Course Plan

- lec. 1: Introduction
- lec. 2: Mathematics for 3D Games
- lec. 3: Scene Graph
- lec. 4: Game 3D Physics +
- lec. 5: Game Particle Systems
- lec. 6: Game 3D Models
- lec. 7: Game Textures
- lec. 9: Game Materials
- lec. 8: Game 3D Animations
- lec. 10: Networking for 3D Games
- lec. 11: 3D Audio for 3D Games
- lec. 12: Rendering Techniques for 3D Games
- lec. 13: Artificial Intelligence for 3D Games

51

Rotation composition? Quaternion multiplication!

$q_0, q_1, p \in \mathbb{H}$
q_0, q_1 represent rotations
p represents a point

p rotated by q1, rotated by q0

p rotated by q1

$$q_0 \cdot (q_1 \cdot p \cdot \bar{q}_1) \cdot \bar{q}_0$$

product is associative (like for complex numbers)

$$=$$

$$(q_0 \cdot q_1) \cdot p \cdot (\bar{q}_1 \cdot \bar{q}_0)$$

$\bar{r} \cdot \bar{s} = \overline{s \cdot r}$ (rules of quaternions) (remember: product is not commutative)

$$=$$

$$(q_0 \cdot q_1) \cdot p \cdot \overline{(q_0 \cdot q_1)}$$

52

3D Rotations as Quaternions

- quaternion $\mathbf{q}$ representing the 3D rotation of angle α around axis $\hat{\mathrm{a}}$:
  - $\mathrm{q} = \left(\sin\left(\frac{\alpha}{2}\right) \hat{\mathrm{a}} \,,\ \cos\left(\frac{\alpha}{2}\right) \right)$

  that is
  - $\mathrm{q} = \sin\left(\frac{\alpha}{2}\right) \hat{\mathrm{a}}_x i + \sin\left(\frac{\alpha}{2}\right) \hat{\mathrm{a}}_y j + \sin\left(\frac{\alpha}{2}\right) \hat{\mathrm{a}}_z k + \cos\left(\frac{\alpha}{2}\right)$
- Observe that $\|\mathrm{q}\|^2 = 1$ ← verify

53

Exercise: are the following quaternions unitary?

- $\mathbf{q}_0 = (0,0,-1,0) = -j$
- $\mathbf{q}_1 = \left(\frac{1}{2},\frac{1}{2},\frac{1}{2},\frac{1}{2}\right) = 0.5i + 0.5j + 0.5k + 0.5$
- $\mathbf{q}_2 = (1,1,1,1) = i + j + k + 1$

54

Quaternions: exercises

- Which quaternion encodes a turnabout?
  - (ita: «*un dietrofront*»: turning 180° around the up vector)
- Apply that quaternion to rotate a point in (x,y,z)
  - Use plain quaternion algebra, and algebraic notation
- Which quaternion encodes the identity rotation?
  - Is it the only one? If not, which other does?
  - Verify by applying it (or them)
- Which quaternion encodes a turn of 90° to the left?
- Uses your previous *two* answers to find the quat. encoding turn 45° to the left, *by using interpolation*
  - Do you need SLERP in this case? Is NLERP enough? Why?
  - Verify that the solution is correct using the axis-angle formula

55

Example: turnabout rotation

- Find the quaternion $\mathbf{r}$ representing the rotation by 180° (π radiants) around axis Y
  - $\hat{a} = (0,1,0)$
  - $\alpha = \pi, \sin\left(\frac{\alpha}{2}\right) = 1, \cos\left(\frac{\alpha}{2}\right) = 0$
  - $\mathbf{r} = (1\,\hat{a}, 0) = 0i + 1j + 0k + 0 = j$

imaginary vector — real scalar

- Find the quaternion $\mathbf{q}$ representing point $\begin{pmatrix}2\\3\\4\end{pmatrix}$:
  - $\mathbf{q} = 2i + 3j + 4k$
- Rotate that point with that rotation:
  - $\mathbf{q}' = \mathbf{r}\,\mathbf{q}\,\bar{\mathbf{r}} = j\,(2i + 3j + 4k)(-j) = \ldots$ *(finish me!)*

