



Robotics Foundations

Henrik I Christensen

(c) Henrik I Christensen

Logistics

- Book - Python (see Piazza/Canvas)
- PodCast Recording? (On podcast.ucsd.edu) or canvas media
- Waitlist - a constraint satisfaction problem (~15 people added)
- If you are not taking the class - a decision earlier is helpful to others
- Tuesday ROS2 tutorial
 - ROS2 is ideally on a Linux computer
 - Consider a virtual machine or container setup
 - UTM, Parallel, ...
 - Tour of ROS - basics, terminal operation, basic ROS programs, debug tools, ...
- Next Tuesday - SW/ROS2 for class and intro to robot and assoc models

(c) Henrik I Christensen

Outline

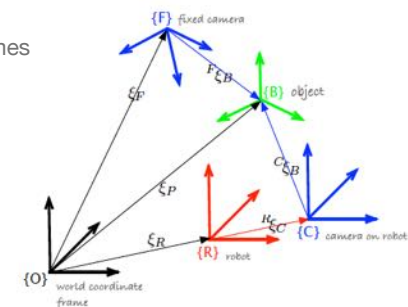
- Representation of position and orientation
 - Reference Frames
 - Representation of position and orientation in 2D and 3D
 - Models for transformations between reference frames
 - Homogenous coordinates
- Representations for time and motions
 - Trajectory models
 - Time varying reference frames
- Summary

Source: Many Illustrations / basic material adopted from P. Corke, Robotics, Vision and Control, STAR Vol. 73, Springer Verlag, 2011

(c) Henrik I Christensen

Reference Frames

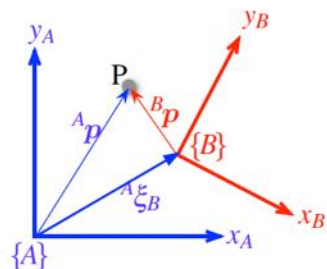
- Robotics is all about management of reference frames
- Perception is about estimation of reference frames
- Planning is how to move reference frames
- Control is the implementation of trajectories for reference frames
- The relation between references frames is essential to a successful system



(c) Henrik I Christensen

Representation of position and orientation

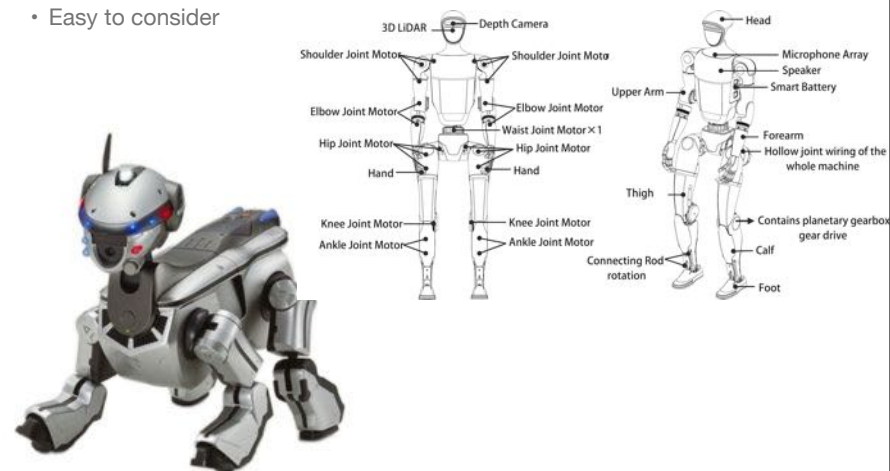
- We will make reference to different frames such as A and B
- The transformation between frames will be denoted by the symbol ξ (ξ)
- If we have a point P then we will use ${}^A P$ to denote that P is represented in reference frame A
- For transformations between reference frames we will use the notation ${}^A \xi_B$



