

Introduction to Cognitive Robotics

Module 4: Robot Manipulators

Lecture 1: Robot programming; coordinate frames of reference and homogenous transformations

David Vernon
Carnegie Mellon University Africa

www.vernon.eu

A Brief Review of Robot Programming

There are, broadly speaking, three main categories of robot programming system which are, in order of the level of sophistication

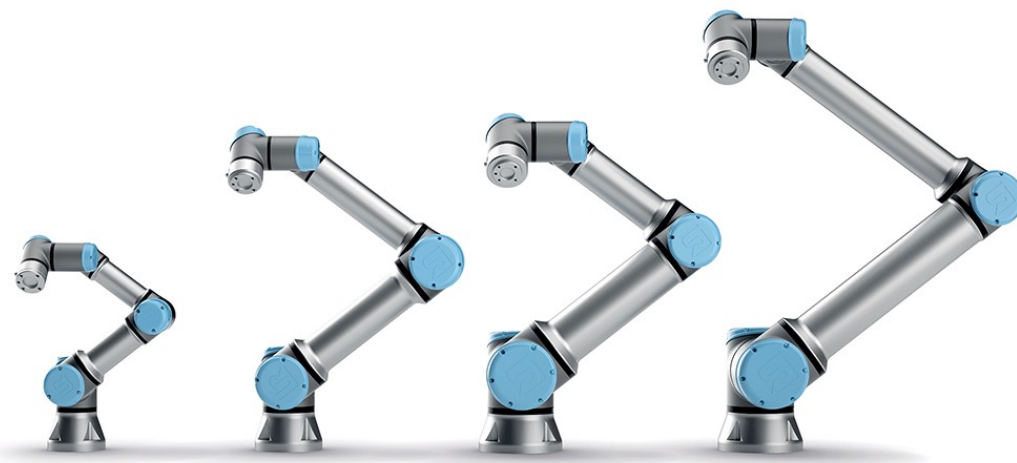
- Guiding Systems
- Robot-Level or Explicit-Level Systems and
- Task Level Systems

- Guiding systems are typified by the manual lead-through approach in which the manipulator is trained by guiding the arm through the appropriate positions using, for example, a **teach-pendant** and recording the individual joint positions
- Task execution is effected by driving the joints to these recorded positions
- This type of manual teaching is the most common of all programming systems



