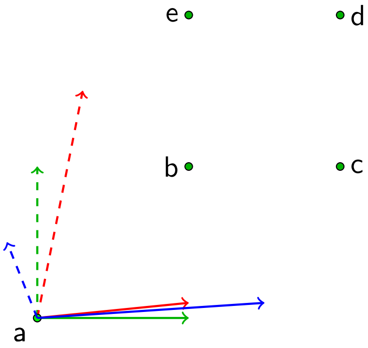
Cartesian Coordinates and Rigid Transformations

Spring 2020

Hasan Poonawala

Same Vector Space, Different Bases



A basis and an origin together form a **coordinate frame** or **reference frame**.

Change Of Basis

The coordinate $(1, 0)$ will produce different points under different bases.

Change Of Basis

The coordinate $(1, 0)$ will produce different points under different bases.

When we use a different basis, the coordinates assigned to a point must change, in order to correctly regenerate that point using the new basis.

Change Of Basis

The coordinate $(1, 0)$ will produce different points under different bases.

When we use a different basis, the coordinates assigned to a point must change, in order to correctly regenerate that point using the new basis.

Let A, B, C, ... be different coordinate frames.

A point p then has coordinates p^A, p^B, p^C ... corresponding to each basis.

Change Of Vector Space Basis

Given p^A, what is p^B, or p^C?

Change Of Vector Space Basis

Given p^A, what is p^B, or p^C? The vector e_B^i has coordinates $\left(e_B^i\right)^A = (T_{1i}, T_{2i}, \ldots, T_{ni})$ in frame A.

Change Of Vector Space Basis

Given p^A, what is p^B, or p^C? The vector e_B^i has coordinates $\left(e_B^i\right)^A = (T_{1i}, T_{2i}, \ldots, T_{ni})$ in frame A.

Let the coordinates of p in frame A be $p^A = (\alpha_1^A, \alpha_2^A, \ldots, \alpha_n^A)$, so that the point p can be expressed as

$$p = \sum_j^n \alpha_j^A e_A^j$$

Note that p is an abstract point equivalent to the coordinate-given combination of the basis $\{e_A^1, e_A^2, \ldots, e_A^n\}$.

Change Of Vector Space Basis

Given p^A, what is p^B, or p^C? The vector e_B^i has coordinates $\left(e_B^i\right)^A = (T_{1i}, T_{2i}, \ldots, T_{ni})$ in frame A.

Let the coordinates of p in frame A be $p^A = (\alpha_1^A, \alpha_2^A, \ldots, \alpha_n^A)$, so that the point p can be expressed as

$$p = \sum_j^n \alpha_j^A e_A^j$$

Note that p is an abstract point equivalent to the coordinate-given combination of the basis $\{e_A^1, e_A^2, \ldots, e_A^n\}$.

Similarly, if $p^B = (\beta_1^B, \beta_2^B, \ldots, \beta_n^B)$, then

$$p = \sum_i^n \beta_i^B e_B^i$$

Change Of Vector Space Basis

So, we can write

$$e_B^i = \sum_i^n T_{ji} e_A^i; \quad p = \sum_i^n \beta_i^B e_B^i; \quad p = \sum_j^n \alpha_j^A e_A^j \tag{1}$$

Change Of Vector Space Basis

So, we can write

$$e_B^i = \sum_i^n T_{ji} e_A^i; \quad p = \sum_i^n \beta_i^B e_B^i; \quad p = \sum_j^n \alpha_j^A e_A^j \tag{1}$$

Combining the first and second equation in (1), we get

$$\begin{aligned} p = \sum_i^n \beta_i^B e_B^i &= \sum_i^n \beta_i^B \left(\sum_j^n T_{ji} e_A^j \right) \\ &= \sum_j^n \left(\sum_i^n \left(\beta_i^B T_{ji} \right) \right) e_A^j \end{aligned} \tag{2}$$

Comparing (2) to the third equation in (1), we get

$$\alpha_j^A = \sum_i^n \left(\beta_i^B T_{ji} \right).$$

Change Of Vector Space Basis

The expression

$$\alpha_j^A = \sum_i^n \left(\beta_i^B T_{ji} \right)$$

represents a linear transformation of the coordinates of a point.

Change Of Vector Space Basis

The expression

$$\alpha_j^A = \sum_i^n \left(\beta_i^B T_{ji} \right)$$

represents a linear transformation of the coordinates of a point.

To see this, we abuse some notation and write:

$$p = \begin{bmatrix} e_A^1 & e_A^2 & \cdots & e_A^n \end{bmatrix} \begin{bmatrix} \alpha_1^A \\ \alpha_2^A \\ \vdots \\ \alpha_n^A \end{bmatrix} = \begin{bmatrix} e_B^1 & e_B^2 & \cdots & e_B^n \end{bmatrix} \begin{bmatrix} \beta_1^B \\ \beta_2^B \\ \vdots \\ \beta_n^B \end{bmatrix}$$

Change Of Vector Space Basis

The expression

$$\alpha_j^A = \sum_i^n \left(\beta_i^B T_{ji} \right)$$

represents a linear transformation of the coordinates of a point.

To see this, we abuse some notation and write:

$$p = \begin{bmatrix} e_A^1 & e_A^2 & \cdots & e_A^n \end{bmatrix} \begin{bmatrix} \alpha_1^A \\ \alpha_2^A \\ \vdots \\ \alpha_n^A \end{bmatrix} = \begin{bmatrix} e_B^1 & e_B^2 & \cdots & e_B^n \end{bmatrix} \begin{bmatrix} \beta_1^B \\ \beta_2^B \\ \vdots \\ \beta_n^B \end{bmatrix}$$

The coordinates of e_B^i in frame A give:

$$e_B^1 = \begin{bmatrix} e_A^1 & \cdots & e_A^n \end{bmatrix} \begin{bmatrix} T_{11} \\ \vdots \\ T_{n1} \end{bmatrix}, e_B^2 = \begin{bmatrix} e_A^1 & \cdots & e_A^n \end{bmatrix} \begin{bmatrix} T_{12} \\ \vdots \\ T_{n2} \end{bmatrix}, \ldots$$

