Spatial Transforms

COMP 4766/6912: Autonomous Robotics

Andrew Vardy

Department of Computer Science
Memorial University of Newfoundland

June 4, 2018

Introduction

- Spatial transforms describe coordinate frames (a.k.a reference frames, coordinate systems, or often just *frames*)
- We have already seen two different reference frames:
  - Inertial (or global) reference: Defined by origin O and axes X_I and Y_I
  - Robot reference frame: Defined **w.r.t.** the inertial frame by origin P and axes X_R and Y_R
  - Both were specialized to motion in the plane, we need a more general representation for motion in 3-D
- We will look at transforms in two different ways: as mappings and as operators
- We will discuss two different orientation representations: rotation matrices and quaternions

Notation

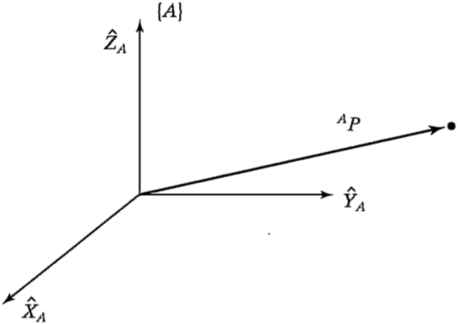
We use the notation from "Introduction to Robotics: Mechanics and Control", 3rd Edition by John J. Craig

Point P described in frame A is denoted as follows:

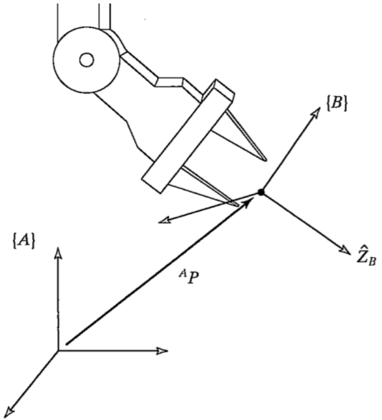
$$^{A}P = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

The A superscript appears to the left of the actual point and means that we are describing the point with respect to A. The same point described in frame B would be ^{B}P.

Frame A is described by three mutually orthogonal unit vectors $\hat{X}_A$, $\hat{Y}_A$, and $\hat{Z}_A$. The components of ${}^AP$ are the projections of the vector corresponding to this point onto these axes.



Lets now say that this point ${}^{A}P$ actually gives the origin of another frame called B. Perhaps this frame gives the position of a robot's end effector.



We can describe one frame w.r.t. to another by specifying the **rotation** and **translation** of the movement that separates them.

Rotation

The **rotation** of frame B w.r.t. frame A is defined through the mutually orthogonal unit vectors ${}^{A}\hat{X}_B$, ${}^{A}\hat{Y}_B$, and ${}^{A}\hat{Z}_B$. If we stack these three vectors as columns of a 3×3 matrix we get the rotation matrix that encodes the orientation of frame B w.r.t. frame A:

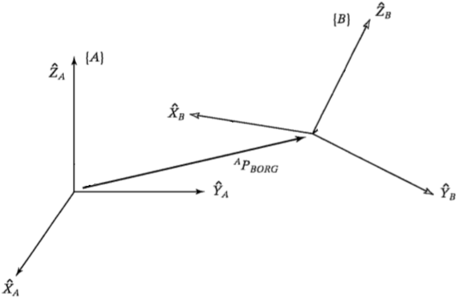
$$ {}^{A}_{B}R = \left[\begin{array}{ccc} {}^{A}\hat{X}_B & {}^{A}\hat{Y}_B & {}^{A}\hat{Z}_B \end{array} \right] = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} $$

For justification, see Craig's book, chapter 2. You see that the rotation matrix is special: it has mutually orthogonal columns. It turns out that the rows are mutually orthogonal too. Also, the inverse of the rotation is just equal to the transpose.

$$ {}^{A}_{B}R = {}^{B}_{A}R^{-1} = {}^{B}_{A}R^{T} $$

Translation

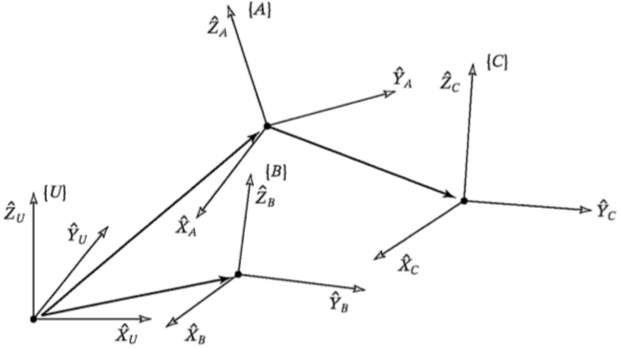
We describe the **translation** between frames by specifying the origin of B w.r.t. A as ${}^{A}P_{BORG}$.



