3D Geometry
with Applications to Computer Games

Prof. Dr. Udo Frese

Frames and Transformations

3D Geometry
with Applications to Computer Games

3rd semester course for
Computer Science, Digital Media, Systems Engineering

Competency Goals

- **ability to model geometric situations using homogenous coordinates, transformations and geometric primitives**
- **ability to model motion and behavior as operations on homogenous coordinates, transformations and geometric primitives**
- **ability for creative implementation of own ideas in the computer using 3D geometry**

1	**vectors and homogenous coordinates; lines and planes**
2	**linear functions and matrices**
3	**frames and transformations**

[1] Christopher Tremblay: **Mathematics for Game Developers**, Course Technology PTR, 2004

[2] James D. Foley, Andries Van Dam, Steven K. Feiner: **Computer Graphics: Principles and Practice,** Addison-Wesley, 1995

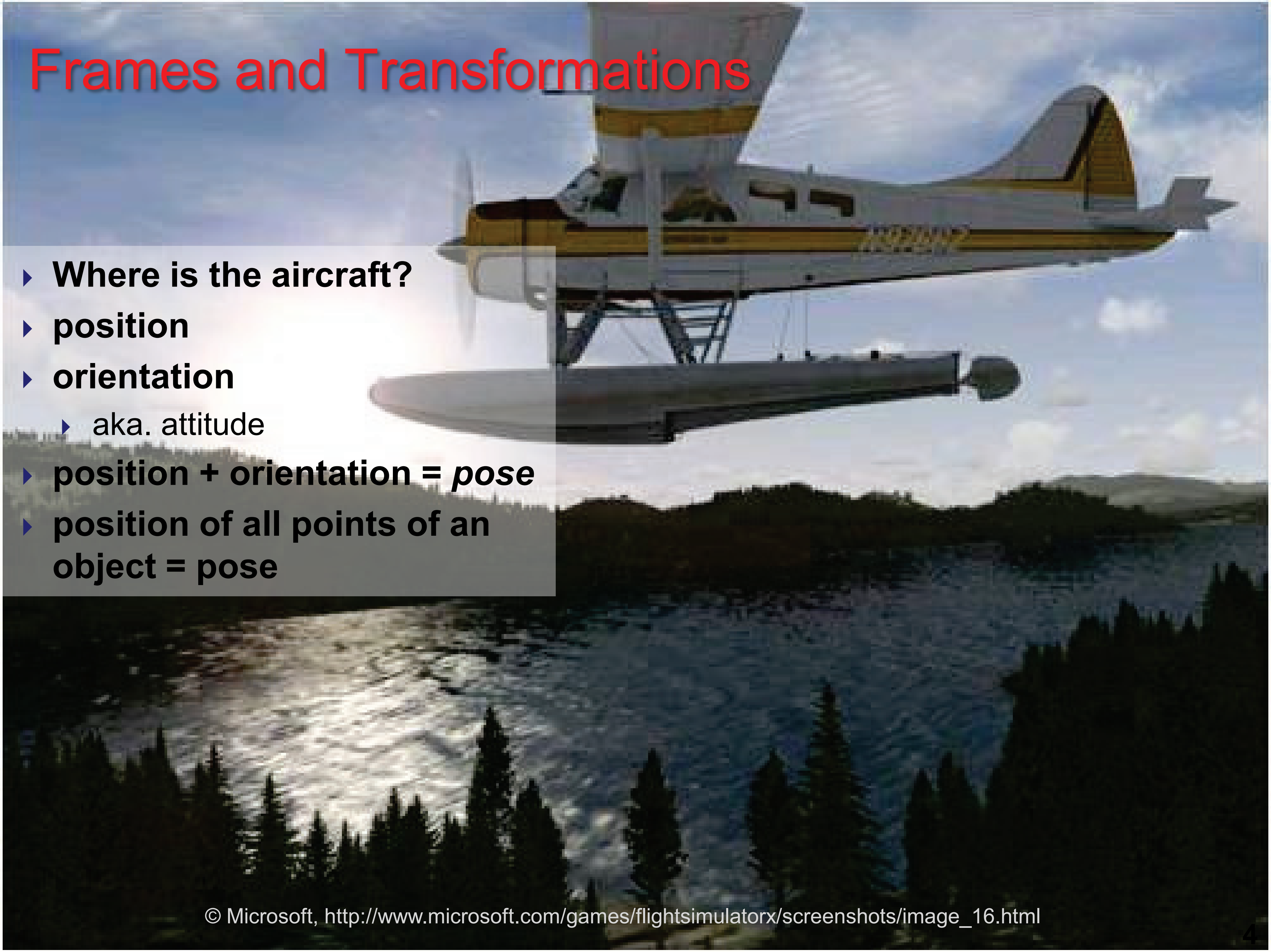
Frames and Transformations

- **Where is the aircraft?**

© Microsoft, http://www.microsoft.com/games/flightsimulatorx/screenshots/image_16.html

Frames and Transformations

- **Where is the aircraft?**
- **position**
- **orientation**
  - aka. attitude
- **position + orientation = *pose***
- **position of all points of an object = pose**