(c) Henrik I Christensen

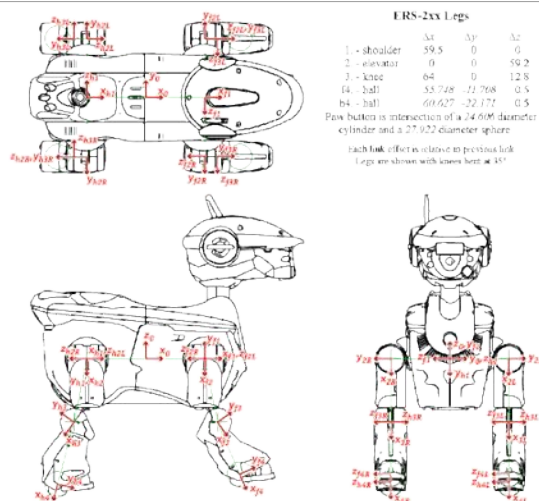
Use of reference frames - example

- Easy to consider



(c) Henrik I Christensen

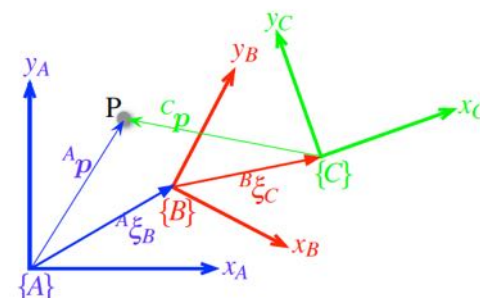
Use of reference frames - example



(c) Henrik I Christensen

Transformations

- We can compose transformation
 - ${}^A \xi_C = {}^A \xi_B \oplus {}^B \xi_C$
- We can represent a point ${}^C P$ in the frame A through the transformation
 - ${}^A P = ({}^A \xi_B \oplus {}^B \xi_C) {}^C P$



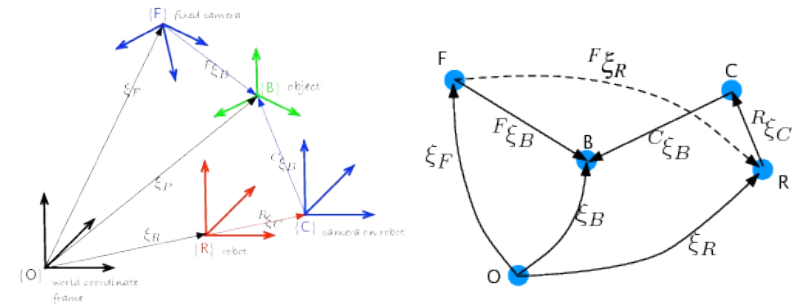
(c) Henrik I Christensen

Basics concepts:

- A point P is described by a vector that specify the displacement from the origin of the reference frame.
- A set of points that represent a rigid object can be described by a single coordinate frame, and its constituent points are described by displacements from that coordinate frame
- The position and orientation of an object's coordinate frame is referred to as its pose
- A relative pose describes the pose of one coordinate frame with respect to another and is denoted by an algebraic variable ξ
- A coordinate vector describing a point can be represented with respect to a different coordinate frame by applying the relative pose to the vector using the $\cdot$ operator
- We can perform algebraic manipulation of expressions written in terms of relative poses

(c) Henrik I Christensen

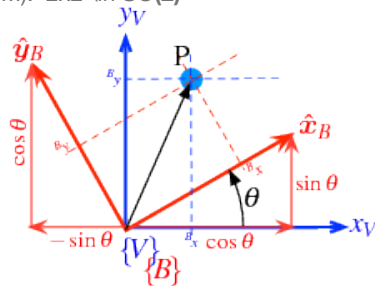
Pose Algebra



(c) Henrik I Christensen

Rotation in 2D

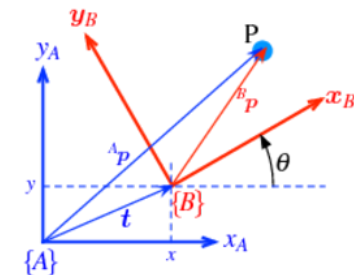
- Rotation Matrices (DCM): $2 \times 2 \in \mathbf{SO}(2)$



- Rotation Matrices (DCM), $2 \times 2 \in \mathbf{SO}(2)$
- Columns of $\mathbf{vRb}$: axes of B in V . In MATLAB notation: $[c \ -s; s \ c]$
- 4 numbers, but only 1D **Manifold**

