact, all of the tangent spaces are isomorphic to each other (they have the same structure): the tangent space at $([\theta]_{\measuredangle})_{\odot}$ is just the tangent space at $I = ([0]_{\measuredangle})_{\odot}$, rotated by θ! We can show this formally by computing the instantaneous velocity for M:

$$\begin{aligned}\lim_{dt\to 0}\frac{([[\theta+[\omega]_{\frac{d}{dt}\measuredangle}\cdot dt]_{\measuredangle})_M-([\theta]_{\measuredangle})_M}{dt} &= \omega\frac{d}{d\theta}\begin{bmatrix}\cos(\theta) & -\sin(\theta)\\ \sin(\theta) & \cos(\theta)\end{bmatrix}_M\\ &= \omega\begin{bmatrix}-\sin(\theta) & -\cos(\theta)\\ \cos(\theta) & -\sin(\theta)\end{bmatrix}\\ &= \omega\begin{bmatrix}\cos(\theta) & -\sin(\theta)\\ \sin(\theta) & \cos(\theta)\end{bmatrix}_M\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}_M\\ &= ([\theta]_{\measuredangle})_M\big([\omega]_{\frac{d}{dt}\measuredangle}\big)_{\frac{d}{dt}M}\end{aligned}$$

where we define:

$$\big([\omega]_{\frac{d}{dt}\measuredangle}\big)_{\frac{d}{dt}M} := \begin{bmatrix}0 & -\omega\\ \omega & 0\end{bmatrix}_{\frac{d}{dt}M}$$

and

$$[\omega]_{\frac{d}{dt}\measuredangle} \equiv \begin{bmatrix}0 & -\omega\\ \omega & 0\end{bmatrix}_{\frac{d}{dt}M}$$

In words, this means that: If an object is moving at a constant velocity $[\omega]_{\frac{d}{dt}\measuredangle}$, then rotating the object by $[\theta]_{\measuredangle}$ also rotates its [global] velocity (in M-space) by $[\theta]_{\measuredangle}$. Or, put another way, even though the object's [global] velocity in M-space is changing, its [local] velocity in M-space remains constant!

[Illustration]

These statements mathematically capture the intuition that the unit circle $\odot$ (and by extension, the group of 2-D rotations) is [rotationally] symmetric. The fact that $[\omega]_{\frac{d}{dt}\theta}$ does not change equivariantly with the angle θ is in fact a sign that the $\measuredangle$ space does [not] share an underlying geometry with the unit circle $\odot$. There are, of course, other kinds of [continuous symmetries] - and we will soon be leveraging the ideas developed here and in the next section in our study of 3-dimensional motion.

4.4.2 Integrating Velocity in M-space

Despite the fact that the [local] velocity vector in M-space is constant, integrating a velocity (applying a certain $[\omega]_{\frac{d}{dt}\measuredangle}$ over a period of time) appears to require working with the [global] velocity - which is not constant. Is the best approach to perform a numerical approximation by repeatedly applying:

$$([\theta]_{\measuredangle})_M \leftarrow ([\theta]_{\measuredangle})_M\big([\omega]_{\frac{d}{dt}\measuredangle}\big)_{\frac{d}{dt}M}$$

[Illustration]

It turns out that we can do better, by leveraging the group property of 2-D rotations! Suppose that an object starts at $R_{t=0} = I$, and moves at a constant velocity $[\omega]_{\frac{d}{dt}\measuredangle}$ for a time of T. If I can compute the rotation $R_{t=\frac{T}{2}}$, then I know that $R_{t=T} = R_{t=\frac{T}{2}}(R_{t=\frac{T}{2}})$: I traveled a distance of $R_{t=\frac{T}{2}}$ during the first half, and because a heading is itself a rotation, I can apply that same rotation again to go the rest of the way!

[Illustration]

This suggests a strategy for computing $R_{t=T}$: if we break T into infinitesimally small dt-sized chunks, we need only compute $R_{t=dt}$ and then

$$R_{t=T} = (R_{t=dt})^{T/dt}$$

As $dt \to 0$, we can use our result[link] from the previous section to calculate $[R_{t=dt}]_M = ([[\omega]_{\frac{d}{dt}\measuredangle} \cdot dt]_\measuredangle)_M$:

$$\lim_{dt\to 0} \frac{([[\theta + [\omega]_{\frac{d}{dt}\measuredangle} \cdot dt]_\measuredangle)_M - ([\theta]_\measuredangle)_M}{dt} = ([\theta]_\measuredangle)_M \left([\omega]_{\frac{d}{dt}\measuredangle}\right)_{\frac{d}{dt}M}$$

Or, letting $\theta = 0$:

$$\lim_{dt\to 0} \frac{([[\omega]_{\frac{d}{dt}\measuredangle} \cdot dt]_\measuredangle)_M - I_M}{dt} = \left([\omega]_{\frac{d}{dt}\measuredangle}\right)_{\frac{d}{dt}M}$$

$$\lim_{dt\to 0} \left[([[\omega]_{\frac{d}{dt}\measuredangle} \cdot dt]_\measuredangle)_M = I_M + \left([\omega]_{\frac{d}{dt}\measuredangle}\right)_{\frac{d}{dt}M} \cdot dt\right]$$

$$\lim_{dt\to 0} \left[([[\omega]_{\frac{d}{dt}\measuredangle} \cdot dt]_\measuredangle)_M = I_M + (dt) \begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix}_{\frac{d}{dt}M}\right]$$

As $dt \to 0$, this approximation becomes exact:

$$\begin{aligned}
[R_{t=T}]_M &= \lim_{dt\to 0} [(R_{t=dt})^{T/dt}]_M \\
&= \lim_{dt\to 0} \left(I_M + (dt) \begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix}_{\frac{d}{dt}M}\right)^{T/dt} \\
&= \exp\left(T \begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix}_{\frac{d}{dt}M}\right)
\end{aligned}$$

This is a remarkable formula: even though the instantaneous [global] velocity of an object moving at a constant velocity in M-space is constantly changing, we can compute its [global] position with only its [local] velocity!

There are several reasons for this, which we will discuss in more detail in the next section: there are several key assumptions that we have made while deriving this formula.

A quick sanity check verifies the formula: if an object starts at $[\theta]_\measuredangle = [0]_\measuredangle$ and moves at a constant velocity $[\omega]_{\frac{d}{dt}\measuredangle}$, then at time T it should be at $([[\omega]_{\frac{d}{dt}\measuredangle} \cdot T]_\measuredangle)_M$.

Letting

$$\widehat{\omega} := \begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix}_{\frac{d}{dt}M}$$

we have:

$$\begin{aligned}
[R_{t=T}]_M &= e^{\widehat{\omega}T} \\
&= \sum_{n=0}^{\infty} \left[\frac{1}{n!} (\widehat{\omega}T)^n \right] \\
&= I + (\omega T) \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} - \frac{1}{2!}(\omega T)^2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\
&\quad - \frac{1}{3!}(\omega T)^3 \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} + \frac{1}{4!}(\omega T)^4 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \cdots \\
&= \sum_{n=0}^{\infty} (-1)^n \left(\frac{(\omega T)^{2n}}{(2n)!} \right) \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\
&\quad + \sum_{n=0}^{\infty} (-1)^n \left(\frac{(\omega T)^{2n+1}}{(2n+1)!} \right) \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \\
&= \cos(\omega T) \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \sin(\omega T) \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \\
&= ([[\omega]_{\frac{d}{dt}\angle} \cdot T]_{\angle})_M
\end{aligned}$$

as expected.

