



Scene Graphs and Barycentric Coordinates

Misha Kazhdan

(601.457/657)



Last Time

2D Transformations

- Basic 2D transformations
- Matrix representation
- Matrix composition

3D Transformations

- Basic 3D transformations
- Matrix representation
- Matrix composition



Homogeneous Coordinates

Represent transformations by 4×4 matrices

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- The top-left 3×3 block represents the linear part of the transformation
- The last column represents the translation
- Transformations (translations/rotations/scales) can be composed using simple matrix multiplication



Overview

Transformation Hierarchies

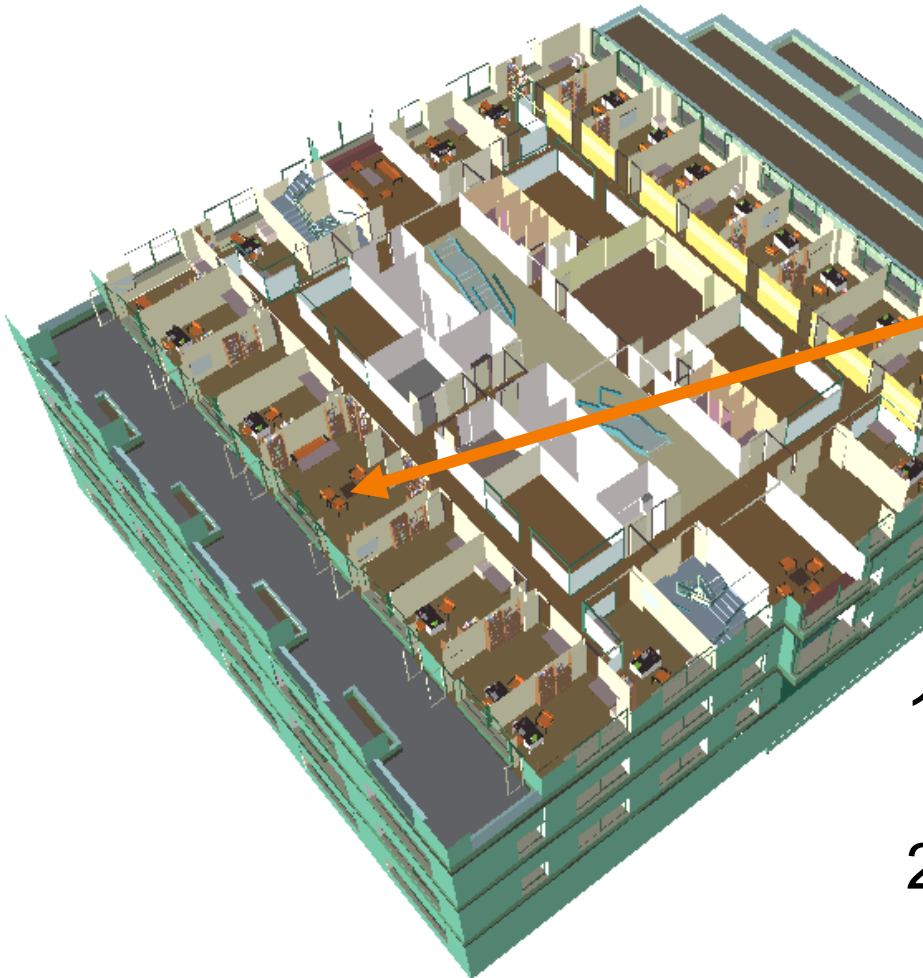
- Scene graphs
- Ray casting

Barycentric Coordinates



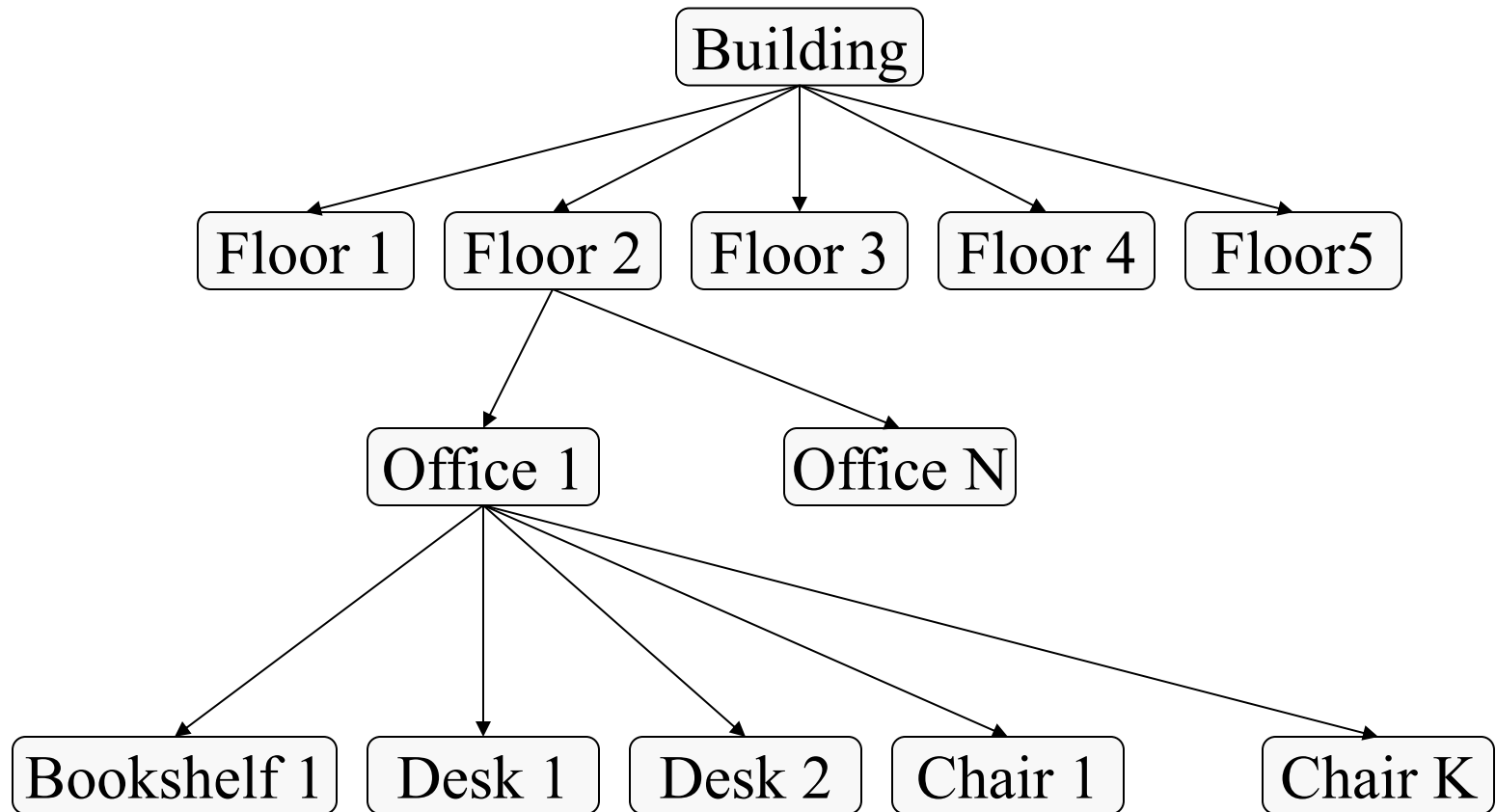
Scene Graphs: Static Environments

An object may appear in a scene multiple times

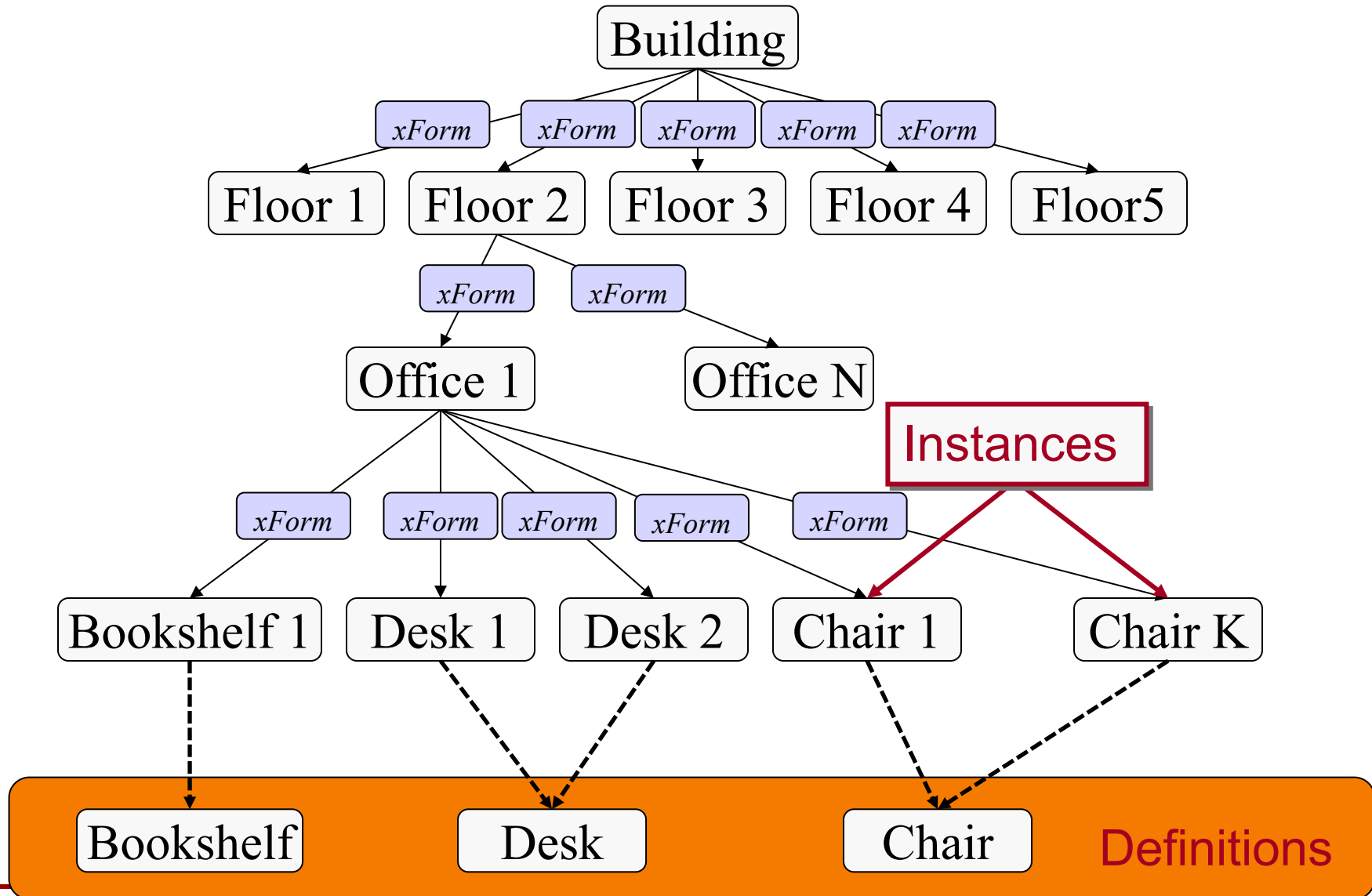


1. Model objects in their own coordinate systems
2. Draw same 3D data with different transformations

Scene Graphs: Static Environments

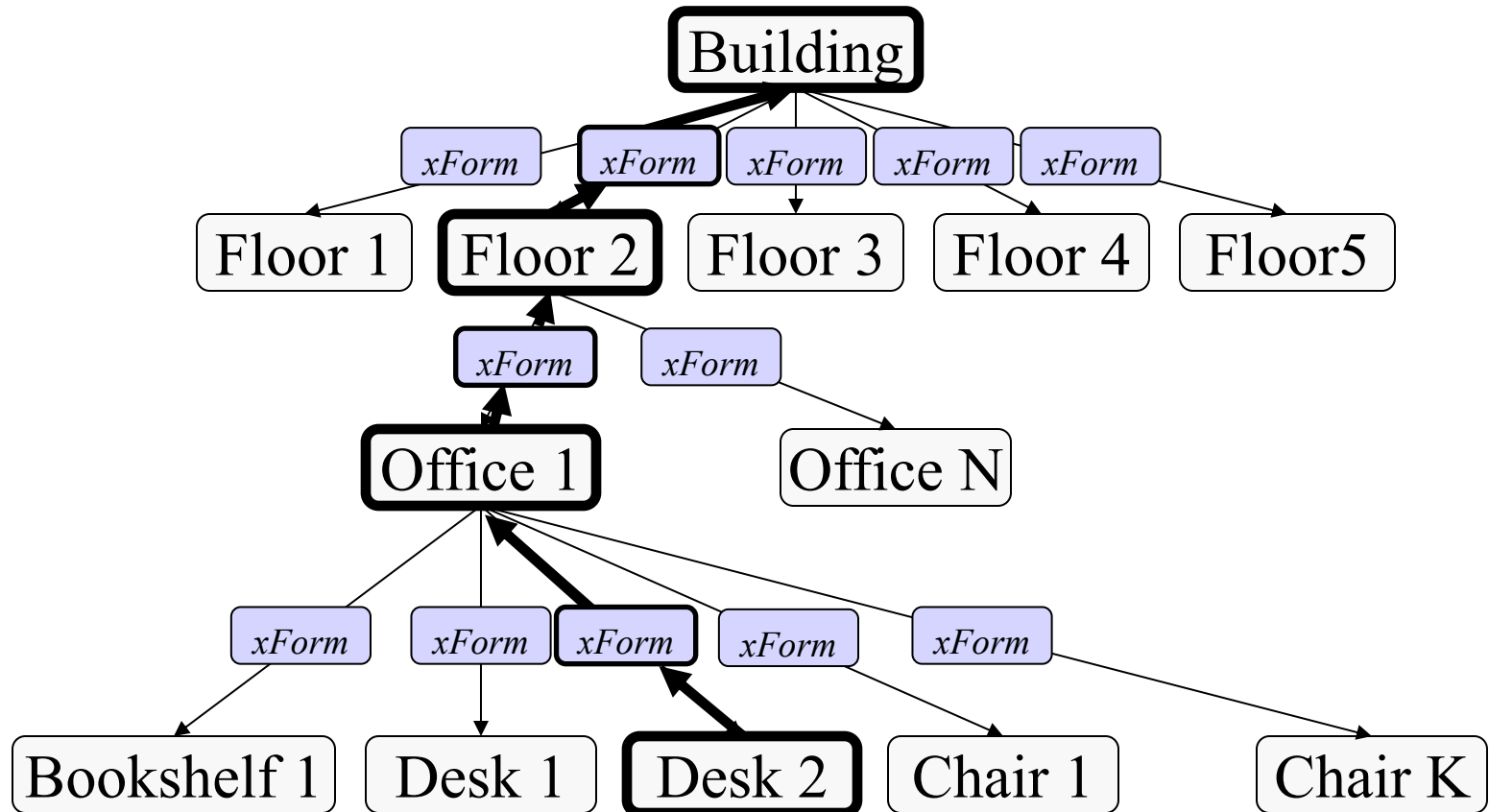


Scene Graphs: Static Environments