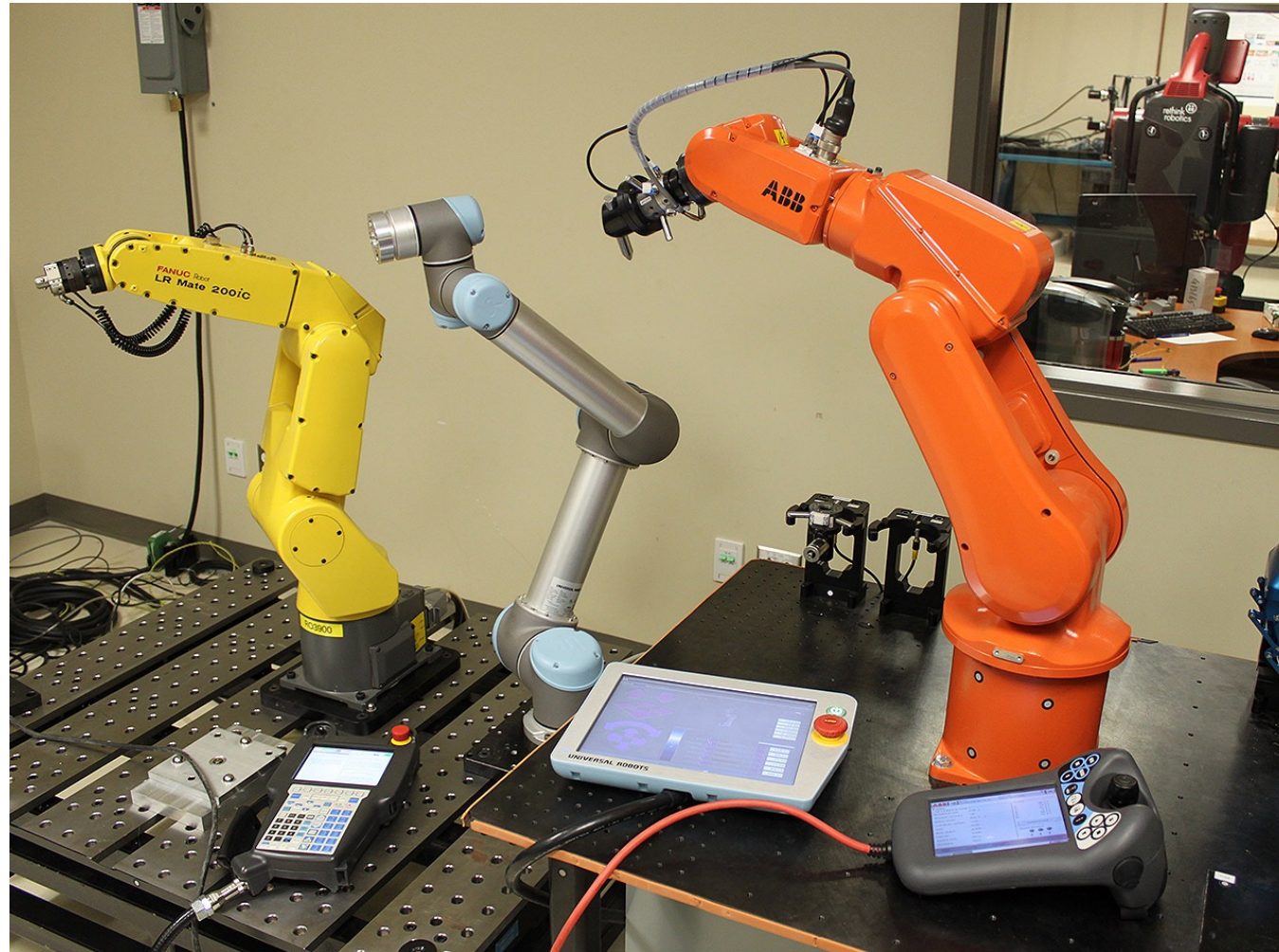
KUKA LBR iiwa

<https://robots.ieee.org/robots/lbriiwa>



Universal Arms

<https://robots.ieee.org/robots/lbriiwa>



FANUC LR Mate 200iC, Universal Robots UR5, ABB IRB 120

<https://robohub.org/what-is-so-special-about-the-robot-arms-of-universal-robots/>

Actively-compliant arms:
They move in response to an externally applied force

This allows the operator to guide the robot by physically
placing the arm at the required positions and orientations

Sometimes referred to as a co-bot:
a robot that is safe to work with in close proximity




Baxter

Baxter is a versatile manufacturing robot. Its cameras and force-sensing actuators let it adapt to changes in the environment, and a user can program a new task simply by moving its arms around.

CREATOR

Rethink Robotics [↗](#)