To summarize, starting at $R_{t=0} = I$ and moving at constant velocity $[\omega]_{\frac{d}{dt}\angle}$, we can compute

$$[R_{t=T}]_M = ([[\omega]_{\frac{d}{dt}\angle} \cdot T]_{\angle})_M = \exp\left(T \begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix}_{\frac{d}{dt}M} \right)$$

If we instead start at some arbitrary $[\theta]_{\angle}$, we can use the group property of 2-D rotations to amend this formula: starting at $[\theta]_{\angle}$, we can advance our position by the same amount we would have travelled if we had started at the identity.

$$[R_{t=T}]_M = ([\theta + [\omega]_{\frac{d}{dt}\angle} \cdot T]_{\angle})_M = [R_{t=0}]_M \cdot \exp\left(T \begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix}_{\frac{d}{dt}M} \right)$$

We now return to the observation that, although the instantaneous velocity in M-space is constantly changing along the path, the final expression for $[R_{t=T}]_M$ only uses the instantaneous velocity at the identity

$$[\omega]_{\frac{d}{dt}\angle} \equiv \begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix}_{\frac{d}{dt}M}$$

What properties about 2-D rotations did we use in coming up with this formula? And does a similar formula exist for any set with such properties? We answer these questions presently.

4.5 Lie Groups and Lie Algebras

In the previous two sections, we derived two interesting properties about the group of 2-D rotations:

First:

> If an object is moving at a constant velocity $[\omega]_{\frac{d}{dt}\angle}$, then rotating the object by $[\theta]_\angle$ also rotates its [global] velocity (in M-space) by $[\theta]_\angle$. Or, put another way, even though the object's [global] velocity in M-space is changing, its [local] velocity in M-space remains constant!

[Illustration]

And second, we derived the formula:

$$[R_{t=T}]_M = \exp\left(T \begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix}\right)$$

which provides a mapping from an object's [local] velocity to its [global] position.

Together, these two statements hint at something deeper: the concept of [continuous symmetry]. As we move in M-space, the global velocity rotates in just the right way so that the local velocity remains constant. Is the property something special about 2-D rotations? Or are there other spaces with similar structure, for which we will be able to derive similar formulae?

One property, which is necessary to even be able to speak about velocities, is that 2-D rotations are [differentiable]: that is, the limit

$$\dot{x} = \lim_{dt \to 0} \frac{x(t+dt) - x(t)}{dt}$$

is always well-defined.

Even embedded within this property is yet another one: that the space of 2-D rotations is a [manifold]. That is, it is [locally Euclidean]: this means that the operation $x(t+dt) - x(t)$ is well-defined for sufficiently small dt.

The final property which enabled these derivations was the fact that 2-D rotations are a [group]. We used this in deriving mapping from local velocity to global position:

$$R_{t=T} = \lim_{dt \to 0} (R_{t=dt})^{T/dt}$$

Specifically, in taking this limit, we used the fact that 2-D rotations are [differentiable with respect to the group operation].

All together, we have following three properties:

1. Be a [group]
2. Be a [manifold]

3. Be [differentiable] with respect to the group operations

Any space which satisfies these three properties is termed a [Lie group], named after Sophus Lie, who studied these groups in the 19th century. It turns out that these three properties are all one needs about a space to derive that it has [continuous symmetry]: the [local] structure of the manifold is the same regardless of one's global position; therefore, moving [locally] in any direction has a symmetric effect regardless of global position; therefore, an object moving at a constant [local] velocity has a [global] velocity that changes [equivariantly] with (in the same way as) its [global] position.

[Illustration (sphere)]

Since the local structure of the manifold does not change with global position, it also has a special name: it is known as the corresponding [Lie algebra] of the Lie group. It is commonly referred to as the "tangent space at the identity": "Tangent space" implies that it can be used to represent infinitesimal positional changes around a point (and hence it is the appropriate space to express velocities), while "at the identity" defines a canonical coordinate system: all of the [local] tangent spaces are isomorphic - but, at the identity, the local velocity and the global velocity share the same coordinate system and are equivalent.

[Illustration (sphere)]

Putting this back into the context of 2-D rotations:

- The Lie Group of 2-D rotations is known as $SO(2)$, and consists of matrices of the form
$$\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix},$$
(or equivalently, matrices R where $R^T R = I_2$ and $\det(R) = +1$). This is indeed a manifold, a group (under matrix multiplication), and is differentiable with respect to the group operation.

- Its Lie algebra $so(2)$ consists of matrices of the form
$$\begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix},$$
(or equivalently, the set of 2x2 skew-symmetric matrices). We recognize this as the space of [local] velocities in M-space: $\left([\omega]_{\frac{d}{dt}\measuredangle}\right)_{\frac{d}{dt}M}$. As expected by the "tangent space at the identity" definition, it is equal to the [global] velocity space if starting at the identity rotation I_M. And, we have previously derived that

> rotating [an] object by $[\theta]_\measuredangle$ also rotates its [global] velocity (in M-space) by $[\theta]_\measuredangle$

just as we would expect from the general properties of a Lie group.

The mapping from the Lie algebra to the Lie group (from local velocity to position) is known as the [exponential map], while the inverse mapping from the

Lie group to the Lie algebra is known as the [logarithmic map]. The logarithmic map is not necessarily unique: for example, for 2-D rotations, we can get from $[0]_{\measuredangle}$ to $[\pi/2]_{\measuredangle}$ in one time-step by rotating at $[\pi/2]_{\frac{d}{dt}\measuredangle}$, but we could also move at $[-3\pi/2]_{\frac{d}{dt}\measuredangle}$ or $[7\pi/2]_{\frac{d}{dt}\measuredangle}$ to achieve the same thing.

[Illustration]

Tying it back to the group of 2-D rotations:

- The exponential map from velocities to positions $so(2) \to SO(2)$ is given by:
$$e^{\widehat{\omega}} = \cos(\omega)\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} + \sin(\omega)\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}$$
- The logarithmic map from positions to velocities $SO(2) \to so(2)$ is given by:
$$\omega = \arctan_2(\sin(\theta), \cos(\theta)) = \arctan_2(R_{21}, R_{11})$$

At the beginning, I also promised that Lie theory would help us to generalize to more complex, higher-dimensional cases. A famous result, by Élie Cartan, shows that, for any [matrix Lie Group] (one where composition is matrix multiplication and inversion is matrix inversion), the exponential map from the Lie Algebra to the Lie Group is given by the matrix exponential:

$$e^{At} = \sum_{n=0}^{\infty}\left[\frac{1}{n!}(At)^n\right]$$

I will make use of this in the subsequent sections to show the equivalence between the geometric and Lie-theoretic formulations for several problems.

5 2-D Pose

Now that we've established the theoretical foundations, let's now apply these tools to the more complex case of 2-D pose, which is the union of both 2-D position and orientation. In this section and the following ones, I'll be exploring the two questions laid out at the beginning of this post: how can I compute the poses along a path that maintains a constant (local) velocity? And, given a starting pose a and an ending pose b, how can I compute the local velocities required to get from a to b in a specified time T?

