



ΠΑΝΕΠΙΣΤΗΜΙΟ
ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

Robotic Systems I

Lecture 4: Orientation Errors, Special Euclidean Group and Trajectory Generation

Konstantinos Chatzilygeroudis - costashatz@upatras.gr

Department of Electrical and Computer Engineering
University of Patras

Template made by Panagiotis Papagiannopoulos



Relation between Lie Algebra and the Manifold

For groups with multiplicative group binary operator:

$$\begin{aligned}\mathbf{v}^\wedge &= \mathcal{X}^{-1} \dot{\mathcal{X}} \\ \dot{\mathcal{X}} &= \mathcal{X} \mathbf{v}^\wedge\end{aligned}$$

This is a linear differential equation. We can solve it:

$$\mathcal{X}(t) = \mathcal{X}(0) \exp(\mathbf{v}^\wedge t)$$

But $\mathcal{X}(0) = \mathbf{\varepsilon} = \mathbf{I}$! Setting $\boldsymbol{\tau}^\wedge = \mathbf{v}^\wedge t$, we get:

$$\mathcal{X} = \exp(\boldsymbol{\tau}^\wedge)$$

Now, we can go back to the manifold using the exponential map!
We can do the inverse operation using the logarithm map:

$$\boldsymbol{\tau}^\wedge = \log(\mathcal{X})$$

Interesting Properties of the \exp Operator

- $\exp((a + b)\tau^\wedge) = \exp(a\tau^\wedge) \exp(b\tau^\wedge)$
- $\exp(a\tau^\wedge) = \exp(\tau^\wedge)^a$
- $\exp(-\tau^\wedge) = \exp(\tau^\wedge)^{-1}$
- $\exp(\mathcal{X}\tau^\wedge \mathcal{X}^{-1}) = \mathcal{X} \exp(\tau^\wedge) \mathcal{X}^{-1}$

We also usually write $\text{Exp}(\tau) \triangleq \exp(\tau^\wedge)$ and $\text{Log}(\mathcal{X}) \triangleq \log(\mathcal{X})^\vee$ to simplify notation.

Right (local) operators

$$\begin{aligned}\mathcal{Y} &= \mathcal{X} \oplus {}^{\mathcal{X}}\boldsymbol{\tau} = \mathcal{X} \circ \text{Exp}({}^{\mathcal{X}}\boldsymbol{\tau}) \in \mathcal{M} \\ {}^{\mathcal{X}}\boldsymbol{\tau} &= \mathcal{Y} \ominus \mathcal{X} = \text{Log}(\mathcal{X}^{-1} \circ \mathcal{Y}) \in T_{\mathcal{X}}\mathcal{M}\end{aligned}$$

Left (global) operators

$$\begin{aligned}\mathcal{Y} &= {}^{\varepsilon}\boldsymbol{\tau} \oplus \mathcal{X} = \text{Exp}({}^{\varepsilon}\boldsymbol{\tau}) \circ \mathcal{X} \in \mathcal{M} \\ {}^{\varepsilon}\boldsymbol{\tau} &= \mathcal{Y} \ominus \mathcal{X} = \text{Log}(\mathcal{Y} \circ \mathcal{X}^{-1}) \in T_{\varepsilon}\mathcal{M}\end{aligned}$$

Adjoint Operator

We have:

$$\mathcal{Y} = \mathcal{X} \oplus {}^{\mathcal{X}}\boldsymbol{\tau}$$

$$\mathcal{Y} = {}^{\varepsilon}\boldsymbol{\tau} \oplus \mathcal{X}$$

This means that $\mathcal{X} \oplus {}^{\mathcal{X}}\boldsymbol{\tau} = {}^{\varepsilon}\boldsymbol{\tau} \oplus \mathcal{X}$

Adjoint Operator

We have:

$$\mathcal{Y} = \mathcal{X} \oplus {}^{\mathcal{X}}\boldsymbol{\tau}$$

$$\mathcal{Y} = {}^{\varepsilon}\boldsymbol{\tau} \oplus \mathcal{X}$$

This means that $\mathcal{X} \oplus {}^{\mathcal{X}}\boldsymbol{\tau} = {}^{\varepsilon}\boldsymbol{\tau} \oplus \mathcal{X}$, and we have:

$$\text{Exp}({}^{\varepsilon}\boldsymbol{\tau}) \circ \mathcal{X} = \mathcal{X} \circ \text{Exp}({}^{\mathcal{X}}\boldsymbol{\tau})$$

$$\exp({}^{\varepsilon}\boldsymbol{\tau}^\wedge) = \mathcal{X} \exp({}^{\mathcal{X}}\boldsymbol{\tau}^\wedge) \mathcal{X}^{-1} = \exp(\mathcal{X} {}^{\mathcal{X}}\boldsymbol{\tau}^\wedge \mathcal{X}^{-1})$$

$${}^{\varepsilon}\boldsymbol{\tau}^\wedge = \mathcal{X} {}^{\mathcal{X}}\boldsymbol{\tau}^\wedge \mathcal{X}^{-1}$$