Frames and Transformations

Overview

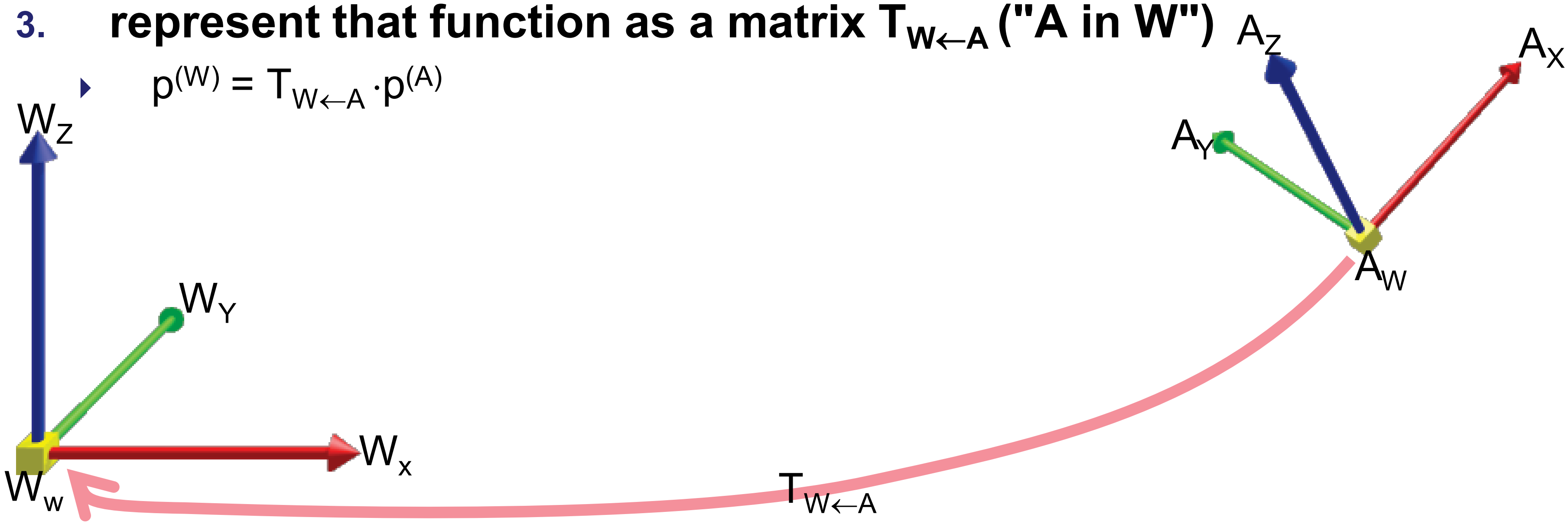
I. Transformation between Frames

I. Transformation between Frames

<u>Overview:</u>

Three steps to represent a geometrical pose in a computer

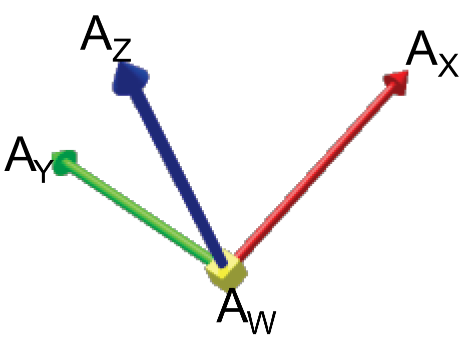
1. **attach frames (coordinate systems) to object and world**
   - aircraft-frame (A) and world-frame (W)
2. **consider the function that maps A-coordinates ($p^{(A)}$) of a point p to W-coordinates ($p^{(W)}$) of the same point**
3. **represent that function as a matrix $T_{W \leftarrow A}$ ("A in W")**
   - $p^{(W)} = T_{W \leftarrow A} \cdot p^{(A)}$



I. Transformation between Frames

1. Attach Frames

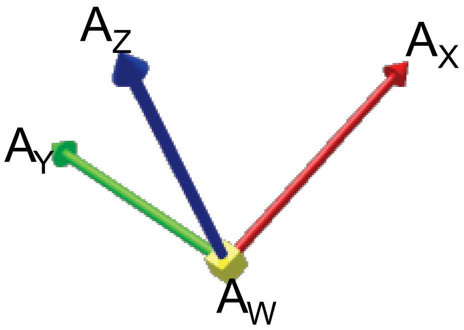
- **frame (coordinate system) A: 3 axes + origin in space**
  - X-axis: A_x (free vector, length 1)
  - Y-axis: A_y (free vector, length 1)
  - Z-axis: A_z (free vector, length 1)
  - A_X, A_Y, A_Z orthogonal
  - origin A_w (point)
- **same for frame W**



I. Transformation between Frames

2. Map A- to W-Coordinates

- **frame A defines a mapping between a geometric point/vector and four coordinates in the computer**
- **A: point/vector $\mapsto$ (x,y,z,w)**
- **$p^{(A)}$ = (x,y,z,w): geometric point/vector p expressed in A-coordinates**
  - point (w=1)
  - free vector (w=0)



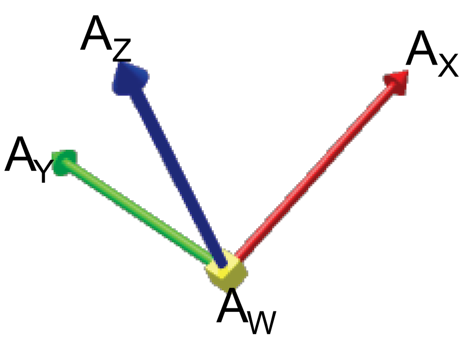
I. Transformation between Frames

2. Map A- to W-Coordinates

- **distinction**
  - p: point / free vector ⇒ geometrical entity
  - $p^{(A)}$: 4 coordinates ⇒ representation in the computer
- **relation between p and $p^{(A)}$**

$$p = \left(p^{(A)}\right)_X \cdot A_X + \left(p^{(A)}\right)_Y \cdot A_Y + \left(p^{(A)}\right)_Z \cdot A_Z + \left(p^{(A)}\right)_W \cdot A_W$$