[Illustration]

The first thing is to show that 2-D poses are a Lie group. If this is true, then we will be able to exploit same symmetry properties as in our analysis of 2-D rotations! I will not prove this, but consider the following intuitive arguments:

- 2-D poses are a group.
  - A pose can also be interpreted as a transformation between poses: the position can be interpreted as a translation and the orientation

as a rotation. Applying rotations and translations to a object does not deform it; hence it will still have a valid 2-D pose at the end of the transformation.

- 2-D poses are a manifold (are locally Euclidean).
  - Consider that 2-D rotations are locally Euclidean and that 2-D translations are Euclidean; it makes sense that their union would inherit this property.
- The group operation (a combined rotation and translation) is differentiable.
  - We know from everyday experience that 2-D poses vary smoothly and can be infinitesimally interpolated. This supports the idea that the space of 2-D poses is differentiable.

[Illustration of group operation]

In particular, the Lie group properties simplify our analysis of the [velocity $\rightarrow$ pose] and [pose $\rightarrow$ velocity] mappings because we can always start at the identity pose!

Suppose that an object starts at the identity pose I, and that traveling at a certain local velocity g for a time T takes it to a pose $G(v, T)$. Then, we can we can compute the pose b achieved after starting a and moving at that same velocity g for a time T simply as $b = a(G(g, T))$.

[Illustration]

Similarly, to compute the local velocity g needed to go between two poses a and b in a certain time T, we can bring a to the identity pose I by composing it with a^{-1}. If we apply this also to b, then the local velocity g needed to get between I and $a^{-1}(b)$ in time T is precisely the same (local) velocity needed to go between a and b in time T!

[Illustration]

Since the space of 2-D poses has a real-world geometric interpretation, I'll first derive the [velocity $\rightarrow$ pose] and [pose $\rightarrow$ velocity] mappings with a geometric approach. I'll then delve into the Lie-theoretic formulation of this setting and show that it yields equivalent results!

5.1 Velocity $\rightarrow$ Pose: The Geometric Approach

The velocity of a 2-D object has two components: an angular velocity ω and a linear velocity v.

If we approximate the path of an object moving at constant linear and angular velocity, we find something that looks vaguely circular:

[Illustration]

In fact, if we were to take this to its limit and make our steps infinitesimally small, we would precisely get an arc of a circle! (A circle is a path of constant curvature.)

Suppose the object moves for a time T. Can we characterize the arc of the object's path?

[Illustration]

Firstly, we know that the object has rotated through an angle of $T\omega$. So we know that the angle of the arc is also $T\omega$:

$$\measuredangle AOB = T\omega$$

[Illustration]

Secondly, we know that the total length of the arc must be Tv, since the object was travelling at a constant linear velocity the whole time! So the radius of the arc is $(Tv)/(T\omega) = v/\omega$:

$$\overline{OA} = \overline{OB} = v/\omega$$

[Illustration]

This gives us all of the information that we need to compute the offset $\overrightarrow{AB} = \overrightarrow{OB} - \overrightarrow{OA}$.

[Illustration]

For a circular orbit, we know that the velocity is perpendicular to the radius. Therefore, if

$$v = \begin{bmatrix} v_x \\ v_y \end{bmatrix}$$

we should rotate it by 90 degrees to the right to get

$$v_\perp = \begin{bmatrix} v_y \\ -v_x \end{bmatrix}$$

Since $\overrightarrow{OA} \perp v$, we then have

$$v_\perp \propto \overrightarrow{OA} = \frac{1}{\omega} \begin{bmatrix} v_y \\ -v_x \end{bmatrix}$$

Then, $\overrightarrow{OB}$ is just $\overrightarrow{OA}$ rotated by $\measuredangle AOB$:

$$\overrightarrow{OB} = (\measuredangle AOB)(\overrightarrow{OA}) = ([T\omega]_\measuredangle)_M(\overrightarrow{OA}) = \frac{1}{\omega} \begin{bmatrix} \cos(T\omega) & -\sin(T\omega) \\ \sin(T\omega) & \cos(T\omega) \end{bmatrix} \begin{bmatrix} v_y \\ -v_x \end{bmatrix}$$

and

$$\begin{aligned} \overrightarrow{AB} &= \overrightarrow{OB} - \overrightarrow{OA} \\ &= (I_M - ([T\omega]_\measuredangle)_M)(\overrightarrow{OA}) \\ &= \frac{1}{\omega} \begin{bmatrix} 1 - \cos(T\omega) & -\sin(T\omega) \\ \sin(T\omega) & 1 - \cos(T\omega) \end{bmatrix} \begin{bmatrix} v_y \\ -v_x \end{bmatrix} \end{aligned}$$

Therefore, if we start at $(x, y) = (0, 0)$, pointing along the $+x$-axis ($\theta = 0$), then at time T we will have heading

$$\theta = T\omega$$

and position

$$\begin{bmatrix} x \\ y \end{bmatrix} = \frac{1}{\omega} \begin{bmatrix} 1-\cos(T\omega) & -\sin(T\omega) \\ \sin(T\omega) & 1-\cos(T\omega) \end{bmatrix} \begin{bmatrix} v_y \\ -v_x \end{bmatrix}$$

5.2 Velocity → Pose: The Lie Group Approach

Concretely, the Lie Group $SE(2)$ is a [matrix Lie group], consisting of matrices of the form:

$$\begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$$

where $R \in SO(2)$ is a 2-D rotation matrix, and t represents a 2-D translation.

[Show closure under inversion and multiplication]

What is the Lie Algebra? $se(2)$ is comprised of matrices of the form:

$$\begin{bmatrix} \widehat{\omega} & v \\ 0 & 0 \end{bmatrix}$$

where $\widehat{\omega}$ is defined as in section [link] for 2-D rotations

$$\widehat{\omega} := \begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix}$$

and v is the local linear velocity.

Applying Cartan's formula for matrix Lie groups, we find the exponential map to be:

$$\begin{aligned} \exp\left(T \begin{bmatrix} \widehat{\omega} & v \\ 0 & 0 \end{bmatrix}\right) &= \sum_{n=0}^{\infty} \frac{1}{n!} \begin{bmatrix} T\widehat{\omega} & Tv \\ 0 & 0 \end{bmatrix}^n \\ &= \begin{bmatrix} I & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} T\widehat{\omega} & Tv \\ 0 & 1 \end{bmatrix} + \frac{1}{2!} \begin{bmatrix} (T\widehat{\omega})^2 & (T\widehat{\omega})Tv \\ 0 & 0 \end{bmatrix} + \frac{1}{3!} \begin{bmatrix} (T\widehat{\omega})^3 & (T\widehat{\omega})^2 Tv \\ 0 & 0 \end{bmatrix} + \cdots \\ &= \begin{bmatrix} e^{T\widehat{\omega}} & t \\ 0 & 1 \end{bmatrix} \end{aligned}$$

where

$$t := 0 + Tv + \frac{1}{2!}(T\widehat{\omega})Tv + \frac{1}{3!}(T\widehat{\omega})^2 Tv + \cdots$$