(c) Henrik I Christensen

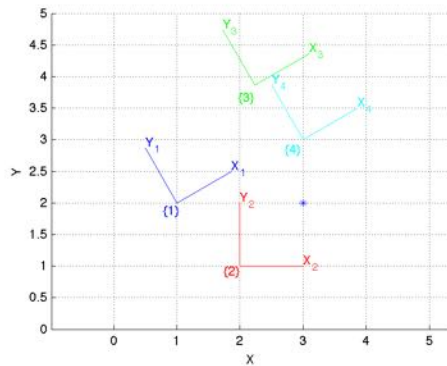
Poses in 2D



- (x, y, θ) or (R, t)
- Better: $\mathbf{SE}(2)$, next slide

(c) Henrik I Christensen

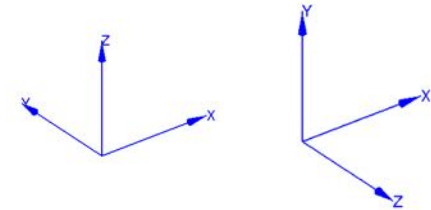
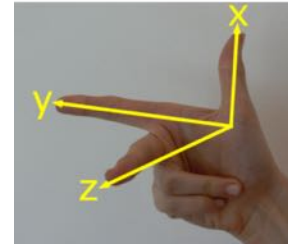
SE(2)



- $[R \ t; 0 \ 0 \ 1] = 3 \times 3$ matrix $\in \mathbf{SE}(2)$

(c) Henrik I Christensen

Rotations in 3D



- Rotation Matrices (DCM), $3 \times 3 \in \mathbf{SO}(3)$
- Columns of ${}^a\mathbf{R}_b$: axes of B in A. In MATLAB notation: $[X_b \ Y_b \ Z_b]$
- 9 numbers, but 3D **Manifold**

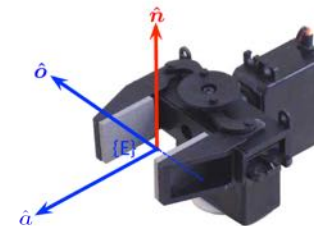
(c) Henrik I Christensen

Two vector representations

- In some applications it is convenient to talk about direction and orientation.
- Can you think of any?

(c) Henrik I Christensen

Representing end-effector Pose



- $Z_E =$ approach vector a
- $Y_E =$ orientation vector o
- $X_E = o \times a$

(c) Henrik I Christensen

Two vector representations

- In some applications it is convenient to talk about direction and orientation

- Grasping
- Docking



- We can define orientation (**o**) and direction (**a**) as two vectors. They complete define the rotation matrix **R** the third column can be defined as the orthogonal vector **n=o x a**

- The resulting rotation matrix is then

$$R = \begin{pmatrix} n_x & o_x & a_x \\ n_y & o_y & a_y \\ n_z & o_z & a_z \end{pmatrix}$$

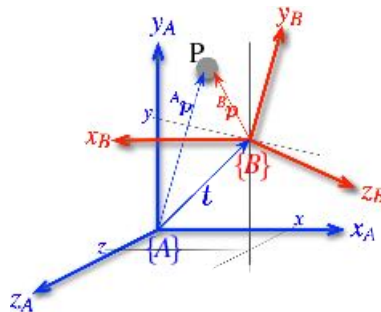
(c) Henrik I Christensen

2D Pose Representation

- In 2D we will represent the pose of an object by (x, y, θ)
- To represent translations we can use simple vector addition
 - $P_{\text{new}} = P_{\text{old}} + T$
- For rotations we can do simple operations
 - $P_{\text{new}} = R P_{\text{old}}$
- To make operations manageable it is easier to use homogenous transformations
- Representing the vector with an additional element
 - $P = (x, y, \theta, 1)^T$
- Or without orientation
 - $P = (x, y, 1)^T$
- We can consider also the 3 dimensional case