We now define the **adjoint** ($\text{Ad}_{\mathcal{X}} : \mathfrak{m} \rightarrow \mathfrak{m}$):

$$\text{Ad}_{\mathcal{X}}(\boldsymbol{\tau}^\wedge) = \mathcal{X} \boldsymbol{\tau}^\wedge \mathcal{X}^{-1}$$

Adjoint Operator

We have:

$$\mathcal{Y} = \mathcal{X} \oplus {}^{\mathcal{X}}\tau$$

$$\mathcal{Y} = {}^{\varepsilon}\tau \oplus \mathcal{X}$$

This means that $\mathcal{X} \oplus {}^{\mathcal{X}}\tau = {}^{\varepsilon}\tau \oplus \mathcal{X}$, and we have:

$$\text{Exp}({}^{\varepsilon}\tau) \circ \mathcal{X} = \mathcal{X} \circ \text{Exp}({}^{\mathcal{X}}\tau)$$

$$\exp({}^{\varepsilon}\tau^\wedge) = \mathcal{X} \exp({}^{\mathcal{X}}\tau^\wedge) \mathcal{X}^{-1} = \exp(\mathcal{X} {}^{\mathcal{X}}\tau^\wedge \mathcal{X}^{-1})$$

$${}^{\varepsilon}\tau^\wedge = \mathcal{X} {}^{\mathcal{X}}\tau^\wedge \mathcal{X}^{-1}$$

We now define the **adjoint** ($\text{Ad}_{\mathcal{X}} : \mathfrak{m} \rightarrow \mathfrak{m}$):

$$\text{Ad}_{\mathcal{X}}(\tau^\wedge) = \mathcal{X} \tau^\wedge \mathcal{X}^{-1}$$

Some interesting properties:

$$\text{Ad}_{\mathcal{X}}(a\tau^\wedge + b\sigma^\wedge) = a\text{Ad}_{\mathcal{X}}(\tau^\wedge) + b\text{Ad}_{\mathcal{X}}(\sigma^\wedge)$$

$$\text{Ad}_{\mathcal{X}}(\text{Ad}_{\mathcal{Y}}(\tau^\wedge)) = \text{Ad}_{\mathcal{X}\mathcal{Y}}(\tau^\wedge)$$

Adjoint Matrix

The Adjoint matrix $\mathbf{Ad}_{\mathcal{X}} : \mathbb{R}^m \rightarrow \mathbb{R}^m$ is defined as:

$${}^\varepsilon \boldsymbol{\tau} = \mathbf{Ad}_{\mathcal{X}} {}^\mathcal{X} \boldsymbol{\tau}$$

$$\mathbf{Ad}_{\mathcal{X}} \boldsymbol{\tau} = (\mathcal{X} \boldsymbol{\tau}^\wedge \mathcal{X}^{-1})^\vee$$

Adjoint Matrix

The Adjoint matrix $\mathbf{Ad}_{\mathcal{X}} : \mathbb{R}^m \rightarrow \mathbb{R}^m$ is defined as:

$${}^\varepsilon \boldsymbol{\tau} = \mathbf{Ad}_{\mathcal{X}} \boldsymbol{\chi} \boldsymbol{\tau}$$

$$\mathbf{Ad}_{\mathcal{X}} \boldsymbol{\tau} = (\boldsymbol{\chi} \boldsymbol{\tau}^\wedge \boldsymbol{\chi}^{-1})^\vee$$

Some interesting properties:

$$\boldsymbol{\chi} \oplus \boldsymbol{\tau} = (\mathbf{Ad}_{\mathcal{X}} \boldsymbol{\tau}) \oplus \boldsymbol{\chi}$$

$$\mathbf{Ad}_{\mathcal{X}}^{-1} = \mathbf{Ad}_{\mathcal{X}^{-1}}$$

$$\mathbf{Ad}_{\mathcal{X}} \mathbf{Ad}_{\mathcal{Y}} = \mathbf{Ad}_{\mathcal{X} \mathcal{Y}}$$

Adjoint Matrix

The Adjoint matrix $\mathbf{Ad}_{\mathcal{X}} : \mathbb{R}^m \rightarrow \mathbb{R}^m$ is defined as:

$${}^\varepsilon \boldsymbol{\tau} = \mathbf{Ad}_{\mathcal{X}} \boldsymbol{\chi} \boldsymbol{\tau}$$

$$\mathbf{Ad}_{\mathcal{X}} \boldsymbol{\tau} = (\boldsymbol{\chi} \boldsymbol{\tau}^\wedge \boldsymbol{\chi}^{-1})^\vee$$

Some interesting properties:

$$\boldsymbol{\chi} \oplus \boldsymbol{\tau} = (\mathbf{Ad}_{\mathcal{X}} \boldsymbol{\tau}) \oplus \boldsymbol{\chi}$$

$$\mathbf{Ad}_{\mathcal{X}}^{-1} = \mathbf{Ad}_{\mathcal{X}^{-1}}$$

$$\mathbf{Ad}_{\mathcal{X}} \boldsymbol{\gamma} = \mathbf{Ad}_{\mathcal{X}} \mathbf{Ad}_{\boldsymbol{\gamma}}$$

For $SO(3)$, we have:

$$\mathbf{Ad}_R = R$$

Orientation Errors

In other words, I am at R_{wc} and I want to end up at R_{wt} . What is the orientation error?