56

3D Rotations as Quaternions: equivalent representations ☹

- Around axis $\hat{a}$ by angle α :

$$q = \left(\sin\left(\frac{\alpha}{2}\right)\hat{a} , \cos\left(\frac{\alpha}{2}\right) \right)$$

- Around axis $-\hat{a}$ by angle $(-\alpha)$ (it's the same rotation!) :

$$q' = \left(-\sin\left(\frac{-\alpha}{2}\right)\hat{a} , \cos\left(\frac{-\alpha}{2}\right) \right) = q$$

same quaternion :-)

Nice! But:

- Around axis $\hat{a}$ by angle $(\alpha + 2\pi)$ (again, it's the same rotation!) : [360°]

$$q'' = \left(\sin\left(\frac{\alpha}{2} + \pi\right)\hat{a} , \cos\left(\frac{\alpha}{2} + \pi\right) \right) =$$

[180°]

$$= \left(-\sin\left(\frac{\alpha}{2}\right)\hat{a} , -\cos\left(\frac{\alpha}{2}\right) \right) = -q$$

different quaternion :-(

- Conclusion:
  quaternion q and quaternion $-q$ encode the same rotation

57

3D Rotations as Quaternions: equivalent representations ☹

Given a quaternion representing a rotation:

- Flip its real part: invert rotation
- Flip its imaginary part (conjugate): invert rotation
- Flip everything: same rotation

Every rotation is encoded
by two different quaternions: q and $-q$.

58

Interpolating two quaternions (that represent two rotations)

Good results, but two *caveats*:

⚠ Take the “shortest path” (as usual):
flip 2nd quaternion first, if this makes them closer

- Distance defined as dot product in 4D
(consider quaternions as 4D unit vectors for this)
(remember: dot product between unitary vectors is a measure of similarity!)

⚠ Loss of normality

- Needs re-normalization (NLERP),
- Or SLERP
(again, just consider quaternions as 4D unit vectors)

59

Shortest path interpolation: the case of quaternions

- Let $\mathbf{p}$ and $\mathbf{q}$ be two rotations
- $\mathbf{q}$ and $-\mathbf{q}$ represent the same rotation.
  - Which one to choose?
- Which one is closer to $\mathbf{p}$?
  - Distance between $\mathbf{p}$ and $\mathbf{q}$ = $\mathrm{dot}(\mathbf{p}, \mathbf{q})$
  - Distance between $\mathbf{p}$ and $-\mathbf{q}$ = $\mathrm{dot}(\mathbf{p}, -\mathbf{q}) = -\mathrm{dot}(\mathbf{p}, \mathbf{q})$
- Conclusion:
  - If $\mathrm{dot}(\mathbf{p}, \mathbf{q})$ is positive, interpolate with $\mathbf{q}$
  - Otherwise, interpolate with $-\mathbf{q}$

60

Quaternion Product

×	a *i*	+	b *j*	+	c *k*	+	d
e *i*							
+							
f *j*							
+							
g *k*							
+							
h							

61

Quaternion Product

$\vec{v}$ (a, b, c); $\vec{w}$ (e, f, g)

×	a *i*	+	b *j*	+	c *k*	+	d	
e *i*	-1 ae	+	*k* be	+	-*j* ce	+	*i* de	+
+								
f *j*	-*k* af	+	-1 bf	+	*i* cf	+	*j* df	+
+								
g *k*	*j* ag	+	-*i* bg	+	-1 cg	+	*k* dg	+
+								
h	*i* ah	+	*j* bh	+	*k* ch	+	hd	