Scene Graphs: Static Environments



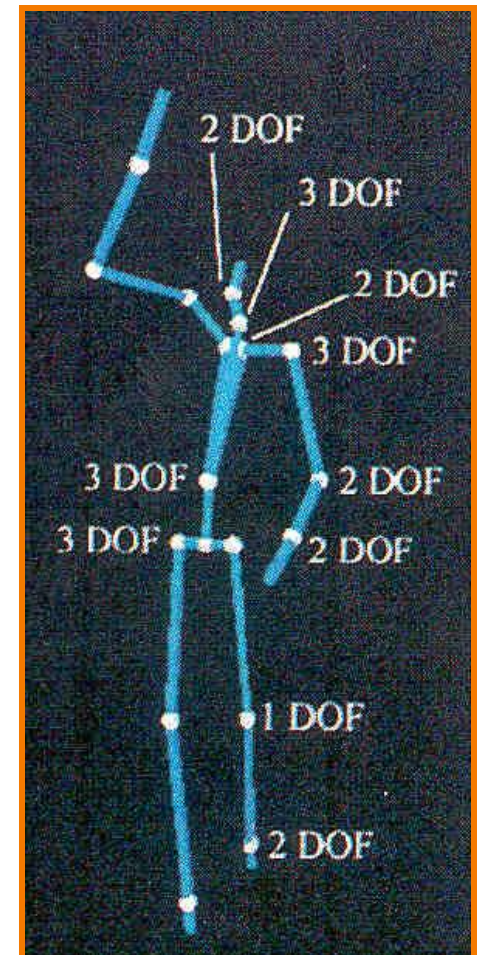
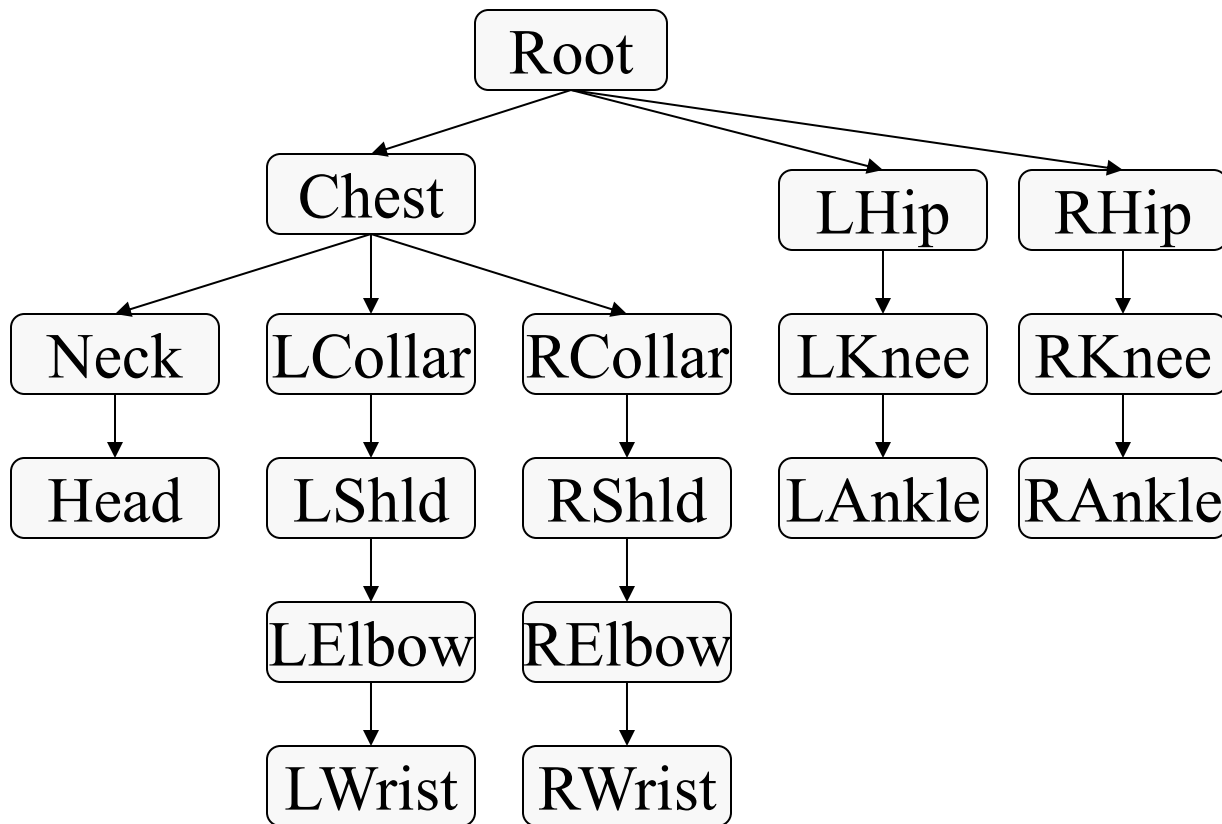
The transformation applied to a node of the scene graph is the composition of transformations along the path to the root.

⇒ Global transformations only requires transforming the root



Scene Graphs: Dynamic Characters

Well-suited for articulated characters



Rose et al. '96



Scene Graphs

Local Modeling

Allows modeling in local coordinates and then placing into a global frame – particularly important for animation

Instancing

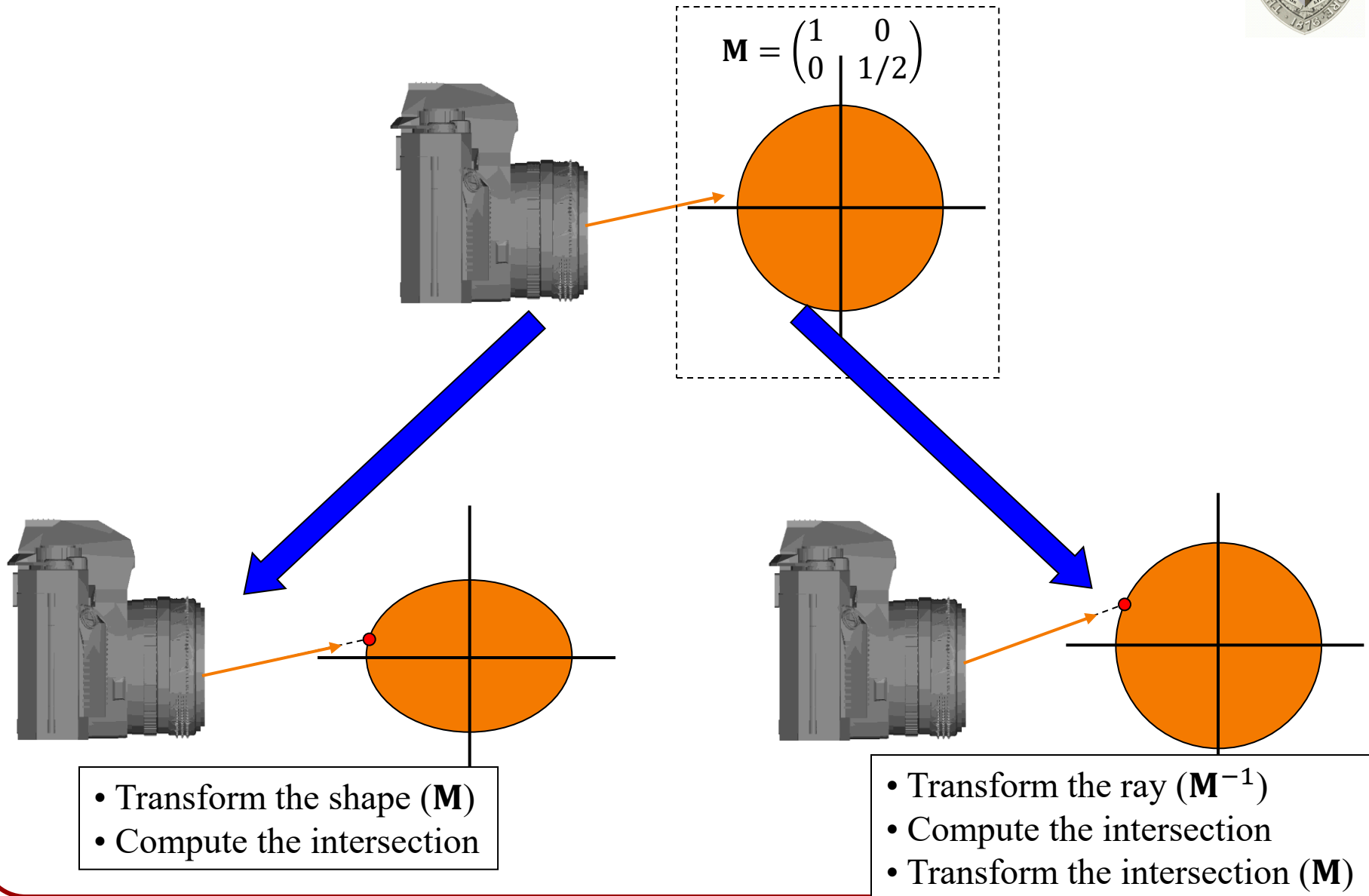
Allows multiple instances of a single model – reducing storage and facilitating making consistent changes

Hierarchical Representation

Facilitates modifying the scene by having geometry moved with its context

Accelerates ray-tracing by providing a hierarchy that can be used for bounding volume testing

Ray Casting





Ray Casting with Hierarchies

Iterating from the root, transform rays:

world $\rightarrow$ model coordinates:

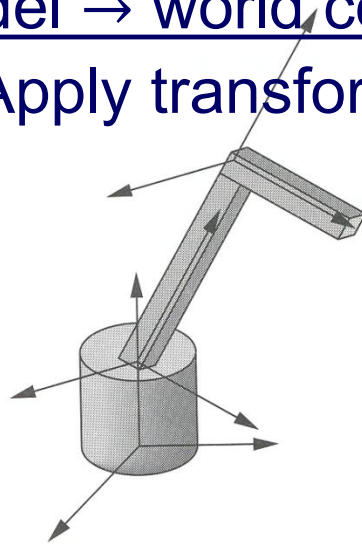
Apply inverse transform to ray

model coordinates:

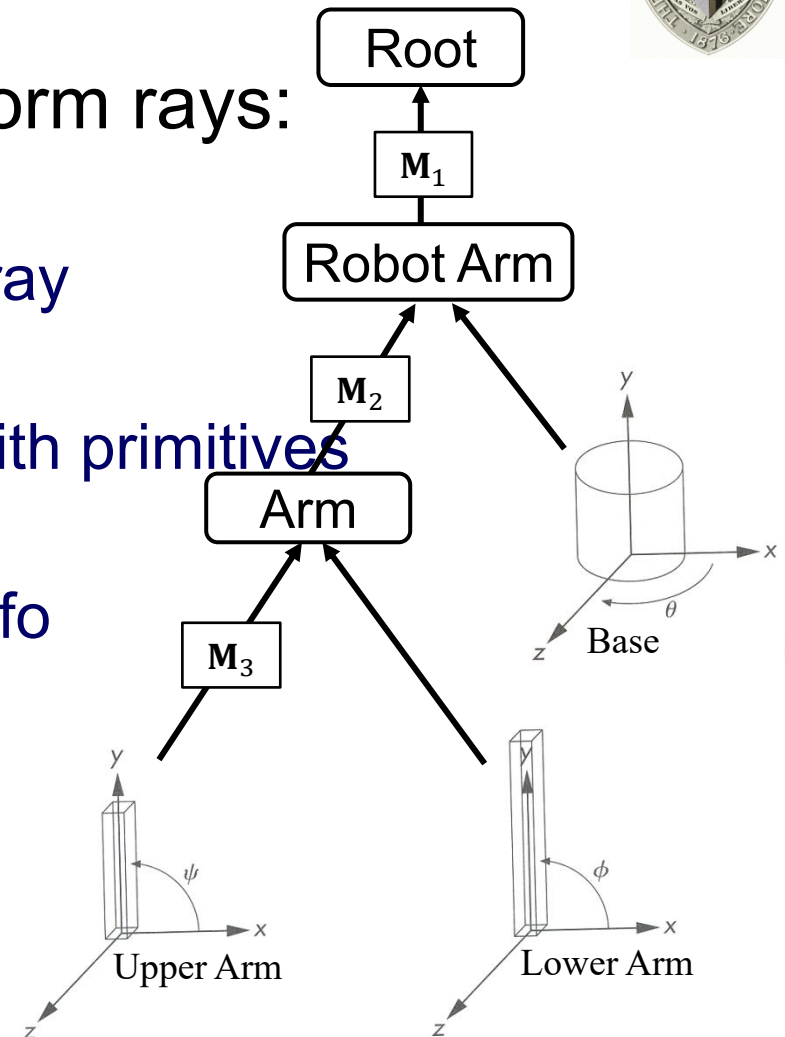
Intersect transformed ray with primitives

model $\rightarrow$ world coordinates:

Apply transform to the hit info



Robot Arm

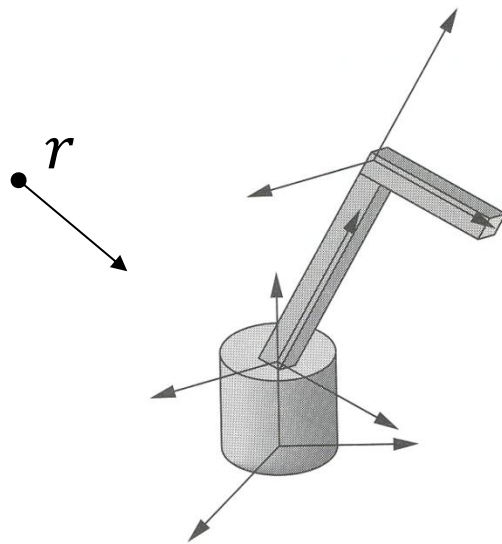




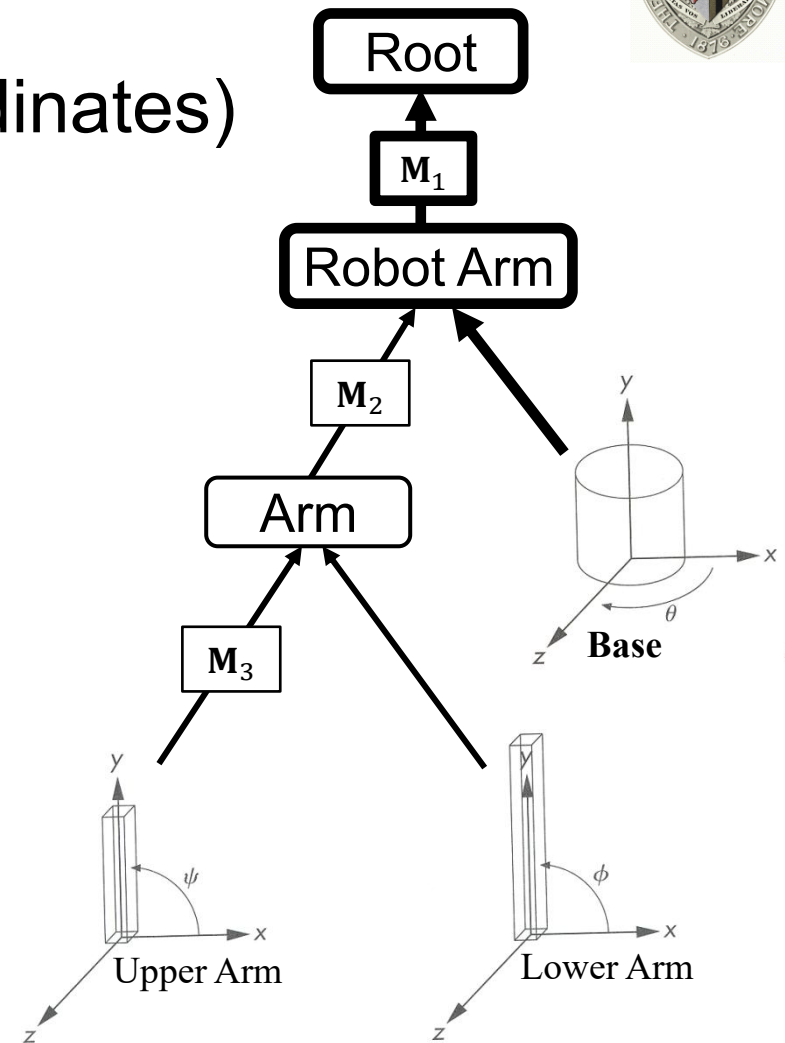
Ray Casting with Hierarchies

Given a ray r (in global coordinates)

- **Base:** Apply M_1^{-1} to the ray and test for intersection



Robot Arm



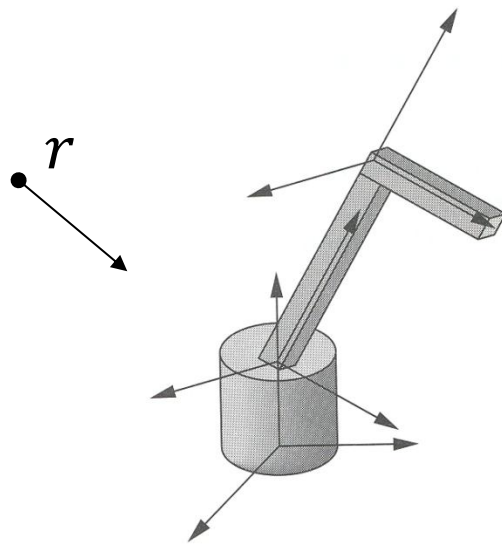
Angel Figures 8.8 & 8.9



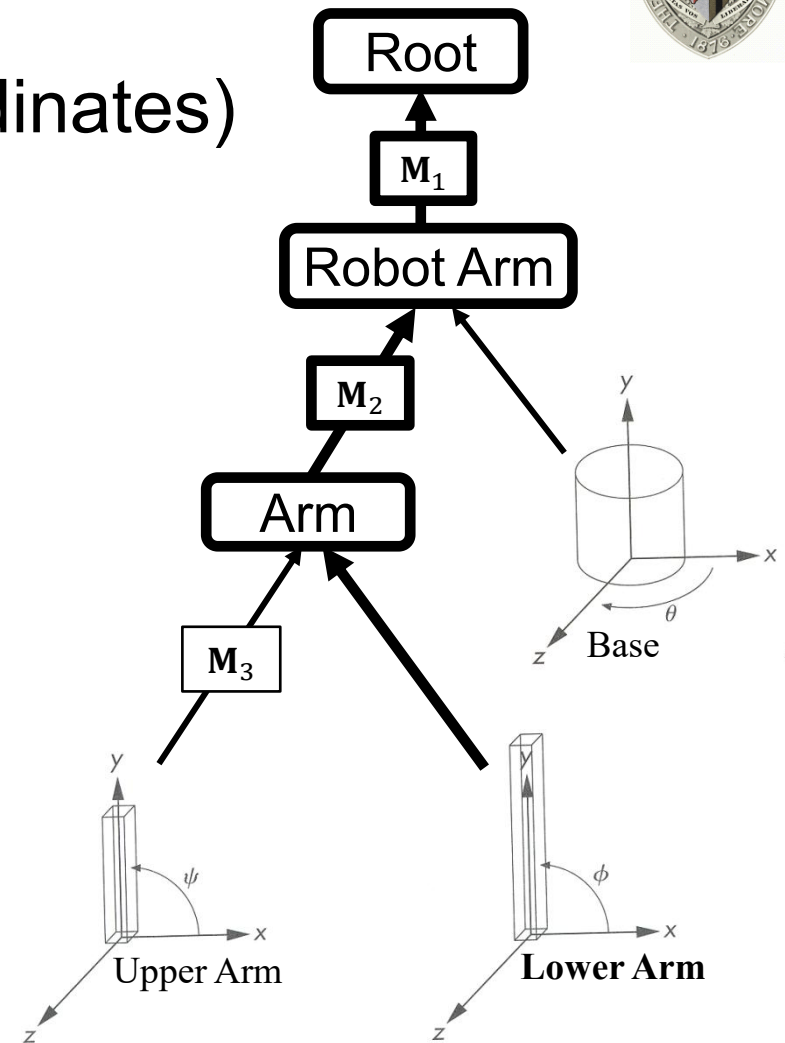
Ray Casting with Hierarchies

Given a ray r (in global coordinates)

- **Base:** Apply M_1^{-1} to the ray and test for intersection
- **Lower Arm:** Apply $(M_1 \cdot M_2)^{-1}$ to the ray and test for intersection



Robot Arm

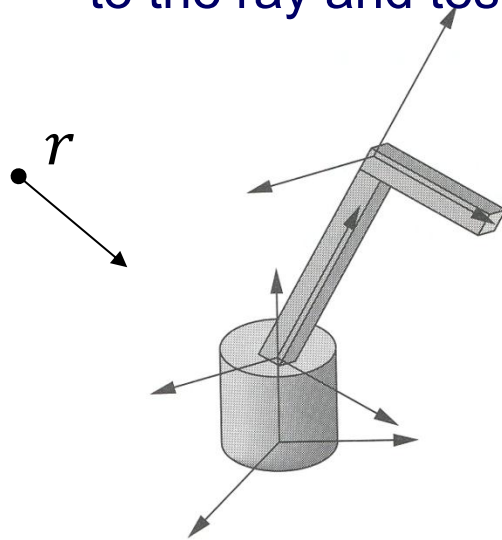




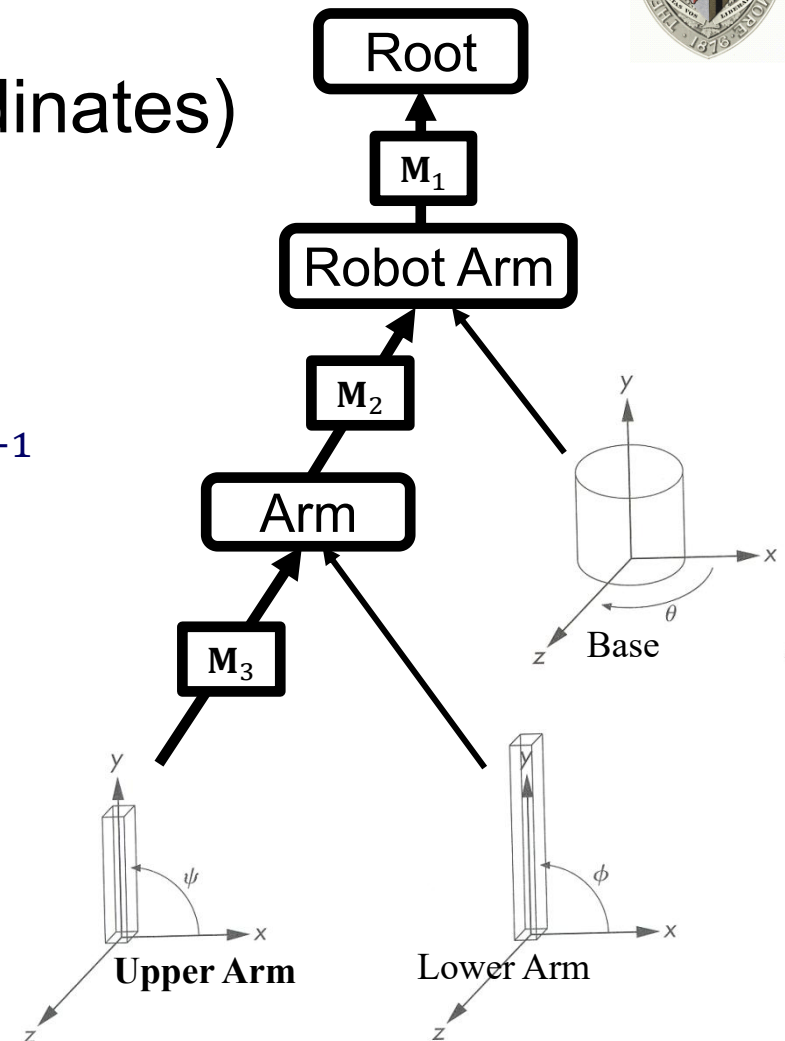
Ray Casting with Hierarchies

Given a ray r (in global coordinates)