Orientation Errors

In other words, I am at R_{wc} and I want to end up at R_{wt} . What is the orientation error?

- First, an orientation difference is another rotation matrix!

Orientation Errors

In other words, I am at R_{wc} and I want to end up at R_{wt} . What is the orientation error?

- First, an orientation difference is another rotation matrix!
- I can compute R_{ct} :

Orientation Errors

In other words, I am at \mathbf{R}_{wc} and I want to end up at \mathbf{R}_{wt} . What is the orientation error?

- First, an orientation difference is another rotation matrix!
- I can compute \mathbf{R}_{ct} :

$$\begin{aligned}\mathbf{R}_{ct} &= \mathbf{R}_{tc}^T \\ &= (\mathbf{R}_{tw} \mathbf{R}_{wc})^T \\ &= (\mathbf{R}_{wt}^T \mathbf{R}_{wc})^T \\ \Rightarrow \mathbf{R}_{ct} &= \mathbf{R}_{wc}^T \mathbf{R}_{wt}\end{aligned}$$

Orientation Errors

In other words, I am at \mathbf{R}_{wc} and I want to end up at \mathbf{R}_{wt} . What is the orientation error?

- First, an orientation difference is another rotation matrix!
- I can compute \mathbf{R}_{ct} :

$$\begin{aligned}\mathbf{R}_{ct} &= \mathbf{R}_{tc}^T \\ &= (\mathbf{R}_{tw} \mathbf{R}_{wc})^T \\ &= (\mathbf{R}_{wt}^T \mathbf{R}_{wc})^T \\ \Rightarrow \mathbf{R}_{ct} &= \mathbf{R}_{wc}^T \mathbf{R}_{wt}\end{aligned}$$

- This is the orientation error expressed in *body frame*
- How can I get the error in *world frame*?

$$\begin{aligned}\mathbf{R}_{ct}^W \mathbf{R}_{wc} &= \mathbf{R}_{wt} \\ \Rightarrow \mathbf{R}_{ct}^W &= \mathbf{R}_{wt} \mathbf{R}_{wc}^T\end{aligned}$$

Orientation Errors and Lie Algebra

In general, the *rotation difference* can be described by:

$$\begin{aligned}\phi_{12} &= \log(\mathbf{R}_1^T \mathbf{R}_2)^\vee = \text{Log}(\mathbf{R}_1^T \mathbf{R}_2) \text{ }_{Right} \ominus \\ \phi_{21} &= \log(\mathbf{R}_2 \mathbf{R}_1^T)^\vee = \text{Log}(\mathbf{R}_2 \mathbf{R}_1^T) \text{ }_{Left} \ominus\end{aligned}$$

Also the *difference angle* is given by:

$$\begin{aligned}\phi_{12} &= \sqrt{\phi_{12}^T \phi_{12}} = |\phi_{12}| \\ \phi_{21} &= \sqrt{\phi_{21}^T \phi_{21}} = |\phi_{21}|\end{aligned}$$

Interpolating Orientations

We **cannot** do:

$$\mathbf{R} = \mathbf{R}_1 + \alpha(\mathbf{R}_2 - \mathbf{R}_1)$$

for $\alpha \in [0, 1]$.

We **cannot** do:

$$\mathbf{R} = \mathbf{R}_1 + \alpha(\mathbf{R}_2 - \mathbf{R}_1)$$

for $\alpha \in [0, 1]$.

We **CAN** do:

$$\begin{aligned}\mathbf{R} &= (\mathbf{R}_2 \mathbf{R}_1^T)^\alpha \mathbf{R}_1 \\ &= \mathbf{R}_1 (\mathbf{R}_1^T \mathbf{R}_2)^\alpha\end{aligned}$$

Why?

We **cannot** do:

$$\mathbf{R} = \mathbf{R}_1 + \alpha(\mathbf{R}_2 - \mathbf{R}_1)$$

for $\alpha \in [0, 1]$.

We **CAN** do:

$$\begin{aligned}\mathbf{R} &= (\mathbf{R}_2 \mathbf{R}_1^T)^\alpha \mathbf{R}_1 \\ &= \mathbf{R}_1 (\mathbf{R}_1^T \mathbf{R}_2)^\alpha\end{aligned}$$

Why?

$$\begin{aligned}\mathbf{R}_2 \mathbf{R}_1^T &= \exp(\phi_{21}^\wedge) \\ \Rightarrow (\mathbf{R}_2 \mathbf{R}_1^T)^\alpha &= \exp(\alpha \phi_{21}^\wedge)\end{aligned}$$

$$\begin{aligned}\boldsymbol{R}_{k+1} &= \boldsymbol{R}_k \exp(\boldsymbol{\omega}_b^\wedge dt) = \boldsymbol{R}_k \oplus (\boldsymbol{\omega}_b dt) \\ &= \exp(\boldsymbol{\omega}_w^\wedge dt) \boldsymbol{R}_k = (\boldsymbol{\omega}_w dt) \oplus \boldsymbol{R}_k\end{aligned}$$