COUNTRY

United States 

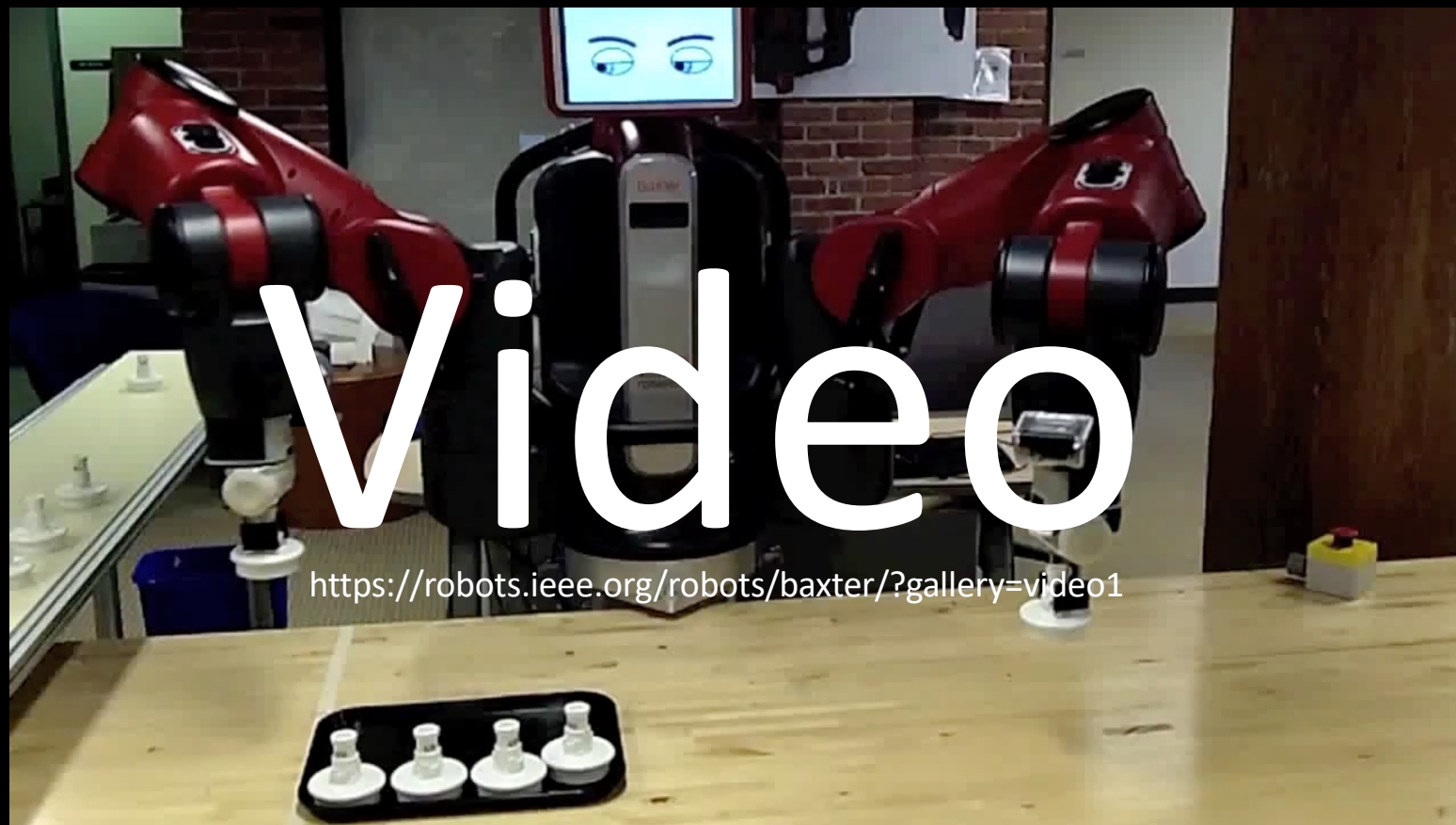
YEAR

2012

TYPE

Industrial

Source: <https://robots.ieee.org/robots/baxter/>



<https://robots.ieee.org/robots/baxter/?gallery=video1>

Source: <https://robots.ieee.org/robots/baxter/>

Robot-level programming systems, for the most part, simply replace the teach pendant with a robot programming language

Manipulator movements are still programmed by explicitly specifying joint positions

However, several languages also facilitate robot control in a **three dimensional Cartesian space**, rather than in the joint space

- **(Forward) Kinematic Solution** of the manipulator arm

Allows you to **compute the pose (position and orientation) of the end-effector** in a 3D Cartesian frame of reference, given the manipulator joint positions

- **Inverse Kinematic Solution**

Allows you to **compute the joint positions** for a given position and orientation of the end-effector

- The more advanced of these languages incorporate structured programming control constructs.
- They make extensive use of **coordinate transformations** and **coordinate frames**

With this approach


- The robot control is defined in terms of transformations on a coordinate frame (a set of XYZ axes) associated with, and embedded in, the robot hand
- Off-line programming is more feasible as long as the transformations representing the relationships between the frames describing the objects in the robot environment are accurate

Task-level robot programming languages attempt to describe assembly tasks as **sequences of goal spatial relationships between objects**

- they focus on the **objects** rather than on the **manipulator**
- the robot is merely a mechanism to achieve these goals
- they typically require the use of task planning, path planning, collision avoidance and world-modelling

Description of Object Pose with Homogeneous Transformations Vectors & Quaternions

Position and Orientation:
Six degrees of freedom



- Robot manipulation is concerned, in essence, with the spatial relationships between several objects, between objects and manipulators, and with the reorganization of these relationships
- We will use **homogeneous transformations** and (later in the course) **vectors & quaternions** to represent these spatial relationships
- We begin by introducing homogeneous transformations showing how they can be used to represent coordinate frames of reference

A 3D vector $\mathbf{v} = a\hat{\mathbf{x}} + b\hat{\mathbf{y}} + c\hat{\mathbf{z}}$, where $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, and $\hat{\mathbf{z}}$ are unit vectors along the X , Y , and Z axes are represented in homogeneous coordinates as

Note the use of the tilde to denote a homogeneous representation of a vector

$$\tilde{\mathbf{v}} = \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$$

Scaling factor, thus a single 3D vector can be represented by several homogeneous coordinates

where $a = \frac{x}{w}$, $b = \frac{y}{w}$, and $c = \frac{z}{w}$

$$a\mathbf{i} + b\mathbf{j} + c\mathbf{k}$$

$$a\hat{\mathbf{i}} + b\hat{\mathbf{j}} + c\hat{\mathbf{k}}$$

$$a\hat{\mathbf{x}} + b\hat{\mathbf{y}} + c\hat{\mathbf{z}}$$

$$a\hat{x} + b\hat{y} + c\hat{z}$$

There are several conventions for unit vectors.

