

Robotics: Principles and Practice

Module 4: Robot Manipulators

Lecture 3: Robot programming by frame-based task specification

David Vernon
Carnegie Mellon University Africa

www.vernon.eu

Robot Programming by Task Specification

By defining a series of manipulator end-effector positions M_n , a task can be described as a sequence of manipulator movements to these defined positions

For example, a task to pick and place an object might be formulated as follows

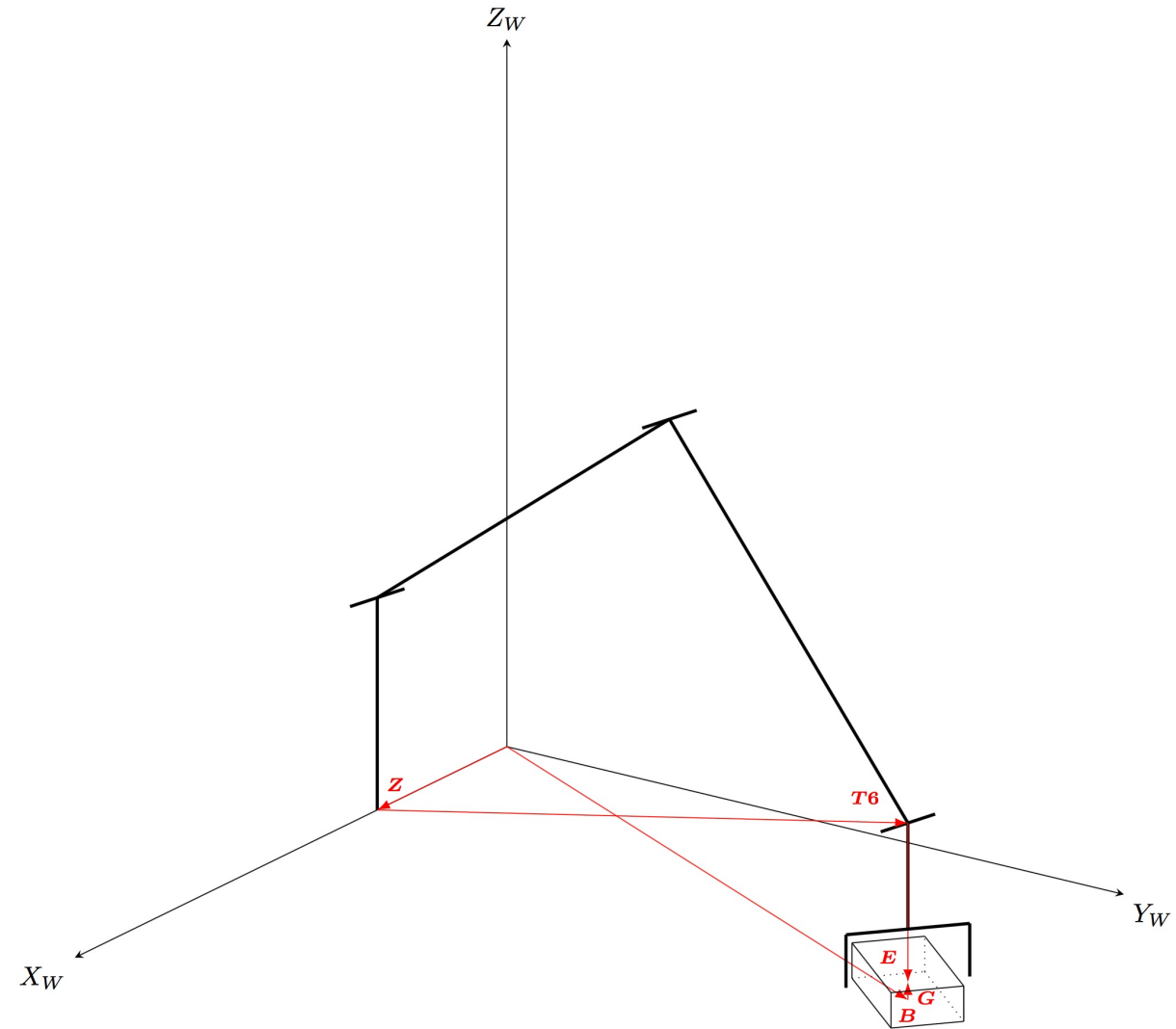
- M_0 : Move out of the field of view of the camera
 Determine the pose of a object and a suitable grasp point (possibly using a camera)
- M_1 : Move to an approach position close to the grasp point
- M_2 : Move to the grasp position
 Grasp the object
- M_3 : Move to the depart position above the grasp point
- M_4 : Move to the approach position in above the destination position
- M_5 : Move to the destination position
 Release the object
- M_6 : Move to the depart position away from the destination position