$$p = \sum_{i=X,Y,Z,W} \left(p^{(A)}\right)_i \cdot A_i$$



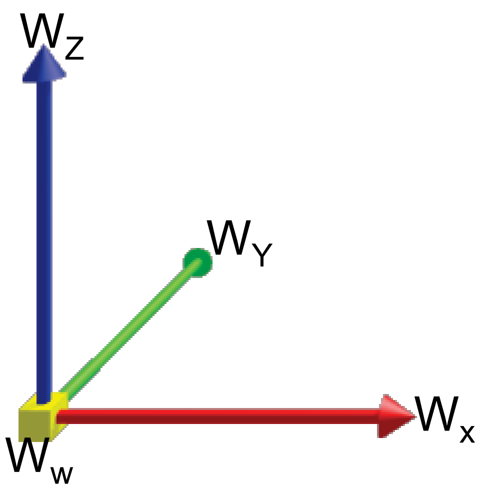
I. Transformation between Frames

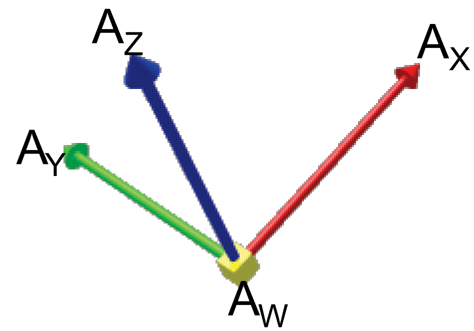
2. Map A- to W-Coordinates

- Question to the audience: Give numbers for these coordinates

$$A_X^{(A)} \quad A_Y^{(A)} \quad A_Z^{(A)} \quad A_W^{(A)}$$

$$\begin{pmatrix} ? \\ ? \\ ? \\ ? \end{pmatrix} \quad \begin{pmatrix} ? \\ ? \\ ? \\ ? \end{pmatrix} \quad \begin{pmatrix} ? \\ ? \\ ? \\ ? \end{pmatrix} \quad \begin{pmatrix} ? \\ ? \\ ? \\ ? \end{pmatrix}$$

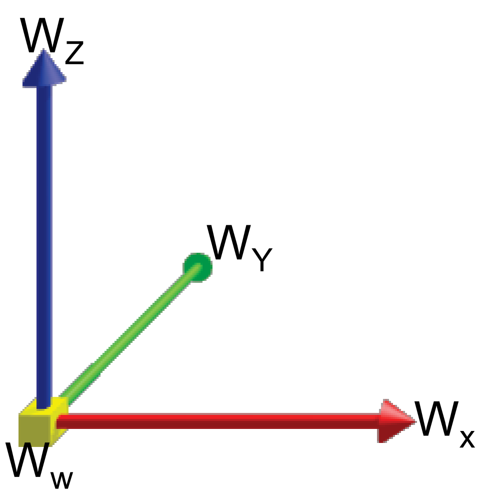




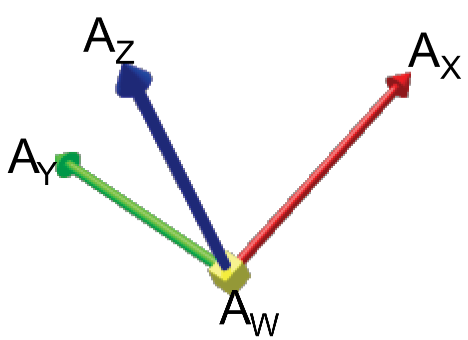
I. Transformation between Frames

2. Map A- to W-Coordinates

- **Question to the audience: Give numbers for these coordinates**



$$A_X^{(A)} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad A_Y^{(A)} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \quad A_Z^{(A)} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \quad A_W^{(A)} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$



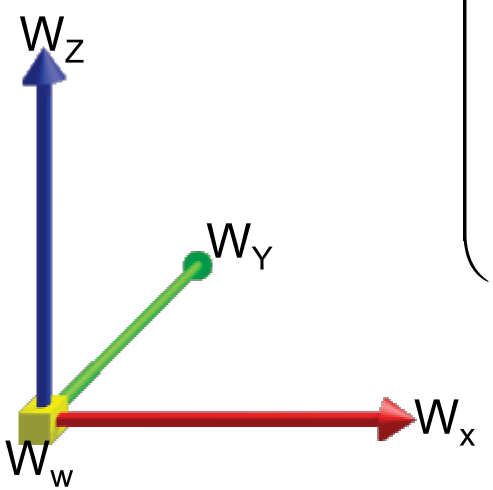
I. Transformation between Frames

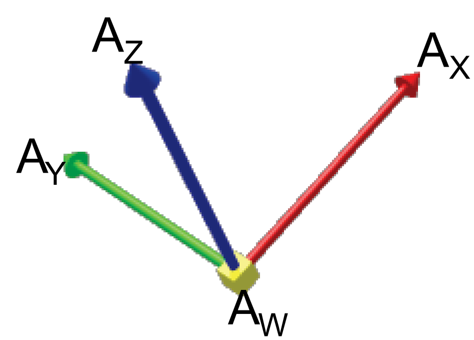
2. Map A- to W-Coordinates

- **Question to the audience: Assign the coordinates to the vectors above.**

$$A_X^{(W)} \qquad A_Y^{(W)} \qquad A_Z^{(W)} \qquad A_W^{(W)}$$

$$\begin{pmatrix} -0.7 \\ +0.3 \\ +0.6 \\ 0 \end{pmatrix} \begin{pmatrix} 0.8m \\ 0.2m \\ 0.2m \\ 1 \end{pmatrix} \begin{pmatrix} +0.7 \\ +0.6 \\ +0.4 \\ 0 \end{pmatrix} \begin{pmatrix} -0.3 \\ +0.7 \\ -0.7 \\ 0 \end{pmatrix}$$





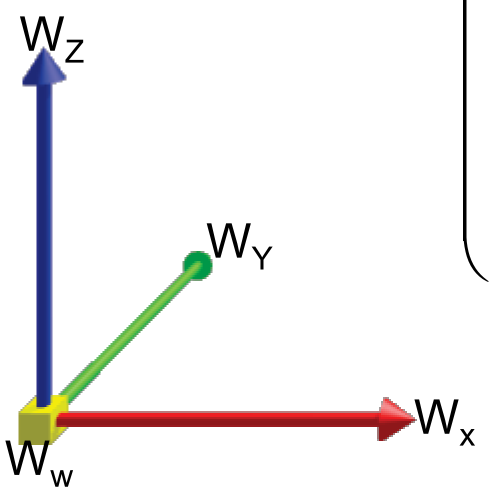
I. Transformation between Frames

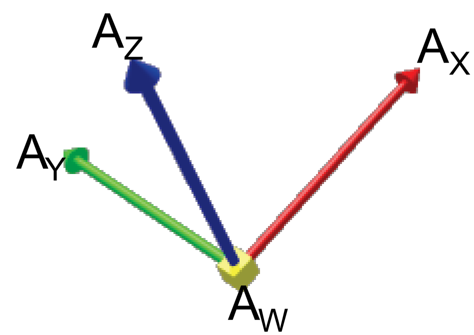
2. Map A- to W-Coordinates

- **Question to the audience: Assign the coordinates to the vectors above.**

$$A_X^{(W)} \qquad A_Y^{(W)} \qquad A_Z^{(W)} \qquad A_W^{(W)}$$

$$\begin{pmatrix} -0.7 \\ +0.3 \\ +0.6 \\ 0 \end{pmatrix} \begin{pmatrix} 0.8m \\ 0.2m \\ 0.2m \\ 1 \end{pmatrix} \begin{pmatrix} +0.7 \\ +0.6 \\ +0.4 \\ 0 \end{pmatrix} \begin{pmatrix} -0.3 \\ +0.7 \\ -0.7 \\ 0 \end{pmatrix}$$





