# Transformations for Virtual Worlds

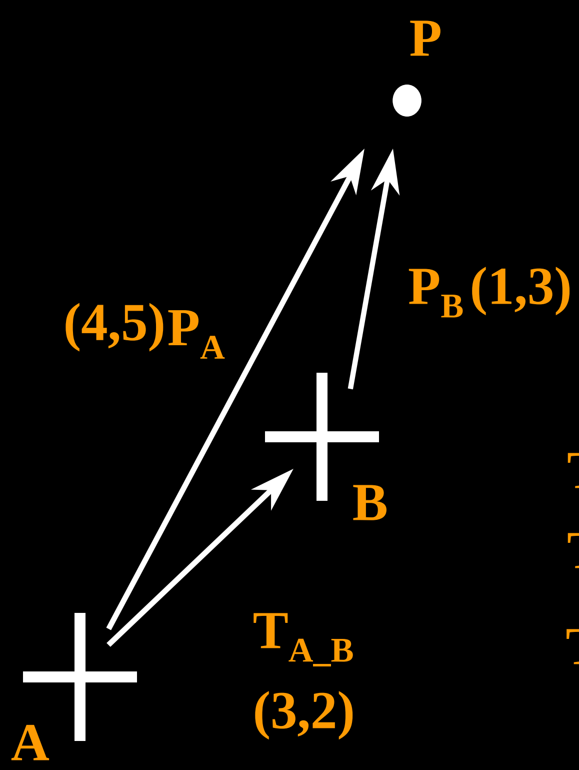
**(based on a talk by Greg Welch)**

# Coordinate System Transformations

## What is a transformation?

- **The instantaneous relationship between a pair of coordinate systems.**
  - **Defines the relative position, orientation and scale of two coordinate systems**
- **Notation: $T_{A\_B}$ is the transformation *from* coordinate system B *to* coordinate system A**

# Simple 2D Example




$P_A = T_{A\_B} \bullet P_B$

$(4,5) = (3,2) \bullet (1,3)$

**$T_{A\_B}$ converts points in B to points in A**

**$T_{A\_B}$ measures the position of B's origin in A**

**The vector runs from A to B**

# Properties of Coordinate System Transformations

- **Used to convert the coordinates of a point specified in one coordinate system to another.**

$$P_A = T_{A\_B} \bullet P_B$$

- **Can be *inverted***

$$\text{Inverse of } T_{A\_B} = T_{B\_A}$$

# Properties of Coordinate System Transformations

- **Can be *composed* to compute the relationship between several coordinate systems**

$$T_{A\_C} = T_{A\_B} \bullet T_{B\_C}$$

  - **Note: Nice property of subscript cancellation**

## Example:

$$T_{Shoulder\_Hand} = T_{Shoulder\_Elbow} \bullet T_{Elbow\_Hand}$$

# Transformation Representations

- **Can be represented by a 4x4 Transformation Matrix**
- **Alternately can use the VQS notation**
  - **Represent transformation as a Vector (translation), Quaternion (rotation), and a (uniform) Scaling factor**

# Transformation Matrices

**TRANSLATION**

$$\begin{bmatrix} 1 & 0 & 0 & T_X \\ 0 & 1 & 0 & T_Y \\ 0 & 0 & 1 & T_Z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

**ROTATION**

$$\begin{bmatrix} R_{11} & R_{12} & R_{13} & 0 \\ R_{21} & R_{22} & R_{23} & 0 \\ R_{31} & R_{32} & R_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

**SCALE**

$$\begin{bmatrix} S_X & 0 & 0 & 0 \\ 0 & S_Y & 0 & 0 \\ 0 & 0 & S_Z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

# VQS Notation

**TRANSLATION:**

$$V = (T_X , T_Y , T_Z )$$

**ROTATION:**

$$Q = (Q_X , Q_Y , Q_Z , Q_W )$$

**SCALE:**

$$S = S_{UNIFORM}$$

# Transformations: Why Quaternions?

- Allow simple interpolation
- More compact
- Angle and axis of rotation easy to extract
- More efficient (composing and inverting)
- More tractable mathematically than matrices or Euler angles