Change Of Vector Space Basis

We can collect these expressions for point e_B^i as

$$\begin{bmatrix} e_B^1 & e_B^2 & \cdots & e_B^n \end{bmatrix} = \begin{bmatrix} e_A^1 & e_A^2 & \cdots & e_A^n \end{bmatrix} \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1n} \\ T_{21} & T_{22} & \cdots & T_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ T_{n1} & T_{n2} & \cdots & T_{nn} \end{bmatrix},$$

So that

$$\begin{bmatrix} e_B^1 & e_B^2 & \cdots & e_B^n \end{bmatrix} \begin{bmatrix} \beta_1^B \\ \beta_2^B \\ \vdots \\ \beta_n^B \end{bmatrix} = \begin{bmatrix} e_A^1 & e_A^2 & \cdots & e_A^n \end{bmatrix} \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1n} \\ T_{21} & T_{22} & \cdots & T_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ T_{n1} & T_{n2} & \cdots & T_{nn} \end{bmatrix} \begin{bmatrix} \beta_1^B \\ \beta_2^B \\ \vdots \\ \beta_n^B \end{bmatrix}$$

Change Of Vector Space Basis

Since

$$p = \begin{bmatrix} e_A^1 & e_A^2 & \cdots & e_A^n \end{bmatrix} \begin{bmatrix} \alpha_1^A \\ \alpha_2^A \\ \vdots \\ \alpha_n^A \end{bmatrix},$$

we find that transforming coordinates is a linear operation represented by matrix operations:

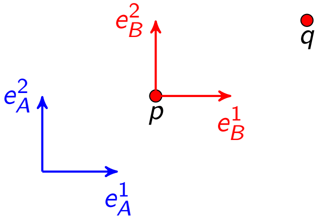
$$\begin{bmatrix} \alpha_1^A \\ \alpha_2^A \\ \vdots \\ \alpha_n^A \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1n} \\ T_{21} & T_{22} & \cdots & T_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ T_{n1} & T_{n2} & \cdots & T_{nn} \end{bmatrix} \begin{bmatrix} \beta_1^B \\ \beta_2^B \\ \vdots \\ \beta_n^B \end{bmatrix}$$

More compactly: $p^B = \left(T_B^A\right)^{-1} p^A$, where to example

$$T_B^A = \left[\left(e_B^1\right)^A \quad \left(e_B^2\right)^A \quad \cdots \quad \left(e_B^n\right)^A \right].$$

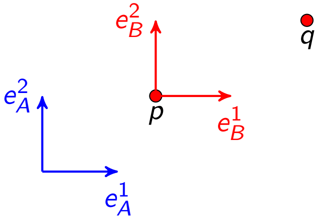
Change Of Origin

Suppose points p, q have coordinates p^A, q^A in a frame A. Consider frame B whose origin is at p, with the same basis elements for its vector space as the frame A. What is q^B?



Change Of Origin

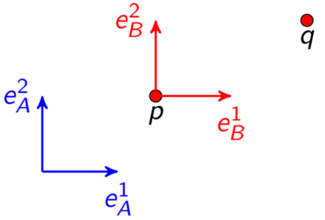
Suppose points p, q have coordinates p^A, q^A in a frame A. Consider frame B whose origin is at p, with the same basis elements for its vector space as the frame A. What is q^B?



The coordinates of q in frame B is the same as coordinates of the vector $v = q - p$ in the basis common to both frame A and B.

Change Of Origin

Suppose points p, q have coordinates p^A, q^A in a frame A. Consider frame B whose origin is at p, with the same basis elements for its vector space as the frame A. What is q^B?

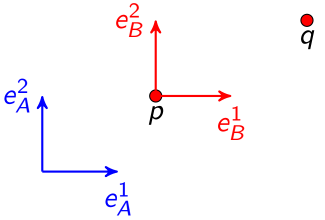


The coordinates of q in frame B is the same as coordinates of the vector $v = q - p$ in the basis common to both frame A and B.

Precisely because vectors are free, the coordinates of v in frame B will be the same as that in frame A.

Change Of Origin

Suppose points p, q have coordinates p^A, q^A in a frame A. Consider frame B whose origin is at p, with the same basis elements for its vector space as the frame A. What is q^B?



The coordinates of q in frame B is the same as coordinates of the vector $v = q - p$ in the basis common to both frame A and B.

Precisely because vectors are free, the coordinates of v in frame B will be the same as that in frame A. So, $q^B = q^A - p^A$.

In general, $q^B = q^A -$ (coordinates of origin of B in A)

Change Of Frames

Combining previous discussions, we get that:

To map coordinates from one frame to another, we express the coordinates of the basis vectors (through, say, matrix T_B^A) and the origin of one frame in another (through, say coordinate vector o_B^A), and use the map

$$p^B = \left(T_B^A\right)^{-1}(p^A - o_B^A)$$

Checkpoint

- We relate points by picking an origin and using a vector space

Checkpoint

- We relate points by picking an origin and using a vector space
- Coordinates of vectors, given a basis, become coordinates of points

Checkpoint

- We relate points by picking an origin and using a vector space
- Coordinates of vectors, given a basis, become coordinates of points
- These coordinates correspond to a frame: origin + basis

Checkpoint

- We relate points by picking an origin and using a vector space
- Coordinates of vectors, given a basis, become coordinates of points
- These coordinates correspond to a frame: origin + basis
- All coordinates are n-tuples, we can't say anything about the basis from the coordinates.

Checkpoint

- We relate points by picking an origin and using a vector space
- Coordinates of vectors, given a basis, become coordinates of points
- These coordinates correspond to a frame: origin + basis
- All coordinates are *n*-tuples, we can't say anything about the basis from the coordinates.
- If we know the bases and origins, we can transform coordinates from one frame to another.

Checkpoint

- We relate points by picking an origin and using a vector space
- Coordinates of vectors, given a basis, become coordinates of points
- These coordinates correspond to a frame: origin + basis
- All coordinates are n-tuples, we can't say anything about the basis from the coordinates.
- If we know the bases and origins, we can transform coordinates from one frame to another.

Checkpoint

- We relate points by picking an origin and using a vector space
- Coordinates of vectors, given a basis, become coordinates of points
- These coordinates correspond to a frame: origin + basis
- All coordinates are n-tuples, we can't say anything about the basis from the coordinates.
- If we know the bases and origins, we can transform coordinates from one frame to another.