Note that:

- “Right” multiplication is *local*, $\boldsymbol{\omega}_b \in T_{\mathcal{X}} \mathcal{M}$
- “Left” multiplication is *global/world*, $\boldsymbol{\omega}_w \in T_{\varepsilon} \mathcal{M}$

- The **Special Euclidean Group**, representing full transformation matrices is the set:

$$SE(3) = \{ \boldsymbol{T} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathbb{R}^{4 \times 4} \mid \boldsymbol{R}\boldsymbol{R}^T = \boldsymbol{I}, \det(\boldsymbol{R}) = 1 \}$$

- The set of all 4×4 matrices is a vectorspace. What does it mean?

- The **Special Euclidean Group**, representing full transformation matrices is the set:

$$SE(3) = \left\{ \boldsymbol{T} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathbb{R}^{4 \times 4} \mid \boldsymbol{R}\boldsymbol{R}^T = \mathbf{I}, \det(\boldsymbol{R}) = 1 \right\}$$

- The set of all 4×4 matrices is a vectorspace. What does it mean?
- $SE(3)$ is not:
 - 1 The zero matrix $\mathbf{0}$ is not part of the space!
 - 2 $\boldsymbol{T}_1, \boldsymbol{T}_2 \in SE(3) \not\Rightarrow \boldsymbol{T}_1 + \boldsymbol{T}_2 \in SE(3)$

$SE(3)$ is a Matrix Lie Group

$SE(3)$ forms a *matrix Lie group* (\circ is the regular matrix multiplication, ε is the identity matrix):

- $\mathbf{T}_1, \mathbf{T}_2 \in SE(3) \Rightarrow \mathbf{T}_1 \mathbf{T}_2 \in SE(3)$
- $(\mathbf{T}_1 \mathbf{T}_2) \mathbf{T}_3 = \mathbf{T}_1 (\mathbf{T}_2 \mathbf{T}_3) = \mathbf{T}_1 \mathbf{T}_2 \mathbf{T}_3$
- $\mathbf{T}, \mathbf{I} \in SE(3)$ and $\mathbf{T}\mathbf{I} = \mathbf{I}\mathbf{T} = \mathbf{T}$
- $\mathbf{T}^{-1} = \begin{bmatrix} \mathbf{R}^T & -\mathbf{R}^T \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \in SE(3)$

Full Poses - $SE(3)$

Recap for the **Special Euclidean Group**(3), $SE(3)$:

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix}, \boldsymbol{T}^{-1} = \begin{bmatrix} \boldsymbol{R}^T & -\boldsymbol{R}^T \boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix}, \boldsymbol{Ad}_{\boldsymbol{T}} = [Ad_{\boldsymbol{T}}] = \begin{bmatrix} \boldsymbol{R} & \mathbf{0} \\ \boldsymbol{t}^\wedge \boldsymbol{R} & \boldsymbol{R} \end{bmatrix}$$

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\phi} \\ \boldsymbol{\rho} \end{bmatrix} \in \mathbb{R}^6, \exp(\boldsymbol{\xi}^\wedge) = \boldsymbol{T}, \log(\boldsymbol{T})^\vee = \boldsymbol{\xi}$$

$$\boldsymbol{\xi}_{12} = \text{Log}(\boldsymbol{T}_1^{-1} \boldsymbol{T}_2), \boldsymbol{\xi}_{12} = \sqrt{\boldsymbol{\xi}_{12}^T \boldsymbol{\xi}_{12}} = |\boldsymbol{\xi}_{12}|$$

$$\boldsymbol{\xi}_{21} = \text{Log}(\boldsymbol{T}_2 \boldsymbol{T}_1^{-1}), \boldsymbol{\xi}_{21} = \sqrt{\boldsymbol{\xi}_{21}^T \boldsymbol{\xi}_{21}} = |\boldsymbol{\xi}_{21}|$$

$$\boldsymbol{T} = (\boldsymbol{T}_2 \boldsymbol{T}_1^{-1})^\alpha \boldsymbol{T}_1 = \boldsymbol{T}_1 (\boldsymbol{T}_1^{-1} \boldsymbol{T}_2)^\alpha$$

$$\begin{aligned} \boldsymbol{T}_{k+1} &= \exp([\mathcal{V}_w]dt) \boldsymbol{T}_k = ({}^\varepsilon \boldsymbol{\xi} dt) \oplus \boldsymbol{T}_k \\ &= \boldsymbol{T}_k \exp([\mathcal{V}_b]dt) = \boldsymbol{T}_k \oplus ({}^{\boldsymbol{T}_k} \boldsymbol{\xi} dt) \end{aligned}$$

$$\text{where } \boldsymbol{\xi}^\wedge = [\boldsymbol{\xi}] = \begin{bmatrix} \boldsymbol{\phi}^\wedge & \boldsymbol{\rho} \\ \mathbf{0} & 0 \end{bmatrix}, \boldsymbol{\phi}, \boldsymbol{\rho} \in \mathbb{R}^3.$$