# VQS Transform from P to P’

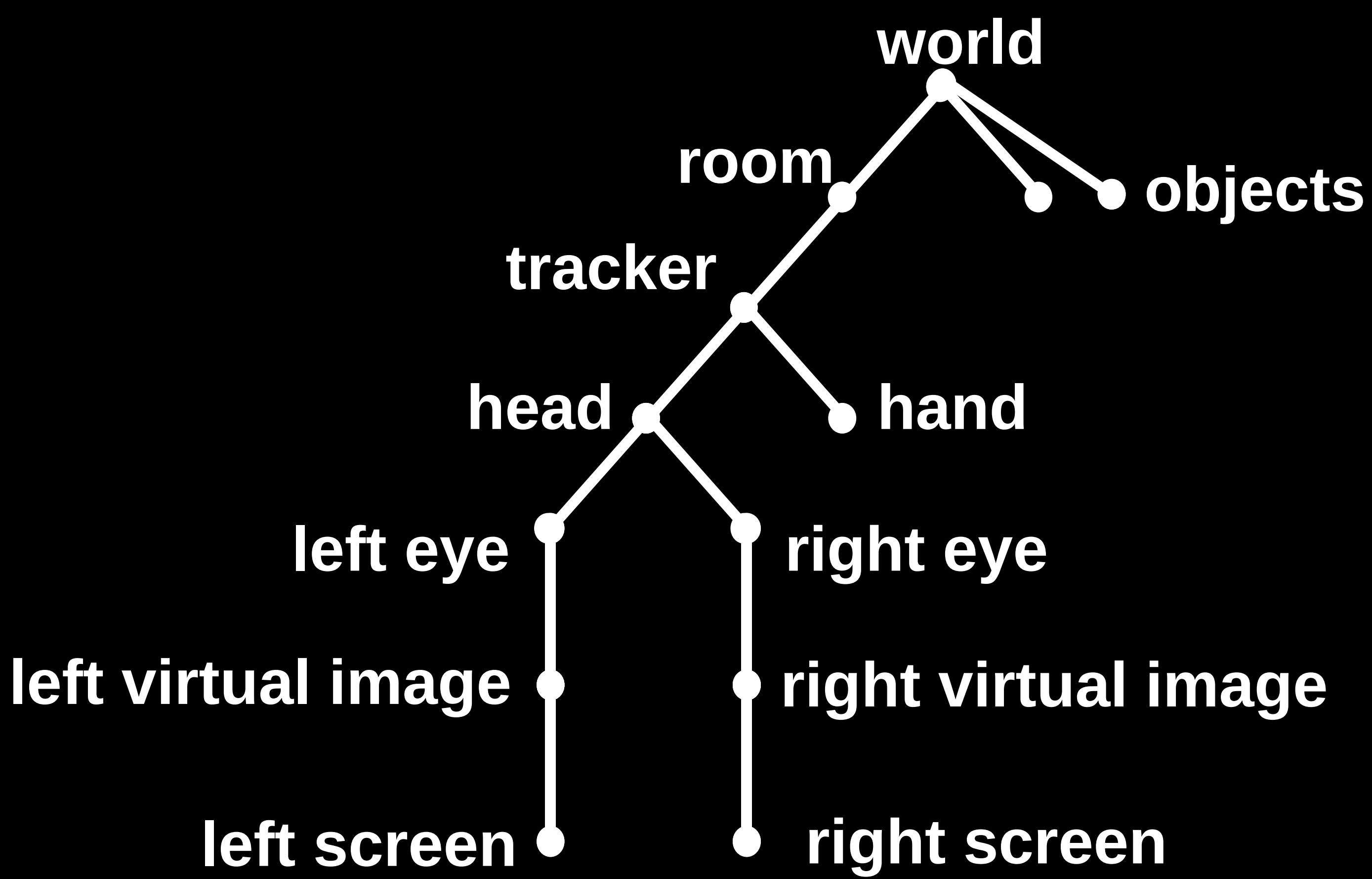
$$p' = [v, q, s] \bullet p = s\,(q * p * q^{-1}) + v$$

**(where p is treated as a quaternion with zero scalar component, and the result has a zero scalar component so can be treated as a vector.)**

# Coordinate System Graphs

- Graphical representation of the coordinate systems in a virtual world and their relationship
- Nodes represent coordinate systems
- Edges represent transformations

# Coordinate System Graphs



# Coordinate System Graphs: How do I use them?

Can be used to determine the transformations involved in converting between coordinate systems.

## Example: Finding world coord of head space point

$$P_{World} = T_{World\_Head} \bullet P_{Head}$$

$$T_{World\_Head} = T_{World\_Room} \bullet T_{Room\_Tracker} \bullet T_{Tracker\_Head}$$

$$P_{World} = T_{World\_Room} \bullet T_{Room\_Tracker} \bullet T_{Tracker\_Head} \bullet P_{Head}$$

# Coordinate System Graphs and Virtual World Interactions

- **Can be used to determine the transformations involved in any virtual world interaction**
  - **Specifying Actions With Invariants**
- **Based on *frame-to-frame invariants***
  - **A relation between a set of transformations in the current frame and a set from the previous frame**

# Specifying Virtual World Interactions

- **Coordinate system hierarchy and frame-to-frame invariants can be used to specify many forms of virtual world interaction:**
  - **Grabbing**
  - **Flying**
  - **Scaling**