- We are specifying the task in terms of **movements of the robot** but the object are what we are really interested in
- The object movements are implicit in the fact that the manipulator has grasped it
- We make up for this when we describe the structure of the task by considering the structure of the task's component objects:
 - the manipulator
 - the end-effector
 - the object being manipulated
 - the object grasp pose
- **In particular, we will use the explicit positional relationships between these objects to describe the task structure**

Since coordinate frames can be used to describe object position and orientation ...

And since we may need to describe a coordinate frame in two or more ways (there is more than one way to reach any given position and orientation) ...

We use transform equations to relate the two descriptions



A manipulator grasping a block

- $\mathbf{Z}$ is the transformation (frame) which describes the position of manipulator with respect to the base co-ordinate reference frame
- ${}^{\mathbf{Z}}\mathbf{T}_6$ describes the end of the manipulator (*i.e.* the wrist) with respect to the base of manipulator, *i.e.* with respect to $\mathbf{Z}$
- ${}^{\mathbf{T}_6}\mathbf{E}$ describes the end-effector with respect to the end of the manipulator, *i.e.* with respect to $\mathbf{T}_6$
- $\mathbf{B}$ describes a block's position with respect to the base coordinate reference frame
- ${}^{\mathbf{B}}\mathbf{G}$ describes the **manipulator end-effector with respect to the block**, *i.e.* with respect to $\mathbf{B}$.

In this example, the end-effector is described in two ways, by the transformations leading from the base to the wrist to the end-effector :

$$\mathbf{Z} * \mathbf{Z} \mathbf{T6} * \mathbf{T6} \mathbf{E}$$

and by the transformations leading from the block to the end-effector grip position:

$$\mathbf{B} * \mathbf{B} \mathbf{G}$$

Equating these descriptions, we get the following transformation equation:

$$\mathbf{Z}^Z \mathbf{T}_6^{T_6} \mathbf{E} = \mathbf{B}^B \mathbf{G}$$

Alternatively, including the explicit composition operator in Corke (2016)

$$\mathbf{Z} \oplus^Z \mathbf{T}_6 \oplus^{T_6} \mathbf{E} = \mathbf{B} \oplus^B \mathbf{G}$$

- Solving for $\mathbf{T}_6$ by multiplying across by the inverse of $\mathbf{Z}$ and ${}^{T_6}\mathbf{E}$

$${}^Z\mathbf{T}_6 = \mathbf{Z}^{-1} \mathbf{B} {}^B\mathbf{G} {}^{T_6}\mathbf{E}^{-1}$$

- $\mathbf{T}_6$ is a function of the joint variables of the manipulator and, if known, then the appropriate joint variables can be computed using the **inverse kinematic solution**

- T_6 then is the coordinate frame which we wish to program in order to effect the manipulation task
- An arm position and orientation specified by T_6 is, thus, equivalent to our previous informal movement M_n

$$\text{Move } M_n = \text{Move } {}^Z T_6$$

- since we can compute T_6 in terms of our known frame we now have an arm movement which is specified in terms of the frames which describe the task structure

- Assigning the appropriate value to T_6 and moving to that position, implicitly using the inverse kinematic solution

$${}^Z T_6 = Z^1 B^B G^{T_6} E^1$$

Move ${}^Z T_6$

- What we have not yet done is to fully specify each of these frames by embedding them in the appropriate objects and specifying the transformations which define them

- Note that the position of the end-effector with respect to the base reference system is represented by