(c) Henrik I Christensen

Poses in 3D

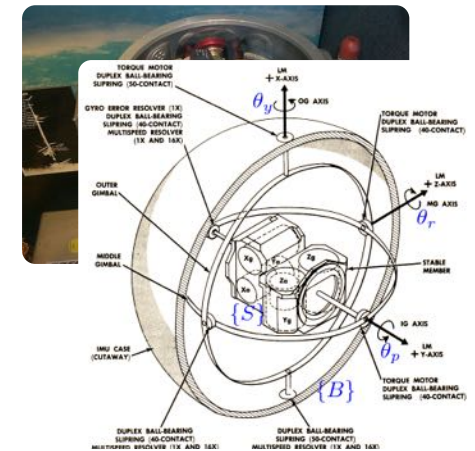


- $[R \ t; 0 \ 0 \ 0 \ 1]$, 4x4 matrix in **SE(3)**

(c) Henrik I Christensen

Representing 3D Rotations

- Rotation Matrices (DCM)
- Euler Angles
 - Eulerian
 - Cardanian
 - Gimbal Lock
- Axis-Angle
- Unit Quaternions



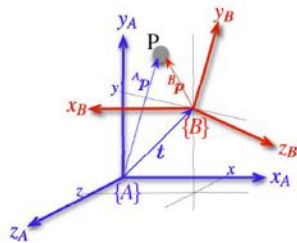
(c) Henrik I Christensen

3D Representation

- Basic model - $P = (x, y, z, 1)^T$
- Structure of a transformation matrix

$$\xi = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix}$$

- The relation between reference frames is exactly the same as before



(c) Henrik I Christensen

Representing rotations

- There are two commonly used rotation representations
- Euler rotations are represented by rotations about axes
 - ZYZ, XYZ, YXY, YZY, ZXZ
- Cardanian type rotations are about all three axes
 - XYZ, XZY, YZX, YXZ, ZXY, ZYX
- The Euler representation is widely used in Aerospace and Mechanical Dynamics. The most common model is ZYZ, i.e.

$$R = R_z(\phi)R_y(\theta)R_z(\psi)$$

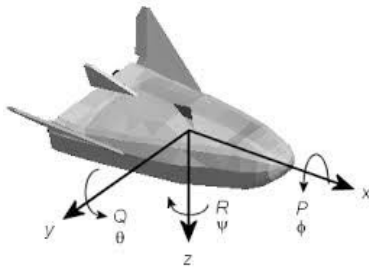
- Represented by the three angles of rotation.
- Another commonly used representation is pitch-roll-yaw

$$R = R_x(\theta_r)R_y(\theta_p)R_z(\theta_y)$$

- Often seen in maritime applications. Typically x direction is forward, y is to the right and z is downward.

(c) Henrik I Christensen

Roll-Pitch-Yaw



$$\begin{aligned} R_v^b(\phi, \theta, \psi) &= R_{v2}^b(\phi)R_{v1}^{v2}(\theta)R_v^{v1}(\psi) \\ &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & \sin \phi \\ 0 & -\sin \phi & \cos \phi \end{pmatrix} \begin{pmatrix} \cos \theta & 0 & -\sin \theta \\ 0 & 1 & 0 \\ \sin \theta & 0 & \cos \theta \end{pmatrix} \begin{pmatrix} \cos \psi & \sin \psi & 0 \\ -\sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} c_\theta c_\psi & c_\theta s_\psi & -s_\theta \\ s_\theta c_\psi & s_\theta s_\psi & c_\theta \\ c_\phi s_\theta c_\psi + s_\phi s_\psi & c_\phi s_\theta s_\psi - s_\phi s_\psi & c_\phi c_\theta \end{pmatrix}, \end{aligned} \quad (2.4)$$

Beard, 2011, Small Unmanned Aircraft

(c) Henrik I Christensen

UNIT Quaternions

- 2D:
 - θ
 - $[c\theta \ -s\theta; s\theta \ c\theta]$
 - $z = (c\theta, s\theta)$
- 3D:
 - $\theta_x, \theta_y, \theta_z$
 - $[r_{11} \ r_{12} \ r_{13}; r_{21} \ r_{22} \ r_{23}; r_{31} \ r_{32} \ r_{33}]$
 - $q = c(\theta/2) \langle s(\theta/2) \mathbf{v} \rangle$