I. Transformation between Frames

3. Represent Mapping as Matrix

- let A and B be two frames
- the desired $T_{B\leftarrow A}$ must satisfy $p^{(B)} = T_{B\leftarrow A}\, p^{(A)}$
- recall: the image of i-th unit vector is the i-th column of the matrix
- substitute $p=A_X$

$$\Rightarrow \quad A_X^{(B)} = T_{B\leftarrow A} \cdot A_X^{(A)} = T_{B\leftarrow A} \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \underbrace{\left(T_{B\leftarrow A}\right)_{\bullet 1}}_{\text{first column of } T_{B\leftarrow A}}$$

- same for $p=A_Y$ (2nd column), $p=A_Z$ (3rd column), $p=A_W$ (4th column)

$$\Rightarrow \quad T_{B\leftarrow A} = \begin{pmatrix} A_X^{(B)} & A_Y^{(B)} & A_Z^{(B)} & A_W^{(B)} \end{pmatrix}$$

I. Transformation between Frames

<u>Theorem</u>:

For coordinate frames A and B, the transformation matrix

$$T_{B \leftarrow A} = \begin{pmatrix} A_X^{(B)} & A_Y^{(B)} & A_Z^{(B)} & A_W^{(B)} \end{pmatrix}$$

converts A- to B-coordinates, i.e. for any point or vector p

$$p^{(B)} = T_{B \leftarrow A} \cdot p^{(A)}.$$

I. Transformation between Frames

Proof

$$T_{B\leftarrow A}\cdot p^{(A)} = \begin{pmatrix} A_X^{(B)} & A_Y^{(B)} & A_Z^{(B)} & A_X^{(B)} \end{pmatrix}\cdot p^{(A)} \qquad \text{definition of } T_{B\leftarrow A}$$

$$= \sum_i A_i^{(B)}\cdot\left(p^{(A)}\right)_i \qquad \text{definition of matrix}$$

$$= \sum_i \left(A_i\cdot\left(p^{(A)}\right)_i\right)^{(B)} \qquad \_^{(B)}\ \text{linear}$$

$$= \left(\overbrace{\sum_i A_i\cdot\left(p^{(A)}\right)_i}^{p}\right)^{(B)} \qquad \_^{(B)}\ \text{linear}$$

$$= p^{(B)} \qquad \text{coordinates}$$

II. Operations on Transformations

II. Operations on Transformations

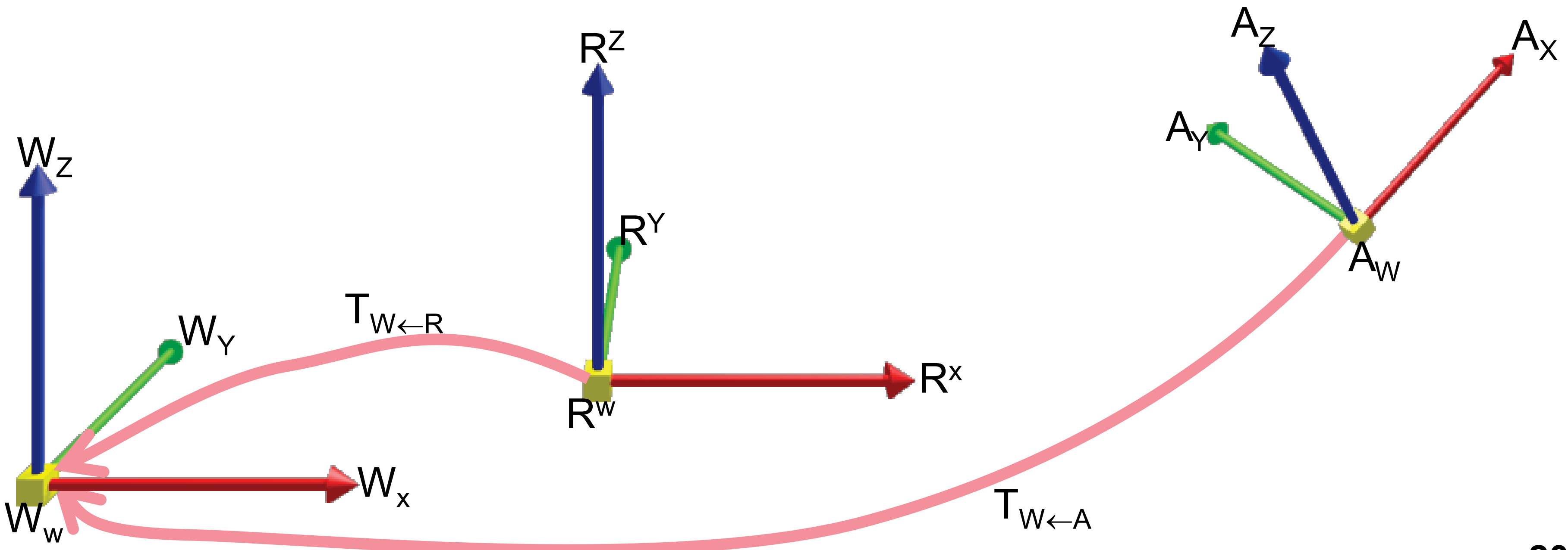
Common Operations

- **let A, B, C be three frames**
- **inverse**
  - $T_{B \leftarrow A} = (T_{A \leftarrow B})^{-1}$
  - proof: $p^{(A)} = T_{A \leftarrow B} \cdot p^{(B)} \Leftrightarrow (T_{A \leftarrow B})^{-1} \cdot p^{(A)} = p^{(B)}$
  - $T_{A \leftarrow B}$ is always invertible (see next lecture)
- **composition**
  - $T_{C \leftarrow A} = T_{C \leftarrow B} \cdot T_{B \leftarrow A}$
  - proof: $T_{C \leftarrow B} \cdot T_{B \leftarrow A} \cdot p^{(A)} = T_{C \leftarrow B} \cdot p^{(B)} = p^{(C)}$
  - read from right to left to check consistency
  - output of right transformation ("B") must be input of left transformation

