

# Octonions

(Just kidding)

Slide 1 6.837 Fall 2006

# Quaternions

"Quaternions came from Hamilton after his really good work had been done; and, though beautifully ingenious, have been an unmixed evil to those who have touched them in any way, including Clark Maxwell."

Lord Kelvin, 1892.

Slide 2 6.837 Fall 2006

## Rotation Interpolation

1-, 2-, 3-DOF rotations as constrained points on 1, 2, 3-spheres in 2D, 3D and 4D



Slide 3 6.837 Fall 2006

## What we know about quaternions

2 constructions:

- $(\cos \theta/2; \sin \theta/2 \mathbf{v})$  represents rotation of  $\theta$  around  $\mathbf{v}$
- Extension of complex numbers:  $d+ia+jb+kc$  with 3 roots of -1

Addition, multiplication by scalar

Multiplication:

$$(d; \mathbf{u}) (d'; \mathbf{u}') = (dd' - \mathbf{u} \cdot \mathbf{u}'; d\mathbf{u}' + d'\mathbf{u} + \mathbf{u} \times \mathbf{u}')$$

- Corresponds to rotation composition
- Non commutative, same as rotations

Identity quaternion is  $(1; \mathbf{0})$

The conjugate  $(d; -\mathbf{u})$  is the inverse rotation

- $q^*q = (1; \mathbf{0})$  for unit quaternions

Slide 4 6.837 Fall 2006

## Sanity checks

$$q = (\cos(\theta/2); \mathbf{v} \sin(\theta/2)) = d+ai+bj+ck$$

What does  $i$  mean?

- $(0, 1, 0, 0) = (\cos \pi/2; \sin \pi/2 (1,0,0))$   
→ rotation of  $\pi$  around the x axis

Similarly

- $j$  is a rotation of  $\pi$  around the y axis
- $k$  is a rotation of  $\pi$  around the z axis

Slide 5 6.837 Fall 2006

## Sanity checks

What does  $i^2$  mean?

- Composition of two rotations of  $\pi$  around the x axis
- Rotation of  $2\pi$  around the x axis
- →  $(\cos 2\pi, \sin 2\pi, 0, 0) = (1, 0, 0, 0) = 1$  (identity) but equivalent to -1 because  $q$  is the same rotation as  $-q$

Similarly

- $j^2$  is a rotation of  $2\pi$  around the y axis  
→  $(\cos 2\pi, 0, \sin 2\pi, 0) = (1, 0, 0, 0) = 1$  (identity)
- $k^2$  is a rotation of  $2\pi$  around the z axis  
→  $(\cos 2\pi, 0, 0, \sin 2\pi) = (1, 0, 0, 0) = 1$  (identity)

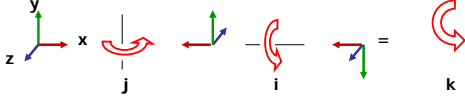
Conclusions: boy, it is complicated that  $q$  and  $-q$  are the same rotation

Slide 6 6.837 Fall 2006

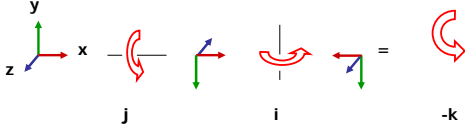
## Sanity checks

What is  $ij$ ?

- Composition rotation of  $\pi$  around x axis and rotation of  $\pi$  around y axis



What is  $ji$ ? (that is going to be confusing)



- The whole  $q$  equivalent to  $-q$  is confusing indeed

Slide 7 6.837 Fall 2006

## So why this $q/-q$ business?

Topology of group of rotations

- But we won't go there

Hint: if we had  $ij=ji$ ;  $jk=kj$ ;  $ik=ki$ , multiplication would be commutative, but we know that rotations composition is not

Slide 8

6.837 Fall 2006

## Other definition, simpler rule

A quaternion is a scalar and a vector

We know how to multiply a scalar by a scalar, and a scalar by vector

New rule: vectors multiply according to

$$\mathbf{uv} = -\mathbf{u} \cdot \mathbf{v} + \mathbf{u} \times \mathbf{v}$$

Slide 9

6.837 Fall 2006

## Question?

Slide 10

6.837 Fall 2006

## What do we need?

What do we want to do with rotations?