We now have everything we need to describe frame B:

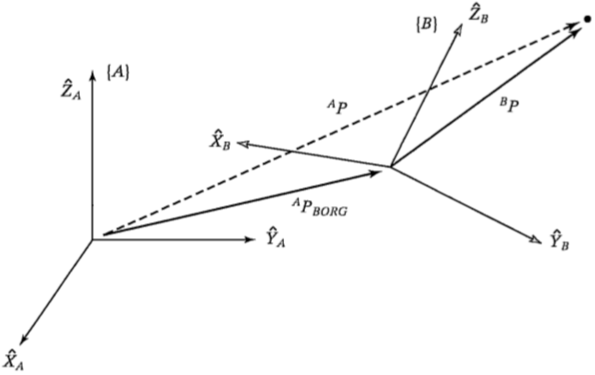
$$\{B\} = \left\{ {}^{A}_{B}R, {}^{A}P_{BORG} \right\}$$

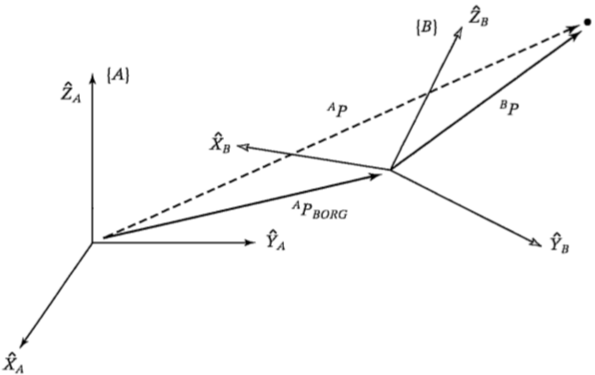
There is usually a universal (a.k.a. global or inertial) frame that others are defined w.r.t.. However, its often the case that we don't have a direct description of a frame with respect to the universal (e.g. for frame C below).



We would like a way of mapping descriptions of points from one frame to another.

A mapping is a change in our description of the same entity from one frame to another. Lets say we know the position of a point w.r.t. B but we want to compute it w.r.t. A. For example, we wish to find ${}^{A}P$ below:





If we have ${}^B P$ and our description of B we can get ${}^A P$.

$${}^A P = {}^A_B R \, {}^B P + {}^A P_{BORG}$$

Here again is this mapping:

$$^{A}P = {}^{A}_{B}R\,{}^{B}P + {}^{A}P_{BORG}$$

It would be convenient to combine both operations together into one big matrix so that we can do the following:

$$^{A}P = {}^{A}_{B}T\,{}^{B}P$$

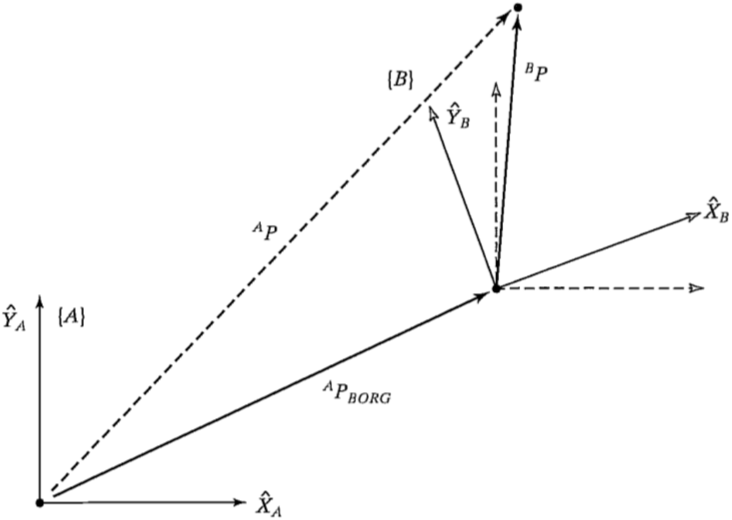
We can do this by moving to **homogeneous coordinates**. In homogeneous coordinates, we just augment our 3×1 vectors with an additional row that is always set to 1.

$$\begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix} = \left[\begin{array}{ccc|c} & {}^{A}_{B}R & & {}^{A}P_{BORG} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \begin{bmatrix} {}^{B}P \\ 1 \end{bmatrix}$$

This yields the equation at the top of this slide, as well as $1 = 1$ (uninformative, but reassuring). We can represent other sorts of transformations (e.g. shearing, scaling, perspective projection) by modifying particular entries in the augmented matrix.

2-D Example

Frame B lies at position (10, 5) w.r.t. frame A and is rotated by 30^o. Given ${}^BP = [3\ 7\ 0]^T$ find ${}^AP$.