- **Base:** Apply M_1^{-1} to the ray and test for intersection
- **Lower Arm:** Apply $(M_1 \cdot M_2)^{-1}$ to the ray and test for intersection
- **Upper Arm:** Apply $(M_1 \cdot M_2 \cdot M_3)^{-1}$ to the ray and test for intersection



Robot Arm

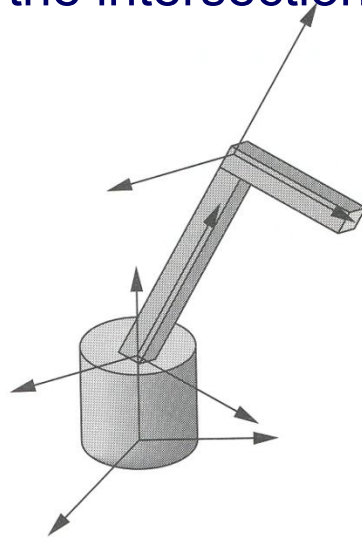




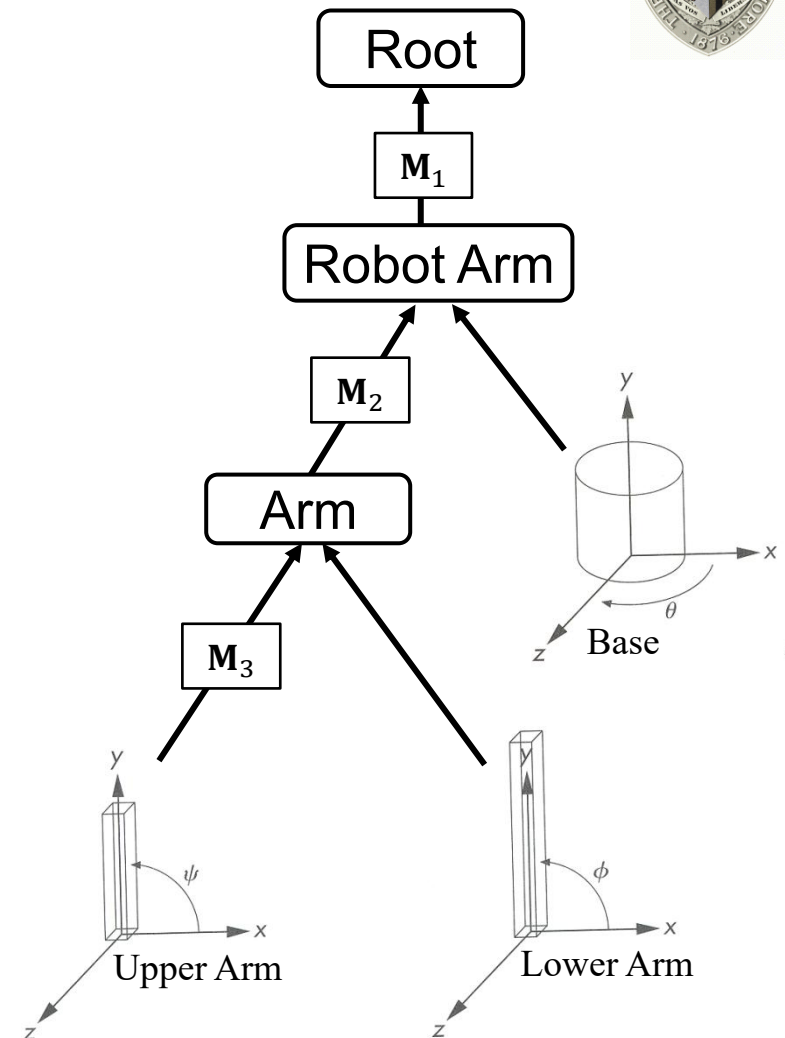
Ray Casting with Hierarchies

If there is an intersection:

- **Base:** Apply M_1 to the intersection information
- **Lower Arm:** Apply $M_1 \cdot M_2$ to the intersection information
- **Upper Arm:** Apply $M_1 \cdot M_2 \cdot M_3$ to the intersection information



Robot Arm



Angel Figures 8.8 & 8.9



Applying a Transformation

Position

Direction

Normal

$$\begin{array}{c} \text{Affine} \\ \begin{pmatrix} a & b & c & t_x \\ d & e & f & t_y \\ g & h & i & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M} \end{array} = \begin{array}{c} \text{Translate} \\ \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M}_T \end{array} \cdot \begin{array}{c} \text{Linear} \\ \begin{pmatrix} a & b & c & 0 \\ d & e & f & 0 \\ g & h & i & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M}_L \end{array}$$



Applying a Transformation

Position

Apply the full affine transformation:

$$\mathbf{p}' = \mathbf{M} \cdot \mathbf{p} = (\mathbf{M}_T \cdot \mathbf{M}_L) \cdot \mathbf{p}$$

Direction

Normal

$$\begin{array}{c} \text{Affine} \\ \begin{pmatrix} a & b & c & t_x \\ d & e & f & t_y \\ g & h & i & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M} \end{array} = \begin{array}{c} \text{Translate} \\ \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M}_T \end{array} \cdot \begin{array}{c} \text{Linear} \\ \begin{pmatrix} a & b & c & 0 \\ d & e & f & 0 \\ g & h & i & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M}_L \end{array}$$



Applying a Transformation

Position

Direction

Apply just the linear component of the transformation:

$$\vec{v}' = \mathbf{M}_L \cdot \vec{v}$$

Normal

$$\begin{array}{c} \text{Affine} \\ \begin{pmatrix} a & b & c & t_x \\ d & e & f & t_y \\ g & h & i & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M} \end{array} = \begin{array}{c} \text{Translate} \\ \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M}_T \end{array} \cdot \begin{array}{c} \text{Linear} \\ \begin{pmatrix} a & b & c & 0 \\ d & e & f & 0 \\ g & h & i & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M}_L \end{array}$$



Applying a Transformation

Position

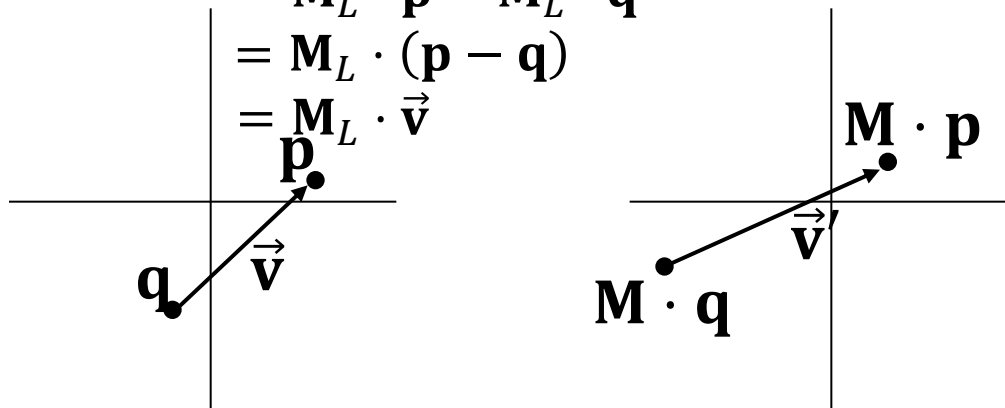
Direction

Apply just the linear component of the transformation:

$$\vec{v}' = \mathbf{M}_L \cdot \vec{v}$$

A direction $\vec{v}$ represents the difference between two positions: $\vec{v} = \mathbf{p} - \mathbf{q}$.
The transformed direction is the difference of transformed positions:

$$\begin{aligned}\vec{v}' &= \mathbf{M} \cdot \mathbf{p} - \mathbf{M} \cdot \mathbf{q} \\ &= (\mathbf{M}_L \cdot \mathbf{p} + \vec{t}) - (\mathbf{M}_L \cdot \mathbf{q} + \vec{t}) \\ &= \mathbf{M}_L \cdot \mathbf{p} - \mathbf{M}_L \cdot \mathbf{q} \\ &= \mathbf{M}_L \cdot (\mathbf{p} - \mathbf{q}) \\ &= \mathbf{M}_L \cdot \vec{v}\end{aligned}$$





Ray Casting With Hierarchies

Iterating from the root, transform rays:

world $\rightarrow$ model coords:

Apply inverse transform to ray

model coordinates coordinates:

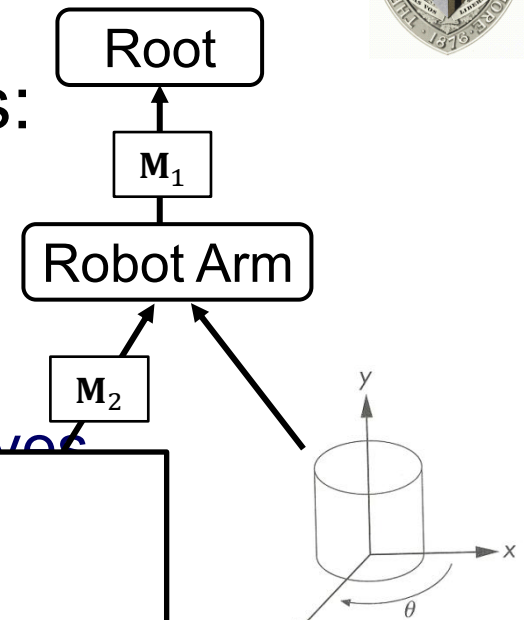
Intersect transformed ray with primitives

model

Apply

Ray Transformation:

$$(\mathbf{p}, \vec{\mathbf{v}}) \rightarrow (\mathbf{M}^{-1} \cdot \mathbf{p}, \mathbf{M}_L^{-1} \cdot \vec{\mathbf{v}})$$



Note:

- When the ray direction has unit-length, time travelled along the ray equals distance travelled along the ray.

[Caution] Even if the original ray direction, $\vec{\mathbf{v}}$, was unit-length, the transformed ray direction, $\mathbf{M}_L^{-1} \cdot \vec{\mathbf{v}}$, may not be.

Robot Arm

Angel Figures 8.8 & 8.9



Applying a Transformation

Position

Direction

Normal

$$\vec{\mathbf{n}}' = ?$$

$$\begin{array}{c} \text{Affine} \\ \begin{pmatrix} a & b & c & t_x \\ d & e & f & t_y \\ g & h & i & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M} \end{array} = \begin{array}{c} \text{Translate} \\ \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M}_T \end{array} \cdot \begin{array}{c} \text{Linear} \\ \begin{pmatrix} a & b & c & 0 \\ d & e & f & 0 \\ g & h & i & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M}_L \end{array}$$



Normal Transformation

Key Idea:

Normals describe a (perpendicularity) relationship to directions, not the directions themselves.

⇒ To transform a normal, transform the relationship.