→ What we need to learn to do with quaternions

- Compose
- Interpolate
  - Multiply by a scalar
  - Add
  - slerp
- Invert
- Apply to a vector
- Convert back and forth
  - Axis-angle to quaternion
  - Quaternion to matrix
  - Also Matrix, Euler

Slide 11

6.837 Fall 2006

## Apply to a vector solution 1

Convert quaternion to matrix

Apply matrix to vector

$$\begin{pmatrix} 1 - 2b^2 - 2c^2 & 2ab - 2dc & 2ac + 2db \\ 2ab + 2dc & 1 - 2a^2 - 2c^2 & 2yz - 2wx \\ 2ca - 2db & 2bc + 2da & 1 - 2a^2 - 2b^2 \end{pmatrix}$$

Slide 12

6.837 Fall 2006

## Rotating a point: 2D/complex case

We can represent a 2D vector  $(x, y)$  with a complex  $re^{i\alpha}$

Rotation of angle  $\theta$  is the complex  $e^{i\theta}$

Rotation application = complex multiplication

**Warning:** This is not so simple for quaternions & 3D rotations

- For one thing, quaternions are 4D and vectors are 3D

Slide 13 6.837 Fall 2006

## Rotating a point

The vector  $\mathbf{p}$  is represented by the quaternion  $(0, \mathbf{p})$

- Makes sense, it's a pure vector, no real part, a pure axis with 0 rotation

To rotate 3D point/vector  $\mathbf{p}$  by rotation/quaternion  $q$ , compute:

$$q (0; \mathbf{p}) q^{-1}$$

Slide 14 6.837 Fall 2006

## Proof

Axis  $\mathbf{v}$ , point  $\mathbf{p}$  decomposed into  $(\mathbf{u}+\mathbf{z}\mathbf{v})$  where  $\mathbf{u}$  is orthogonal to  $\mathbf{v}$  and  $\mathbf{z}$  is a scalar.  $q = (\cos \theta/2; \sin \theta/2 \mathbf{v}) = (c; \mathbf{sv})$

Recall:  $(d; \mathbf{u}) (d'; \mathbf{u}') = (dd' - \mathbf{u} \cdot \mathbf{u}'; d\mathbf{u}' + d'\mathbf{u} + \mathbf{u} \times \mathbf{u}')$

$$q p q^{-1} = (c; \mathbf{sv}) (0; \mathbf{u} + \mathbf{z}\mathbf{v}) (c; -\mathbf{sv})$$

$$= (c; \mathbf{sv}) (0c + (\mathbf{u} + \mathbf{z}\mathbf{v})\mathbf{sv}; 0(-\mathbf{sv}) + c(\mathbf{u} + \mathbf{z}\mathbf{v}) + (\mathbf{u} + \mathbf{z}\mathbf{v}) \times (-\mathbf{sv}))$$

$$= (c; \mathbf{sv}) (\mathbf{zs}; c\mathbf{u} + \mathbf{z}\mathbf{c}\mathbf{v} - \mathbf{su} \times \mathbf{v})$$

$$= (c\mathbf{zs} - \mathbf{sv}(c\mathbf{u} + \mathbf{z}\mathbf{c}\mathbf{v} - \mathbf{su} \times \mathbf{v}); \text{vector part } tbd)$$

$$= (c\mathbf{zs} - \mathbf{sz}\mathbf{c}\mathbf{v} \cdot \mathbf{v}; \text{vector part } tbd)$$

$$= (0; c(\mathbf{u} + \mathbf{z}\mathbf{c}\mathbf{v} - \mathbf{su} \times \mathbf{v}) + \mathbf{zs}(\mathbf{sv}) + \mathbf{sv} \times (c\mathbf{u} + \mathbf{z}\mathbf{c}\mathbf{v} - \mathbf{su} \times \mathbf{v}))$$

$$= (0; c^2\mathbf{u} + \mathbf{z}\mathbf{c}^2\mathbf{v} - \mathbf{scu} \times \mathbf{v} + \mathbf{zs}^2\mathbf{v} + \mathbf{scv} \times \mathbf{u} - \mathbf{s}^2\mathbf{v} \times \mathbf{u} \times \mathbf{v})$$

$$= (0; \mathbf{z}\mathbf{c}^2\mathbf{v} + \mathbf{z}\mathbf{s}^2\mathbf{v} + 2\mathbf{scv} \times \mathbf{u} + c^2\mathbf{u} - \mathbf{s}^2(\mathbf{v} \cdot \mathbf{v})\mathbf{u} + \mathbf{s}^2(\mathbf{v} \cdot \mathbf{u})\mathbf{v})$$