II. Operations on Transformations

Example: Aircraft and Runway

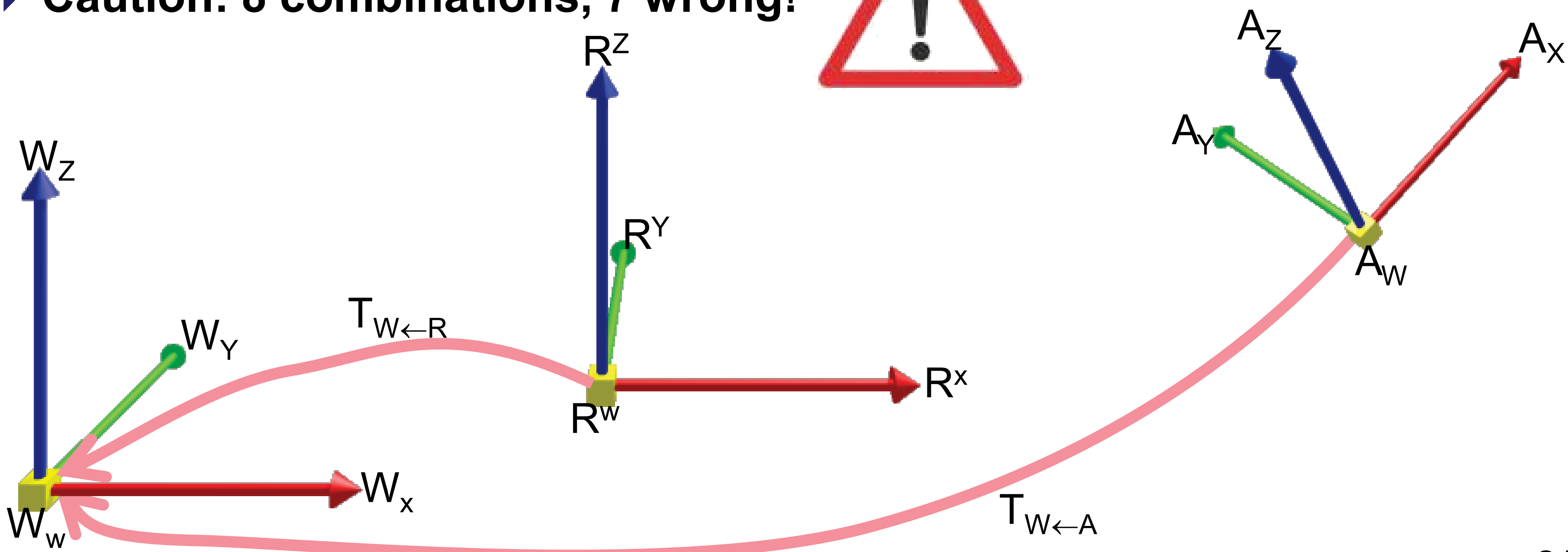
- **Question to the audience: We have aircraft in world ($T_{W \leftarrow A}$) and runway in world ($T_{W \leftarrow R}$). How do we compute "aircraft in runway" ($T_{R \leftarrow A}$)?**
- **Remember: $T_{B \leftarrow A} = (T_{A \leftarrow B})^{-1}$, $T_{C \leftarrow A} = T_{C \leftarrow B} \cdot T_{B \leftarrow A}$**



II. Operations on Transformations

Example: Aircraft and Runway

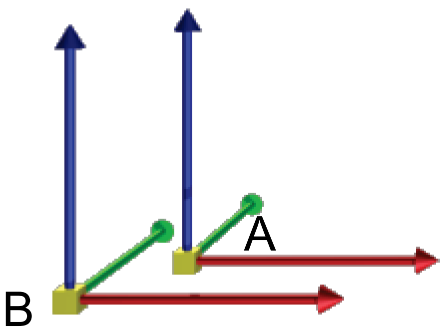
- **Question to the audience: We have aircraft in world ($T_{W \leftarrow A}$) and runway in world ($T_{W \leftarrow R}$). How do we compute "aircraft in runway" ($T_{R \leftarrow A}$)?**
- **Remember: $T_{B \leftarrow A} = (T_{A \leftarrow B})^{-1}$, $T_{C \leftarrow A} = T_{C \leftarrow B} \cdot T_{B \leftarrow A}$**
- **$T_{R \leftarrow A} = (T_{W \leftarrow R})^{-1} \cdot T_{W \leftarrow A}$**
- **Caution: 8 combinations, 7 wrong!**



II. Operations on Transformations

Common Transformations: Translation

- **let A and B be two frames with**
- **same axes**
- **origin of frame A at (x,y,z,1)$^{(B)}$**