We can simplify t with a little bit of algebra:

$$\begin{aligned} (T\widehat{\omega})t &= (T\widehat{\omega})Tv + \frac{1}{2!}(T\widehat{\omega})^2 Tv + \frac{1}{3!}(T\widehat{\omega})^3 Tv + \cdots \\ \widehat{\omega}t &= (T\widehat{\omega})v + \frac{1}{2!}(T\widehat{\omega})^2 v + \frac{1}{3!}(T\widehat{\omega})^3 v + \cdots \\ \widehat{\omega}t + v &= v + (T\widehat{\omega})v + \frac{1}{2!}(T\widehat{\omega})^2 v + \frac{1}{3!}(T\widehat{\omega})^3 v + \cdots \\ &= \left(I + T\widehat{\omega} + \frac{1}{2!}T\widehat{\omega}^2 + \frac{1}{3!}T\widehat{\omega}^3\right) v \\ &= e^{T\widehat{\omega}} v \\ t &= \widehat{\omega}^{-1}(e^{T\widehat{\omega}} - I)v \end{aligned}$$

Therefore we have:

$$\exp\left(T\begin{bmatrix}\widehat{\omega} & v\\ 0 & 0\end{bmatrix}\right) = \begin{bmatrix}e^{T\widehat{\omega}} & \widehat{\omega}^{-1}(I - e^{T\widehat{\omega}})v\\ 0 & 1\end{bmatrix}$$

where $e^{T\widehat{\omega}}$ was previously found to be

$$e^{T\widehat{\omega}} = \begin{bmatrix}\cos(T\omega) & -\sin(T\omega)\\ \sin(T\omega) & \cos(T\omega)\end{bmatrix}$$

Note that the expression for t is equivalent to the expression for $\overrightarrow{AB}$ found in the previous section! And the expression for R is equivalent to $[T\omega]_{\measuredangle}$. Therefore, using the geometric and Lie-theoretic approaches has leads us to the same answer for integrating velocities to positions, as one would expect.

5.3 Pose → Velocity: The Geometric Approach

Let's now consider the inverse problem: we have a target pose (θ, t) and need to compute the local velocity that would be needed to go from I to that target pose in time T.

Since the linear velocity v does not affect the angle θ, we must have

$$\omega = \theta/T$$

From section [link], we know that the path is a circular arc with angle θ.

[Illustration]

Letting M be the midpoint of AB, we have $\overrightarrow{AM} = t/2$.

Since $\overline{MO} \perp \overline{AM}$, we know that it is in the direction of $\begin{bmatrix}-t_y\\ t_x\end{bmatrix}$. And trigonometry gives us that $\overline{AM}/\overline{MO} = \tan(\theta/2)$. Putting these together gives:

$$\overrightarrow{MO} = \frac{1}{\tan(\theta/2)}\left(\frac{1}{2}\begin{bmatrix}-t_y\\ t_x\end{bmatrix}\right)$$

So,

$$\begin{aligned}\overrightarrow{AO} &= \overrightarrow{AM} + \overrightarrow{MO}\\ &= \frac{1}{2}\left(\begin{bmatrix}t_x\\ t_y\end{bmatrix} + \frac{1}{\tan(\theta/2)}\begin{bmatrix}-t_y\\ t_x\end{bmatrix}\right)\end{aligned}$$

[Illustration]

As before, we can use the fact that the velocity is perpendicular to the radius: this gives us the direction of v. And we know its magnitude: the total path length is $\theta(\overline{AO})$, so the speed is $\theta(\overline{AO})/T$.

Putting this together, we have

$$v = \frac{\theta}{T}(\overrightarrow{AO})_{\perp} = \frac{\theta}{2T}\left(\begin{bmatrix}t_y\\ -t_x\end{bmatrix} + \frac{1}{\tan(\theta/2)}\begin{bmatrix}t_x\\ t_y\end{bmatrix}\right)$$

5.4 Pose → Velocity: The Lie Theory Approach

This is pretty easy! Recall that $SE(2)$ was defined as the space of rotations

$$\begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$$

where R was the matrix representation of an angle θ.

We previously found that the mapping from velocity to pose was given by:

$$\exp\left(T\begin{bmatrix} \widehat{\omega} & v \\ 0 & 0 \end{bmatrix}\right) = \begin{bmatrix} e^{T\widehat{\omega}} & \widehat{\omega}^{-1}(I - e^{T\widehat{\omega}})v \\ 0 & 1 \end{bmatrix}$$

Inverting this, we find that, if $R = e^{\theta}$, we must have $\omega = \theta/T$.

Going from t to v requires a bit more algebra:

$$\begin{aligned}
t &= \widehat{\omega}^{-1}(I - e^{T\widehat{\omega}})v \\
&= (\widehat{\theta/T})^{-1}(I - e^{T(\widehat{\theta/T})})v \\
&= T(\widehat{\theta})^{-1}(I - e^{\widehat{\theta}})v \\
v &= (I - e^{\widehat{\theta}})^{-1}(\widehat{\theta})T^{-1}t \\
&= \begin{bmatrix} 1-\cos(\theta) & -\sin(\theta) \\ \sin(\theta) & 1-\cos(\theta) \end{bmatrix}^{-1} \begin{bmatrix} 0 & -\theta \\ \theta & 0 \end{bmatrix} T^{-1}t \\
&= \frac{1}{(1+\cos(\theta))^2 + \sin^2(\theta)} \begin{bmatrix} 1-\cos(\theta) & \sin(\theta) \\ -\sin(\theta) & 1-\cos(\theta) \end{bmatrix} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \frac{\theta}{T}t \\
&= \frac{\theta}{T}\left(\frac{1}{2-2\cos(\theta)}\right) \begin{bmatrix} \sin(\theta) & \cos(\theta)-1 \\ 1-\cos(\theta) & \sin(\theta) \end{bmatrix} \begin{bmatrix} t_x \\ t_y \end{bmatrix} \\
&= \frac{\theta}{2T}\left(\frac{1}{1-1\cos(\theta)}\right) \begin{bmatrix} \sin(\theta)t_x - (1-\cos(\theta))t_y \\ (1-\cos(\theta))t_x + \sin(\theta)t_y \end{bmatrix} \\
&= \frac{\theta}{2T}\left(\begin{bmatrix} -t_y \\ t_x \end{bmatrix} + \frac{\sin(\theta)}{1-\cos(\theta)} \begin{bmatrix} t_x \\ t_y \end{bmatrix}\right)
\end{aligned}$$

This is equivalent to the expression found in the previous section! We have now shown the equivalence of the geometric and Lie-theoretic approaches for the space of 2-D poses.

6 3-D Rotations

Working towards 3-D poses, we know add one more [spatial] dimension to 3-D rotations. However, in moving from 2-D to 3-D rotations, we have in fact added [two] degrees of freedom!

[Illustration]

This is pretty weird. In fact, it requires re-thinking the way that we think about rotations.

6.1 Problems with the ∡ Representation

In the section on 2-D rotations [link], I hinted that using the ∡ representation would extend poorly to multiple dimensions. Let's take a moment to see why.

Extending the ∡ representation to multiple dimensions, we can try to represent an object's 3-D orientation in terms of three angles θ_x, θ_y, and θ_z.