$$= (0; \mathbf{z}\mathbf{v} + 2\mathbf{scv} \times \mathbf{u} + c^2\mathbf{u} - \mathbf{s}^2\mathbf{u})$$

$$= (0; \mathbf{z}\mathbf{v} + \sin 2\theta/2 \mathbf{v} \times \mathbf{u} + \cos 2\theta/2 \mathbf{u})$$

the component along  $\mathbf{v}$  is unaffected, and in the frame  $(\mathbf{v}, \mathbf{u}, \mathbf{v} \times \mathbf{u})$ , the new coordinates are  $\cos \theta, \sin \theta$ : it's a rotation

Slide 15 6.837 Fall 2006

## Other example (in coordinates)

$$q (0; \mathbf{p}) q^{-1} \quad \& \quad (d; \mathbf{u}) (d'; \mathbf{u}') = (dd' - \mathbf{u} \cdot \mathbf{u}', d\mathbf{u}' + d'\mathbf{u} + \mathbf{u} \times \mathbf{u}')$$

Example:  $\mathbf{p} = (x, y, z)$

$$q = (\cos(\theta/2), 0, 0, \sin(\theta/2)) = (c, 0, 0, s)$$

$$q^{-1} = (\cos(\theta/2), 0, 0, -\sin(\theta/2)) = (c, 0, 0, -s)$$

$$q (0; \mathbf{p}) q^{-1} = (c, 0, 0, s) (0, x, y, z) (c, 0, 0, -s)$$

$$= (c^2 0 - \mathbf{zs}; \mathbf{cp} + 0(0,0,s) + (0,0,s) \times \mathbf{p}) (c, 0, 0, -s)$$

$$= (-\mathbf{zs}; \mathbf{cp} + (-\mathbf{sy}, \mathbf{sx}, 0)) (c, 0, 0, -s)$$

$$= (-\mathbf{zsc}(\mathbf{cp} + (-\mathbf{sy}, \mathbf{sx}, 0)); (0,0,-s); -\mathbf{zs}(0,0,-s)$$

$$+ c(\mathbf{cp} + (-\mathbf{sy}, \mathbf{sx}, 0)) + (\mathbf{cp} + (-\mathbf{sy}, \mathbf{sx}, 0)) \times (0,0,-s))$$

$$= (0; (0,0,\mathbf{zs}^2) + c^2\mathbf{p} + (-\mathbf{csy}, \mathbf{csx}, 0) +$$

$$(-\mathbf{csy}, \mathbf{csx}, 0) + (\mathbf{s}^2\mathbf{x}, \mathbf{s}^2\mathbf{y}, 0))$$

$$= (0, c^2\mathbf{x} - 2\mathbf{csy} - \mathbf{s}^2\mathbf{x}, c^2\mathbf{y} + 2\mathbf{csx} - \mathbf{s}^2\mathbf{y}, \mathbf{zs}^2 + \mathbf{sc}^2))$$

$$= (0, x \cos(q/2) - y \sin(q/2), x \sin(q/2) + y \cos(q/2), z)$$

Slide 16 6.837 Fall 2006

## Joke

<http://mathworld.wolfram.com/CrossProduct.html>

What do you get when you cross an elephant and a grape?"  
The answer is "Elephant grape sine-of-theta."

Slide 17 6.837 Fall 2006

## Questions?

Slide 18 6.837 Fall 2006

## Back to multiplication

Now we can prove that multiplication is composition  
 Consider  $q_0$  and  $q_1$  corresponding to rotation  $R_0$  and  $R_1$   
 Let's apply the product  $q_0q_1$  to a point  $p$

$$\begin{aligned} (q_0q_1) p & (q_0q_1)^{-1} \\ &= (q_0q_1) p (q_0q_1)^{-1} \\ &= q_0q_1 p q_1^{-1}q_0^{-1} \\ &= q_0 (q_1 p q_1^{-1})q_0^{-1} \end{aligned}$$

That is, inside the parenthesis we rotate by  $q_1$  and outside we rotate by  $q_0$ ; this is the composition!