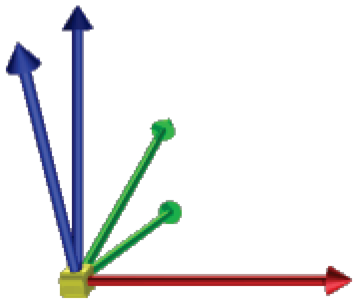


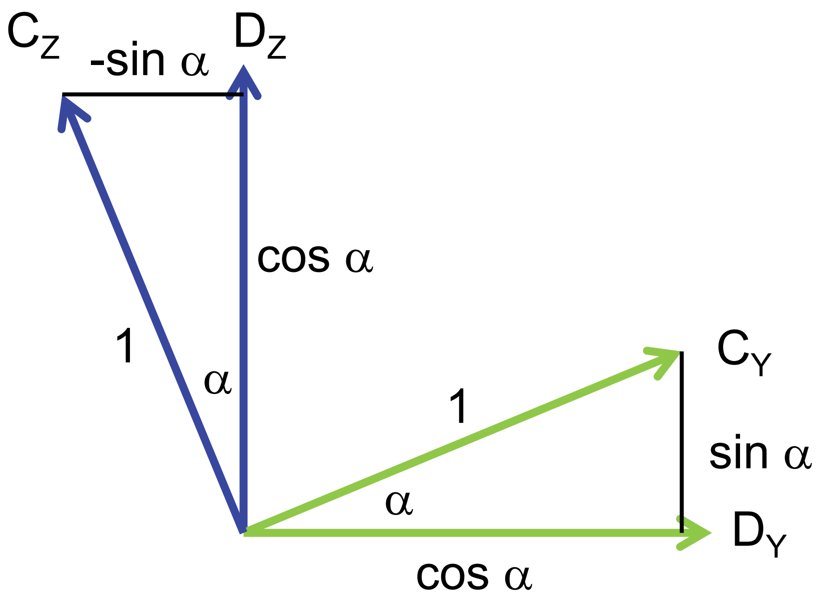
$$T_{B\leftarrow A} = \begin{pmatrix} 1 & & & x \\ & 1 & & y \\ & & 1 & z \\ & & & 1 \end{pmatrix}$$

II. Operations on Transformations

Common Transformations: Cardinal Rotations

- let C and D be two frames
- with same origin
- frame C is rotated around D_X by α



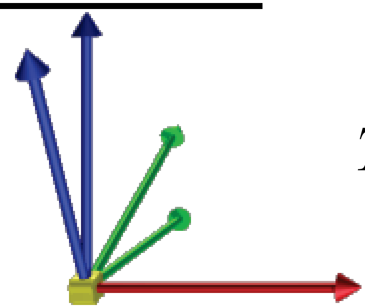


$$T_{D\leftarrow C} = \begin{pmatrix} 1 & & & \\ & \cos\alpha & -\sin\alpha & \\ & \sin\alpha & \cos\alpha & \\ & & & 1 \end{pmatrix}$$

II. Operations on Transformations

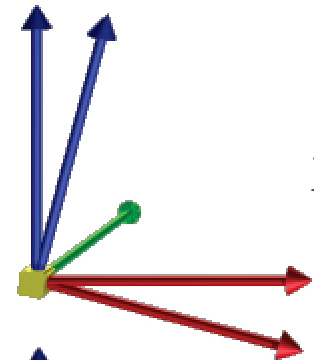
Common Transformations: Cardinal Rotations

- let C and D be two frames
- with same origin
- frame C is rotated around D_X by α



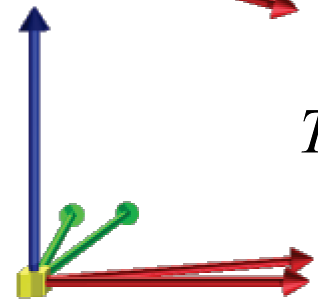
$$T_{D \leftarrow C} = \begin{pmatrix} 1 & & & \\ & \cos\alpha & -\sin\alpha & \\ & \sin\alpha & \cos\alpha & \\ & & & 1 \end{pmatrix}$$

- let E and F be two frames
- with same origin
- frame E is rotated around F_Y by α



$$T_{F \leftarrow E} = \begin{pmatrix} \cos\alpha & & \sin\alpha & \\ & 1 & & \\ -\sin\alpha & & \cos\alpha & \\ & & & 1 \end{pmatrix}$$

- let G and H be two frames
- with same origin
- frame G is rotated around H_Z by α



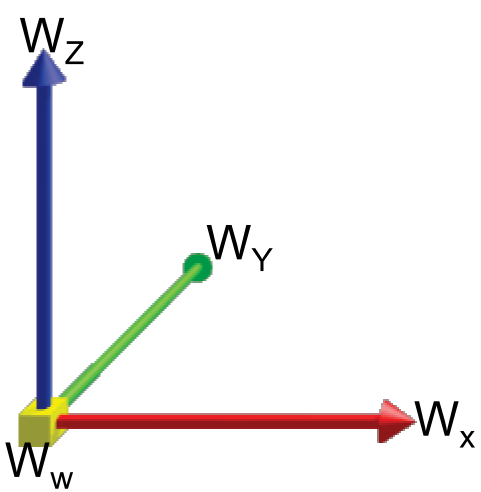
$$T_{H \leftarrow G} = \begin{pmatrix} \cos\alpha & -\sin\alpha & & \\ \sin\alpha & \cos\alpha & & \\ & & 1 & \\ & & & 1 \end{pmatrix}$$

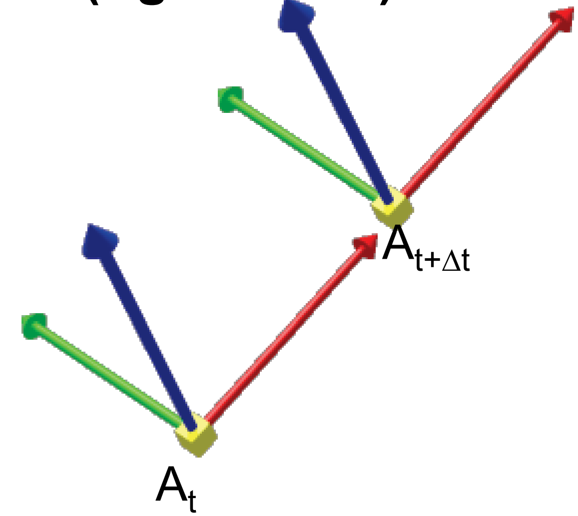
III. Representation of Motion

III. Representation of Motion

Motion as Transformation between Frames

- body-fixed frames move in space
- one frame for each time step $A_{t-\Delta t}$, A_t, $A_{t+\Delta t, \ldots}$
- define motion as $T_{At \leftarrow At+\Delta t}$ transformation "new in old"
- $T_{W \leftarrow At+\Delta t} = T_{W \leftarrow At} \cdot T_{At \leftarrow At+\Delta t}$
- "new in old times old in world is new in world. " (right to left)





III. Representation of Motion

Example: Simplistic Aircraft Motion Model

Question to the audience:

Which of the transformations should we use?