If all bases for the plane give us two numbers, what's special about a basis where the two elements are at 90 degrees , and have the same 'length'?

Norms and Distances

Let's reconsider our hard won example:

$$T_B^A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}; \quad q^A = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \implies q^B = \left(T_B^A\right)^{-1} q^A = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Norms and Distances

Let's reconsider our hard won example:

$$T_B^A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}; \quad q^A = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \implies q^B = \left(T_B^A\right)^{-1} q^A = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

REMEMBER: We're talking about the same two points in Euclidean space.

Norms and Distances

Let's reconsider our hard won example:

$$T_B^A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}; \quad q^A = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \implies q^B = \left(T_B^A\right)^{-1} q^A = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

REMEMBER: We're talking about the same two points in Euclidean space.

$\|q^A\|_A = \sqrt{2}$. $\|q^B\|_B = 1$.

Norms and Distances

Let's reconsider our hard won example:

$$T_B^A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}; \quad q^A = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \implies q^B = \left(T_B^A\right)^{-1} q^A = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

REMEMBER: We're talking about the same two points in Euclidean space.

$\|q^A\|_A = \sqrt{2}$. $\|q^B\|_B = 1$. What gives?

Norms and Distances

Let's reconsider our hard won example:

$$T_B^A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}; \quad q^A = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \implies q^B = \left(T_B^A\right)^{-1} q^A = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

REMEMBER: We're talking about the same two points in Euclidean space.

$\|q^A\|_A = \sqrt{2}$. $\|q^B\|_B = 1$. What gives?

Note that $\|q^B\|_B = \|\left(T_B^A\right)^{-1} q^A\|_A$.

Q: What kinds of matrices preserve the norms of the vectors they act upon?

Special Orthogonal Group in Three Dimensions

if $T_B^A \in SO(3)$, then we'd have norm-preservation.

Definition ($SO(3)$)

The Special Orthogonal Group $SO(3)$ is the set of matrices $R \in \mathbb{R}^{3\times 3}$ such that

$$R^T R = Id, \text{and } \det R = 1$$

.

$SO(3)$ is known as the orientation group **and** the rotation group.

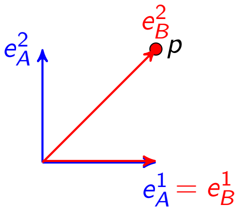
Example

Problem: Find p^B if $p^A = (1, 1)$.
Solution: From the diagram,

$$e_B^1 = e_A^1$$
$$e_B^2 = e_A^1 + e_A^2$$
$$\implies T_B^A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

Apply the formula:

$$p^B = \left(T_B^A\right)^{-1} p^A$$
$$= T_A^B p^A = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$



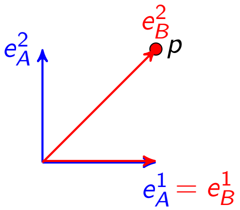
The columns of T_B^A tell us how to draw the basis of B in A

to defn

Example

Problem: Find p^B if $p^A = (1,1)$.
Solution: From the diagram,

$$e_B^1 = e_A^1$$
$$e_B^2 = e_A^1 + e_A^2$$
$$\implies T_B^A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$



Not norm-preserving.

Apply the formula:

$$p^B = \left(T_B^A\right)^{-1} p^A$$
$$= T_A^B p^A = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\left(T_A^B\right)^T T_A^B = \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix}$$

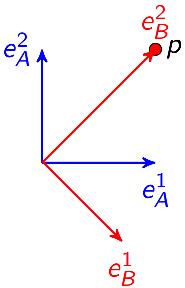
Example

Problem: Find p^B if $p^A = (1, 1)$.
Solution: From the diagram,

$$e_B^1 = \frac{1}{\sqrt{2}} e_A^1 - \frac{1}{\sqrt{2}} e_A^2$$
$$e_B^2 = e_A^1 + e_A^2$$
$$\implies T_B^A = \begin{bmatrix} \frac{1}{\sqrt{2}} & 1 \\ -\frac{1}{\sqrt{2}} & 1 \end{bmatrix}$$

Apply the formula:

$$p^B = \left(T_B^A\right)^{-1} p^A$$
$$= T_A^B p^A = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$



The columns of T_B^A tell us how to draw the basis of B in A

to defn

Example

Problem: Find p^B if $p^A = (1,1)$.
Solution: From the diagram,

$$e_B^1 = \frac{1}{\sqrt{2}} e_A^1 - \frac{1}{\sqrt{2}} e_A^2$$

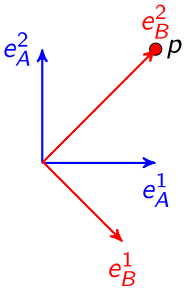
$$e_B^2 = e_A^1 + e_A^2$$

$$\implies T_B^A = \begin{bmatrix} \frac{1}{\sqrt{2}} & 1 \\ -\frac{1}{\sqrt{2}} & 1 \end{bmatrix}$$

Apply the formula:

$$p^B = \left(T_B^A\right)^{-1} p^A$$

$$= T_A^B p^A = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$



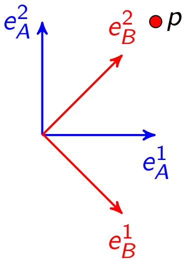
Not norm-preserving.

$$\left(T_A^B\right)^T T_A^B = \begin{bmatrix} 0.75 & -0.25 \\ -0.25 & 0.75 \end{bmatrix}$$

Example

Problem: Find p^B if $p^A = (1, 1)$.
Solution: From the diagram,

$$e_B^1 = \frac{1}{\sqrt{2}} e_A^1 - \frac{1}{\sqrt{2}} e_A^2$$
$$e_B^2 = \frac{1}{\sqrt{2}} e_A^1 + \frac{1}{\sqrt{2}} e_A^2$$
$$\implies T_B^A = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$$



Apply the formula:

$$p^B = \left(T_B^A\right)^{-1} p^A$$
$$= T_A^B p^A = \begin{bmatrix} 0 \\ \sqrt{2} \end{bmatrix}$$

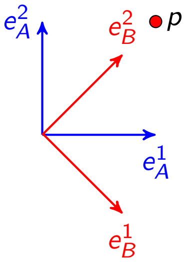
Example

Problem: Find p^B if $p^A = (1,1)$.
Solution: From the diagram,

$$e_B^1 = \frac{1}{\sqrt{2}} e_A^1 - \frac{1}{\sqrt{2}} e_A^2$$
$$e_B^2 = \frac{1}{\sqrt{2}} e_A^1 + \frac{1}{\sqrt{2}} e_A^2$$
$$\implies T_B^A = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$$