For the case of quaternions our generalized pose is $\xi \sim \hat{q} \in \mathbb{Q}$ and

$$\hat{q}_1 \oplus \hat{q}_2 \mapsto s_1 s_2 - \mathbf{v}_1 \cdot \mathbf{v}_2, \langle s_1 \mathbf{v}_2 + s_2 \mathbf{v}_1 + \mathbf{v}_1 \times \mathbf{v}_2 \rangle$$

which is known as the quaternion or Hamilton product, and

$$\ominus \hat{q} \mapsto \hat{q}^{-1} = s, \langle -\mathbf{v} \rangle$$

which is the quaternion conjugate. The zero pose $0 \mapsto 1 \langle 0, 0, 0 \rangle$ which is the identity quaternion. A vector $\mathbf{v} \in \mathbb{R}^3$ is rotated $\hat{q} \cdot \mathbf{v} \mapsto \hat{q} \hat{q}(\mathbf{v}) \hat{q}^{-1}$ where $\hat{q}(\mathbf{v}) = 0, \langle \mathbf{v} \rangle$ is known as a pure quaternion.

(c) Henrik I Christensen

Quaternion rotation around an axis

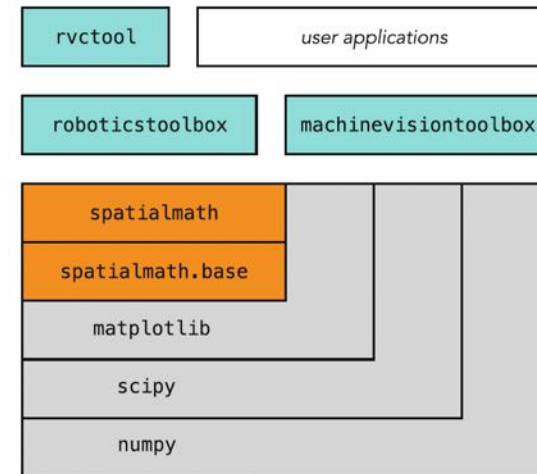
angle θ around the vector $\vec{a}$ expressed as a quaternion:

$$\vec{e} = \cos \frac{\theta}{2} + a_x \sin \frac{\theta}{2} \vec{i} + a_y \sin \frac{\theta}{2} \vec{j} + a_z \sin \frac{\theta}{2} \vec{k}$$

$$\vec{e} = \left(\cos \frac{\theta}{2}, \vec{a} \sin \frac{\theta}{2} \right)$$

(c) Henrik I Christensen

Representations of Pose in Toolbox



(c) Henrik I Christensen



(c) Henrik I Christensen

Time and motion

- Robotics is often about movement from a position A to B ($P_A \rightarrow P_B$)
- Typically we have additional constraints
 - Smooth trajectory
 - Time constraints
 - Maximum speed
 - Acceleration
 - Jerk
- How can we formulate trajectories that satisfy these constraints?

(c) Henrik I Christensen

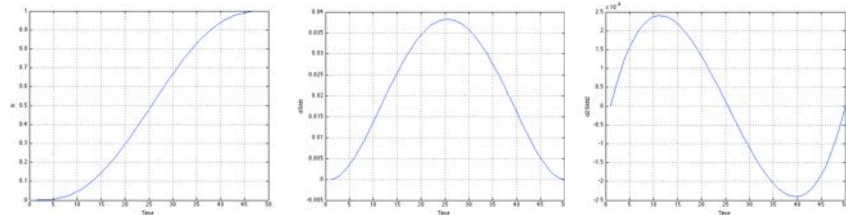
Paths (Locus) vs. Trajectories (Locus + Time)

Smooth trajectories in 1-D

- For representation of trajectories we can use polynomials
- A typical example is a quintic polynomial
 - $S(t) = A t^5 + B t^4 + C t^3 + D t^2 + E t + F$ with $t \in [0, T]$
- It will have smooth first and second order derivatives
- An example of such models is the clothoid model for roads in Europe
- An alternative is use of linear segments with parabolic blend
 - Smooth trajectories, with linear velocity variations
 - The motion has step changes in acceleration
 - Where do we see this type of trajectories?
- Lets look at a couple of examples