$(\vec{w}, h)$

$\cdot$

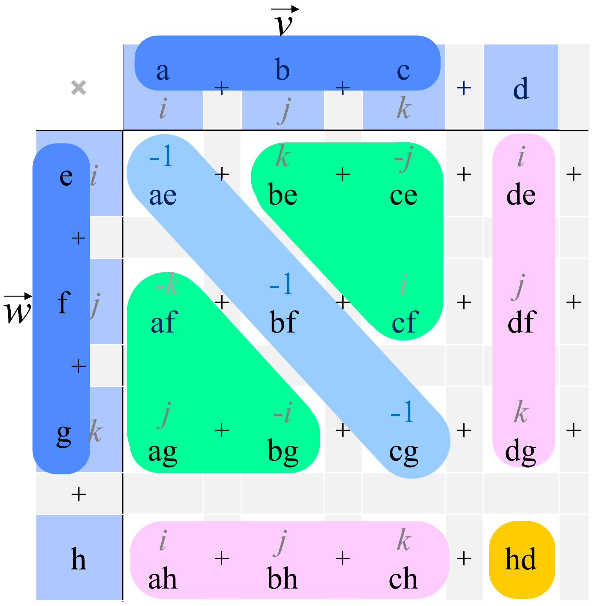
$(\vec{v}, d)$

$=$

(some vector , some scalar)

62

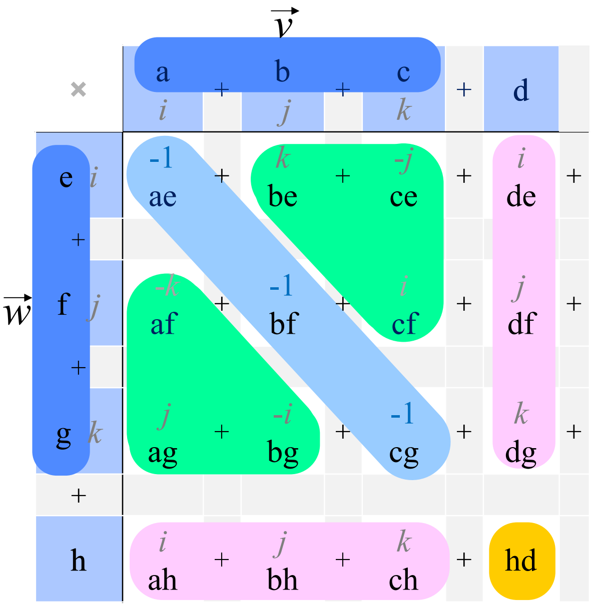
Quaternion Product



$(\vec{w}, h) \cdot (\vec{v}, d) =$ (some vector , some scalar)

63

Quaternion Product



$$(\vec{w}, h) \cdot (\vec{v}, d) = (\vec{w}\,d + \vec{v}\,h + \vec{w} \times \vec{v} \,,\; h\,d - \vec{w} \cdot \vec{v})$$

64

Marco Tarini
Unviersità degli studi di Milano

Exercise: quaternion norm as a quaternion product

- As you may remember,
  given a complex number $\mathbf{c} \in \mathbb{C}$, $\mathbf{c} = a + ib$
  its magnitude $\|\mathbf{c}\| = \sqrt{a^2 + b^2}$
  can be expressed as
  $$\|\mathbf{c}\|^2 = \mathbf{c}\,\bar{\mathbf{c}}$$
- Does the same hold for quaternions?
  Given $\mathbf{q} \in \mathbb{H}$:
  $$\|\mathbf{q}\|^2 = \mathbf{q}\,\bar{\mathbf{q}}$$
- Verify, using the multiplication formula seen above

65

Quaternions as rotations: summary

- Compact to store (4 scalars, almost the minimum)
- Trivial to invert (just conjugate)
- Fast to composite (just multiply)
- Fast to apply
- Easy to enforce it’s still a rotations (just renormalize)
  - Even after long sequences of cumulations, unlike matrices
- Behaves well under interpolation
  - Even with just NLERP – better with SLERP
  - Remember to take the shortest path (flip sign if necessary)
- The favourite representation in 3D games
  - but, other solutions still useful in one context or another