Apply the formula:

$$p^B = \left(T_B^A\right)^{-1} p^A$$
$$= T_A^B p^A = \begin{bmatrix} 0 \\ \sqrt{2} \end{bmatrix}$$



norm-preserving!

$$\left(T_A^B\right)^T T_A^B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Orthonormal Vectors

We have seen that

$$T_B^A = \begin{bmatrix} \left(e_B^1\right)^A & \left(e_B^2\right)^A & \cdots & \left(e_B^n\right)^A \end{bmatrix}.$$

Therefore,

$$\left(T_B^A\right)^T T_B^A = \begin{bmatrix} \left(\left(e_B^1\right)^A\right)^T \\ \left(\left(e_B^2\right)^A\right)^T \\ \vdots \\ \left(\left(e_B^1\right)^A\right)^T \end{bmatrix} \begin{bmatrix} \left(e_B^1\right)^A & \left(e_B^2\right)^A & \cdots & \left(e_B^n\right)^A \end{bmatrix}$$

Orthonormal Vectors

$$\left(T_B^A\right)^T T_B^A = \begin{bmatrix} \left(\left(e_B^1\right)^A\right)^T \left(e_B^1\right)^A & \left(\left(e_B^1\right)^A\right)^T \left(e_B^2\right)^A & \cdots & \left(\left(e_B^1\right)^A\right)^T \left(e_B^n\right)^A \\ \left(\left(e_B^2\right)^A\right)^T \left(e_B^1\right)^A & \left(\left(e_B^2\right)^A\right)^T \left(e_B^2\right)^A & \cdots & \left(\left(e_B^2\right)^A\right)^T \left(e_B^n\right)^A \\ \vdots & \vdots & \ddots & \vdots \\ \left(\left(e_B^n\right)^A\right)^T \left(e_B^1\right)^A & \left(\left(e_B^n\right)^A\right)^T \left(e_B^2\right)^A & \cdots & \left(\left(e_B^n\right)^A\right)^T \left(e_B^n\right)^A \end{bmatrix}.$$

$$= \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

Effectively, the coordinates of basis vectors of B in frame A are unit length and perpendicular to each other.

Checkpoint

- Frames are origin+basis

Checkpoint

- Frames are origin+basis
- Frames define vector coordinates for points in Euclidean space, relative to the frame

Checkpoint

- Frames are origin+basis
- Frames define vector coordinates for points in Euclidean space, relative to the frame
- Can transform vector coordinates of a point in different frames using an affine map

Checkpoint

- Frames are origin+basis
- Frames define vector coordinates for points in Euclidean space, relative to the frame
- Can transform vector coordinates of a point in different frames using an affine map
- To preserve distance, the linear part of the affine map must be in SO(3)

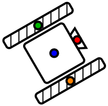
Checkpoint

- Frames are origin+basis
- Frames define vector coordinates for points in Euclidean space, relative to the frame
- Can transform vector coordinates of a point in different frames using an affine map
- To preserve distance, the linear part of the affine map must be in SO(3)
- $T_B^A \in SO(3)$ when basis vectors are all unit length, mutually perpendicular.

Checkpoint

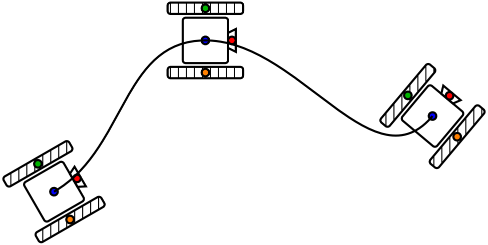
- Frames are origin+basis
- Frames define vector coordinates for points in Euclidean space, relative to the frame
- Can transform vector coordinates of a point in different frames using an affine map
- To preserve distance, the linear part of the affine map must be in SO(3)
- $T_B^A \in SO(3)$ when basis vectors are all unit length, mutually perpendicular.
- The coordinate transformation is then $p^B = \left(R_B^A\right)^{-1}\left(p^A - o_B^A\right)$ ▸ mobile robot

Coordinate Transformation Vs Rigid Motion



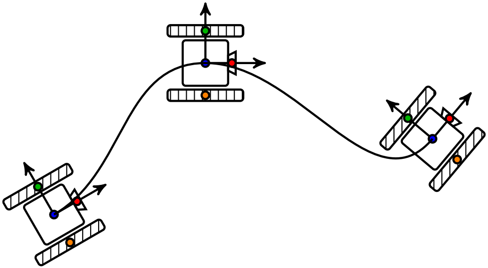
Consider a robot with a center, a camera in ‘front’, and two wheels to the side.

Coordinate Transformation Vs Rigid Motion



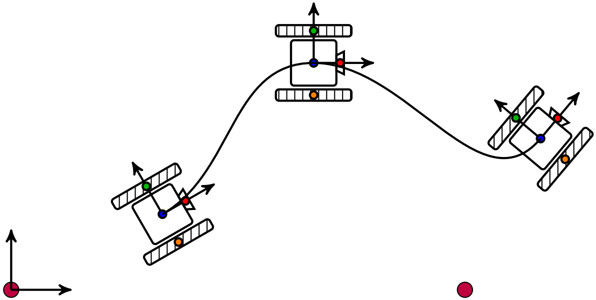
Whenever we move the robot, the distances between these points don't change.

Coordinate Transformation Vs Rigid Motion



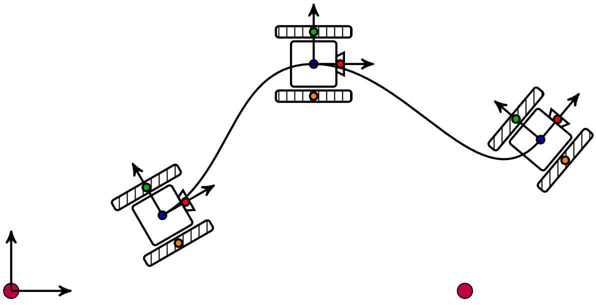
As the robot moves, we can take a snapshot of these points, and they each define a coordinate frame for Euclidean space.