1. **The aircraft should fly forward.**

$$T_{W \leftarrow A_{t+\Delta t}} = T_{W \leftarrow A_t} \cdot \begin{pmatrix} & & & \\ & & & \\ & & & \\ & & & \end{pmatrix} \cdot \begin{pmatrix} & & & \\ & & & \\ & & & \\ & & & \end{pmatrix} \cdot \begin{pmatrix} & & & \\ & & & \\ & & & \\ & & & \end{pmatrix}$$

$$\begin{pmatrix} 1 & & & x \\ & 1 & & y \\ & & 1 & z \\ & & & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & & & \\ & \cos\alpha & -\sin\alpha & \\ & \sin\alpha & \cos\alpha & \\ & & & 1 \end{pmatrix} \quad \begin{pmatrix} \cos\alpha & & \sin\alpha & \\ & 1 & & \\ -\sin\alpha & & \cos\alpha & \\ & & & 1 \end{pmatrix} \quad \begin{pmatrix} \cos\alpha & -\sin\alpha & & \\ \sin\alpha & \cos\alpha & & \\ & & 1 & \\ & & & 1 \end{pmatrix}$$

III. Representation of Motion

Example: Simplistic Aircraft Motion Model

Question to the audience:

Which of the transformations should we use?

1. **The aircraft should fly forward.**
2. **Joystick left/right (j_x) should roll the aircraft.**

$$T_{W \leftarrow A_{t+\Delta t}} = T_{W \leftarrow A_t} \cdot \begin{pmatrix} 1 & & & v \cdot \Delta t \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{pmatrix} \cdot \begin{pmatrix} & & & \\ & & & \\ & & & \\ & & & \end{pmatrix} \cdot \begin{pmatrix} & & & \\ & & & \\ & & & \\ & & & \end{pmatrix}$$

$$\begin{pmatrix} 1 & & & x \\ & 1 & & y \\ & & 1 & z \\ & & & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & & & \\ & \cos\alpha & -\sin\alpha & \\ & \sin\alpha & \cos\alpha & \\ & & & 1 \end{pmatrix} \quad \begin{pmatrix} \cos\alpha & & \sin\alpha & \\ & 1 & & \\ -\sin\alpha & & \cos\alpha & \\ & & & 1 \end{pmatrix} \quad \begin{pmatrix} \cos\alpha & -\sin\alpha & & \\ \sin\alpha & \cos\alpha & & \\ & & 1 & \\ & & & 1 \end{pmatrix}$$

III. Representation of Motion

Example: Simplistic Aircraft Motion Model

Question to the audience:

Which of the transformations should we use?

1. **The aircraft should fly forward.**
2. **Joystick left/right (j_x) should roll the aircraft.**
3. **Joystick up/down (j_y) should pitch the aircraft.**

$$T_{W \leftarrow A_{t+\Delta t}} = T_{W \leftarrow A_t} \cdot \begin{pmatrix} 1 & & & v \cdot \Delta t \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & & & \\ & \cos j_x \cdot \Delta t & -\sin j_x \cdot \Delta t & \\ & \sin j_x \cdot \Delta t & \cos j_x \cdot \Delta t & \\ & & & 1 \end{pmatrix} \cdot \begin{pmatrix} & & & \\ & & & \\ & & & \\ & & & \end{pmatrix}$$

$$\begin{pmatrix} 1 & & & x \\ & 1 & & y \\ & & 1 & z \\ & & & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & & & \\ & \cos\alpha & -\sin\alpha & \\ & \sin\alpha & \cos\alpha & \\ & & & 1 \end{pmatrix} \quad \begin{pmatrix} \cos\alpha & & \sin\alpha & \\ & 1 & & \\ -\sin\alpha & & \cos\alpha & \\ & & & 1 \end{pmatrix} \quad \begin{pmatrix} \cos\alpha & -\sin\alpha & & \\ \sin\alpha & \cos\alpha & & \\ & & 1 & \\ & & & 1 \end{pmatrix}$$

III. Representation of Motion

Example: Simplistic Aircraft Motion Model

Question to the audience:

Which of the transformations should we use?

1. **The aircraft should fly forward.**
2. **Joystick left/right (j_x) should roll the aircraft.**
3. **Joystick up/down (j_y) should pitch the aircraft.**

$$T_{W \leftarrow A_{t+\Delta t}} = T_{W \leftarrow A_t} \cdot \begin{pmatrix} 1 & & & v \cdot \Delta t \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & & & \\ & \cos j_x \cdot \Delta t & -\sin j_x \cdot \Delta t & \\ & \sin j_x \cdot \Delta t & \cos j_x \cdot \Delta t & \\ & & & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos j_y \cdot \Delta t & & \sin j_y \cdot \Delta t & \\ & 1 & & \\ -\sin j_y \cdot \Delta t & & \cos j_y \cdot \Delta t & \\ & & & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & & & x \\ & 1 & & y \\ & & 1 & z \\ & & & 1 \end{pmatrix} \quad \begin{pmatrix} 1 & & & \\ & \cos\alpha & -\sin\alpha & \\ & \sin\alpha & \cos\alpha & \\ & & & 1 \end{pmatrix} \quad \begin{pmatrix} \cos\alpha & & \sin\alpha & \\ & 1 & & \\ -\sin\alpha & & \cos\alpha & \\ & & & 1 \end{pmatrix} \quad \begin{pmatrix} \cos\alpha & -\sin\alpha & & \\ \sin\alpha & \cos\alpha & & \\ & & 1 & \\ & & & 1 \end{pmatrix}$$