66

Recap: representing rotations 1/2

	3x3 Matrix	Euler Angles
Space efficient? (in RAM, GPU, storage...)	★☆☆☆☆ 9 scalars	★★★★★ 3 scalars (even as small int!)
Efficient / easy to: Apply (to points/vectors)	★★★★☆ 9 products (3 dot products)	★★☆☆☆ requires trigonometry sin/cos
Efficient / easy to: Invert (produce inverse)	★★★★★ just transpose	★☆☆☆☆
Efficient / easy to: Composite (with another rotation)	★★☆☆☆ Matrix multipl (9 dots) Numerical errors	★☆☆☆☆
Efficient / easy to: Interpolate (with another rotation)	★☆☆☆☆ Introduces shear/scale	★☆☆☆☆ easy to do, unintuitive result (⚠ shortest-path required!)
Intuitive? (e.g. to manually set)	★★★☆☆	★★★★★ roll yaw pitch
Notes...	Free extra shear + scale. Useful to extract local axes.	⚠ GIMBAL LOCK

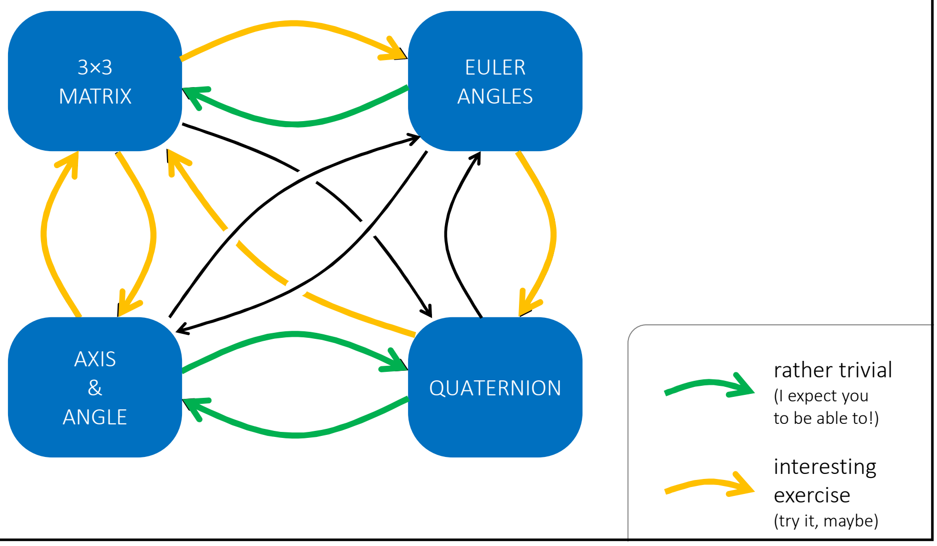
68

Recap: representing rotations 2/2

	axis , angle	(unitary) quaternion
Space efficient? (in RAM, GPU, storage...)	★★★★☆ 4 scalars (or 3) (precision needed)	★★★★☆ 4 scalars (precision needed)
Efficient / easy to: Apply (to points/vectors)	★★★☆☆ Requires trigonometry	★★★★★ Just 2 quat product
Efficient / easy to: Invert (produce inverse)	★★★★★ Just flip axis OR angle	★★★★★ super easy flip imaginary or real part
Efficient / easy to: Composite (with another rotation)	★☆☆☆☆	★★★★★ super easy: 1 quat product
Efficient / easy to: Interpolate (with another rotation)	★★★★★	★★★★☆ easy + good result (NLERP or SLERP)
Intuitive? (e.g. to manually set)	★☆☆☆☆ no	★☆☆☆☆ no
Notes...	⚠ *two* representations for each rotation (flip all → no effect)	(for different reasons) Require shortest path!

69

Switching between representations



72

What defines a rotation, for you?

« Roll, pitch, and yaw! »

then you are… a pilot, or an astronaut

« X-angle, Y-angle, and Z-angle! »

then you are… a digital artist (an animator, or a scener)

« An angle! »

then you are… a flatland citizen

« A vector! the dir is the axis the magnitude the angle »

then you are… a physicist

« A 3x3 matrix! the submatrix of a 4x4 transform »

then you are… a computer graphicist, or a Graphics API

« A quaternion! »

then you are… a game developer 

73