Coordinate Transformation Vs Rigid Motion



Q1: How would the robot compare observations of either purple point over time? A1: Coordinate transformations

Coordinate Transformation Vs Rigid Motion



Q1: How would the robot compare observations of either purple point over time? A1: Coordinate transformations

Q2: How do we keep track of all the points on the robots?

A2: Coordinate transformations, but reinterpreted. ▸ rigid motion

Coordinate Transformation Vs Rigid Motion

We know how to express all points in the robot's frame in any other frame: Use a distance-preserving coordinate transformation.

Coordinate Transformation Vs Rigid Motion

We know how to express all points in the robot's frame in any other frame: Use a distance-preserving coordinate transformation.

This transformation itself becomes a representative for all points on the robot.

Coordinate Transformation Vs Rigid Motion

We know how to express all points in the robot's frame in any other frame: Use a distance-preserving coordinate transformation.

This transformation itself becomes a representative for all points on the robot.

We have seen that

$$p^B = \left(T_B^A\right)^{-1}\left(p^A - o_B^A\right) \tag{3}$$

Let $d = o_B^A$ and $R = T_B^A$.

Coordinate Transformation Vs Rigid Motion

We know how to express all points in the robot's frame in any other frame: Use a distance-preserving coordinate transformation.

This transformation itself becomes a representative for all points on the robot.

We have seen that

$$p^B = \left(T_B^A\right)^{-1}\left(p^A - o_B^A\right) \tag{3}$$

Let $d = o_B^A$ and $R = T_B^A$. From (3), we can derive

$$p^B = R^{-1}\left(p^A - d\right) \tag{4}$$

$$p^A = R\ p^B + d. \tag{5}$$

Coordinate Transformation Vs Rigid Motion

We know how to express all points in the robot's frame in any other frame: Use a distance-preserving coordinate transformation.

This transformation itself becomes a representative for all points on the robot.

We have seen that

$$p^B = \left(T_B^A\right)^{-1} \left(p^A - o_B^A\right) \tag{3}$$

Let $d = o_B^A$ and $R = T_B^A$. From (3), we can derive

$$p^B = R^{-1} \left(p^A - d\right) \tag{4}$$

$$p^A = R \; p^B + d. \tag{5}$$

Rigid Body Pose

We refer to the pair (d, R) as the pose – relative to frame A – of the rigid body to which frame B is attached.

Rigid Body Pose

We refer to the pair (d, R) as the pose – relative to frame A – of the rigid body to which frame B is attached.

Let's reinterpret the two affine transformations associated with (d, R). Consider vector v in frame A:

$$u_1 = R^{-1}(v - d) \qquad \text{(Change of Basis)} \qquad (6)$$

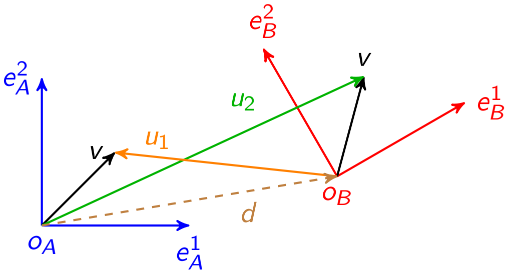
$$u_2 = R\ v + d. \qquad \text{(Rigid motion)} \qquad (7)$$

Rigid Body Pose

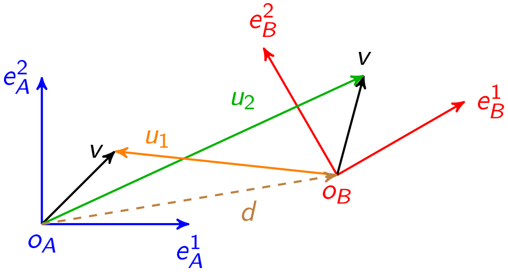
Let's reinterpret the two affine transformations associated with (d, R). Consider vector v in frame A:

$$u_1 = R^{-1}(v - d) \qquad \text{(Change of Basis)} \tag{6}$$

$$u_2 = R\ v + d. \qquad \text{(Rigid motion)} \tag{7}$$

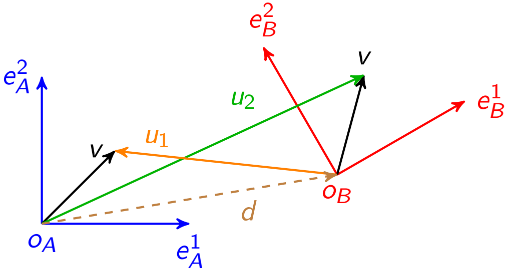


Rigid Body Pose



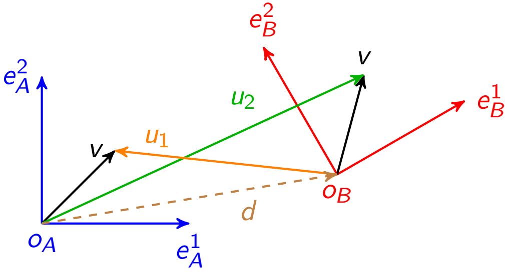
- If we view u_1 as coordinates in frame B, we've changed coordinates of v from world to body frame.
- If we view u_2 as coordinates in frame A, we've moved the point $o_A \oplus v$ relative to frame A.

Rigid Body Pose



The pair $(d, R) \in \mathbb{R}^3 \times \mathrm{SO}(3)$ tells us how to move points in frame A to achieve the same coordinates in frame B.