(c) Henrik I Christensen

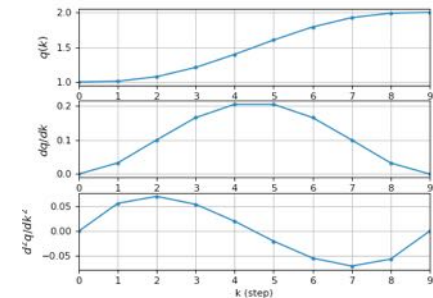
5th order polynomial trajectory



(c) Henrik I Christensen

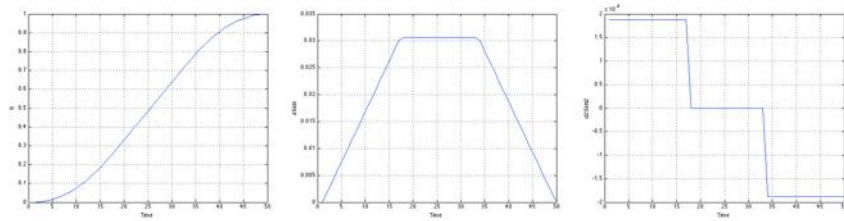
Example code from the RTB

```
from roboticstoolbox import quintic
tg = quintic(1, 2, 10)
tg
# Trajectory created by quintic: 10 time steps x 1 axes
len(tg)
# 10
tg.q
# array([1.          , 1.0115, 1.0764, 1.2099, 1.3967, 1.6033, 1.7901,
#        1.9236, 1.9885, 2.          ])
tg.plot()
```



(c) Henrik I Christensen

Linear trajectory with parabolic blends



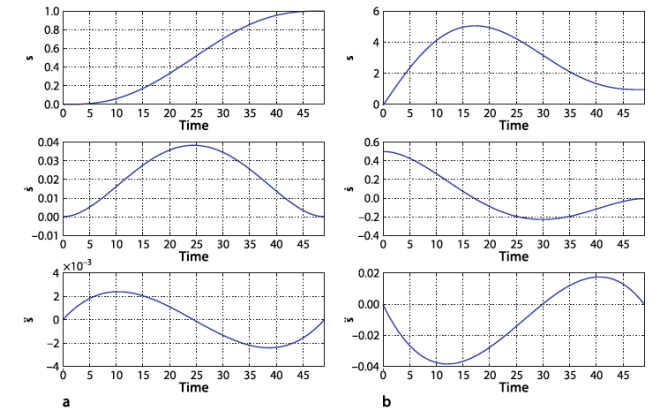
(c) Henrik I Christensen

Quintics

$$\hat{S}(t) = At^5 + Bt^4 + Ct^3 + Dt^2 + Et + F$$

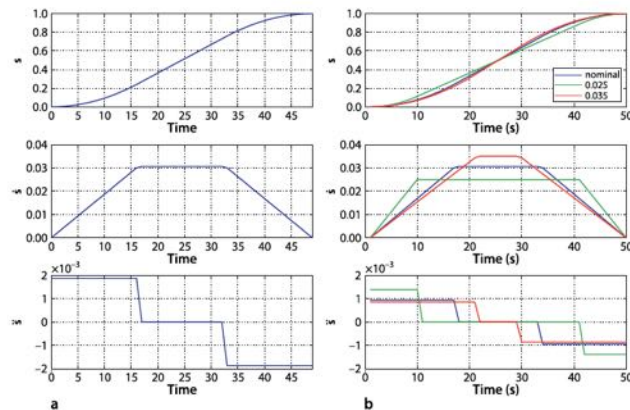
$$\hat{\dot{S}}(t) = 5At^4 + 4Bt^3 + 3Ct^2 + 2Dt + E$$