Exp/Log Map for $SE(3)$

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\phi} \\ \boldsymbol{\rho} \end{bmatrix} \in \mathbb{R}^6, \boldsymbol{\phi} = \phi \mathbf{r}$$

$$\exp(\boldsymbol{\xi}^\wedge) = \mathbf{T} = \begin{bmatrix} \exp(\boldsymbol{\phi}^\wedge) & \mathbf{J}_I(\boldsymbol{\phi})\boldsymbol{\rho} \\ \mathbf{0} & 1 \end{bmatrix}$$

$$\log(\mathbf{T})^\vee = \boldsymbol{\xi} = \begin{bmatrix} \text{Log}(\mathbf{R}) \\ \mathbf{J}_I^{-1}(\boldsymbol{\phi})\mathbf{t} \end{bmatrix}$$

where

$$\mathbf{J}_I(\boldsymbol{\phi}) = \mathbf{I} + \frac{1 - \cos \phi}{\phi^2} \boldsymbol{\phi}^\wedge + \frac{\phi - \sin \phi}{\phi^3} \boldsymbol{\phi}^\wedge \boldsymbol{\phi}^\wedge$$

$$\mathbf{J}_I^{-1}(\boldsymbol{\phi}) = \mathbf{I} - \frac{1}{2} \boldsymbol{\phi}^\wedge + \left(\frac{1}{\phi^2} - \frac{1 + \cos \phi}{2\phi \sin \phi} \right) \boldsymbol{\phi}^\wedge \boldsymbol{\phi}^\wedge$$

Manifolds and Lie Algebra - Code Example

jupyter 3-so3 Last Checkpoint: 6 minutes ago

File Edit View Run Kernel Settings Help

Code

```
def angular_velocity_local_to_deriv_rotation(omega_b, R):
    return R @ hat(omega_b)

# Some testing
R = RotX(1.)

omega_b = np.array([[1., 0., 0.]]).T

# Integrate naively for a few steps
dt = 0.01
R1 = np.copy(R)
R2 = np.copy(R)
R3 = np.copy(R)
R4 = np.copy(R)

# Try increasing the steps!
```

Trajectory Generation

- How can we generate a full trajectory (with timings) given an initial, x_s and target, x_g , state?
- Let's do the simplest one! *Straight line*:

Trajectory Generation

- How can we generate a full trajectory (with timings) given an initial, \mathbf{x}_s and target, \mathbf{x}_g , state?
- Let's do the simplest one! *Straight line*:

$$\mathbf{x}(t) = \mathbf{x}_s + t(\mathbf{x}_g - \mathbf{x}_s)$$

- What is the issue with this?

Trajectory Generation

- How can we generate a full trajectory (with timings) given an initial, \mathbf{x}_s and target, \mathbf{x}_g , state?
- Let's do the simplest one! *Straight line*:

$$\mathbf{x}(t) = \mathbf{x}_s + t(\mathbf{x}_g - \mathbf{x}_s)$$

- What is the issue with this? What if $t > 1$?
- We also need to specify the final time T :

$$\mathbf{x}(t) = \mathbf{x}_s + \frac{t}{T}(\mathbf{x}_g - \mathbf{x}_s)$$

Trajectory Generation

- How can we generate a full trajectory (with timings) given an initial, \mathbf{x}_s and target, \mathbf{x}_g , state?
- Let's do the simplest one! *Straight line*:

$$\mathbf{x}(t) = \mathbf{x}_s + t(\mathbf{x}_g - \mathbf{x}_s)$$

- What is the issue with this? What if $t > 1$?
- We also need to specify the final time T :

$$\mathbf{x}(t) = \mathbf{x}_s + \frac{t}{T}(\mathbf{x}_g - \mathbf{x}_s)$$

- What if $t > T$ or if I do not start from $t = 0$?

$$\mathbf{x}(t) = \mathbf{x}_s + \max(1, \frac{t - t_0}{T})(\mathbf{x}_g - \mathbf{x}_s)$$

Trajectory Generation (2)

- What if x contains velocities? Aka, $x = \begin{bmatrix} q \\ \dot{q} \end{bmatrix}$
- Can we use the previous?

Trajectory Generation (2)

- What if x contains velocities? Aka, $x = \begin{bmatrix} q \\ \dot{q} \end{bmatrix}$
- Can we use the previous? **We can only have constant velocity!** Why?

Trajectory Generation (2)

- What if x contains velocities? Aka, $x = \begin{bmatrix} q \\ \dot{q} \end{bmatrix}$
- Can we use the previous? **We can only have constant velocity!** Why?
- Using the above, we model the path as $q(t) = c_1 t + c_0$. Thus $\dot{q}(t) = c_1$!!! What can we do?

Trajectory Generation (2)

- What if x contains velocities? Aka, $x = \begin{bmatrix} q \\ \dot{q} \end{bmatrix}$
- Can we use the previous? **We can only have constant velocity!** Why?
- Using the above, we model the path as $q(t) = c_1 t + c_0$. Thus $\dot{q}(t) = c_1$!!! What can we do?
- Let's use higher order polynomial: $q(t) = c_2 t^2 + c_1 t + c_0$
- Now $\dot{q}(t) = 2c_2 t + c_1$
- Much better!