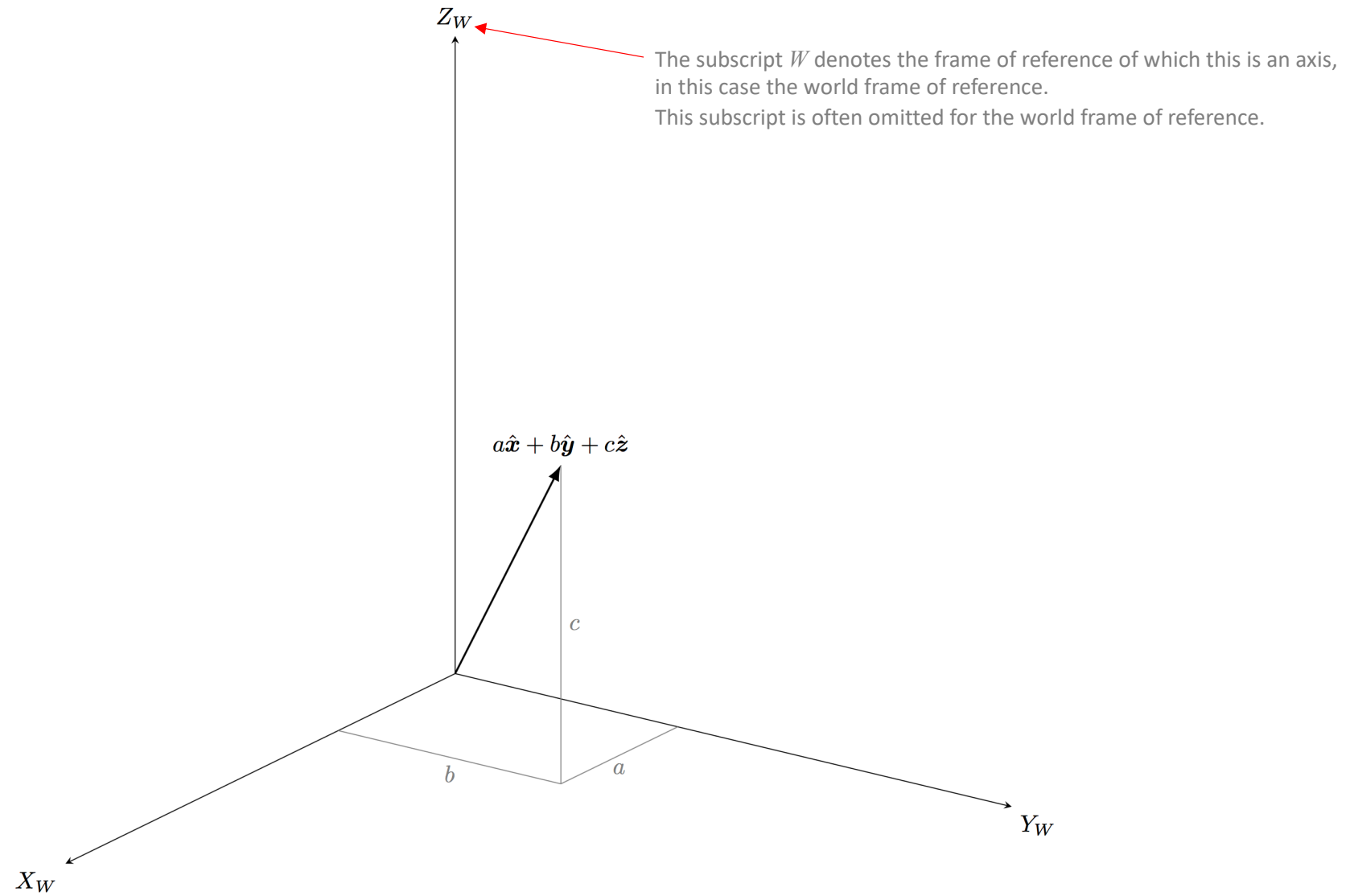
Here, we use the convention adopted in

P. Corke, Robotics, Vision and Control, 2nd Edition, Springer, 2016.

A 3D vector $\mathbf{v} = a\hat{\mathbf{x}} + b\hat{\mathbf{y}} + c\hat{\mathbf{z}}$, where $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, and $\hat{\mathbf{z}}$ are unit vectors along the X , Y , and Z axes are represented in homogeneous coordinates as

$$\tilde{\mathbf{v}} = \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$$

where $a = \frac{x}{w}$, $b = \frac{y}{w}$, and $c = \frac{z}{w}$.



For example, $\boldsymbol{v} = 3\hat{\boldsymbol{x}} + 4\hat{\boldsymbol{y}} + 5\hat{\boldsymbol{z}}$ can be represented by $\begin{bmatrix} 3 \\ 4 \\ 5 \\ 1 \end{bmatrix}$ or $\begin{bmatrix} 6 \\ 8 \\ 10 \\ 2 \end{bmatrix}$

Since division by zero is indeterminate, the vector $\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ is undefined

A general transformation \mathbf{H} , in 3D space, representing translation, rotation, stretching and perspective distortions, is a 4×4 matrix in homogeneous formulation