1. The first problem is shared by the 2-D case, is that the representation is redundant: for each dimension, $\theta \equiv \theta + 2\pi k$ for any integer k. Canonical?

2. The second problem is that the ordering matters: rotating around the z-axis and then the x-axis is [not] the same as rotating around the x-axis and then the z-axis.

   [Illustration]

   Intuitively, the axes should be symmetric. But, for the $\theta_{x,y,z}$ representation, we have to define an ordering where one comes first.

3. The third problem, which is connected to the second, is that of further redundancy: rotating -90 degrees around x and then 90 degrees around y is equal to rotating 90 degrees around y and then 90 degrees around z.

   [Illustration]

   In the case of the cyclical 2π redundancy, we could choose a canonical range of $-\pi$ to π. But, it would be very difficult to choose a "canonical" representation for an orientation under this degree of redundancy, if such a thing is even possible.

4. The fourth problem is the famous phenomenon of "gimbal lock". Strictly speaking, this does not apply to the representation that I have described here, as the representation which results in gimbal lock is sightly different. However, because this problem is so ubiquitous in robotics, aeronautics, and gaming and animation, it's worth a quick detour: it showcases the practical implications of using a representation which does not respect the underlying geometry of 3-D rotations.

   Gimbal lock occurs when we try to implement a physical system with 3 degrees of freedom. Recall that addressing the second problem requires defining a canonical order of rotations. Deciding to perform rotations in the order x, then y, then z, we first add a joint which rotates in the x-axis. Then, we add a joint which rotates in the [local] y-axis, and then another joint which rotates in the [local] z-axis. This is known as the [Euler angle] representation; the latter two axes being local rather than global differentiates the Euler angle representation from the $\theta_{x,y,z}$ representation discussed above.

   [Illustration]

   Unfortunately, although any 3-D rotation is achievable with this configuration, it has a special case where we actually lose one degree of freedom.

If we rotate by -90 degrees around the y-axis, then the x-axis of the first joint and the [local] z-axis of the second joint are exactly aligned! So, we have lost the ability to rotate about the [global] z-axis.

[Illustration]

The practical implications of this can be fatal. A pilot would not be able to steer in one direction. A robot starting at this configuration and commanded to rotate about the [global] z-axis could output infinite velocities: because nothing the controller does is able to effect the desired rotation, the output is arbitrary and could be strongly impacted by numerical instabilities.

This illustrates the difficulties in using triple-axis representations for 3-D orientation. Of course, it can be done, and Euler angles are intuitive and used extensively, but doing so requires special care and attention. Furthermore, because these representations do not respect the underlying geometry, we cannot use them to express local velocities or use any of our tools from Lie theory.

Is there another way to think about 3-D rotations, one which does respect the underlying geometry? In order to do this, we must think more deeply about the space of 3-D rotations itself.

6.2 What [are] 3-D Rotations?

There are a few ways to define 3-D rotations: a geometric interpretation is to define 3-D rotations as [the group of symmetries of the sphere]. Any rotation of a sphere does not change it; this is clearly not true for other transformations, such as translation or any non-rigid shearing, squashing, or scaling.

However, this definition alone does not yield a representation for 3-D rotations: the group of 3-D rotations is not isomorphic to the sphere in three dimensions. This is quickly shown: pick any point p on the sphere. Then, any rotation around the diameter passing through p does not change p. Therefore, a point p on the sphere does not uniquely identify a rotation.

[Illustration]

But, this does lead us to one representation: a 3-D rotation can be represented by a 3-D axis of rotation α and an angle θ, where we constrain $||\alpha|| = 1$. This is known as the [axis-angle] representation, and it is the proper 3-D analogue to the 2-D $\measuredangle$ representation. Note that the axis of rotation α is defined by the characteristic that any point along the axis of rotation is left unchanged by the rotation.

This representation, however, does suffer from a number of redundancies:

1. As in the $\measuredangle$ representation, $\theta \equiv \theta + 2\pi$.
2. $(\alpha, \theta) \equiv (-\alpha, -\theta)$.
3. $(\alpha_1, 0) \equiv (\alpha_2, 0)$ for any α_1, α_2.

Therefore, despite its neat geometrical interpretation, the axis-angle representation is not a Lie-theoretic representation which shares a geometry with the space of 3-D rotations.

A different representation comes from analyzing the action of a rotation on the three coordinate axes. This derivation follows [MLS].

One way to represent a linear operation is by its action on the standard basis vectors $e_1 = [1, 0, 0]$, $e_2 = [0, 1, 0]$, and $e_3 = [0, 0, 1]$.

Suppose that, after a rotation, $[1, 0, 0]$ has been mapped to $\boldsymbol{x}$, $[0, 1, 0]$ has been mapped to $\boldsymbol{y}$, and $[0, 0, 1]$ has been mapped to $\boldsymbol{z}$. That is, the rotation is represented in matrix form as

$$R = \begin{bmatrix} \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} \end{bmatrix}$$

Because rotations preserve angles, we must still have that $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{z}$ are mutually orthogonal. And because rotations preserve lengths, we know that $||x|| = ||y|| = ||z|| = 1$.

Put in other terms,

$$\begin{aligned} R^T R &= \begin{bmatrix} \boldsymbol{x^T} \\ \boldsymbol{y^T} \\ \boldsymbol{z^T} \end{bmatrix} \begin{bmatrix} \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= I_3 \end{aligned}$$

There is one additional requirement for R to be a rotation matrix: consider the reflection over the xz-plane:

$$\begin{bmatrix} \boldsymbol{x} & \boldsymbol{y} & \boldsymbol{z} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

This also satisfies $R^T R = I$: the axes remain mutually orthogonal and lengths are preserved. Therefore, in order to ensure that R is a rotation and not a reflection, we need the additional constraint that $\det(R) = +1$. (Recall that the determinant is a [signed] volume and therefore contains information about the relative orientation of the axes).

This is the [matrix] representation for rotations. By construction, it is convenient apply a rotation R to a point p: one simply needs to perform the matrix multiplication Rp. Composition is also performed via matrix multiplication: since $R_a(R_b p) = (R_a R_b)p$, $R_a(R_b) = R_a R_b$. And, the matrix representation is the one used in Lie theory, as it is isomorphic to the underlying geometry. Specifically, the group of 3-D rotations $SO(3)$ is defined as:

$$SO(3) := \{R : R^T R = I_3, \det(R) = +1\}$$

We will be working with these two representations (axis-angle and matrix) in the next sections.

(Aside: there is another major representation for 3-D rotations, known as the [unit quaternions]. These are isomorphic to the sphere in four dimensions, and "double-cover" the space of 3-D rotations, in the sense that there are exactly two quaternions associated with each 3-D rotation. Like the Euler angles, they have also found extensive usage in computer graphics and robotics. However, they are outside the scope of this post.)

6.3 The Axis-Angle Representation

The axis-angle representation is unfortunately not isomorphic to the geometry of 3-D rotations, due to the fact that $\theta \equiv \theta + 2\pi$, as in the $\measuredangle$ representation.

However, like the $\measuredangle$ representation, the axis-angle representation for 3-D orientations is well-suited to working with velocities. In this representation, an angular velocity is identified by an axis of rotation α and a rotational velocity ω; then the rotation after time T is that with axis of rotation α and angle $T\omega$. Inversion is similarly simple.