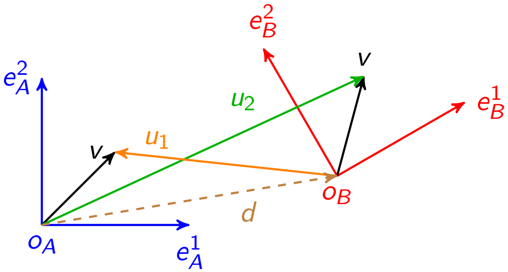
Rigid Body Pose



The pair $(d, R) \in \mathbb{R}^3 \times \mathrm{SO}(3)$ tells us how to move points in frame A to achieve the same coordinates in frame B.

d is a vector, the coordinates of origin of frame B, and R is a matrix containing the coordinates of axes of B, both relative to A

Rigid Body Pose

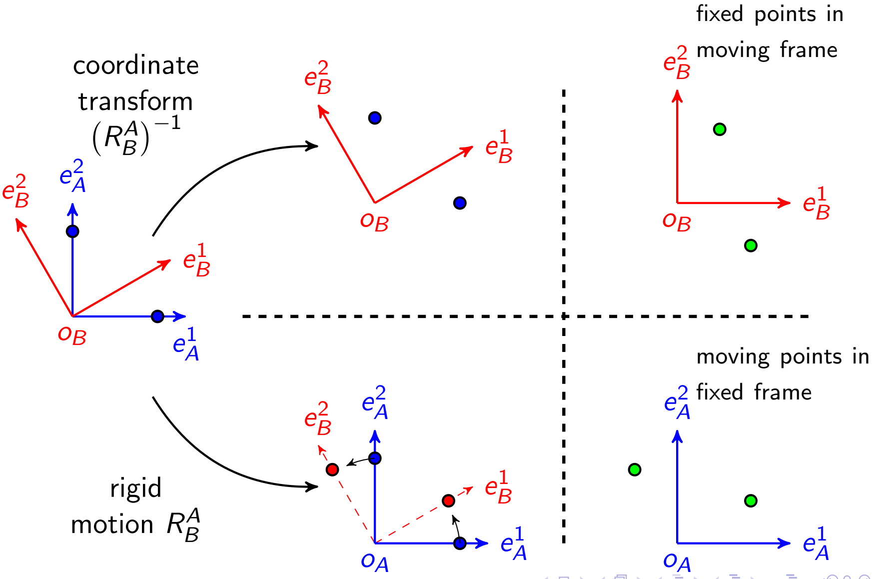


The pair $(d, R) \in \mathbb{R}^3 \times \mathrm{SO}(3)$ tells us how to move points in frame A to achieve the same coordinates in frame B.

d is a vector, the coordinates of origin of frame B, and R is a matrix containing the coordinates of axes of B, both relative to A

Move in frame A = reorient by R and then move by d : $Rv + d$

Example



coordinate
transform
$\left(R_B^A\right)^{-1}$
e_B^2
e_A^2
e_B^1
o_B
e_A^1
e_B^2
e_B^1
o_B
fixed points in
moving frame
e_B^2
o_B
e_B^1
rigid
motion R_B^A
e_B^2
e_A^2
e_B^1
o_A
e_A^1
moving points in
fixed frame
e_A^2
o_A
e_A^1

Special Euclidean Group $\mathrm{SE}(3)$

Coordinates of points in 3D Euclidean space $= p^A \in \mathbb{R}^3$
Coordinates of cartesian frames in 3D Euclidean space $=$
$(d, R) \in \mathbb{R}^3 \times \mathrm{SO}(3)$

Points: Euclidean Space :: Cartesian Frames : Special Euclidean Group

Special Euclidean Group $\mathrm{SE}(3)$

Coordinates of points in 3D Euclidean space $= p^A \in \mathbb{R}^3$
Coordinates of cartesian frames in 3D Euclidean space $=$
$(d, R) \in \mathbb{R}^3 \times \mathrm{SO}(3)$

Points: Euclidean Space :: Cartesian Frames : Special Euclidean Group

Affine Space : Euclidean Space :: G-Torsor : Special Euclidean Group

Special Euclidean Group $\mathrm{SE}(3)$

Coordinates of points in 3D Euclidean space $= p^A \in \mathbb{R}^3$
Coordinates of cartesian frames in 3D Euclidean space $=$
$(d, R) \in \mathbb{R}^3 \times \mathrm{SO}(3)$

Points: Euclidean Space :: Cartesian Frames : Special Euclidean Group

Affine Space : Euclidean Space :: G-Torsor : Special Euclidean Group

G-Torsor: A group G with an action that maps a group element to another group element

Special Euclidean Group $\mathrm{SE}(3)$

Coordinates of points in 3D Euclidean space $= p^A \in \mathbb{R}^3$
Coordinates of cartesian frames in 3D Euclidean space $=$
$(d, R) \in \mathbb{R}^3 \times \mathrm{SO}(3)$

Points: Euclidean Space :: Cartesian Frames : Special Euclidean Group

Affine Space : Euclidean Space :: G-Torsor : Special Euclidean Group

G-Torsor: A group G with an action that maps a group element to another group element

Again, no coordinate frame is unique.
For a G-Torsor, we don't define origin+basis (not a vector space).

Special Euclidean Group $\mathrm{SE}(3)$

Coordinates of points in 3D Euclidean space $= p^A \in \mathbb{R}^3$
Coordinates of cartesian frames in 3D Euclidean space $=$
$(d, R) \in \mathbb{R}^3 \times \mathrm{SO}(3)$

Points: Euclidean Space :: Cartesian Frames : Special Euclidean Group

Affine Space : Euclidean Space :: G-Torsor : Special Euclidean Group

G-Torsor: A group G with an action that maps a group element to another group element

Again, no coordinate frame is unique.
For a G-Torsor, we don't define origin+basis (not a vector space).

Instead, we define an identity element (it's a group): the reference coordinate frame.

Homogenous Transformations

We can convert the affine map between two Euclidean spaces of dimension 3 into a linear map between two subsets of $\mathbb{R}^4$.

Define a homogenization $h\colon \mathbb{R}^3 \mapsto \mathbb{R}^4$ as $h\left(p^A\right) = \begin{bmatrix} p^A \\ 1 \end{bmatrix}$.

If $p^A = Rp^B + d$, then

$$h\left(p^A\right) = \begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix} h\left(p^B\right). \tag{6}$$

The matrix $\begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix}$ represents a homogenous transformation, and forms a group.
You show the group structure in HW2.