Given a point represented by the vector $\tilde{\mathbf{u}}$, its transformation $\tilde{\mathbf{v}}$ is represented by the matrix product

$$\tilde{\mathbf{v}} = \mathbf{H}\tilde{\mathbf{u}}$$

The transformation H corresponding to a translation by a vector $\begin{bmatrix} a \\ b \\ c \end{bmatrix}$ is

$$\mathbf{H} = \mathbf{Trans}(a, b, c) = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For example : to transform $\tilde{\mathbf{u}} = \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$ by \mathbf{H}

$$\tilde{\mathbf{v}} = \mathbf{H}\tilde{\mathbf{u}} = \mathbf{Trans}(a, b, c)\tilde{\mathbf{u}} = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = \begin{bmatrix} x + aw \\ y + bw \\ z + cw \\ w \end{bmatrix} = \begin{bmatrix} x/w + a \\ y/w + b \\ z/w + c \\ 1 \end{bmatrix}$$

The transformations corresponding to rotations about X , Y and Z axes by an angle θ are:

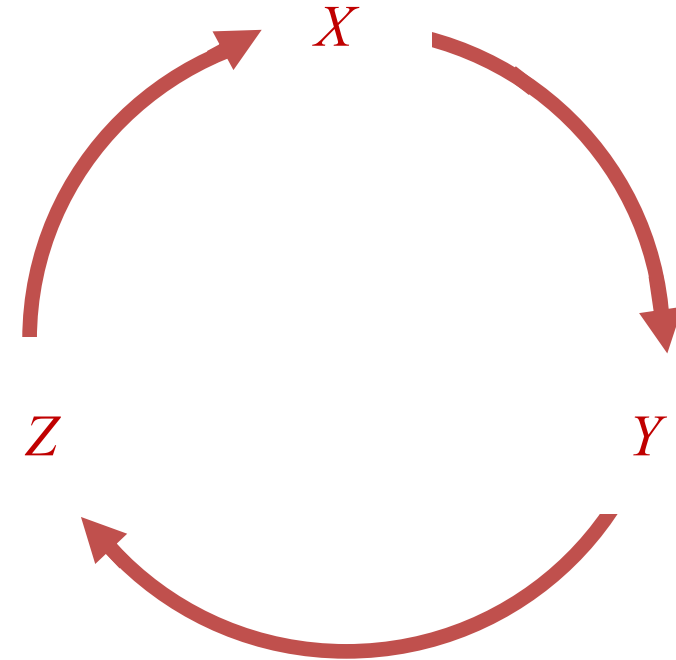
$$\mathbf{Rot}(X, \theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta & 0 \\ 0 & \sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{Rot}(Y, \theta) = \begin{bmatrix} \cos \theta & 0 & \sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{Rot}(Z, \theta) = \begin{bmatrix} \cos \theta & \sin \theta & 0 & 0 \\ -\sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Remember when deciding in which sense to make a rotation that:

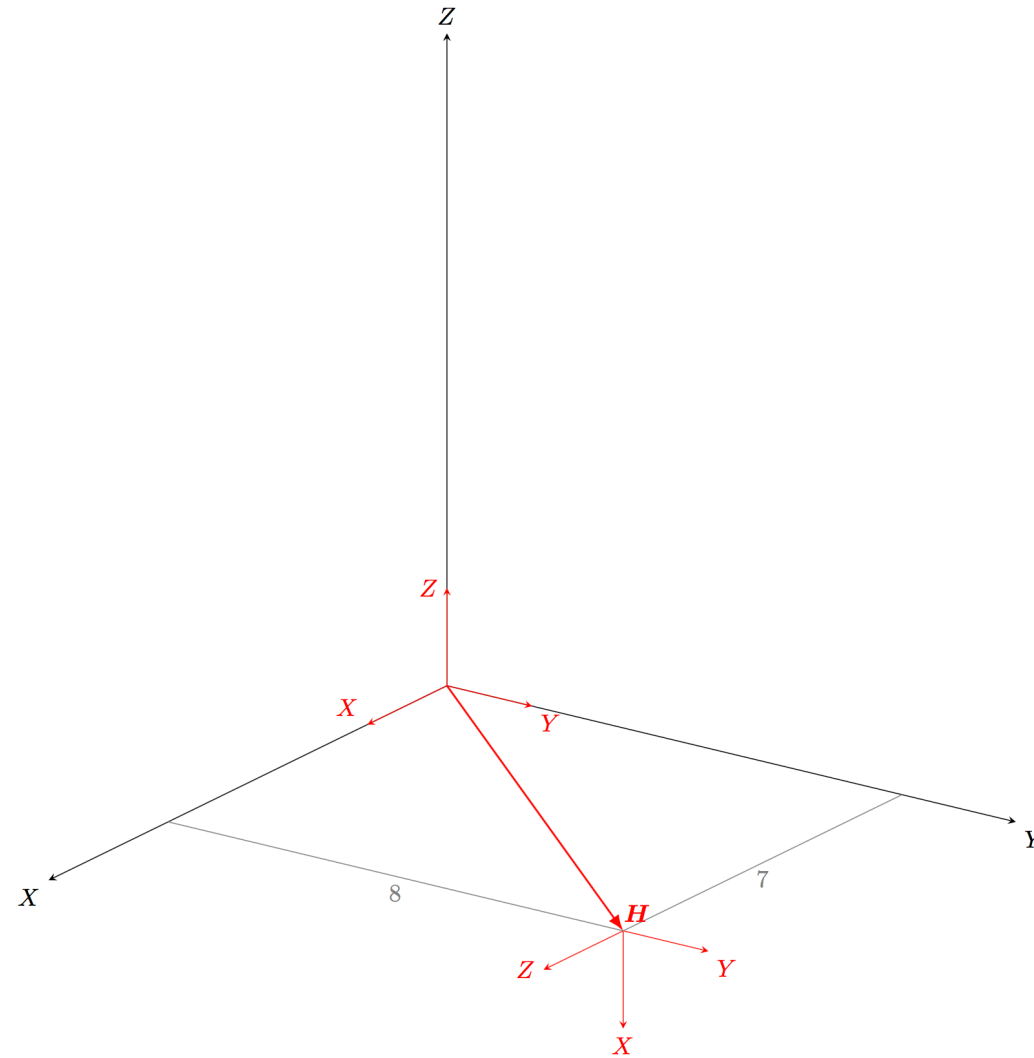
- a positive rotation about the X axis takes the Y axis **toward** the Z axis
- a positive rotation about the Y axis takes the Z axis **toward** the X axis
- a positive rotation about the Z axis takes the X axis **toward** the Y axis