The major difficulty with this representation comes from the question: How does one compose rotations? Even the proof that composing two axis-angle rotations yields another valid axis-angle rotation is non-trivial (and is known as Euler's rotation theorem). Concretely, given two rotations (α_1, θ_1) and (α_2, θ_2), it is unclear how to compute the axis of rotation α_3 and rotation angle θ_3 that comes from composing $(\alpha_1, \theta_1)(\alpha_2, \theta_2)$.

This operation is so tantalizingly simple in the matrix representation. Is there any way to convert between the two?

It turns out that there is! We know that applying a rotation R to a point p is achieved by Rp. If we can figure out how to apply an axis-angle rotation (α, θ) to a point p (yielding p'), then we can hopefully use that to reverse-engineer the R which maps from $p \to p'$.

First, recall that points along the axis of rotation α are unaffected by the rotation. If we decompose p into a component parallel to α $p_{||}$ and a component perpendicular to α $p_\perp$, then $p_{||}$ will be unaffected by the rotation and we only need to worry about the $p_\perp$ component.

[Illustration]

If we let the rotation of $p_\perp$ around α $p'_\perp$, then we have that $p' = p_{||} + p'_\perp$.

But how do we rotate $p_\perp$ around an arbitrary axis α? To tackle this, we need another tool: the cross product $\times$.

Geometrically speaking, the cross product $a \times b$ is an operation which yields a vector c which is perpendicular to both a and b with length $||c|| = ||a||||b|| \sin(\theta)$, where θ is the angle from a to b. The orientation of c is given by the so-called right-hand-rule: if you curl your fingers from a towards b and stick your thumb up, then c will be in the direction that your thumb is pointing.

[Illustration]

Algebraically, speaking $a \times b$ is implemented as follows:

$$a \times b = \begin{bmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{bmatrix} = \begin{bmatrix} 0 & -a_3 & a_1 \\ a_3 & 0 & -a_2 \\ -a_1 & a_2 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \widehat{a}b$$

where

$$\widehat{a} := \begin{bmatrix} 0 & -a_3 & a_1 \\ a_3 & 0 & -a_2 \\ -a_1 & a_2 & 0 \end{bmatrix}$$

It is left to the inquisitive reader to show the equivalence of the geometric and algebraic interpretations.

Let's now return to our original problem: how can we rotate $p_\perp$ by θ around α? The vector yielded by the cross product $\alpha \times p_\perp$ has length $||p_\perp||$ (because $||\alpha|| = 1$ and $\alpha \perp p_\perp$ by construction) and is precisely $p_\perp$ rotated by $\pi/2$ around α.

[Illustration]

We now have two orthogonal vectors in the plane of rotation. Thinking back to the case of 2-D rotations, we can think of $p_\perp$ as being in the direction the $+x$-axis, and $\alpha \times p_\perp$ as being in the direction of the $+y$-axis.

[Illustration]

This lets us $p'_\perp$ as a linear combination of $p_\perp$ and $\alpha \times p_\perp$!

$$\begin{aligned} p'_\perp &= \cos(\theta)p_\perp + \sin(\theta)(\alpha \times p_\perp) \\ p' &= p_{||} + \cos(\theta)p_\perp + \sin(\theta)(\alpha \times p_\perp) \\ &= (p \cdot \alpha)\alpha + \cos(\theta)(p - (p \cdot \alpha)\alpha) + \sin(\theta)(\alpha \times (p - (p \cdot \alpha)\alpha)) \\ &= p\cos(\theta) + (1 - \cos(\theta))(p \cdot \alpha)\alpha + (\alpha \times p)\sin(\theta) \end{aligned}$$

This is known as [Rodrigues' rotation formula], named after the French mathematician Olinde Rodrigues.

It still remains, however, to recover the matrix R which yields the same rotation from $p \to p'$. Luckily, this is achieved with a few more algebraic tricks.

$$\begin{aligned} p' &= p\cos(\theta) + (1 - \cos(\theta))(p \cdot \alpha)\alpha + (\alpha \times p)\sin(\theta) \\ &= p\cos(\theta) + (1 - \cos(\theta))(p - p_\perp) + \widehat{\alpha}p\sin(\theta) \\ &= p\cos(\theta) + (1 - \cos(\theta))(p + \alpha \times (\alpha \times p)) + \widehat{\alpha}p\sin(\theta) \\ &= p\cos(\theta) + (1 - \cos(\theta))p + (1 - \cos(\theta))(\widehat{\alpha}\widehat{\alpha}p) + \widehat{\alpha}p\sin(\theta) \\ &= (I + \widehat{\alpha}\sin(\theta) + (1 - \cos(\theta))\widehat{\alpha}\widehat{\alpha})p \end{aligned}$$

Thus rotating a point p around an axis α by an angle θ is equivalent to rotating it by the matrix

$$R = (I + \widehat{\alpha}\sin(\theta) + (1 - \cos(\theta))\widehat{\alpha}\widehat{\alpha})$$

Therefore, to compose two axis-angle rotations (α_1, θ_1) and (α_2, θ_2), one need only compute the equivalent rotation matrices R_1 and R_2, multiply them to get $R_3 = R_1 R_2$, ...and then convert this back into the axis-angle form (α_3, θ_3).

What is this inverse formula, the one that brings a rotation matrix R to an axis-angle (α, θ)? Geometrically speaking, we know that the rotation R leaves the axis of rotation α unchanged, that is, that $R\alpha = \alpha$.

We can use some clever algebra to show that $\widehat{\alpha} \propto R - R^T$:

$$\begin{aligned} R\alpha &= \alpha \\ R^T R\alpha &= R^T \alpha \\ \alpha &= R^T \alpha \\ R\alpha &= R^T \alpha \\ (R - R^T)\alpha &= 0 \end{aligned}$$

Since $R - R^T$ is a skew-symmetric matrix, then $(R - R^T)\alpha = (\widehat{a})\alpha = a \times \alpha$ for some vector a.

But, since $(R - R^T)\alpha = 0$,

$$||a \times \alpha|| = ||a||||\alpha|| \sin(\measuredangle(a, \alpha)) = 0$$

Assuming that $R - R^T \neq 0$, we then have that $\sin(\measuredangle(a, \alpha)) = 0$, or that $\measuredangle(a, \alpha) = 0$, or that a is parallel to α! Therefore, $\widehat{\alpha} \propto R - R^T$. Or, equivalently,

$$\alpha \propto (R - R^T)^\vee$$

where $\vee$ is defined to be the inverse of the $\wedge$ operator:

$$(\widehat{a})^\vee := a$$

Then, to get the angle of rotation, we can simply pick any point p not on the axis of rotation α. Then, the angle θ is equal to the angle between $p_\perp$ and $Rp_\perp$, where $p_\perp \perp \alpha$ and may be obtained by $p_\perp \leftarrow p - p_{||} = p - (p \cdot \alpha)\alpha$ or $p_\perp \leftarrow p \times \alpha$.

Thus, for the axis-angle representation for 3-D rotations, we have shown how to:

- convert from velocities to rotations and vice-versa,
- apply rotations $p' = (\alpha, \theta)p$ via conversion to the equivalent matrix representation R, and
- compose rotations $(\alpha_3, \theta_3) = (\alpha_1, \theta_1)(\alpha_2, \theta_2)$ via conversion to the equivalent matrix representations R_1 and R_2, composing $R_3 = R_1 R_2$ in the matrix space, and then converting R_3 back to an axis-angle.