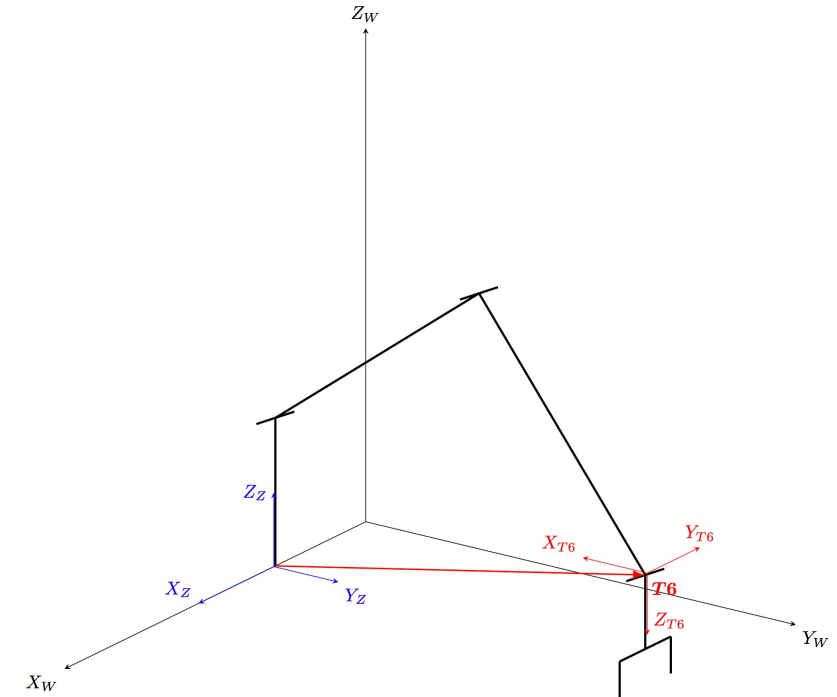
$$\mathbf{Z} \mathbf{T}_6^Z \mathbf{E}$$

- This allows you to generate general-purpose and reusable robot programs
- In particular, the calibration of the manipulator to the workstation is represented by $\mathbf{Z}$, while if the task is to be performed with a change of tool, only $\mathbf{E}$ need be altered

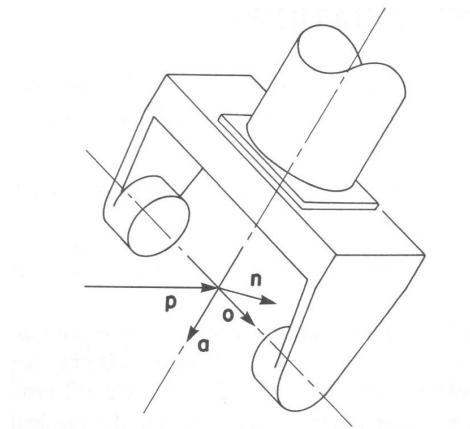
- As we have seen, we specify the orientation of T_6 by solving for it in terms of other frames/transformations in the task specification ...
- We do this by
 1. Embedding a frame in an object (or a desired point in space)
 2. Specifying the position of the origin of the frame by applying a translation
 3. Specifying the orientation of the frame by applying one or more rotations

There is a **convention** that the **T6** frame should be embedded in the manipulator

- with the origin at the wrist
- with the Z axis directed outward from the wrist to the gripper
- with the Y axis directed in the plane of movement of the gripper when it is opening and closing
- with the X axis making up a right-hand system



The same convention applies to the E frame that is embedded in a two-finger gripper [end-effector ... hence E]

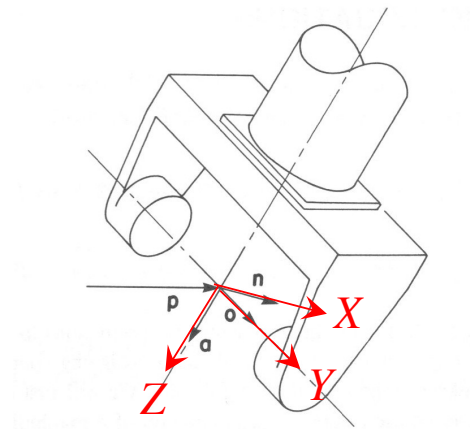


(Paul, 1981)

$$E = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

n Normal
 o Orientation
 a Approach

The same convention applies to the ***E*** frame that is embedded in a two-finger gripper [end-effector ... hence ***E***]