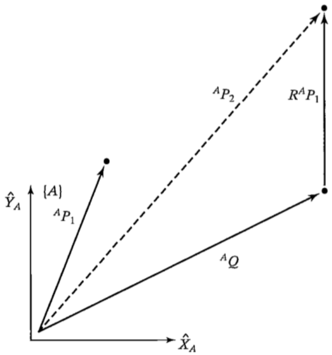
We get the following transform matrix:

$$
{}^A_B T = \begin{bmatrix} 0.866 & -0.500 & 0.000 & 10.0 \\ 0.500 & 0.866 & 0.000 & 5.0 \\ 0.000 & 0.000 & 1.000 & 0.0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

We now apply the transform (notice how the B's "cancel"):

$$
\begin{aligned}
{}^A P &= {}^A_B T \, {}^B P \\
&= {}^A_B T \begin{bmatrix} 3 \\ 7 \\ 0 \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} 9.098 \\ 12.562 \\ 0 \\ 1 \end{bmatrix}
\end{aligned}
$$

We have used transforms to relate the mapping of a point from one frame to another. We can also view a transform as an **operator** which translates and/or rotates points. For example, lets say we have a point ${}^AP_1 = [3\ 7\ 0]^T$ and we wish to rotate it by 30^o about $\hat{Z}$ and translate it by $[10\ 5\ 0]^T$. Now we wish to find the transformed point ${}^AP_2$, pictured below:

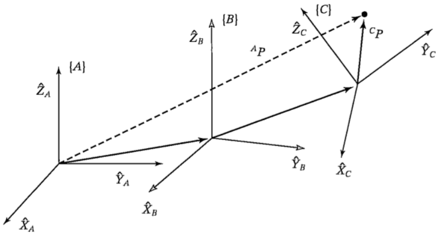


This example is identical to the previous one, except that we view this as an operator, not as a mapping. In this case, there is only one frame, A. The transformation matrix is the same as before, although we have no preceding super- or subscripts because the operation takes place within the same frame.

$$T = \begin{bmatrix} 0.866 & -0.500 & 0.000 & 10.0 \\ 0.500 & 0.866 & 0.000 & 5.0 \\ 0.000 & 0.000 & 1.000 & 0.0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{aligned} {}^{A}P_2 &= T\,{}^{A}P_1 \\ &= T \begin{bmatrix} 3 \\ 7 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 9.098 \\ 12.562 \\ 0 \\ 1 \end{bmatrix} \end{aligned}$$

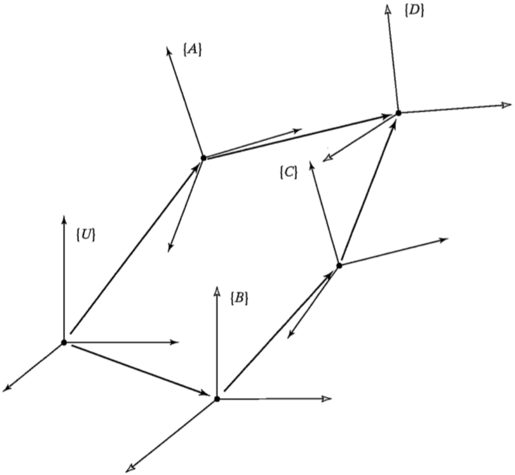
Compound Transforms



Considering the above with ${}^{C}P$ given, we can express this point in frames B and then A,

$$\begin{aligned}
{}^{B}P &= \left({}^{B}_{C}T\right){}^{C}P \\
{}^{A}P &= \left({}^{A}_{B}T\right){}^{B}P \\
&= \left({}^{A}_{B}T\right)\left({}^{B}_{C}T\right){}^{C}P
\end{aligned}$$

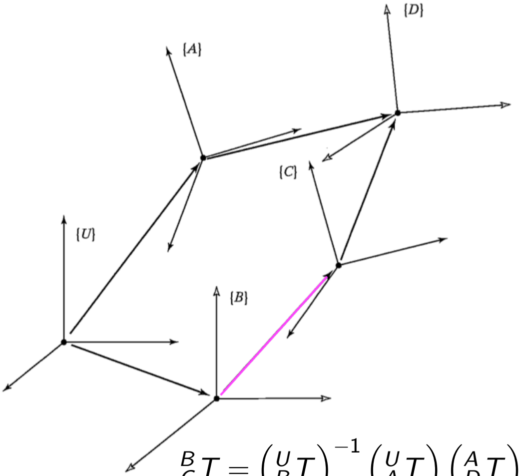
We can define a direct $C \rightarrow A$ transform: ${}^{A}_{C}T = \left({}^{A}_{B}T\right)\left({}^{B}_{C}T\right)$.