An Example

On Board: Three robot poses

Checkpoint

- The coordinate transformation is $p^B = \left(R_B^A\right)^{-1}\left(p^A - o_B^A\right)$
- Norm-preserving coordinate transformation $=$ rigid motion of points within the same coordinate frame.
- Set of rigid body poses/rigid motions forms a group: $\mathrm{SE}(3)$
- After choosing a reference frame, we assign coordinates – aka rigid body pose – (d, R) to frame (Torsor structure)

Back to SO(3)

- Rigid bodies correspond to cartesian frames

Back to $\mathbf{SO}(3)$

- Rigid bodies correspond to cartesian frames
- Cartesian frames have a position and orientation

Back to SO(3)

- Rigid bodies correspond to cartesian frames
- Cartesian frames have a position and orientation
- Orientations are also a G-Torsor

Back to SO(3)

- Rigid bodies correspond to cartesian frames
- Cartesian frames have a position and orientation
- Orientations are also a G-Torsor
- Rotations ($\sim$ vector) help us get from one orientation ($\sim$ point) to another

Back to SO(3)

- Rigid bodies correspond to cartesian frames
- Cartesian frames have a position and orientation
- Orientations are also a G-Torsor
- Rotations ($\sim$ vector) help us get from one orientation ($\sim$ point) to another
- The rotation relative to a reference that produces an orientation becomes the 'orientation' coordinate of that frame.

Back to SO(3)

- Rigid bodies correspond to cartesian frames
- Cartesian frames have a position and orientation
- Orientations are also a G-Torsor
- Rotations ($\sim$ vector) help us get from one orientation ($\sim$ point) to another
- The rotation relative to a reference that produces an orientation becomes the 'orientation' coordinate of that frame.
- We've called this matrix T_B^A, R_B^A, R, T

Back to SO(3)

- Rigid bodies correspond to cartesian frames
- Cartesian frames have a position and orientation
- Orientations are also a G-Torsor
- Rotations ($\sim$ vector) help us get from one orientation ($\sim$ point) to another
- The rotation relative to a reference that produces an orientation becomes the 'orientation' coordinate of that frame.
- We've called this matrix T_B^A, R_B^A, R, T
- The G-Torsor nature is why SO(3) is called both the rotation group and the orientation group.

Back to SO(3)

- Rigid bodies correspond to cartesian frames
- Cartesian frames have a position and orientation
- Orientations are also a G-Torsor
- Rotations ($\sim$ vector) help us get from one orientation ($\sim$ point) to another
- The rotation relative to a reference that produces an orientation becomes the 'orientation' coordinate of that frame.
- We've called this matrix T_B^A, R_B^A, R, T
- The G-Torsor nature is why SO(3) is called both the rotation group and the orientation group.
- Assigning coordinates to an orientation is the same as defining the rotation that generates that frame.

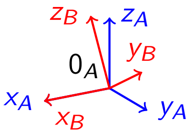
Basic Rotations

Consider three frames rotated about each one of the world frame axes by an angle θ.

Basic Rotations

Consider three frames rotated about each one of the world frame axes by an angle θ. Each rotation is given by

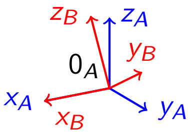
$$R_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix}$$



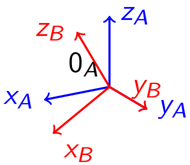
Basic Rotations

Consider three frames rotated about each one of the world frame axes by an angle θ. Each rotation is given by

$$R_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix}$$



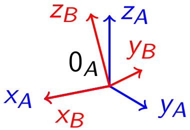
$$R_{y,\theta} = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}$$



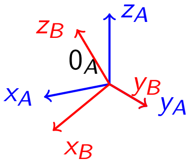
Basic Rotations

Consider three frames rotated about each one of the world frame axes by an angle θ. Each rotation is given by

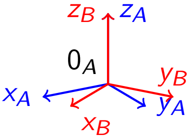
$$R_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix}$$



$$R_{y,\theta} = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}$$



$$R_{z,\theta} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



General Rotations

We can construct a general rotation using a sequence of basic rotations. (Compare to Euclidean space)

General Rotations

We can construct a general rotation using a sequence of basic rotations. (Compare to Euclidean space)

So, orientation coordinates can be derived by sequences of basic rotations.

General Rotations

We can construct a general rotation using a sequence of basic rotations. (Compare to Euclidean space)

So, orientation coordinates can be derived by sequences of basic rotations.

For Euclidean vector spaces, the order of a sequence of (vector space) operations didn't matter: $v + w = w + v$.

General Rotations

We can construct a general rotation using a sequence of basic rotations. (Compare to Euclidean space)

So, orientation coordinates can be derived by sequences of basic rotations.

For Euclidean vector spaces, the order of a sequence of (vector space) operations didn't matter: $v + w = w + v$.

For rotations, they do. In general, $R_1 R_2 \neq R_2 R_1$.

Change-of-Identity For Rotations

Suppose we define an orientation B relative to a oriention A through a rotation R_B^A.

Change-of-Identity For Rotations

Suppose we define an orientation B relative to a oriention A through a rotation R_B^A.

Now, someone decides to change the identity element to be orientation C, with coordinate R_C^A (in frame A).

Change-of-Identity For Rotations

Suppose we define an orientation B relative to a oriention A through a rotation R_B^A.

Now, someone decides to change the identity element to be orientation C, with coordinate R_C^A (in frame A).

Which rotation R_B^C below correctly defines orientation of B relative to orientation C?

1. $R_B^C = R_B^A R_A^C$
2. $R_B^C = R_B^A R_C^A$
3. $R_B^C = R_A^C R_B^A$
4. $R_B^C = R_C^A R_B^A$

Change-of-Identity For Rotations

Suppose we define an orientation B relative to a oriention A through a rotation R_B^A.

Now, someone decides to change the identity element to be orientation C, with coordinate R_C^A (in frame A).

Which rotation R_B^C below correctly defines orientation of B relative to orientation C?

1. $R_B^C = R_B^A R_A^C$
2. $R_B^C = R_B^A R_C^A$
3. $R_B^C = R_A^C R_B^A$
4. $R_B^C = R_C^A R_B^A$

How would you pick the right transformation? Why did we not consider R_A^B?

Change-of-Identity For Rotations

Rotation matrix R_B^A gives the coordinates of the basis vectors of frames B in A.

Change-of-Identity For Rotations

Rotation matrix R_B^A gives the coordinates of the basis vectors of frames B in A.

We want to change the frame of these coordinates to frame C.

Change-of-Identity For Rotations

Rotation matrix R_B^A gives the coordinates of the basis vectors of frames B in A.

We want to change the frame of these coordinates to frame C.

To change the coordinates of vectors from A to C, we must pre-multiply by $\left(R_C^A\right)^{-1} = R_A^C$. So,

$$R_B^C = R_A^C R_B^A$$

.