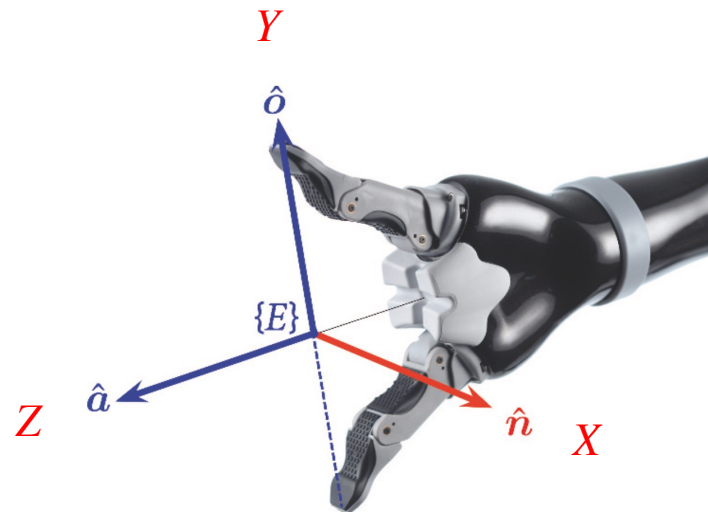
(Paul, 1981)

Direction of X axis
Direction of Y axis
Direction of Z axis

$$E = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

n Normal
o Orientation
a Approach

The same convention applies to the E frame that is embedded in a two-finger gripper [end-effector ... hence E]