We can interpret the homogeneous transformation as a **coordinate reference frame**

In particular, a homogeneous transformation describes the **position and orientation of a coordinate frame with respect to another previously defined coordinate frame**

Thus, the homogeneous transformation **represents**, not only transformations of vectors (points), but also **positions and orientations**



$$H = \text{Trans}(7, 8, 0) \text{Rot}(Y, 90)$$

Interpreting a homogeneous transformation as a coordinate frame

Specifically, a coordinate frame is defined by **four** things: the position of its **origin** and the **direction** of its **X** , **Y** and **Z** axes.

- the **first three columns** of the homogeneous transformation represent the **direction** of the **X** , **Y** and **Z** axes of the coordinate frame with respect to the base coordinate reference frame
- the fourth column represents the position of the origin

- A homogeneous transformation, which can be a combination of many simpler homogeneous transformations, **applies** equally to **other homogeneous transformations** as it does to vectors
- Thus, we can **take a coordinate reference frame** and **move it** elsewhere **by applying** an appropriate homogeneous transformation

If the coordinate frame to be “moved” is originally **aligned** with the so-called base coordinate reference frame

the homogeneous transformation is

- a description of **how to transform** the base coordinate frame to the new coordinate frame **and ...**
- **a description of this new coordinate frame** with respect to the base coordinate reference frame.

For example, consider the following transformation:

$$\begin{aligned} \mathbf{H} &= \mathbf{Trans}(7, 8, 0) \mathbf{Rot}(Y, 90) \\ &= \begin{bmatrix} 1 & 0 & 0 & 7 \\ 0 & 1 & 0 & 8 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 & 1 & 7 \\ 0 & 1 & 0 & 8 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

Let's apply this transformation to some special vectors

The transformation of the **null vector**, *i.e.* the vector which performs no translation and thus defines the origin of the base coordinate frame, is given by

$$\begin{bmatrix} 0 & 0 & 1 & 7 \\ 0 & 1 & 0 & 8 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 7 \\ 8 \\ 0 \\ 1 \end{bmatrix}$$

This is the origin of the new coordinate frame

The transformation of the three vectors corresponding to the **unit vectors** along the ***X***, ***Y*** and ***Z*** axes are

$$\begin{aligned}
 \begin{bmatrix} 0 & 0 & 1 & 7 \\ 0 & 1 & 0 & 8 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} &= \begin{bmatrix} 7 \\ 8 \\ -1 \\ 1 \end{bmatrix} \\
 \begin{bmatrix} 0 & 0 & 1 & 7 \\ 0 & 1 & 0 & 8 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} &= \begin{bmatrix} 7 \\ 9 \\ 0 \\ 1 \end{bmatrix} \\
 \begin{bmatrix} 0 & 0 & 1 & 7 \\ 0 & 1 & 0 & 8 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} &= \begin{bmatrix} 8 \\ 8 \\ 0 \\ 1 \end{bmatrix}
 \end{aligned}$$