Trajectory Generation (3)

$$\mathbf{q}(t) = \mathbf{c}_2 t^2 + \mathbf{c}_1 t + \mathbf{c}_0$$

$$\dot{\mathbf{q}}(t) = 2\mathbf{c}_2 t + \mathbf{c}_1$$

- What now? How can we compute the \mathbf{c}_i s?
- Given $\mathbf{x}_s = \begin{bmatrix} \mathbf{q}_s \\ \dot{\mathbf{q}}_s \end{bmatrix}$, $\mathbf{x}_g = \begin{bmatrix} \mathbf{q}_g \\ \dot{\mathbf{q}}_g \end{bmatrix}$ and a total time T :

Trajectory Generation (3)

$$\mathbf{q}(t) = \mathbf{c}_2 t^2 + \mathbf{c}_1 t + \mathbf{c}_0$$

$$\dot{\mathbf{q}}(t) = 2\mathbf{c}_2 t + \mathbf{c}_1$$

- What now? How can we compute the \mathbf{c}_i s?
- Given $\mathbf{x}_s = \begin{bmatrix} \mathbf{q}_s \\ \dot{\mathbf{q}}_s \end{bmatrix}$, $\mathbf{x}_g = \begin{bmatrix} \mathbf{q}_g \\ \dot{\mathbf{q}}_g \end{bmatrix}$ and a total time T :

$$\mathbf{q}(0) = \mathbf{q}_s = \mathbf{c}_0$$

$$\dot{\mathbf{q}}(0) = \dot{\mathbf{q}}_s = \mathbf{c}_1$$

- Nice! Let's find the last coefficient!

Trajectory Generation (3)

$$\mathbf{q}(t) = \mathbf{c}_2 t^2 + \mathbf{c}_1 t + \mathbf{c}_0$$

$$\dot{\mathbf{q}}(t) = 2\mathbf{c}_2 t + \mathbf{c}_1$$

- What now? How can we compute the \mathbf{c}_i s?
- Given $\mathbf{x}_s = \begin{bmatrix} \mathbf{q}_s \\ \dot{\mathbf{q}}_s \end{bmatrix}$, $\mathbf{x}_g = \begin{bmatrix} \mathbf{q}_g \\ \dot{\mathbf{q}}_g \end{bmatrix}$ and a total time T :

$$\mathbf{q}(0) = \mathbf{q}_s = \mathbf{c}_0$$

$$\dot{\mathbf{q}}(0) = \dot{\mathbf{q}}_s = \mathbf{c}_1$$

- Nice! Let's find the last coefficient!

$$\mathbf{q}(T) = \mathbf{q}_g = \mathbf{c}_2 T^2 + \mathbf{c}_1 T + \mathbf{c}_0$$

$$= \mathbf{c}_2 T^2 + \dot{\mathbf{q}}_s T + \mathbf{q}_s \Rightarrow \mathbf{c}_2 = \frac{\mathbf{q}_g - \dot{\mathbf{q}}_s T - \mathbf{q}_s}{T^2}$$

Trajectory Generation (3)

$$\mathbf{q}(t) = \mathbf{c}_2 t^2 + \mathbf{c}_1 t + \mathbf{c}_0$$

$$\dot{\mathbf{q}}(t) = 2\mathbf{c}_2 t + \mathbf{c}_1$$

- What now? How can we compute the \mathbf{c}_i s?
- Given $\mathbf{x}_s = \begin{bmatrix} \mathbf{q}_s \\ \dot{\mathbf{q}}_s \end{bmatrix}$, $\mathbf{x}_g = \begin{bmatrix} \mathbf{q}_g \\ \dot{\mathbf{q}}_g \end{bmatrix}$ and a total time T :

$$\mathbf{q}(0) = \mathbf{q}_s = \mathbf{c}_0$$

$$\dot{\mathbf{q}}(0) = \dot{\mathbf{q}}_s = \mathbf{c}_1$$

- Nice! Let's find the last coefficient!

$$\mathbf{q}(T) = \mathbf{q}_g = \mathbf{c}_2 T^2 + \mathbf{c}_1 T + \mathbf{c}_0$$

$$= \mathbf{c}_2 T^2 + \dot{\mathbf{q}}_s T + \mathbf{q}_s \Rightarrow \mathbf{c}_2 = \frac{\mathbf{q}_g - \dot{\mathbf{q}}_s T - \mathbf{q}_s}{T^2}$$

- OR

$$\dot{\mathbf{q}}(T) = \dot{\mathbf{q}}_g = 2\mathbf{c}_2 T + \dot{\mathbf{q}}_s \Rightarrow \mathbf{c}_2 = \frac{\dot{\mathbf{q}}_g - \dot{\mathbf{q}}_s}{2T}$$

Cubic Splines

- Even-ordered polynomials are not good! We are always left with overparameterized systems!
- Let's do 3rd order:

$$\mathbf{q}(t) = \mathbf{c}_3 t^3 + \mathbf{c}_2 t^2 + \mathbf{c}_1 t + \mathbf{c}_0$$