(Corke, 2017), p. 41

Direction of X axis
 Direction of Y axis
 Direction of Z axis

$$E = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

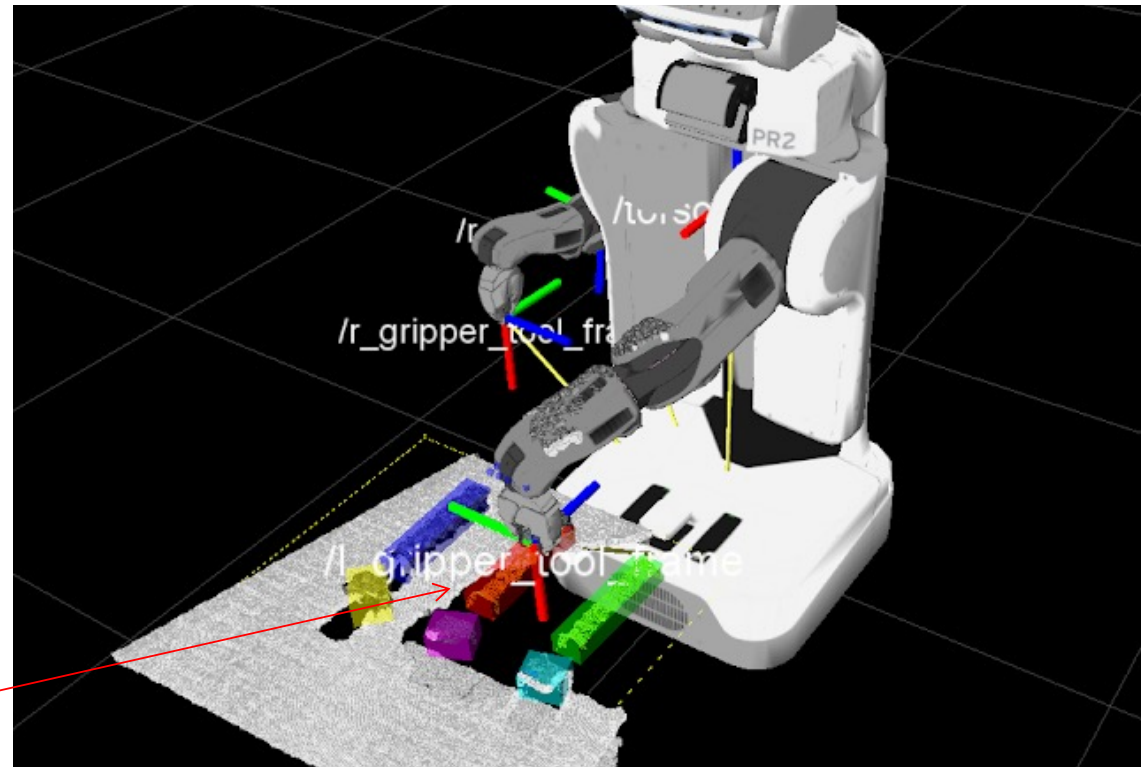
n Normal
 o Orientation
 a Approach

ROS uses a different convention

"If the end effector is a grasping device, the frame should be located at the recommended object grasping location. The frame orientation is defined as **X** the axis going 'toward' the object. **Y** the main dimension in which the grasping device moves and **Z** orthogonal to **X** and **Y** axes."

<https://www.ros.org/reps/rep-0120.html#l-gripper-and-r-gripper>

This approach is consistent with the convention of embedding a frame in a vehicle, with the *X* axis aligned with the direction of travel; see conventions on specifying orientation using roll, pitch, and yaw in the following slides.

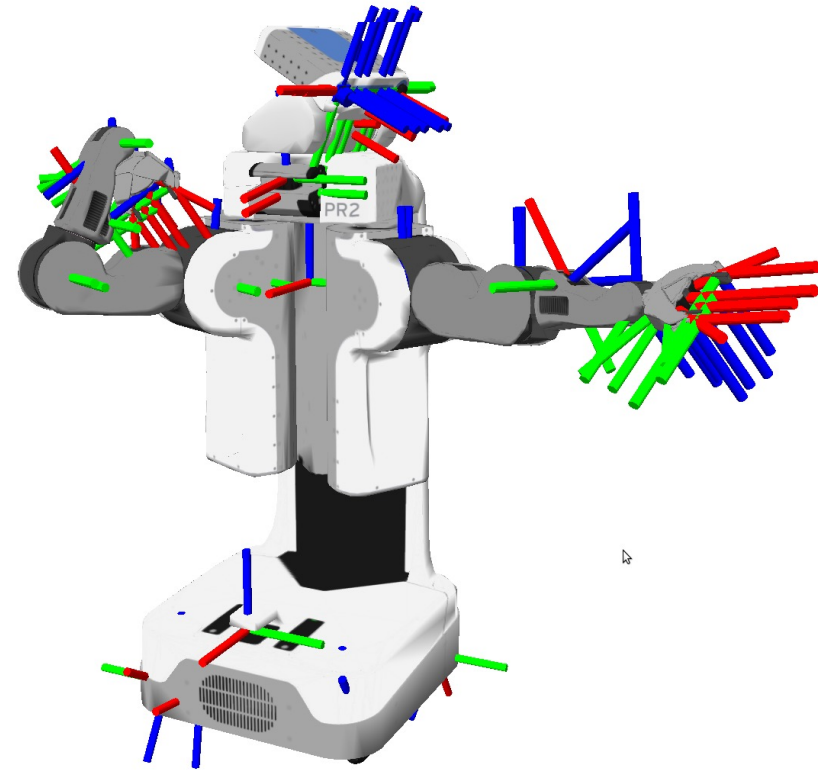


<https://alliance.seas.upenn.edu/~meam620/wiki/index.php?n=IanMcMahon2011.Final>

ROS uses a different convention

"If the end effector is a grasping device, the frame should be located at the recommended object grasping location. The frame orientation is defined as **X** the axis going 'toward' the object. **Y** the main dimension in which the grasping device moves and **Z** orthogonal to **X** and **Y** axes."

<https://www.ros.org/reps/rep-0120.html#l-gripper-and-r-gripper>

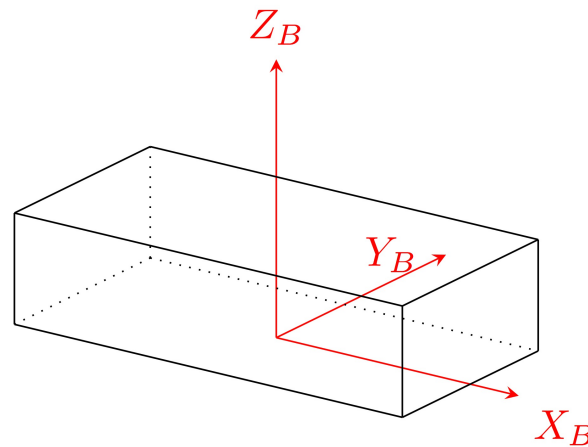


<http://library.isr.ist.utl.pt/docs/roswiki/tf2.html>

Specifying Pose

We have seen that the pose of an object can be specified by embedding a frame in the object in some appropriate manner ... **for example**:

- Placing the origin at the centre of the object
- Aligning the axes with the major and minor axes of the object

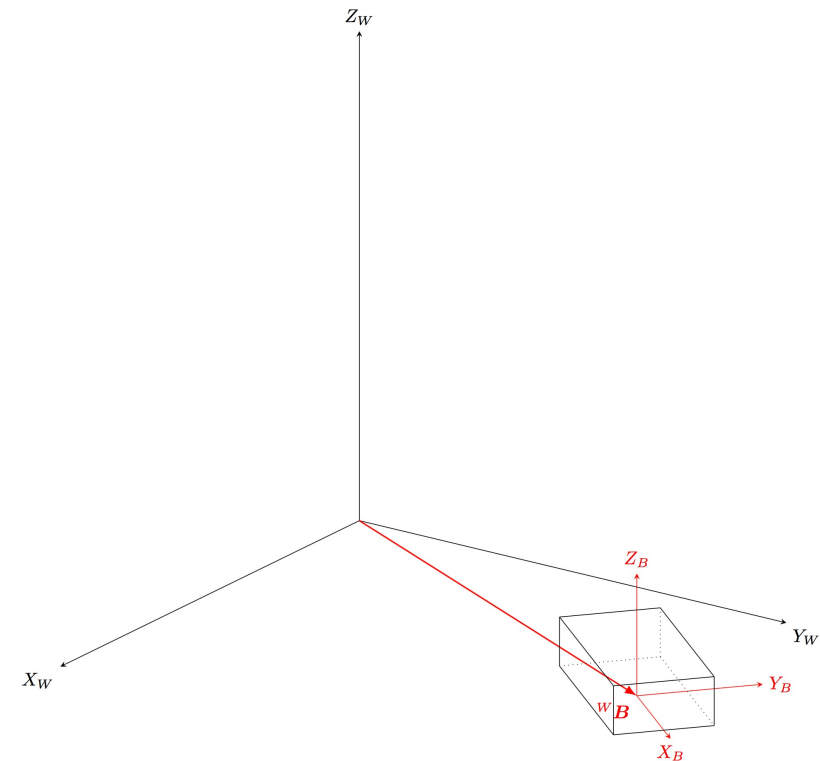


Specifying Pose

Then applying a homogenous transformation, e.g. $\mathbf{B} = \text{Trans}(10, 20, 0) \text{Rot}(Z, 50)$

- Translation part
 - Possibly several translations, applied in turn
- Rotation part
 - Possibly several rotations, applied in turn

You can specify them in whatever order you like,
yielding a valid transform equation such as
 $\mathbf{B} = \text{Trans}(10, 20, 0) \text{Rot}(Z, 50) \text{Rot}(X, 10) \text{Rot}(Z, 30)$



Specifying Orientation

That said, there are several conventions for the way these rotations are specified

One is **Roll-Pitch-Yaw (RPY)** ... sometimes referred to as Cardan angles

RPY can be confusing. There are two reasons.

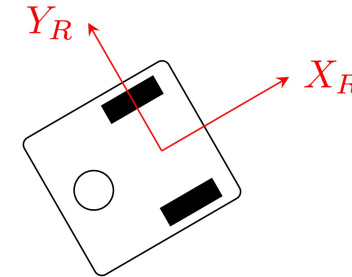
1. There are two conventions, each specifying a different sequence of axes about which to rotate:
 - $Z Y X$ normally used with vehicles
 - $X Y Z$ normally used with end-effectors
2. The angles are specified in the order **yaw**, **pitch**, **roll** (despite the name roll-pitch-yaw)

Specifying Orientation

That said, there are several conventions for the way these rotations are specified