The direction of these (transformed) unit vectors is formed by

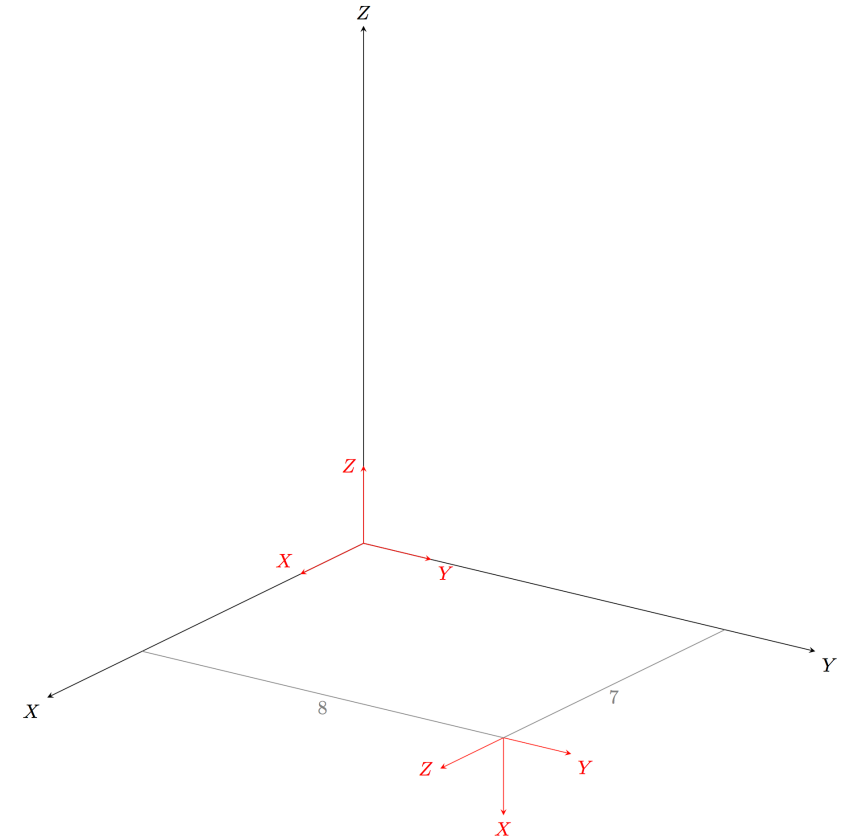
- subtracting the vector representing the origin of this coordinate frame
- extending the vectors to infinity by reducing the scale factor to zero

Thus, the direction of the X , Y and Z axes of this (new) frame are

$$\begin{bmatrix} 0 \\ 0 \\ -1 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \text{ and } \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

These four results show us that

- the new origin is at coordinates $[7, 8, 0]$
- the new X axis is directed along the Z axis of the base coordinate reference frame in the negative direction
- the new Y axis is directed along the Y axis of the base coordinate reference frame in the positive direction
- and the new Z axis is directed along the X axis of the base coordinate reference frame in the positive direction



The rotations and translations we have been describing have all been made relative to the fixed base frame of reference

Thus, in the transformation given by

$$\mathbf{H} = \mathbf{Trans}(7, 8, 0)\mathbf{Rot}(Y, 90)$$

1. First, the frame is first rotated by 90° around the Y axis of the base frame of reference
2. Then translated by $7\hat{x} + 8\hat{y} + 0\hat{z}$ in the base frame of reference

This operation may also be interpreted in reverse order, from **left to right**, viz.

1. First, the object (frame) is first translated by $7\hat{x} + 8\hat{y} + 0\hat{z}$
 2. Then rotated by 90° around the **station frame** Y axis (i.e. the translated frame of reference)
- This second interpretation is more intuitive since we can forget about the base reference frame and just remember “where we are”: **our current station coordinate reference frame**
 - We then just need to decide what transformations are necessary to **get us to where we want to be** based on the orientation of the **station axes**

In this way, we can get from pose to pose by incrementally identifying the appropriate station transformations,

$$H_1, H_2, H_3, \dots H_n$$

which we apply sequentially, as we go, and the final pose is defined with respect to the base simply as

$$H = H_1 H_2 H_3 \dots H_n$$



Sometimes we include an explicit composition operator, e.g. Corke (2017):

$$H = H_1 \oplus H_2 \oplus H_3 \oplus \dots H_n$$

In order to clarify the relative nature of these transformations

- Each of these frames/transformations is normally written with a leading superscript
- This superscript identifies the coordinate frame with respect to which the (new) frame/transformation is defined
- If the leading superscript is omitted, it is assumed to be the base (or world) frame

$$H = H_1^{H_1} H_2^{H_2} H_3 \dots H_{n-1} H_n \leftarrow \text{or } H = H_1 \oplus^{H_1} H_2 \oplus^{H_2} H_3 \oplus \dots \oplus^{H_{n-1}} H_n$$

As a general rule:


- If we **post-multiply** a transform representing a frame by a second transformation describing a rotation and/or translation we make that rotation/transformation **with respect to the frame axis described by the first transformation**
- If we **pre-multiply** the frame transformation representing a rotation/transformation then the rotation/transformation is made **with respect to the base reference coordinate frame**