$$\hat{\ddot{S}}(t) = 20At^3 + 12Bt^2 + 6Ct + 2D$$



(c) Henrik I Christensen

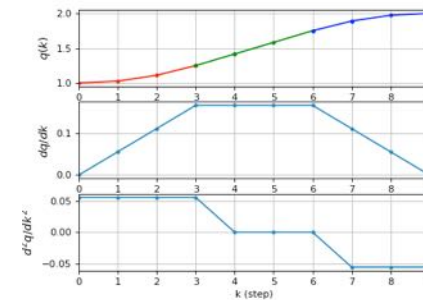
Linear Segment with Parabolic Blend



(c) Henrik I Christensen

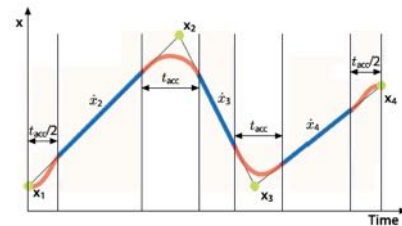
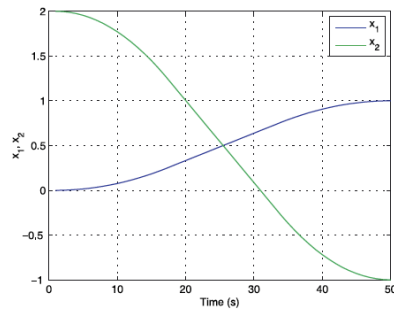
Pure trapezoidal

```
from roboticstoolbox import trapezoidal
tg = trapezoidal(1, 2, 10)
tg
# Trajectory created by trapezoidal: 10 time steps x 1 axes
len(tg)
# 10
tg.q
# array([1. , 1.0278, 1.1111, 1.25 , 1.4167, 1.5833, 1.75 , 1.8889, 1.9722, 2. ])
```



(c) Henrik I Christensen

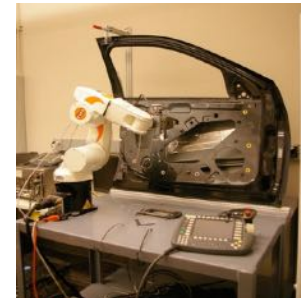
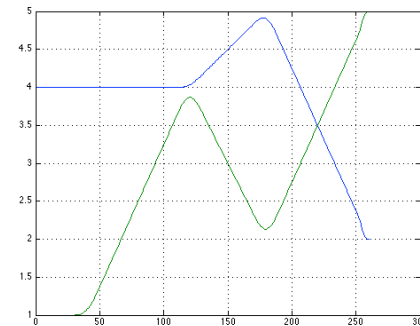
Multiple dimensions, Multi-SEGMENT



(c) Henrik I Christensen

Multi-segment trajectories

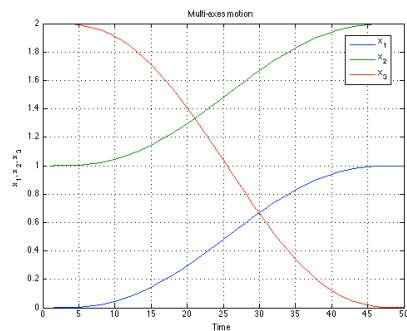
- Frequently the motion is not simply a motion from A to B, but a more complex motion to cover a set of trajectories.
- We can define a set of via points.
- We can “blend” at the via points to achieve acceleration around those areas, but closer to constant velocity elsewhere



(c) Henrik I Christensen

Multi-dimensional case

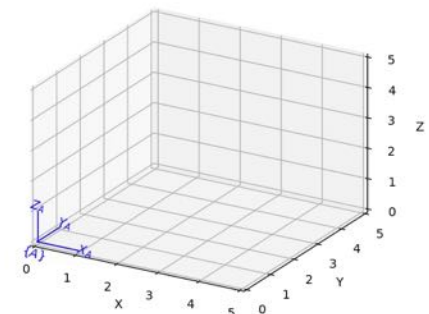
- The single dimensional case generalizes directly to multiple dimensions / multiple axes.
- Most robot controllers will directly generate trajectories given a specification of given constraints
- Example:



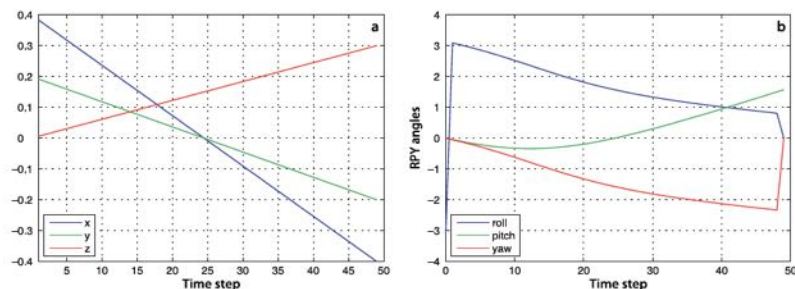
(c) Henrik I Christensen

Cartesian motion

- We could do the same in Cartesian space
- We can define points and do linear interpolation between them
- Animate1 - step change
- Animate2 - smooth motion



Cartesian Motion



(c) Henrik I Christensen

Examples of use of moving reference frames

- Most common example
 - Inertial Navigation System (INS)
 - Inertial Measurement Unit (IMU)
 - Estimation of ego-motion
 - Estimation of vehicle motion from GPS + INS/IMU
 - Estimation of end-effector motion
 - Estimation of motion of people (and gestures)

(c) Henrik I Christensen

Inertial Navigation

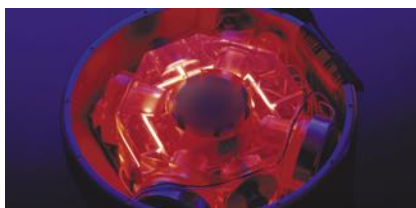
- Measure angular velocity with gyroscope, acceleration with accelerometer
- Integrate over time:

$$\mathbf{R}_{\langle k+1 \rangle} = \delta_t \mathbf{S}(\boldsymbol{\omega}) \mathbf{R}_{\langle k \rangle} + \mathbf{R}_{\langle k \rangle}$$

$${}^0\mathbf{a} = {}^0\mathbf{R}_B {}^B\mathbf{a}$$



USS Alabama

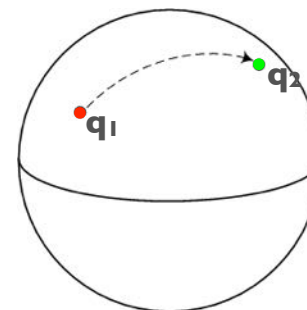


Ring-laser Gyro

(c) Henrik I Christensen

Interpolation in 3D

- Suppose $\delta = c(\theta/2) \langle s(\theta/2) \mathbf{v} \rangle$ is rotation between q_1 and q_2
- Slerp(q_1, q_2, t) = $q_1 \oplus c(t\theta/2) \langle s(t\theta/2) \mathbf{v} \rangle$



(c) Henrik I Christensen

Time-varying coordinate frames

- Handling rotation of coordinate frames is a little more complex
- We can write the derivative of the rotation matrix

$$\dot{R}(s) = S(\omega)R(t)$$

- Where

$$S(\omega) = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$$

- where omega describes the rotation around each axis
- We can do the transformation directly in homogenous coordinates, i.e.

$$T = \begin{bmatrix} S(\delta_\theta) & \delta_d \\ 0 & 0 \end{bmatrix} + I_{4 \times 4}$$

(c) Henrik I Christensen

Summary

- A few foundational concepts on position, orientation, time and trajectories
- Many different models for representation
- Position is relative easy
- Rotation has a number of issues to consider
- Using homogenous transformations often pays off
- We will use reference frames extensively.
- Robotics is about estimation, control and planning across references frames
- Many more foundational concepts covered in the:
 - Handbook of Robotics, B. Siciliano & O. Khatib, Springer Verlag, 2010
 - Robotics, Vision and Control, P. Corke, Springer Verlag, STAR Volume 73, Oct 2023 (incl. Python toolbox)

(c) Henrik I Christensen

Potential MS Projects

- VLA for Autonomous Driving under Resource Constraints x 2
- Perception and Planning for Autonomous Scooter
- Radar-Camera fusion for Perception
- A bank concierge robot - <https://www.youtube.com/watch?v=VNC4iFCMs1I>
- Projects - Winter 2027 (the project names are temporary):
 - Detection for Autonomous Driving
 - Radar-Camera fusion for Perception
 - Mothership

(c) Henrik I Christensen