# Specifying Actions With Invariants

## Example: Grabbing a Virtual Object

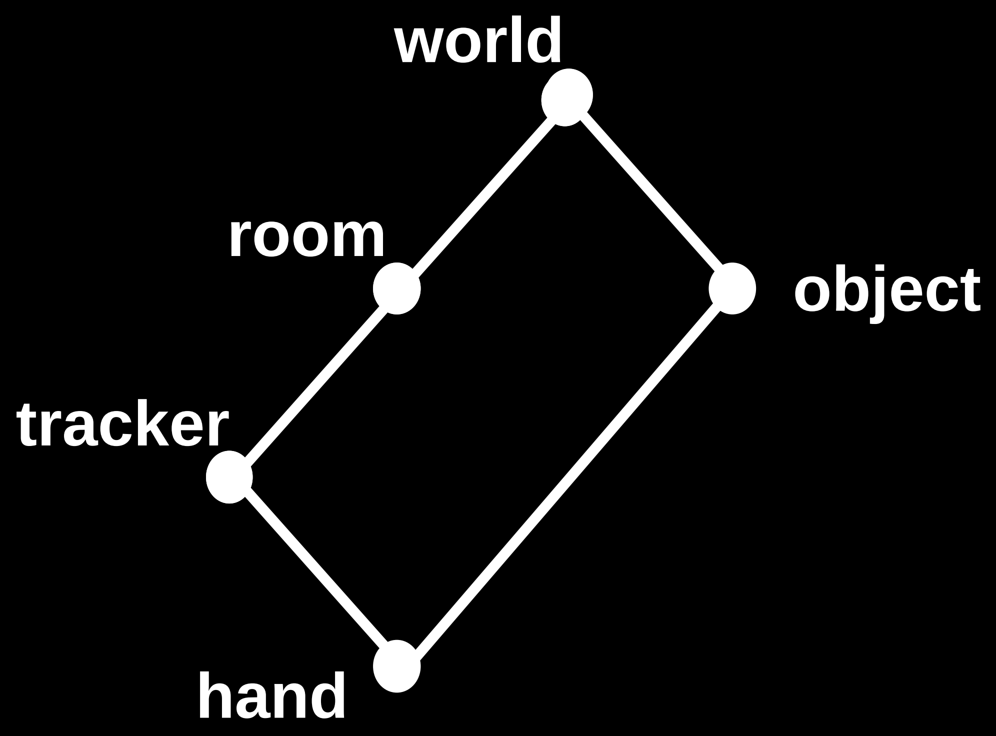
- **Objective: Keep an object "fixed" to the user's hand**
- **Frame-to-frame invariant:**

$$T^{n+1}_{Object\_Hand} = T^{n}_{Object\_Hand}$$

- **Result: Updated object position**

$$T^{n+1}_{World\_Object}$$

# Coordinate System Graph



# Specifying Actions With Invariants

- Use coordinate system graph to expand:

$$T_{Object\_Hand} = T_{Object\_World} \bullet T_{World\_Room} \bullet T_{Room\_Tracker} \bullet T_{Tracker\_Hand}$$

- Substitute:

$$T^2_{Object\_World} \bullet T^2_{World\_Room} \bullet T^2_{Room\_Tracker} \bullet T^2_{Tracker\_Hand} = T^1_{Object\_World} \bullet T^1_{World\_Room} \bullet T^1_{Room\_Tracker} \bullet T^1_{Tracker\_Hand}$$

# Specifying Actions With Invariants

- **Solve:**

$$\mathbf{T^2_{Object\_World} = T^1_{Object\_World} \bullet T^1_{World\_Room} \bullet T^1_{Room\_Tracker} \bullet T^1_{Tracker\_Hand} \bullet T^2_{Hand\_Tracker} \bullet T^2_{Tracker\_Room} \bullet T^2_{Room\_World}}$$

- **Invert $T^2_{Object\_World}$ to obtain $T^2_{World\_Object}$**

# Other Common Operations

- **Flying**
  - **Modify $T_{world\_room}$**
- **Scale world/user**
  - **Also modify $T_{world\_room}$**
  - **Often scale about hand or head**
- **Scale object**
  - **Scale about hand or about centroid**

# Where do I learn more?

- Computer graphics texts (e.g. Foley, vanDam, Feiner, and Hughes)
  - probably on reserve at MSE for Kumar's Computer Graphics class
- 1994 Paper by Robinett and Holloway
  - READ IT!
- Paper on quaternions by Shoemake and Chou
- Quaternion/transformation support provided by quatlib

# References


- Foley, J., A. van Dam, S. Feiner, J. Hughes (1990). *Computer Graphics: Principles and Practice* (2nd ed.). Addison-Wesley Publishing Co., Reading MA.
- Robinett, W., R. Holloway (1992). Implementation of flying, scaling, and grabbing in virtual worlds, *ACM Symposium on Interactive 3D Graphics,* Cambridge MA, March
- Shoemake, K. (1985). Animating rotations using quaternion curves, *Computer Graphics: Proc. of SIGGRAPH '85.*
- Chou, J. (1992). Quaternion Kinematic and Dynamic Differential Equations, *IEEE Transactions on Robotics and Automation*, Vol. 8, No. 1, Feb. 1992.