In the preceding treatment

- The same symbol is used to represent the homogeneous transformation and the frame it is defining
- Using subscripts 1, 2, 3, ... to distinguish different frames/transformations

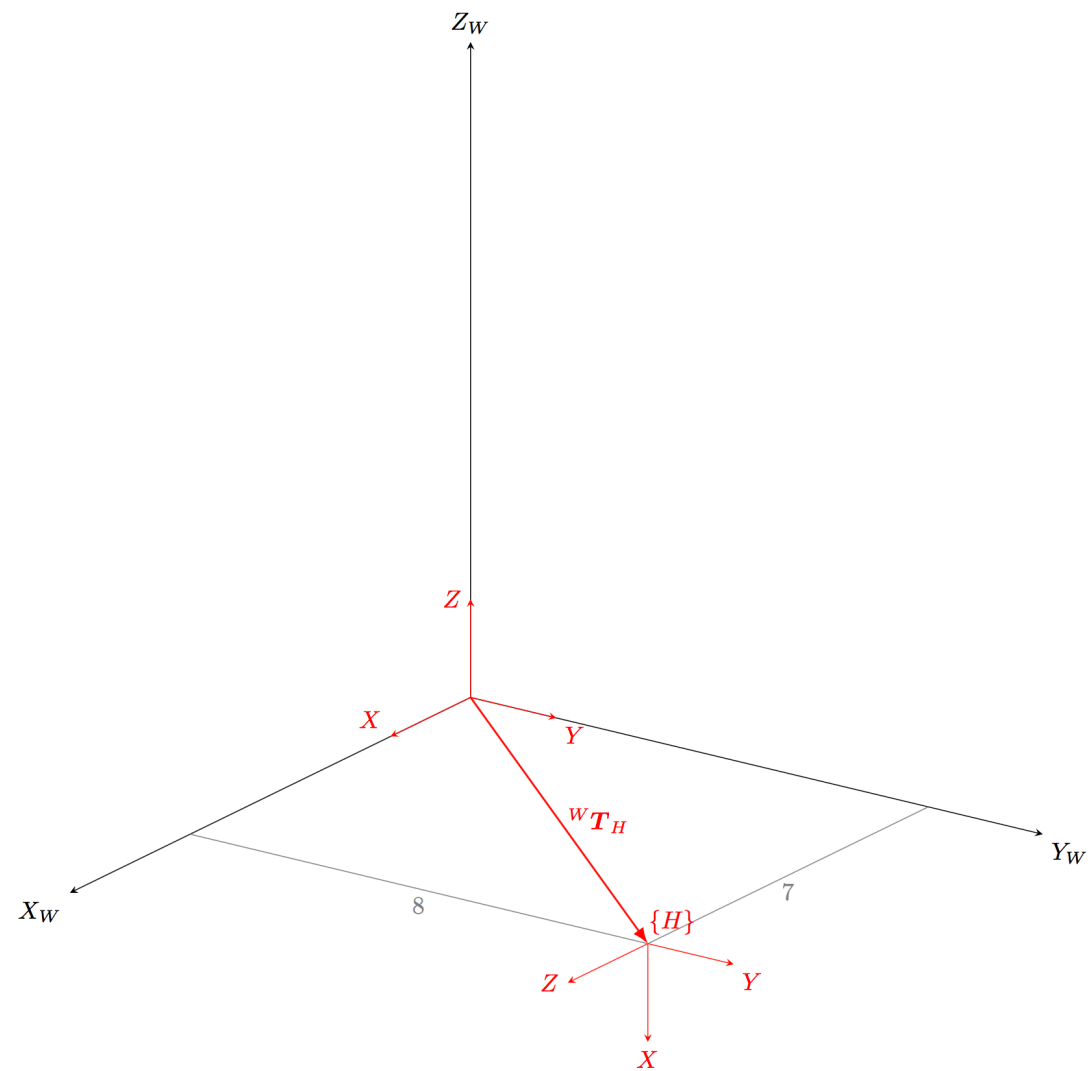
- There is a common **alternative convention** that uses the subscript to identify the frame that is being defined

Homogeneous transformation T representing frame $\{B\}$
with respect to frame $\{A\}$


$${}^A\mathbf{T}_B$$

- In ROS and CRAM documents, you will see this written A_T_B (or a_T_b or aTb)
- The frames are also often written in curly braces $\{A\}$ $\{B\}$

- Note that
- $${}^A\mathbf{T}_B = \left({}^B\mathbf{T}_A\right)^{-1}$$



$${}^wT_H = \text{Trans}(7, 8, 0)\text{Rot}(Y, 90)$$

Interpreting a homogeneous transformation as a coordinate frame

At this stage, we have developed a system where we can

specify the position and orientation of coordinate reference frames anywhere

with respect to each other

w.r.t. station frame of reference



or

with respect to a given base frame

w.r.t. fixed world frame of reference



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.