Normal Transformation

2D Motivating Example:

$$\begin{matrix} & \text{Translate} & \text{Scale} \\ \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 1 \\ 0 & 0 & 1 \end{pmatrix} & = & \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M} & & \mathbf{M}_T \quad \mathbf{M}_L \end{matrix}$$

If $\vec{v}$ is a direction in 2D, and $\vec{n}$ is perpendicular to $\vec{v}$, we want the transformed $\vec{n}$ to be perpendicular to the transformed $\vec{v}$:

$$\begin{aligned} \langle \vec{v}, \vec{n} \rangle = 0 & \Rightarrow \langle \vec{v}', \vec{n}' \rangle = 0 \\ & \Updownarrow \\ \langle \vec{v}, \vec{n} \rangle = 0 & \Rightarrow \langle \mathbf{M}_L \cdot \vec{v}, \vec{n}' \rangle = 0 \end{aligned}$$

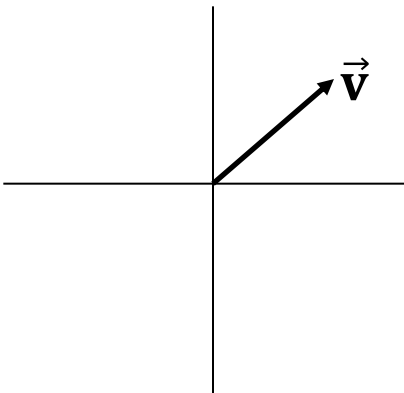


Normal Transformation

2D Motivating Example:

$$\begin{matrix} & \text{Translate} & \text{Scale} \\ \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 1 \\ 0 & 0 & 1 \end{pmatrix} & = & \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M} & & \mathbf{M}_T \quad \mathbf{M}_L \end{matrix}$$

Say $\vec{v} = (2,2)$



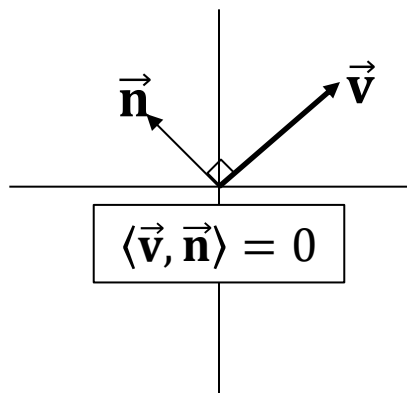


Normal Transformation

2D Motivating Example:

$$\begin{matrix} & \text{Translate} & \text{Scale} \\ \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 1 \\ 0 & 0 & 1 \end{pmatrix} & = & \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M} & \mathbf{M}_T & \mathbf{M}_L \end{matrix}$$

Say $\vec{v} = (2,2)$... then $\vec{n} = (-\sqrt{.5}, \sqrt{.5})$





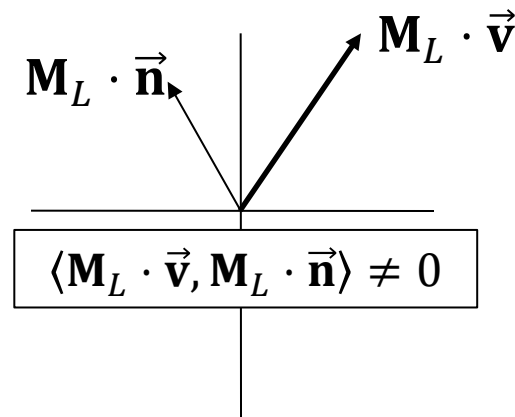
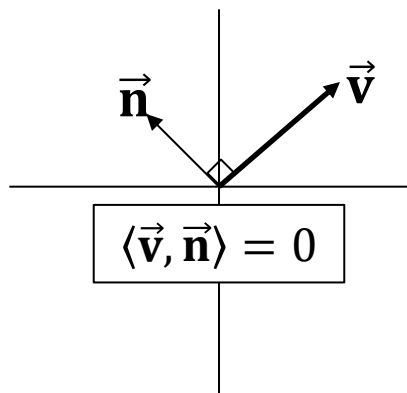
Normal Transformation

2D Motivating Example:

$$\begin{matrix} & \text{Translate} & \text{Scale} \\ \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 1 \\ 0 & 0 & 1 \end{pmatrix} & = & \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M} & & \mathbf{M}_T \quad \mathbf{M}_L \end{matrix}$$

Say $\vec{v} = (2,2)$... then $\vec{n} = (-\sqrt{.5}, \sqrt{.5})$

Transforming $\mathbf{M}_L \cdot \vec{v} = (2,4)$ and $\mathbf{M}_L \cdot \vec{n} = (-\sqrt{.5}, \sqrt{2})$





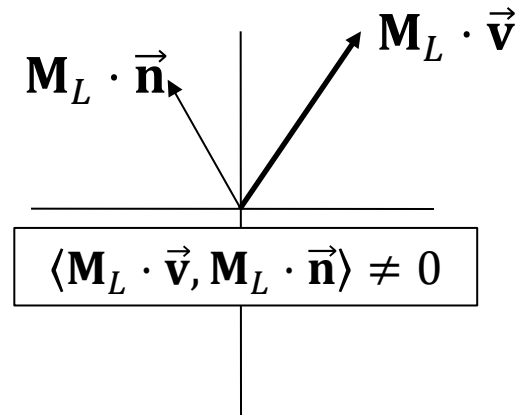
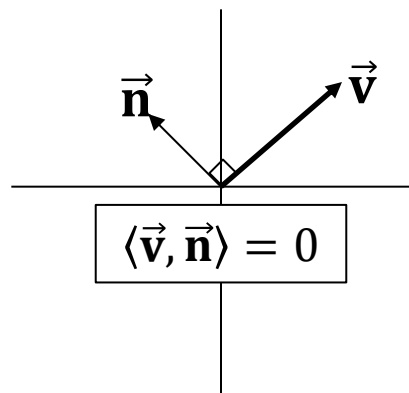
Normal Transformation

2D Motivating Example:

$$\begin{matrix} & \text{Translate} & \text{Scale} \\ \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 1 \\ 0 & 0 & 1 \end{pmatrix} & = & \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M} & & \mathbf{M}_T \quad \mathbf{M}_L \end{matrix}$$

Say $\vec{v} = (2, 2)$... then $\vec{n} = (-\sqrt{.5}, \sqrt{.5})$

Transforming $\vec{n}$ as a direction does not give a vector perpendicular to the transformed $\vec{v}$! $(\sqrt{.5}, \sqrt{2})$





Recall

Transposes:

The transpose of a matrix $\mathbf{M}$ is the matrix $\mathbf{M}^T$ whose (i, j) -th coeff. is the (j, i) -th coeff. of $\mathbf{M}$:

$$\mathbf{M} = \begin{pmatrix} m_{11} & m_{21} & m_{31} \\ m_{12} & m_{22} & m_{32} \\ m_{13} & m_{23} & m_{33} \end{pmatrix} \quad \mathbf{M}^T = \begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix}$$

Recall:

For matrix $\mathbf{M}$, the transpose of the transpose is $\mathbf{M}$:

$$(\mathbf{M}^T)^T = \mathbf{M}$$



Recall

Transposes:

The transpose of a matrix $\mathbf{M}$ is the matrix $\mathbf{M}^T$ whose (i, j) -th coeff. is the (j, i) -th coeff. of $\mathbf{M}$:

$$\mathbf{M} = \begin{pmatrix} m_{11} & m_{21} & m_{31} \\ m_{12} & m_{22} & m_{32} \\ m_{13} & m_{23} & m_{33} \end{pmatrix} \quad \mathbf{M}^T = \begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix}$$

Recall:

For matrices $\mathbf{M}$ and $\mathbf{N}$, the transpose of the product is the reversed product of the transposes:

$$(\mathbf{M} \cdot \mathbf{N})^T = \mathbf{N}^T \cdot \mathbf{M}^T$$



Recall

Dot-Products:

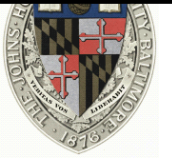
The dot product of two vectors $\vec{v} = (v_x, v_y, v_z)^T$ and $\vec{w} = (w_x, w_y, w_z)^T$ is obtained by summing the product of the coefficients:

$$\langle \vec{v}, \vec{w} \rangle = v_x \cdot w_x + v_y \cdot w_y + v_z \cdot w_z$$

We can also express this as a matrix product:

$$\langle \vec{v}, \vec{w} \rangle = \vec{v}^T \cdot \vec{w} = (v_x \quad v_y \quad v_z) \cdot \begin{pmatrix} w_x \\ w_y \\ w_z \end{pmatrix}$$

$$\langle \vec{v}, \vec{w} \rangle = \vec{v}^T \cdot \vec{w}$$



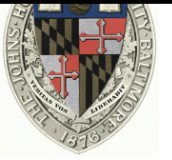
Recall

Transposes and Dot-Products:

If $\mathbf{M}$ is a matrix, and $\vec{v}$ and $\vec{w}$ are vectors, then:

$$\begin{aligned}\langle \vec{v}, \mathbf{M} \cdot \vec{w} \rangle &= \vec{v}^T \cdot (\mathbf{M} \cdot \vec{w}) \\ &= (\vec{v}^T \cdot \mathbf{M}) \cdot \vec{w} \\ &= ((\vec{v}^T \cdot \mathbf{M})^T)^T \cdot \vec{w} \\ &= (\mathbf{M}^T \cdot \vec{v})^T \cdot \vec{w} \\ &= \langle \mathbf{M}^T \vec{v}, \vec{w} \rangle\end{aligned}$$

$$\langle \vec{v}, \vec{w} \rangle = \vec{v}^T \cdot \vec{w}$$



Recall

Example: Orthogonal Matrices

A matrix $\mathbf{R} \in \mathbb{R}^{3 \times 3}$ is **orthogonal** if it preserves the dot-product:

$$\langle \mathbf{R} \cdot \vec{v}, \mathbf{R} \cdot \vec{w} \rangle = \langle \vec{v}, \vec{w} \rangle \quad \forall \vec{v}, \vec{w} \in \mathbb{R}^3$$

Expressing the inner-product as a matrix product:

$$\vec{v}^T \cdot \mathbf{R}^T \cdot \mathbf{R} \cdot \vec{w} = \vec{v}^T \cdot \vec{w} \quad \forall \vec{v}, \vec{w} \in \mathbb{R}^3$$

$$\Leftrightarrow$$

$$\mathbf{R}^T \cdot \mathbf{R} = \mathbf{id.}$$

$$\Leftrightarrow$$

$$\mathbf{R}^T = \mathbf{R}^{-1}$$

$$\langle \vec{v}, \mathbf{M} \cdot \vec{w} \rangle = \langle \mathbf{M}^T \cdot \vec{v}, \vec{w} \rangle$$



Normal Transformation

A normal $\vec{n}$ is defined by having a fixed (zero) dot-product with some direction vector(s) $\vec{v}$.

We need the dot-product of the transformed normal $\vec{n}'$ with the transformed direction(s) to stay fixed:

$$\begin{aligned} \langle \vec{n}, \vec{v} \rangle &= \langle \vec{n}', \mathbf{M}_L \cdot \vec{v} \rangle \\ &= \langle \mathbf{M}_L^T \cdot \vec{n}', \vec{v} \rangle \end{aligned}$$

$$\Updownarrow$$

$$\vec{n} = \mathbf{M}_L^T \cdot \vec{n}'$$

$$\Updownarrow$$

$$\vec{n}' = (\mathbf{M}_L^T)^{-1} \cdot \vec{n}$$

If the linear transformation $\mathbf{M}_L$ is orthogonal (i.e. no scaling) then $(\mathbf{M}_L^T)^{-1} = (\mathbf{M}_L^{-1})^{-1} = \mathbf{M}_L$.



Applying a Transformation

Position

$$\mathbf{p}' = \mathbf{M} \cdot \mathbf{p}$$

Direction

$$\vec{\mathbf{v}}' = \mathbf{M}_L \cdot \vec{\mathbf{v}}$$

Normal

$$\vec{\mathbf{n}}' = \mathbf{M}_L^{-\top} \cdot \vec{\mathbf{n}}$$

$$\begin{array}{c} \text{Affine} \\ \begin{pmatrix} a & b & c & t_x \\ d & e & f & t_y \\ g & h & i & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M} \end{array} = \begin{array}{c} \text{Translate} \\ \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M}_T \end{array} \cdot \begin{array}{c} \text{Linear} \\ \begin{pmatrix} a & b & c & 0 \\ d & e & f & 0 \\ g & h & i & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \mathbf{M}_L \end{array}$$



Ray Casting With Hierarchies

Iterating from the root, transform rays:

world $\rightarrow$ model coordinates:

Apply inverse transform to ray

model coordinates:

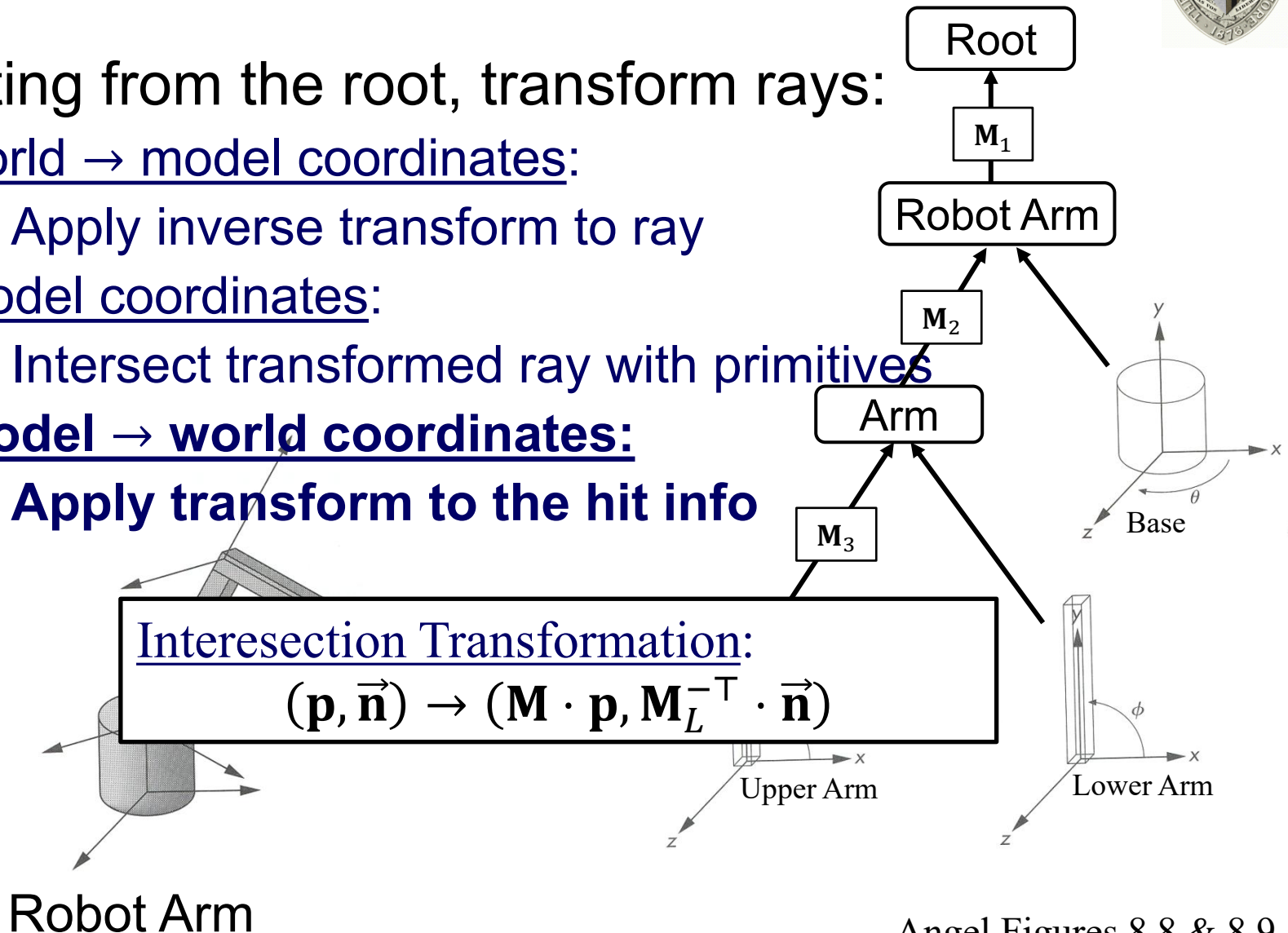
Intersect transformed ray with primitives

model $\rightarrow$ world coordinates:

Apply transform to the hit info

Intersection Transformation:

$$(\mathbf{p}, \vec{\mathbf{n}}) \rightarrow (\mathbf{M} \cdot \mathbf{p}, \mathbf{M}_L^{-T} \cdot \vec{\mathbf{n}})$$





Overview

Transformation Hierarchies

- Scene graphs
- Ray casting

Barycentric Coordinates

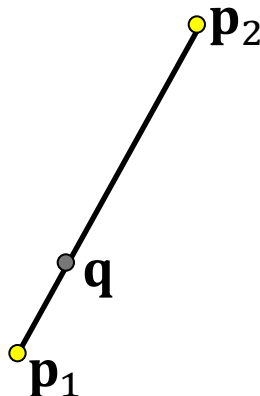


Barycentric Coordinates

Recall:

Given vertices $\mathbf{p}_1$ and $\mathbf{p}_2$, a point $\mathbf{q}$ on the *line segment* between the vertices is the (non-negatively) weighted average of $\mathbf{p}_1$ and $\mathbf{p}_2$:

$$LS = \{\alpha \mathbf{p}_1 + \beta \mathbf{p}_2 \mid \alpha + \beta = 1, \alpha, \beta \geq 0\}$$



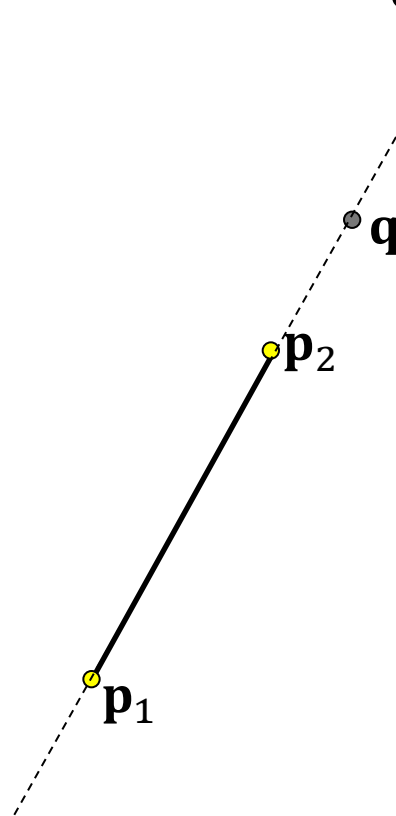
$$\begin{array}{l} \alpha + \beta = 1 \\ \alpha, \beta \geq 0 \end{array}$$



Barycentric Coordinates

Recall:

If the weights sum to one but are not positive, $\mathbf{q}$ is on the line but not on the segment (extrapolation)



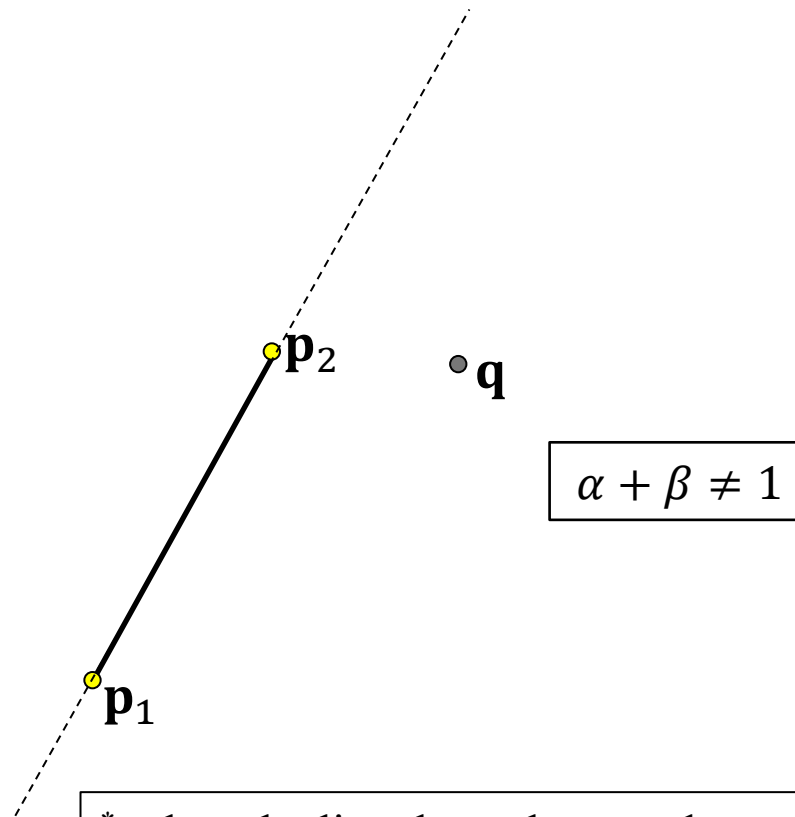
$$\begin{aligned}\alpha + \beta &= 1 \\ \alpha &< 0\end{aligned}$$



Barycentric Coordinates

Recall:

If the weights do not sum to one, $\mathbf{q}$ will not (in general*) be on the line



*unless the line through $\mathbf{p}_1$ and $\mathbf{p}_2$ passes through the origin

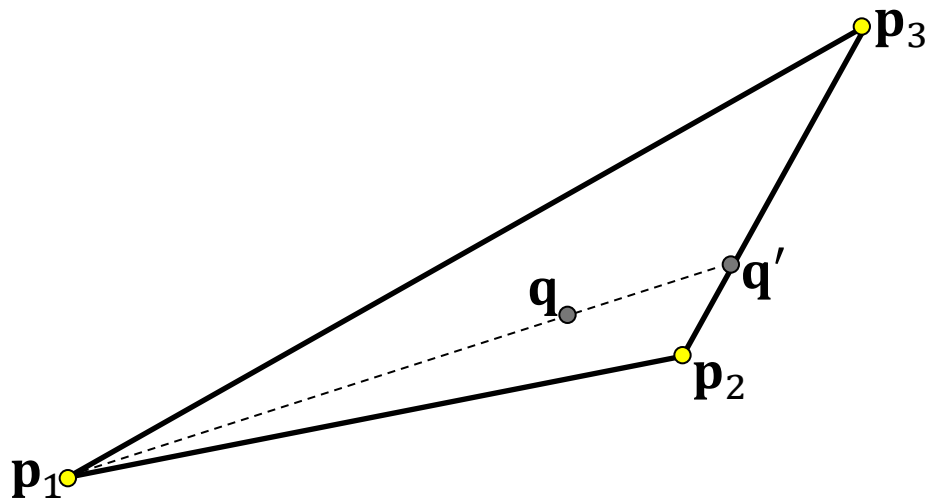


Barycentric Coordinates

A triangle is defined by three (non-collinear) vertices

Note:

A point $\mathbf{q}$ is inside the triangle if and only if $\mathbf{q}$ is on the line segment between (w.l.o.g.*) $\mathbf{p}_1$ and a point $\mathbf{q}'$ on the opposite edge $\overline{\mathbf{p}_2\mathbf{p}_3}$.



*w.l.o.g. $\equiv$ without loss of generality



Inside Triangle $\Rightarrow$ Positive Weights

Claim:

A point $\mathbf{q}$ inside the triangle, iff.* it be expressed as:

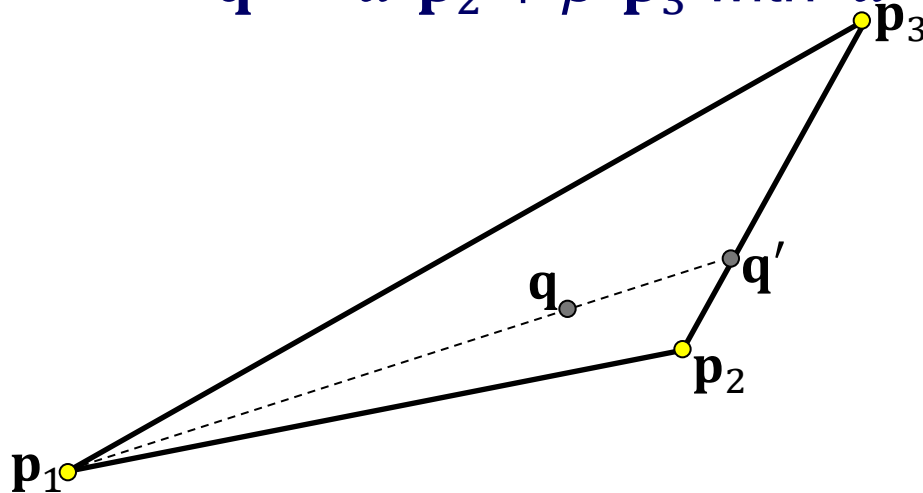
$$\mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{p}_2 + \gamma \mathbf{p}_3 \quad \text{with} \quad \alpha + \beta + \gamma = 1 \quad \text{and} \quad \alpha, \beta, \gamma \geq 0$$

Proof:

The point $\mathbf{q}$ is between $\mathbf{p}_1$ and $\mathbf{q}'$ on edge $\overline{\mathbf{p}_2\mathbf{p}_3}$:

$$\Rightarrow \mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{q}' \quad \text{with} \quad \alpha + \beta = 1 \quad \text{and} \quad \alpha, \beta \geq 0$$

$$\Rightarrow \mathbf{q}' = \alpha' \mathbf{p}_2 + \beta' \mathbf{p}_3 \quad \text{with} \quad \alpha' + \beta' = 1 \quad \text{and} \quad \alpha', \beta' \geq 0$$



*iff. $\equiv$ if and only if

Inside Triangle $\Rightarrow$ Positive Weights



Claim:

Any point $\mathbf{q}$ inside the triangle, can be expressed as:

$$\mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{p}_2 + \gamma \mathbf{p}_3 \quad \text{with} \quad \alpha + \beta + \gamma = 1 \quad \text{and} \quad \alpha, \beta, \gamma \geq 0$$

Proof:

The point $\mathbf{q}$ is between $\mathbf{p}_1$ and $\mathbf{q}'$ on edge $\overline{\mathbf{p}_2 \mathbf{p}_3}$:

$$\Rightarrow \mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{q}' \quad \text{with} \quad \alpha + \beta = 1 \quad \text{and} \quad \alpha, \beta \geq 0$$

$$\Rightarrow \mathbf{q}' = \alpha' \mathbf{p}_2 + \beta' \mathbf{p}_3 \quad \text{with} \quad \alpha' + \beta' = 1 \quad \text{and} \quad \alpha', \beta' \geq 0$$

This gives $\mathbf{q}$ as the linear sum:

$$\circ \quad \mathbf{q} = \alpha \mathbf{p}_1 + (\beta \alpha') \mathbf{p}_2 + (\beta \beta') \mathbf{p}_3$$

Inside Triangle $\Rightarrow$ Positive Weights



Claim:

Any point $\mathbf{q}$ inside the triangle, can be expressed as:

$$\mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{p}_2 + \gamma \mathbf{p}_3 \quad \text{with} \quad \alpha + \beta + \gamma = 1 \quad \text{and} \quad \alpha, \beta, \gamma \geq 0$$

Proof:

The point $\mathbf{q}$ is between $\mathbf{p}_1$ and $\mathbf{q}'$ on edge $\overline{\mathbf{p}_2 \mathbf{p}_3}$:

$$\Rightarrow \mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{q}' \quad \text{with} \quad \alpha + \beta = 1 \quad \text{and} \quad \alpha, \beta \geq 0$$

$$\Rightarrow \mathbf{q}' = \alpha' \mathbf{p}_2 + \beta' \mathbf{p}_3 \quad \text{with} \quad \alpha' + \beta' = 1 \quad \text{and} \quad \alpha', \beta' \geq 0$$

This gives $\mathbf{q}$ as the linear sum:

$$\circ \quad \mathbf{q} = \alpha \mathbf{p}_1 + (\beta \alpha') \mathbf{p}_2 + (\beta \beta') \mathbf{p}_3$$

» Positive weights: $\alpha, \beta \alpha', \beta \beta' \geq 0$

Inside Triangle $\Rightarrow$ Positive Weights



Claim:

Any point $\mathbf{q}$ inside the triangle, can be expressed as:

$$\mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{p}_2 + \gamma \mathbf{p}_3 \quad \text{with} \quad \alpha + \beta + \gamma = 1 \quad \text{and} \quad \alpha, \beta, \gamma \geq 0$$

Proof:

The point $\mathbf{q}$ is between $\mathbf{p}_1$ and $\mathbf{q}'$ on edge $\overline{\mathbf{p}_2 \mathbf{p}_3}$:

$$\Rightarrow \mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{q}' \quad \text{with} \quad \alpha + \beta = 1 \quad \text{and} \quad \alpha, \beta \geq 0$$

$$\Rightarrow \mathbf{q}' = \alpha' \mathbf{p}_2 + \beta' \mathbf{p}_3 \quad \text{with} \quad \alpha' + \beta' = 1 \quad \text{and} \quad \alpha', \beta' \geq 0$$

This gives $\mathbf{q}$ as the linear sum:

- $\mathbf{q} = \alpha \mathbf{p}_1 + (\beta \alpha') \mathbf{p}_2 + (\beta \beta') \mathbf{p}_3$

- » Positive weights: $\alpha, \beta \alpha', \beta \beta' \geq 0$

- » Summing to one:

$$\alpha + \beta \alpha' + \beta \beta'$$

Inside Triangle $\Rightarrow$ Positive Weights



Claim:

Any point $\mathbf{q}$ inside the triangle, can be expressed as:

$$\mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{p}_2 + \gamma \mathbf{p}_3 \quad \text{with} \quad \alpha + \beta + \gamma = 1 \quad \text{and} \quad \alpha, \beta, \gamma \geq 0$$

Proof:

The point $\mathbf{q}$ is between $\mathbf{p}_1$ and $\mathbf{q}'$ on edge $\overline{\mathbf{p}_2 \mathbf{p}_3}$:

$$\Rightarrow \mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{q}' \quad \text{with} \quad \alpha + \beta = 1 \quad \text{and} \quad \alpha, \beta \geq 0$$

$$\Rightarrow \mathbf{q}' = \alpha' \mathbf{p}_2 + \beta' \mathbf{p}_3 \quad \text{with} \quad \alpha' + \beta' = 1 \quad \text{and} \quad \alpha', \beta' \geq 0$$

This gives $\mathbf{q}$ as the linear sum:

- $\mathbf{q} = \alpha \mathbf{p}_1 + (\beta \alpha') \mathbf{p}_2 + (\beta \beta') \mathbf{p}_3$

- » Positive weights: $\alpha, \beta \alpha', \beta \beta' \geq 0$

- » Summing to one:

$$\alpha + \beta \alpha' + \beta \beta' = \alpha + \beta(\alpha' + \beta')$$

Inside Triangle $\Rightarrow$ Positive Weights



Claim:

Any point $\mathbf{q}$ inside the triangle, can be expressed as:

$$\mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{p}_2 + \gamma \mathbf{p}_3 \quad \text{with} \quad \alpha + \beta + \gamma = 1 \quad \text{and} \quad \alpha, \beta, \gamma \geq 0$$

Proof:

The point $\mathbf{q}$ is between $\mathbf{p}_1$ and $\mathbf{q}'$ on edge $\overline{\mathbf{p}_2 \mathbf{p}_3}$:

$$\Rightarrow \mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{q}' \quad \text{with} \quad \alpha + \beta = 1 \quad \text{and} \quad \alpha, \beta \geq 0$$

$$\Rightarrow \mathbf{q}' = \alpha' \mathbf{p}_2 + \beta' \mathbf{p}_3 \quad \text{with} \quad \boxed{\alpha' + \beta' = 1} \quad \text{and} \quad \alpha', \beta' \geq 0$$

This gives $\mathbf{q}$ as the linear sum:

- $\mathbf{q} = \alpha \mathbf{p}_1 + (\beta \alpha') \mathbf{p}_2 + (\beta \beta') \mathbf{p}_3$

- » Positive weights: $\alpha, \beta \alpha', \beta \beta' \geq 0$

- » Summing to one:

$$\alpha + \beta \alpha' + \beta \beta' = \alpha + \beta(\alpha' + \beta') = \alpha + \beta$$

Inside Triangle $\Rightarrow$ Positive Weights



Claim:

Any point $\mathbf{q}$ inside the triangle, can be expressed as:

$$\mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{p}_2 + \gamma \mathbf{p}_3 \quad \text{with} \quad \alpha + \beta + \gamma = 1 \quad \text{and} \quad \alpha, \beta, \gamma \geq 0$$

Proof:

The point $\mathbf{q}$ is between $\mathbf{p}_1$ and $\mathbf{q}'$ on edge $\overline{\mathbf{p}_2 \mathbf{p}_3}$:

$$\Rightarrow \mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{q}' \quad \text{with} \quad \boxed{\alpha + \beta = 1} \quad \text{and} \quad \alpha, \beta \geq 0$$

$$\Rightarrow \mathbf{q}' = \alpha' \mathbf{p}_2 + \beta' \mathbf{p}_3 \quad \text{with} \quad \alpha' + \beta' = 1 \quad \text{and} \quad \alpha', \beta' \geq 0$$

This gives $\mathbf{q}$ as the linear sum:

- $\mathbf{q} = \alpha \mathbf{p}_1 + (\beta \alpha') \mathbf{p}_2 + (\beta \beta') \mathbf{p}_3$

- » Positive weights: $\alpha, \beta \alpha', \beta \beta' \geq 0$

- » Summing to one:

$$\alpha + \beta \alpha' + \beta \beta' = \alpha + \beta(\alpha' + \beta') = \alpha + \beta = 1$$



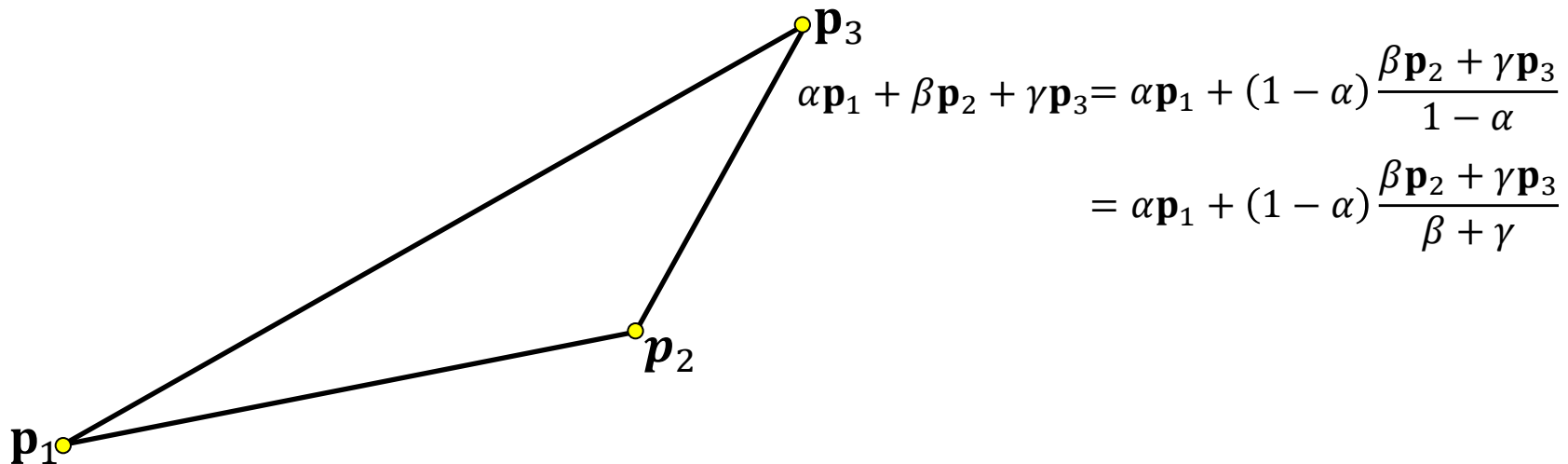
Inside Triangle $\Leftarrow$ Positive Weights

Claim:

If a point $\mathbf{q}$ can be expressed as:

$\mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{p}_2 + \gamma \mathbf{p}_3$ with $\alpha + \beta + \gamma = 1$ and $\alpha, \beta, \gamma \geq 0$ then $\mathbf{q}$ is in the triangle.

Proof:





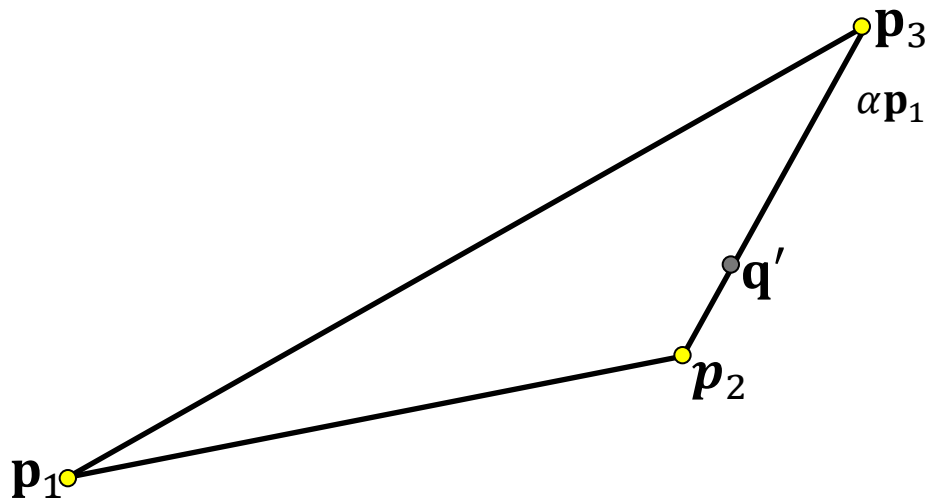
Inside Triangle $\Leftarrow$ Positive Weights

Claim:

If a point $\mathbf{q}$ can be expressed as:

$\mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{p}_2 + \gamma \mathbf{p}_3$ with $\alpha + \beta + \gamma = 1$ and $\alpha, \beta, \gamma \geq 0$
then $\mathbf{q}$ is in the triangle.