- And:

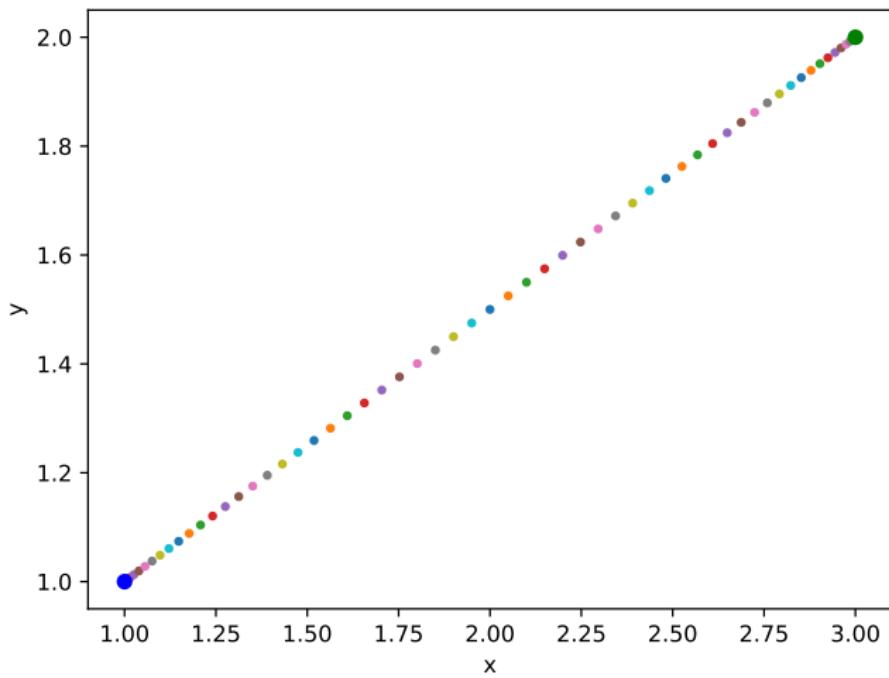
$$\mathbf{c}_0 = \mathbf{q}_s$$

$$\mathbf{c}_1 = \dot{\mathbf{q}}_s$$

$$\mathbf{c}_2 = \frac{3\mathbf{q}_g}{T^2} - \frac{3\mathbf{q}_s}{T^2} - \frac{2\dot{\mathbf{q}}_s}{T} - \frac{\ddot{\mathbf{q}}_g}{T}$$

$$\mathbf{c}_3 = -\frac{2\mathbf{q}_g}{T^3} + \frac{2\mathbf{q}_s}{T^3} + \frac{\dot{\mathbf{q}}_s}{T^2} + \frac{\ddot{\mathbf{q}}_g}{T^2}$$

Splines - Code Example



Trajectory Generation with SO(3)?

- What happens when we are in $\text{SO}(3)$?

Trajectory Generation with SO(3)?

- What happens when we are in SO(3)?
- Let's first see the straight line path. We **CANNOT** do:

$$\mathbf{x}(t) = \mathbf{x}_s + t(\mathbf{x}_g - \mathbf{x}_s)$$

- First we need to replace \mathbf{x} with \mathbf{R} . Now what?

Trajectory Generation with SO(3)?

- What happens when we are in SO(3)?
- Let's first see the straight line path. We **CANNOT** do:

$$\mathbf{x}(t) = \mathbf{x}_s + t(\mathbf{x}_g - \mathbf{x}_s)$$

- First we need to replace \mathbf{x} with \mathbf{R} . Now what?
- Easy! We just interpolate!

$$\begin{aligned}\mathbf{R}(t) &= \mathbf{R}_s (\mathbf{R}_s^T \mathbf{R}_g)^t \\ &= \mathbf{R}_s \exp(t \log(\mathbf{R}_s^T \mathbf{R}_g)^\vee)\end{aligned}$$

Cubic Splines with SO(3)?

- We can use the equations as before

Cubic Splines with SO(3)?

- We can use the equations as before, almost:

$$\mathbf{R}(t) = \exp(\mathbf{c}_3^\wedge t^3 + \mathbf{c}_2^\wedge t^2 + \mathbf{c}_1^\wedge t + \mathbf{c}_0^\wedge)$$

$$\dot{\mathbf{R}}(t) = (3\mathbf{c}_3^\wedge t^2 + 2\mathbf{c}_2^\wedge t + \mathbf{c}_1^\wedge) \mathbf{R}(t)$$

- Thus if we start at $t = 0$ from (\mathbf{R}_s, ω_s) and we want to arrive at $t = T$ to (\mathbf{R}_g, ω_g) , we have:

Cubic Splines with SO(3)?