We can express the transform of D with respect to U in two different ways:

$$ {}^{U}_{D}T = \left({}^{U}_{A}T\right)\left({}^{A}_{D}T\right) \quad \text{and} \quad {}^{U}_{D}T = \left({}^{U}_{B}T\right)\left({}^{B}_{C}T\right)\left({}^{C}_{D}T\right) $$

The direction of the arrows indicates how existing transforms are defined. Lets say that all transforms but ${}^{B}_{C}T$ are known and we wish to solve for it.



$$ {}^{B}_{C}T = \left({}^{U}_{B}T\right)^{-1} \left({}^{U}_{A}T\right) \left({}^{A}_{D}T\right) \left({}^{C}_{D}T\right)^{-1} $$

Representing 3-D Orientation

Rotation matrices must be orthonormal, meaning that their rows and columns are orthogonal unit vectors. Even further, rotation matrices must have a determinant of +1. Cayley's formula for orthonormal matrices states that any such matrix can be rewritten as follows:

$$R = (I - S)^{-1}(I + S)$$

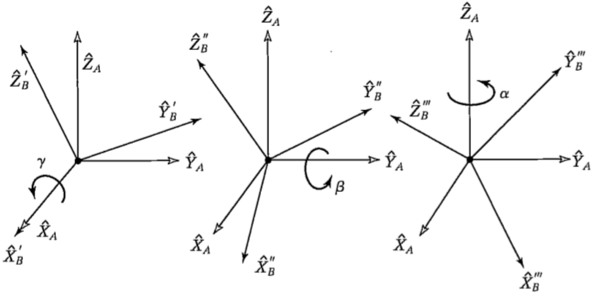
where

$$S = \begin{bmatrix} 0 & -s_z & s_y \\ s_z & 0 & -s_x \\ -s_y & s_x & 0 \end{bmatrix}$$

The important point is that a rotation matrix actually has only three free parameters. In fact any 3-D rotation can be represented by as little as three numbers. We will look at a couple of example representations...

X-Y-Z Fixed Angles

We can describe the rotation of a frame B with respect to a frame A as a sequence of rotations about $\hat{X}_A$, $\hat{Y}_A$, and $\hat{Z}_A$ (other orders beside X-Y-Z are possible). We are rotating about A which is fixed.

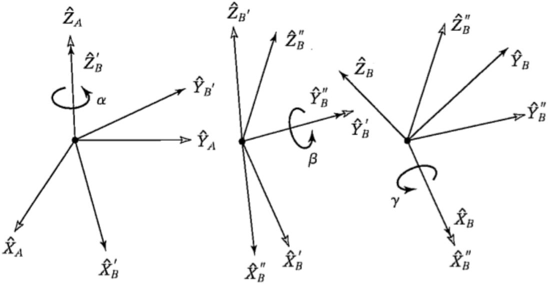


The rotation angles are γ, β, and α. We can generate a rotation matrix by combining all three:

$$
\begin{aligned}
{}^A_B R_{XYZ}(\gamma, \beta, \alpha) &= R_Z(\alpha) R_Y(\beta) R_X(\gamma) \\
&= \begin{bmatrix} c\alpha & -s\alpha & 0 \\ s\alpha & c\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c\beta & 0 & s\beta \\ 0 & 1 & 0 \\ -s\beta & 0 & c\beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\gamma & -s\gamma \\ 0 & s\gamma & c\gamma \end{bmatrix}
\end{aligned}
$$

Euler Angles

Euler angles are defined by rotating the "moving" frame, B, about its own axes. The following is for the order Z-Y-X:



We can view this as a series of rotations: $A \rightarrow B'$, $B' \rightarrow B''$, and $B'' \rightarrow B$, where B' and B'' are intermediary frames.

$$ {}^{A}_{B}R = \left({}^{A}_{B'}R \right) \left({}^{B'}_{B''}R \right) \left({}^{B''}_{B}R \right) $$

From this perspective the rotation by γ happens first, then β, then α. Interestingly, this is the same order and the same angles as for the X-Y-Z fixed angle representation—the two are equivalent!

Various Conventions

- So the X-Y-Z fixed angle convention is equivalent to the Z-Y-X Euler angle convention. There are 6 equivalent pairs of conventions.
- These different conventions were popular in the past but they all have an issue:
    - Gimbal lock: In certain configurations one of the three degrees-of-freedom is lost. e.g. in a plane pitched upwards, the roll and yaw rotations yield the same motion.
- **Unit quaternions** have emerged as another way of representing rotation:
    - They do not exhibit gimbal lock
    - They are more efficient computationally

Unit Quaternions and the Axis-Angle Representation

- SEE SEPARATE NOTES ON QUATERNIONS
- Note that the relationship between rotation and unit quaternions comes through the **axis-angle** representation of rotation:
  - The rotation is represented via a single axis of rotation specified by a unit vector $\hat{N}$ (2 free parameters) and an angle θ.
- Unit quaternions are related to these quantities as follows:

$$q = \cos\frac{\theta}{2} + \sin\frac{\theta}{2}\hat{N}$$