Roll-Pitch-Yaw (RPY) with **vehicles** $Z\ Y\ X$

- The frame embedded in a vehicle normally has
 - X axis in the direction of travel
 - Z axis directly up
 - Y axis specified a right-hand system
- The orientation is specified by $\mathbf{RPY}(\theta_y, \theta_p, \theta_r) = \mathbf{Rot}(Z, \theta_y) \mathbf{Rot}(Y, \theta_p) \mathbf{Rot}(X, \theta_r)$
 - First, rotate the yaw angle θ_y about the Z axis (i.e. about the vertical, thus specifying the direction of travel)
 - Second, rotate the pitch angle θ_p about the Y axis (thus specifying the angle of ascent or descent)
 - Third, rotate the roll angle θ_r about the X axis (thus specifying the banking angle)

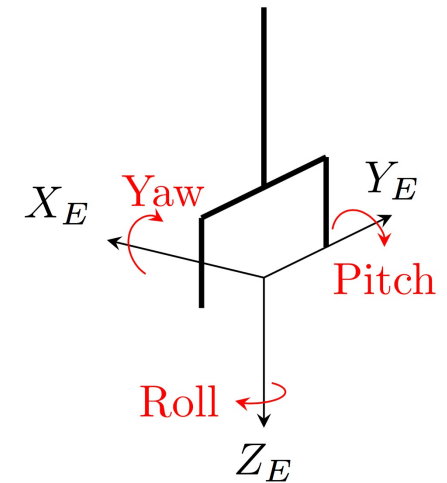


Specifying Orientation

That said, there are several conventions for the way these rotations are specified

Roll-Pitch-Yaw (RPY) with end-effectors $X Y Z$

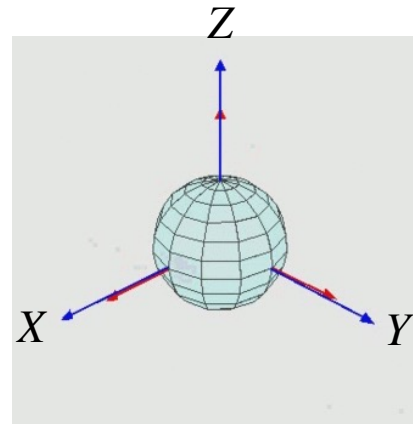
- The frame embedded in an end-effector or two-finger gripper normally has
 - X axis in the **normal** direction [i.e. normal to the movement of the fingers]
 - Z axis directed in the **approach** direction
 - Y axis direction in the orientation direction [i.e. parallel to the movement of the fingers]
- The orientation is specified by $\mathbf{RPY}(\theta_y, \theta_p, \theta_r) = \mathbf{Rot}(X, \theta_y) \mathbf{Rot}(Y, \theta_p) \mathbf{Rot}(Z, \theta_r)$
 - First, rotate the yaw angle θ_y about the X axis [i.e. about the normal]
 - Second, rotate the pitch angle θ_p about the Y axis [about the orientation]
 - Third, rotate the roll angle θ_r about the Z axis [about the approach]



Specifying Orientation

Euler Angles

- There are other commonly-used **conventions** for specifying the orientation of objects/frames
 - For example: **Euler angles** (e.g. rotation about Z, X, Z axes, in that order)



https://en.wikipedia.org/wiki/Euler_angles

- Note that there are twelve Euler angle conventions; this is just one of them
- We also use **quaternions**, especially in ROS

Recommended Reading

D. Vernon, Machine Vision – Automated Visual Inspection and Robot Vision, Prentice Hall International, 1991. Chapter 8.

http://vernon.eu/publications/91_Vernon_Machine_Vision.pdf

Similar material to that presented in this lecture.

R. P. Paul, Robot Manipulators – Mathematics, Programming, and Control, MIT Press, 1981. Chapter 1.

https://books.google.rw/books?id=UzZ3LAYqvRkC&printsec=frontcover&source=gbs_ViewAPI&redir_esc=y#v=onepage&q&f=false

Similar material to that presented in this lecture but complete comprehensive treatment.

P. Corke, Robotics, Vision and Control, 2nd Edition, Springer, 2017.

Comprehensive contemporary treatment; highly recommended.