IV. Examples

IV. Examples

Virtuasphere

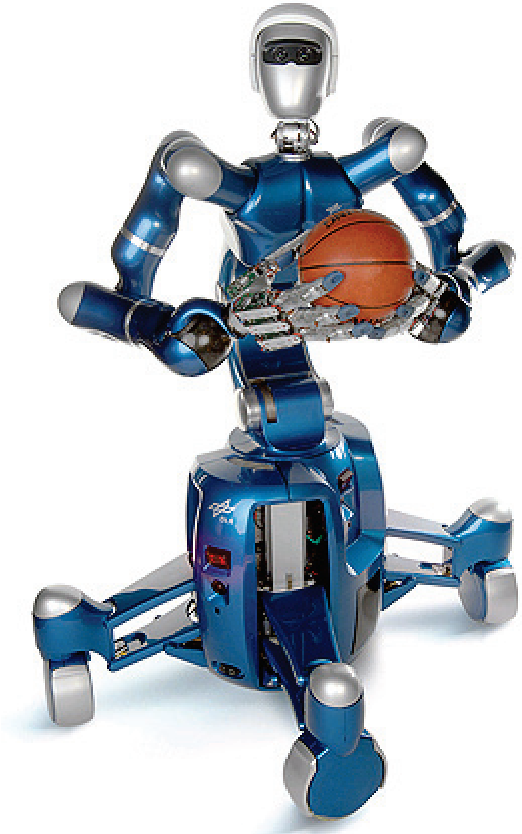
- **Frames**
  - World
  - Player
  - Head
- **$T_{World \leftarrow Player}$ accumulated from turning sphere**
- **position of $T_{Player \leftarrow Head}$ fixed**
- **orientation of $T_{Player \leftarrow Head}$ from compass and inclination sensor**
- **frames in reality vs. virtuality**

Source: Uni. Bremen, Kognitive Neuroinformatics, Tim Hantel

IV. Examples

Complex Humanoid Justin

- **50 joints**
  - 4 torso
  - 2×7 arms
  - 2×4×3 fingers
  - 4×2 wheels
- **one frame before and after each joint**
- **100 frames**



V. Tips for 3D Programming

V. Tips for 3D Programming

General Tips

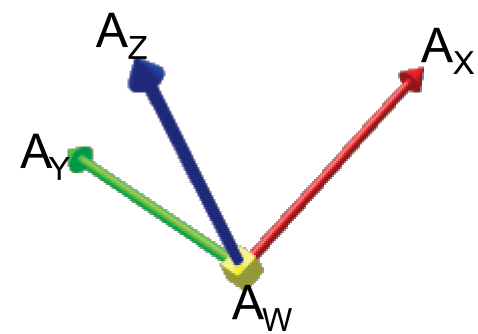
- **different frames and transformation can get *very* confusing**
- **formal naming sheme for variables avoids *many* errors**
- **name points / vectors $p^{(A)}$ with coordinate frame**
  - e.g. `pInAircraft`
- **name poses / transformations $T_{W \leftarrow A}$ with two coordinate frames**
  - e.g. `aircraftInWorld` for the pose of the aircraft in world-coordinates
- **consistency of formulas can be checked**
  - e.g. `aircraftInRunway` = `runwayInWorld.inverse()*aircraftInWorld`
- **prefer `objectInWorld` over `worldInObject`**
- **stick to conventions for choice of X-Y-Z axes**
  - domain specific conventions
  - general: Z pointing up
  - at least be self-consistent

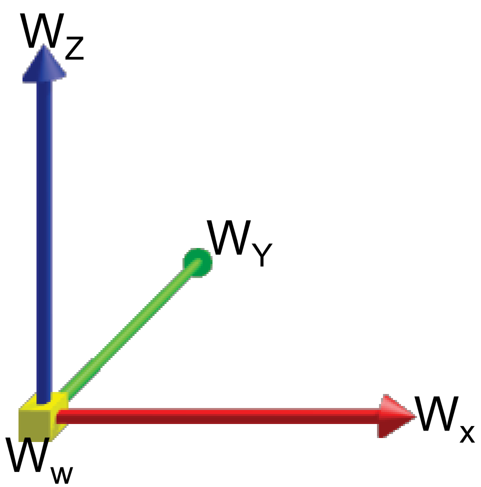
V. Tips for 3D Programming

Tips for Physical Machines

- **choose physical existing axes / planes**
  - e.g. X-Y plane on the ground / table
- **if necessary mark axes on machine**
- **document all frames with a sketch**

VI. Exercise Sheet





- $T_{B \leftarrow A} = (A_X^{(B)}, A_Y^{(B)}, A_Z^{(B)}, A_W^{(B)})$
- $T_{C \leftarrow A} = T_{C \leftarrow B}\, T_{B \leftarrow A}$
- $T_{A \leftarrow B} = (T_{B \leftarrow A})^{-1}$

VII. Summary

- **transformation between frames**
  - goal: represent object poses in space
    - *attach frames to every object and the world*
    - *represent transformation between frame A and B*
    - *by 4*4 matrix* $T_{B \leftarrow A}$
  - $T_{B \leftarrow A} = (A_X^{(B)}, A_Y^{(B)}, A_Z^{(B)}, A_W^{(B)})$
  - $p^{(B)} = T_{B \leftarrow A} \cdot p^{(A)}$
- **operations on transformations**
  - $T_{C \leftarrow A} = T_{C \leftarrow B} \, T_{B \leftarrow A}$
  - $T_{A \leftarrow B} = (T_{B \leftarrow A})^{-1}$
  - read from right to left to check consistency (also in code)
  - common transformations:
    - *translation*
    - *rotation around x, y, or z*
- **motion means time dependent frames (A_t)**

Lehre

- **Didaktische Schwerpunkte**
  - verständliche Erklärungen im Detail, starker Anwendungsbezug
  - Interaktion durch Fragen an das Auditorium während der Vorlesung
  - mathematisch modellieren: Anschauung und Formalismus verknüpfen
- **Übungsprogramm**
  - motivierende und integrierte Übungen mit selbständigem Problemlösen
  - fachliche Gedanken in zusammenhängenden Texten ausdrücken
  - kreativer Freiraum in der Aufgabenstellung
- **Regelmäßige Lehrveranstaltungen**
  - Echtzeitbildverarbeitung (Sommer 05-12)
  - Informatik für Gestalter und General Studies (Winter 09-11)
  - Theorie der Sensorfusion (Winter 06, 08, 10, mit Lutz Schröder)
  - geplant: Medieninformatik 1 / 2
  - Angebot: 3D Geometry with Applications to Computer Games