6.4 The Matrix Representation

As previously mentioned, the matrix representation $SO(3)$ is the one used in Lie theory because it reflects the underlying geometry of 3-D rotations.

The primary question, therefore, is what does the Lie algebra $so(3)$ look like for this space?

It turns out that $so(3)$ is the space of 3-dimensional skew-symmetric matrices! This can be seen through a clever bit of algebraic trickery. Suppose that the rotation R is a function of time t. Then, for any time t, we have:

$$R(t)^T R(t) = I$$

Differentiating this with respect to time t yields:

$$\dot{R}(t)^T \dot{R}(t) + R(t)^T R(t) = 0$$

Or,

$$R(t)^T \dot{R}(t) = -\left(R(t)^T \dot{R}(t)\right)^T$$

This implies that $R(t)^T \dot{R}(t)$ is skew-symmetric, or that

$$R(t)^T \dot{R}(t) = \widehat{a}$$

for some vector a. Recall that $\widehat{a}$ was defined as:

$$\widehat{a} := \begin{bmatrix} 0 & -a_3 & a_1 \\ a_3 & 0 & -a_2 \\ -a_1 & a_2 & 0 \end{bmatrix}$$

If we now let $R(t) = I$, we then have that the instantaneous velocity vector $\dot{R}(t)$ is equal to a skew-symmetric matrix $\widehat{a}$. Therefore, the instantaneous velocity vector at the identity (and hence the Lie algebra) has the form of a skew-symmetric matrix!rendering

Let us now apply Cartan's formula for the exponential map to find how to integrate

One of the interesting characteristics about 3-D pose is that any transformation between two poses can themselves be represented by another pose. What do I mean by this?

To get from one 3-D pose to another (let's say 'A' and 'B'), I can first rotate 'A' so that it is aligned with the rotation of 'B'. I can then translate this rotationally aligned version of 'A' to the position of 'B' to complete the transformation. So a transformation between 3-D poses also consists of a rotation and translation, just the same as a 3-D pose itself! [Illustration]

This characteristic, that the transformation between 3-D poses is itself a kind of 3-D pose,

The special Euclidean group $SE(3)$ is the set of all rigid-body motions in three dimensions. We are all intimately familiar with it: whenever we interact

with any rigid object, by picking up dishes or playing with a toy duck, we are sub-consciously dealing with elements of $SE(3)$. It is no surprise, then, that $SE(3)$ has been extensively studied in robotics, graphics, and physics.

We deal with $SE(3)$ so often, and in such a casual and everyday way, that we take our intuitions about $SE(3)$ granted. For example, it turns out that $SE(3)$ is a mathematical group, which just means that it has the following three properties:

1. Identity: There is an identity rigid-body motion. That is, it is possible to not move a rubber duck.

2. Inversion: A rigid-body motion can be un-done. That is, if I move a rubber duck without deforming it, I can move it back to its starting pose without deforming it.

3. Composition: Any rigid-body motion, followed by another rigid-body motion, is itself a valid rigid-body motion. That is, if I move a rubber duck from pose A to pose B without deforming it, and then move it from pose B to pose C without deforming it, I can move it from pose A to pose C without deforming it.

4. Associativity: If I have three rigid body motions A, B, and C, then $(A \odot B) \odot C = A \odot (B \odot C)$.

Figure 1. A rubber duck undergoing moving in $SE(3)$.

These properties are so obvious to us that we never consciously think about them in everyday life. However, the fact that they are all true allows us to apply results from mathematical group theory to the study of rigid-body motions.

In addition to being a group, $SE(3)$ is also a differentiable manifold. This means that our intuitions about smooth surfaces and Euclidean space apply: a manifold is defined as a space which is *locally* Euclidean; and the fact that it is differentiable means that the Inversion and Composition operations are smooth. For example, since the Inversion operation is smooth, if I move my rubber duck from pose A to pose B, the movement to go from B back to A is similar to the movement going from B' to A for any B' that is "close" to B. A more formal treatment is given in Appendix A of [MLS].

These two properties of $SE(3)$ (that is is a group, and that it is a differentiable manifold), means that it is a **Lie group**. This means that it has an associated **Lie algebra** $se(3)$, which is simply the tangent space at the identity element; the Lie algebra fully captures the local structure and symmetries of the group. The Lie algebra is therefore the natural space for expressing velocities and infinitesimal motions in a Lie group (Figure 2). A full treatment of Lie group theory is far beyond the scope of this work [citation needed]. However, I will give one motivating example to illustrate its usefulness.

Figure 2. The Lie Group S^1 and its Lie Algebra.

6.5 The Duck Game

Suppose that I am playing a game: I have a motorized toy duck, and it needs to "rescue" some bread from a burning building at precisely $t = 3$. At $t = 0$, I put the duck on the ground and set moving at a constant linear and angular velocity. At $t = 8$, the duck stops moving. Unfortunately, I was busy "rescuing" a sandwich at $t = 3$ and I forgot to record the linear and angular velocities of the duck! But I can measure the duck's pose at $t = 8$. How can I determine if the duck was close to the bread at $t = 3$?

Figure 3. A motorized duck "rescues" bread.

One way to solve this problem is to compare the the duck's poses at $t = 0$ and $t = 8$: let these be $G_{t=0} \in SE(2)$ and $G_{t=8} \in SE(2)$. I can find the point $g_{t=8} \in se(2)$ which corresponds to $G_{t=8}$ in the tangent space of $G_{t=0}$, and do linear interpolation between $g_{t=0}$ and $g_{t=8}$ to get $g_{t=3}$. This can then be mapped back into a pose $G_{t=3}$, which gives me the position of my duck at $t = 3$!

Figure 4. Solving the duck problem using the Lie algebra of $SE(2)$.

7 Preliminaries

7.1 Representation of $SE(3)$

Elements of $SE(3)$ can be represented as

$$G = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix},$$

where R is a rotation matrix with the properties: $R^T R = I$, $\det(R) = +1$.

This is convenient, because the group properties under this representation are familiar: the identity element is I_4, inversion is performed by taking the matrix inverse, and composition is achieved through matrix multiplication.

The action of this rigid body motion is a rotation about an axis ω by an angle θ (where ω passes through the origin), followed by a translation of t.

7.2 Representation of $se(3)$

Elements of $se(3)$, the Lie algebra of $SE(3)$, can be represented as

$$g = \begin{bmatrix} \theta\widehat{\omega} & v \\ 0 & 0 \end{bmatrix},$$

where $||\omega||_2 = 1$ and $\widehat{\omega}$ is defined as:

$$\widehat{\omega} := \begin{bmatrix} 0 & -\omega[2] & \omega[1] \\ \omega[2] & 0 & -\omega[0] \\ -\omega[1] & \omega[0] & 0 \end{bmatrix}.$$

ω and θ have semantic meanings: the corresponding element of $G \in SE(3)$ has axis of rotation ω with an angle of rotation θ.