Change-of-Identity For Rotations

Rotation matrix R_B^A gives the coordinates of the basis vectors of frames B in A.

We want to change the frame of these coordinates to frame C.

To change the coordinates of vectors from A to C, we must pre-multiply by $\left(R_C^A\right)^{-1} = R_A^C$. So,

$$R_B^C = R_A^C R_B^A$$

.

Alternatively, The rigid motion in A corresponding to frame B is R_B^A; the rigid motion in frame C corresponding to frame A is R_A^C.

The combined rigid motion in C is achieved by first moving by R_B^A in C, then moving the result by R_A^C.
Therefore,

$$R_B^C = R_A^C R_B^A$$

.

Change-of-Identity For Rotations

Rotation matrix R_B^A gives the coordinates of the basis vectors of frames B in A.

We want to change the frame of these coordinates to frame C.

To change the coordinates of vectors from A to C, we must pre-multiply by $\left(R_C^A\right)^{-1} = R_A^C$. So,

$$R_B^C = R_A^C R_B^A$$

.

Alternatively, The rigid motion in A corresponding to frame B is R_B^A; the rigid motion in frame C corresponding to frame A is R_A^C.

The combined rigid motion in C is achieved by first moving by R_B^A in C, then moving the result by R_A^C.
Therefore,

$$R_B^C = R_A^C R_B^A$$

.

Change-of-Identity For Rotations

- The rotation R^A is relative to frame A.
- A general orientation P has coordinates R_P^A in frame A
- Rotating this point results in an orientation $R^A R_P^A$ in frame A:

$$R_P^A \mapsto R^A R_P^A$$

- But note that $R_P^A = R_C^A R_P^C$
- Therefore :

$$R_C^A R_P^C \mapsto R^A R_C^A R_P^C, \text{or}$$

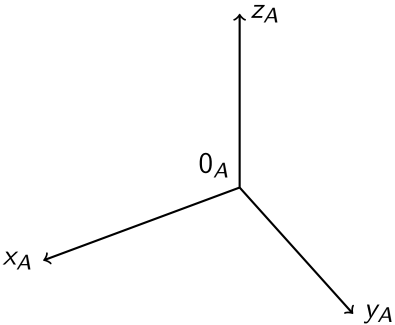
$$R_P^C \mapsto \left(R_C^A\right)^{-1} R^A R_C^A R_P^C, \text{or}$$

- Therefore, a rotation R^A in frame A becomes a rotation

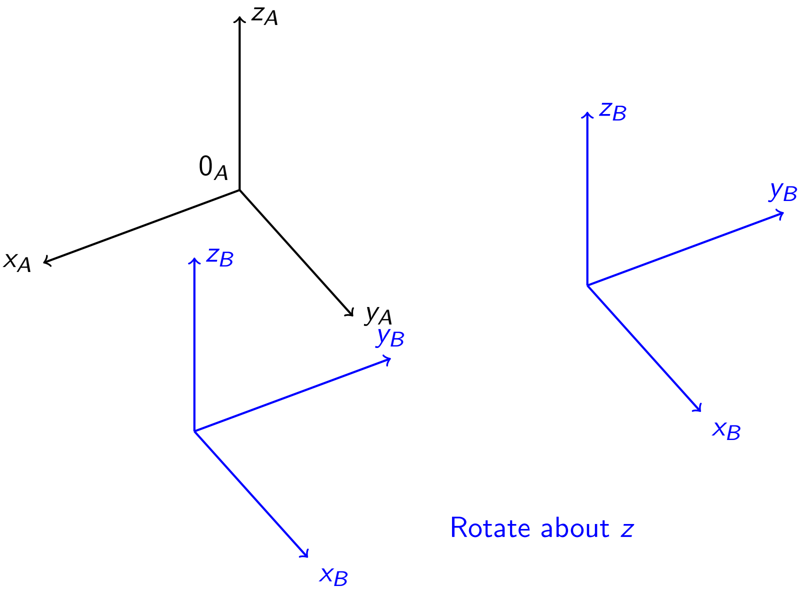
$$R^B = \left(R_C^A\right)^{-1} R^A R_C^A$$

in frame C.

Extrinsic vs Intrinsic Rotations

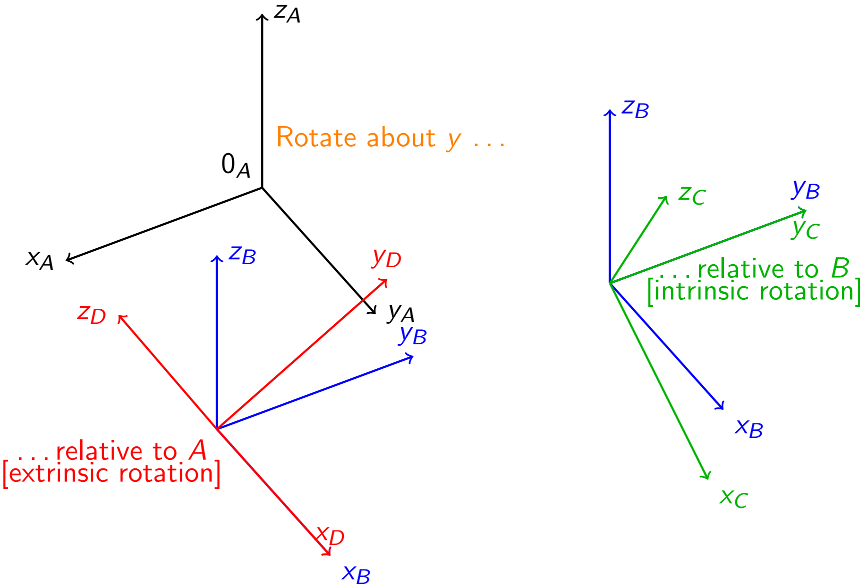


Extrinsic vs Intrinsic Rotations



Rotate about z

Extrinsic vs Intrinsic Rotations



Extrinsic vs Intrinsic Rotations

- A first rigid motion corresponding to rotation R_1 relative to a frame A produces frame B
- A second rigid motion rotation R_2 can be applied relative to either A or B.
- When applied relative to B, the second rotation is an intrinsic rotation. $R = R_1 R_2$.
- When applied relative to A, the second rotation is an extrinsic rotation. $R = R_2 R_1$.