Proof:



$$\alpha \mathbf{p}_1 + \beta \mathbf{p}_2 + \gamma \mathbf{p}_3 = \alpha \mathbf{p}_1 + (1 - \alpha) \frac{\beta \mathbf{p}_2 + \gamma \mathbf{p}_3}{1 - \alpha}$$

$$= \alpha \mathbf{p}_1 + (1 - \alpha) \boxed{\frac{\beta \mathbf{p}_2 + \gamma \mathbf{p}_3}{\beta + \gamma}}$$

A point $\mathbf{q}'$ on the segment
between $\mathbf{p}_2$ and $\mathbf{p}_3$



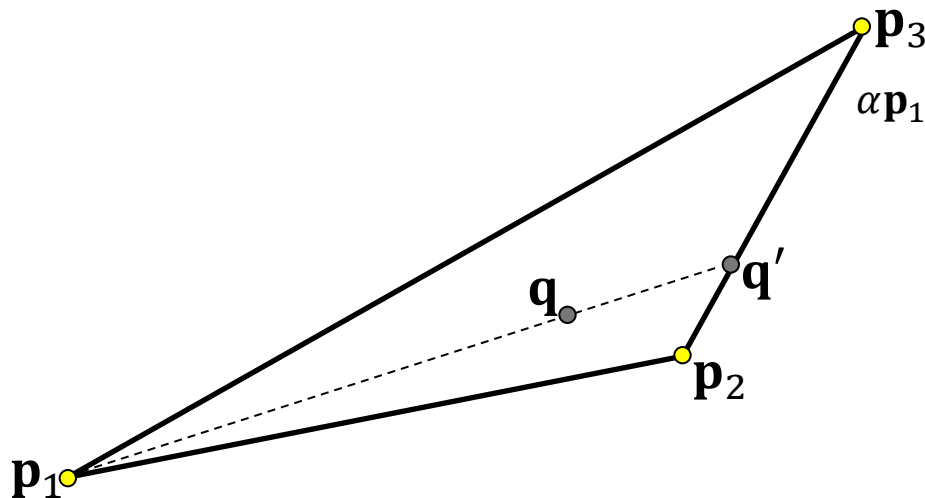
Inside Triangle $\Leftarrow$ Positive Weights

Claim:

If a point $\mathbf{q}$ can be expressed as:

$\mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{p}_2 + \gamma \mathbf{p}_3$ with $\alpha + \beta + \gamma = 1$ and $\alpha, \beta, \gamma \geq 0$
then $\mathbf{q}$ is in the triangle.

Proof:



$$\alpha \mathbf{p}_1 + \beta \mathbf{p}_2 + \gamma \mathbf{p}_3 = \alpha \mathbf{p}_1 + (1 - \alpha) \frac{\beta \mathbf{p}_2 + \gamma \mathbf{p}_3}{1 - \alpha}$$

$$= \alpha \mathbf{p}_1 + (1 - \alpha) \frac{\beta \mathbf{p}_2 + \gamma \mathbf{p}_3}{\beta + \gamma}$$

A point $\mathbf{q}$ on the segment
between $\mathbf{p}_1$ and $\mathbf{q}'$



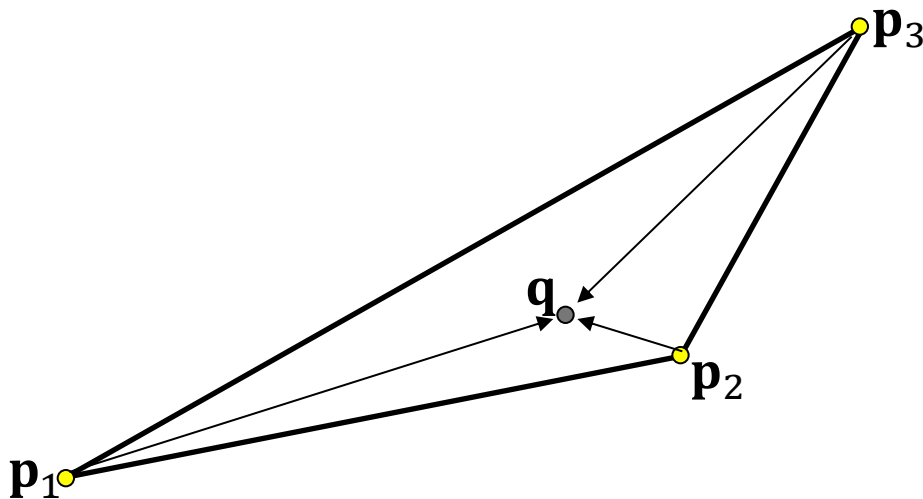
Barycentric Coordinates

The barycentric coordinates of a point $\mathbf{q}$:

$$\mathbf{q} = \alpha \mathbf{p}_1 + \beta \mathbf{p}_2 + \gamma \mathbf{p}_3$$

let us express $\mathbf{q}$ as the weighted average of the triangle vertices.

The weights α , β , and γ tell us, relatively, how close the point $\mathbf{q}$ is to each of $\mathbf{p}_1$, $\mathbf{p}_2$, and $\mathbf{p}_3$.





Barycentric Coordinates

Barycentric coordinates are needed in:

Ray-tracing, to test for intersection

Rendering, to interpolate triangle information



Barycentric Coordinates

Barycentric coordinates are needed in:

Ray-tracing, to test for intersection

Rendering, to interpolate triangle information

```
Float TriangleIntersect( Ray r , Triangle tgl )
{
    Plane p = PlaneContaining( tgl );
    float t = IntersectionDistance( r, p );
    if( t < 0 ) return  $\infty$ ;
    else
    {
        (  $\alpha$  ,  $\beta$  ,  $\gamma$  ) = BarycentricCoordinates( r(t) , tgl );
        if(  $\alpha$  < 0 or  $\beta$  < 0 or  $\gamma$  < 0 ) return  $\infty$ ;
        else return t;
    }
}
```



Barycentric Coordinates

Barycentric coordinates are needed in:

Ray-tracing, to test for intersection

Rendering, to interpolate triangle information

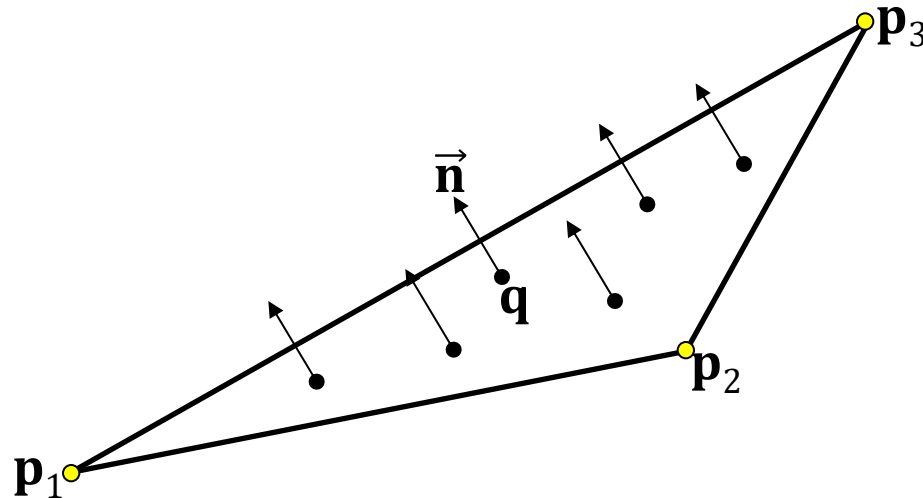
Information (e.g. colors, normals, etc.) is typically defined at vertices rather than triangles



Barycentric Coordinates

We can associate the same **geometric** normal to every point on the face of a triangle by computing:

$$\vec{n} = \frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)}{\|(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)\|}$$





Barycentric Coordinates

We can associate the same **geometric** normal to every point on the face of a triangle by computing:

$$\vec{n} = \frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)}{\|(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)\|}$$



Results in flat shading
across the faces

Triangle Normals



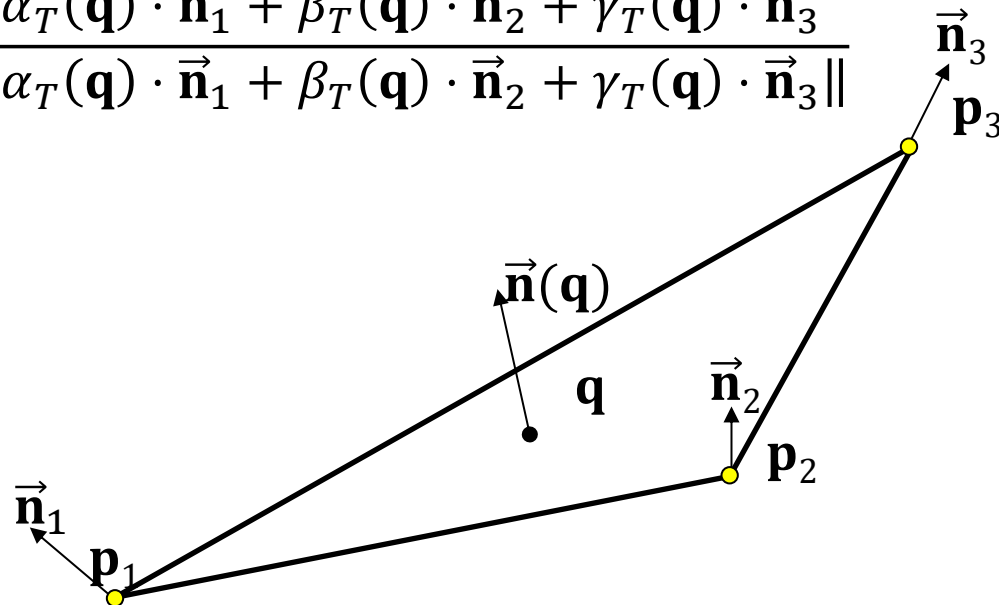
Barycentric Coordinates

Instead, we can associate different **rendering** normals to the vertices:

$$T = ((\mathbf{p}_1, \vec{\mathbf{n}}_1), (\mathbf{p}_2, \vec{\mathbf{n}}_2), (\mathbf{p}_3, \vec{\mathbf{n}}_3))$$

⇒ The normal at a point $\mathbf{q}$ in the triangle is the interpolation of the normals at the vertices:

$$\vec{\mathbf{n}}(\mathbf{q}) = \frac{\alpha_T(\mathbf{q}) \cdot \vec{\mathbf{n}}_1 + \beta_T(\mathbf{q}) \cdot \vec{\mathbf{n}}_2 + \gamma_T(\mathbf{q}) \cdot \vec{\mathbf{n}}_3}{\|\alpha_T(\mathbf{q}) \cdot \vec{\mathbf{n}}_1 + \beta_T(\mathbf{q}) \cdot \vec{\mathbf{n}}_2 + \gamma_T(\mathbf{q}) \cdot \vec{\mathbf{n}}_3\|}$$





Barycentric Coordinates

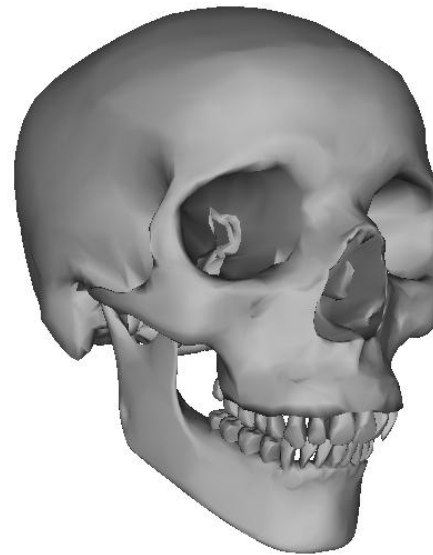
Instead, we can associate different **rendering** normals to the vertices:

$$T = ((\mathbf{p}_1, \vec{\mathbf{n}}_1), (\mathbf{p}_2, \vec{\mathbf{n}}_2), (\mathbf{p}_3, \vec{\mathbf{n}}_3))$$

⇒ The normal we use to render at $\mathbf{q}$ is the interpolation of the normals from the vertices:



Triangle Normals



Interpolated Point Normals



Barycentric Coordinates

Note:

Inst Don't confuse the two normals

- nor
- Geometric normal (for intersections)
 - Rendering normal (for rendering)

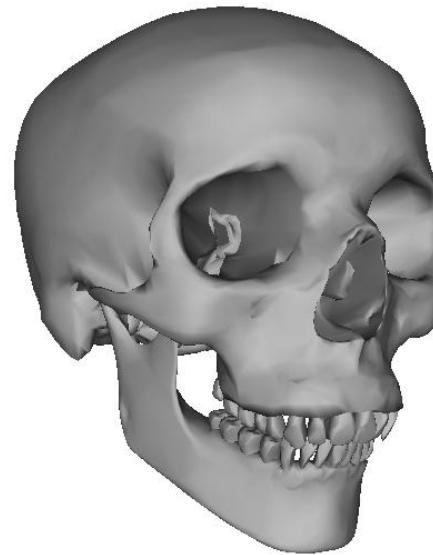
Using a different normal for rendering won't change the geometry.



Silhouettes are unchanged



Triangle Normals



Interpolated Point Normals