8 The Exponential map: $se(3) \to SE(3)$

8.1 Definition

Under this representation for $SE(3)$, we can compute the mapping from $se(3)$ to $SE(3)$ using the matrix exponential:

$$\exp\left(\begin{bmatrix}\theta\widehat{\omega} & v\\ 0 & 0\end{bmatrix}\right) = I_4 + \begin{bmatrix}\theta\widehat{\omega} & v\\ 0 & 0\end{bmatrix} + \frac{1}{2!}\left(\begin{bmatrix}\theta\widehat{\omega} & v\\ 0 & 0\end{bmatrix}\right)^2 + \frac{1}{3!}\left(\begin{bmatrix}\theta\widehat{\omega} & v\\ 0 & 0\end{bmatrix}\right)^3 + \cdots$$
$$= \begin{bmatrix}I_3 & 0\\ 0 & 1\end{bmatrix} + \begin{bmatrix}\theta\widehat{\omega} & v\\ 0 & 0\end{bmatrix} + \frac{1}{2!}\begin{bmatrix}(\theta\widehat{\omega})^2 & (\theta\widehat{\omega})v\\ 0 & 0\end{bmatrix} + \frac{1}{3!}\begin{bmatrix}(\theta\widehat{\omega})^3 & (\theta\widehat{\omega})^2 v\\ 0 & 0\end{bmatrix} + \cdots$$

The bottom row is trivial: the terms are 0 and 1.

The top-left component expands to $\exp(\theta\widehat{\omega})$, which has a closed-form given by Rodrigues' formula:

$$R = I_3 + \widehat{\omega}\sin(\theta) + \widehat{\omega}^2(1-\cos(\theta))$$

.

The tricky part is the top-right component:

$$v + \frac{1}{2!}(\theta\widehat{\omega})v + \frac{1}{3!}(\theta\widehat{\omega})^2 v) + \cdots$$

.

8.2 The tricky part

The problem is that we are missing a $\theta\widehat{\omega}$ term: multiplying by $\theta\widehat{\omega}$ yields:

$$(\theta\widehat{\omega})v + \frac{1}{2!}(\theta\widehat{\omega})v^2 + \frac{1}{3!}(\theta\widehat{\omega})^3 v) + \cdots = (\exp(\theta\widehat{\omega}) - I_3)v$$

.

However, we can't simply factor out $\theta\widehat{\omega}$ because $\widehat{\omega}$ is not invertable. It has rank 2, and ω is in its nullspace (thus it spans $\omega^\perp$).

But, we can do the next best thing. Let's factor v into components parallel to ω and perpendicular to ω:

$$\begin{aligned} v &:= v_{||} + v_\perp \\ &:= c\omega + \theta\widehat{\omega}v'. \end{aligned}$$

Then we have:

$$\begin{aligned} v + \frac{1}{2!}(\theta\widehat{\omega})v + \frac{1}{3!}(\theta\widehat{\omega})^2 v) + \cdots &= (c\omega + \theta\widehat{\omega}v') + \frac{1}{2!}(\theta\widehat{\omega})(c\omega + \theta\widehat{\omega}v') + \frac{1}{3!}(\theta\widehat{\omega})^2(c\omega + \theta\widehat{\omega}v')) + \cdots \\ &= c\omega + (\theta\widehat{\omega})v' + \frac{1}{2!}(\theta\widehat{\omega})v'^2 + \frac{1}{3!}(\theta\widehat{\omega})^3 v') + \cdots \\ &= c\omega + (\exp(\theta\widehat{\omega}) - I_3)v' \\ &= c\omega + (R - I)v' \end{aligned}$$

8.3 Finding v'

How can we efficiently compute v'?

First, note that $\theta\widehat{\omega}v' = \theta(\omega \times v') = v_\perp$. This implies that $v' \perp v_\perp$.

Secondly, note that we can let $v' \perp \omega$ without loss of generality: If v' has a component in the direction of ω, it will be zeroed when computing $\omega \times v'$ in Equation(X). Note also that this will not affect the $(R - I)v' = Rv' - v'$ term in Equation(X): Rv' is v' rotated about the axis ω by θ. The component of v' parallel to ω is not affected by this; hence that term is zeroed by subtracting v'.

Figure 5. $Rv' - v'$.

Therefore, $v' \propto v_\perp \times \omega$. But we know that:

$$\begin{aligned} v_\perp &= \theta\widehat{\omega}v' \\ &:= \theta(\omega \times (kv_\perp \times \omega)) \\ &= k\theta v_\perp, \end{aligned}$$

where the last equality can be verified using the fact that $v_\perp \perp \omega$ and following the right-hand rule.

So we finally have:

$$v' = \theta^{-1}(v_\perp \times \omega) = \theta^{-1}(v \times \ \omega)$$

.

8.4 Final result

Therefore, the final result of this section is:

$$\exp\left(\begin{bmatrix} \theta\widehat{\omega} & v \\ 0 & 0 \end{bmatrix}\right) = \begin{bmatrix} R & \theta^{-1}(R - I)(v \times \omega) + (v^T\omega)\omega \\ 0 & 1 \end{bmatrix},$$

where R is computed according to Rodrigues' formula:

$$R = I_3 + \widehat{\omega}\sin(\theta) + \widehat{\omega}^2(1 - \cos(\theta))$$

.

This is equivalent to the result in Equation (2.36) of [MLS]; however, it is not the form provided in Appendix A of [MLS] or the widely-cited [Jose Luis Blanco Clarco (2010)]. It is provided here to increase adoption, as it is both numerically stable and faster than the methods provided in those works.

9 The Logarithmic map: $SE(3) \to se(3)$

This is simply the inverse of the operation defined in section(X).

We are now given

$$t := \theta^{-1}(R - I)(v \times \omega) + (v^T\omega)\omega$$

and tasked with finding v.

As before, let us decompose t into components parallel to and perpendicular to ω:

$$t := t_{||} + t_{\perp}.$$

Since ω is in the nullspace of $R - I$ (as shown in Figure(X)), we have that $t_{||} = (v^T\omega)\omega$, or that $v_{||} = t_{||}$.

Now returning to $t_{\perp}$, let $t_{\perp} := (R - I)t'$. Since $(R - I)\omega = 0$, we may let $t' \perp \omega$ without loss of generality. Following Figure(X), we compute t' as follows:

$$t' = \frac{1}{\tan(\frac{\theta}{2})}\frac{t_{\perp}}{2} \times \omega - \frac{t_{\perp}}{2}$$

.

Figure 7. Geometrical computation of t'.

Substituting into Equation(X), we have that $v_{\perp} \times \omega = \theta t'$. Since $v_{\perp} \perp \omega$, we then have $v_{\perp} = \theta(\omega \times t')$, where we have used the fact that t' and ω are orthogonal, so $||\omega \times t'|| = ||t'||$.

Therefore,

$$\begin{aligned} v &= \theta\left(\omega \times \left(\frac{1}{\tan(\frac{\theta}{2})}\frac{t_{\perp}}{2} \times \omega - \frac{t_{\perp}}{2}\right)\right) + (t^T\omega)\omega \\ &= \frac{\theta}{2}\left(\frac{1}{\tan(\frac{\theta}{2})}t_{\perp} + t_{\perp} \times \omega\right) + (t^T\omega)\omega. \end{aligned}$$

This is again faster and more numerically stable than the methods given in [MLS] and [Blanco (2010)].