- We can use the equations as before, almost:

$$\mathbf{R}(t) = \exp(\mathbf{c}_3^\wedge t^3 + \mathbf{c}_2^\wedge t^2 + \mathbf{c}_1^\wedge t + \mathbf{c}_0^\wedge)$$

$$\dot{\mathbf{R}}(t) = (3\mathbf{c}_3^\wedge t^2 + 2\mathbf{c}_2^\wedge t + \mathbf{c}_1^\wedge) \mathbf{R}(t)$$

- Thus if we start at $t = 0$ from (\mathbf{R}_s, ω_s) and we want to arrive at $t = T$ to (\mathbf{R}_g, ω_g) , we have:

$$\mathbf{R}(0) = \exp(\mathbf{c}_0^\wedge) = \mathbf{R}_s \Rightarrow \mathbf{c}_0 = \log(\mathbf{R}_s)^\vee = \phi_s$$

$$\dot{\mathbf{R}}(0) = \mathbf{c}_1^\wedge \mathbf{R}(0) \Rightarrow \omega_s^\wedge \mathbf{R}_s = \mathbf{c}_1^\wedge \mathbf{R}_s \Rightarrow \mathbf{c}_1 = \omega_s$$

$$\mathbf{c}_2 = \frac{3\phi_g}{T^2} - \frac{3\phi_s}{T^2} - \frac{2\omega_s}{T} - \frac{\omega_g}{T}$$

$$\mathbf{c}_3 = -\frac{2\phi_g}{T^3} + \frac{2\phi_s}{T^3} + \frac{\omega_s}{T^2} + \frac{\omega_g}{T^2}$$

where $\mathbf{R}_s = \exp(\phi_s^\wedge)$, $\mathbf{R}_g = \exp(\phi_g^\wedge)$.

Cubic Splines with SE(3)

- We do the same as before:

$$\mathbf{T}(t) = \exp(\mathbf{c}_3^\wedge t^3 + \mathbf{c}_2^\wedge t^2 + \mathbf{c}_1^\wedge t + \mathbf{c}_0^\wedge)$$

$$\dot{\mathbf{T}}(t) = (3\mathbf{c}_3^\wedge t^2 + 2\mathbf{c}_2^\wedge t + \mathbf{c}_1^\wedge) \mathbf{T}(t)$$

- Thus if we start at $t = 0$ from $(\mathbf{T}_s, \mathcal{V}_s)$ and we want to arrive at $t = T$ to $(\mathbf{T}_g, \mathcal{V}_g)$, we have:

Cubic Splines with SE(3)

- We do the same as before:

$$\mathbf{T}(t) = \exp(\mathbf{c}_3^\wedge t^3 + \mathbf{c}_2^\wedge t^2 + \mathbf{c}_1^\wedge t + \mathbf{c}_0^\wedge)$$

$$\dot{\mathbf{T}}(t) = (3\mathbf{c}_3^\wedge t^2 + 2\mathbf{c}_2^\wedge t + \mathbf{c}_1^\wedge) \mathbf{T}(t)$$

- Thus if we start at $t = 0$ from $(\mathbf{T}_s, \mathcal{V}_s)$ and we want to arrive at $t = T$ to $(\mathbf{T}_g, \mathcal{V}_g)$, we have:

$$\mathbf{T}(0) = \exp(\mathbf{c}_0^\wedge) = \mathbf{T}_s \Rightarrow \mathbf{c}_0 = \log(\mathbf{T}_s)^\vee = \boldsymbol{\tau}_s$$

$$\dot{\mathbf{T}}(0) = \mathbf{c}_1^\wedge \mathbf{T}(0) \Rightarrow [\mathcal{V}_s] \mathbf{T}_s = \mathbf{c}_1^\wedge \mathbf{T}_s \Rightarrow \mathbf{c}_1 = \mathcal{V}_s$$

$$\mathbf{c}_2 = \frac{3\boldsymbol{\tau}_g}{T^2} - \frac{3\boldsymbol{\tau}_s}{T^2} - \frac{2\mathcal{V}_s}{T} - \frac{\mathcal{V}_g}{T}$$

$$\mathbf{c}_3 = -\frac{2\boldsymbol{\tau}_g}{T^3} + \frac{2\boldsymbol{\tau}_s}{T^3} + \frac{\mathcal{V}_s}{T^2} + \frac{\mathcal{V}_g}{T^2}$$

where $\mathbf{T}_s = \exp(\boldsymbol{\tau}_s^\wedge)$, $\mathbf{T}_g = \exp(\boldsymbol{\tau}_g^\wedge)$.

Bibliography

A micro Lie theory for state estimation in robotics, *Joan Sola, Jeremie Deray, Dinesh Atchuthan*, 2018, arXiv. [pdf](#)

Sections 7.1.1-7.1.3 from **State Estimation for Robotics**,
Timothy D. Barfoot, 2022, Cambridge University Press. [ebook](#)

Chapters 3, and 9 from **Modern Robotics: Mechanics, Planning, and Control**, *Kevin M. Lynch and Frank C. Park*, 2017, Cambridge University Press. [ebook](#)

Thank you

- Any Questions?

- Office Hours:

- Tue-Wed (10:00-12:00)

- 24/7 by email (costashatz@upatras.gr, subject: *ECE_RSI_AM*)

- Material and Announcements



Laboratory of Automation & Robotics