Slide 19

6.837 Fall 2006

## What we know about quaternions

2 constructions:

- $(\cos \theta/2; \sin \theta/2 \mathbf{v})$  represents rotation of  $\theta$  around  $\mathbf{v}$
- Extension of complex numbers:  $d+ia+jb+kc$  with 3 roots of -1

Addition, multiplication by scalar

Multiplication:  $(d; \mathbf{u}) (d'; \mathbf{u}') = (dd' - \mathbf{u} \cdot \mathbf{u}'; d\mathbf{u}' + d'\mathbf{u} + \mathbf{u} \times \mathbf{u}')$

- Corresponds to rotation composition
- Non commutative, same as rotations

Identity quaternion is  $(1; \mathbf{0})$

The conjugate  $(d; -\mathbf{u})$  is the inverse rotation

- $q^*q = (1; \mathbf{0})$  for unit quaternions

To apply a rotation to  $\mathbf{p}$ , compute  $q (0; \mathbf{p}) q^{-1}$

Slide 20

6.837 Fall 2006

## What do we need?

What do we want to do with rotations?

→ What we need to learn to do with quaternions

- Compose
- **Interpolate**
  - Multiply by a scalar
  - Add
  - **slerp**
- Invert
- **Apply to a vector**
- Convert back and forth
  - Axis-angle to quaternion
  - Quaternion to matrix
  - Also Matrix, Euler

Slide 21

6.837 Fall 2006

## Question?

Slide 22

6.837 Fall 2006

## Interpolation: two rotations

Given two rotations  $\mathbf{v} \theta$  and  $\mathbf{v}' \theta'$ , perform linear interpolation:

Convert to quaternions  $q$  &  $q'$

Use **slerp** to interpolate in 4D quaternion space

- You get a rotation for each time step

Slide 23

6.837 Fall 2006

## Why we need an exponential

Recall complex plane & 2dof rotation

The exponential form of complex number  $re^{i\theta}$  is the canonical way to express rotations.

- Enables **slerp** directly
- $\text{slerp}(c_0, c_1, t) = (c_1c_0^{-1})^t c_0$

The same is true of quaternions

- $\text{slerp}(q_0, q_1, t) = (q_1q_0^{-1})^t q_0$ 
  - we know how to multiply
  - We know how to inverse
  - We need to learn power/exponentials

Slide 24

6.837 Fall 2006

## Why are rotation & exp related?

Because when you compose rotations, you add the angles.  
Exponential/Log allow you to turn a product into an addition

Slide 25 6.837 Fall 2006

## Recall complex exponential

$\exp(\log r + i\theta) = r \cos \theta + i r \sin \theta$   
 $\log(c) = \log(|c|) + i \arg(c)$

Slide 26 6.837 Fall 2006

## Exponential form of quaternions

$\exp(d, \mathbf{mu}) = \exp(d) (\cos m; \mathbf{u} \sin m)$   
where  $\mathbf{u}$  is a unit vector

- Can be derived with usual series expression of exponential

Exponential form of quaternions:  
every quaternion  $q$  can be written as

$$q = R \cdot \exp((0; \mathbf{u})\theta)$$

where  $R$  is real,  $\mathbf{u}$  is unit length and  $\theta$  is real

- similar to  $c = r e^{i\theta}$  for complex

$$\log(q = (c, \mathbf{su})) = [\log ||q||, \mathbf{u} \operatorname{argtan2}(c, s)]$$

- $\operatorname{argtan2}$  takes two numbers to resolve ambiguities in  $\operatorname{arccos}$  and  $\operatorname{arcsin}$
- Singularities might occur for angles around  $2\pi$ , but we'll ignore them

Slide 27 6.837 Fall 2006

## slerp in quaternion exponential

In complex:  $\operatorname{slerp}(c_0, c_1, t) = (c_1 c_0^{-1})^t c_0$

Same in quaternion:  $\operatorname{slerp}(q_0, q_1, t) = (q_1 q_0^{-1})^t q_0$   
With the power  $t$  easy to compute in exponential form

Slide 28 6.837 Fall 2006

## Interpolation: two rotations

Given two rotations  $v, \theta$  and  $v', \theta'$ , perform linear interpolation:

Convert to quaternions  $q$  &  $q'$

- Choose between  $q$  &  $-q$ : minimize angle with  $q'$
- Use exponential form

Use  $\operatorname{slerp}$  to interpolate in 4D quaternion space

$$\operatorname{slerp}(q_0, q_1, t) = (q_1 q_0^{-1})^t q_0$$

You get a rotation for each time step

Slide 29 6.837 Fall 2006

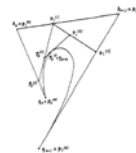
## Interpolation: splines

With the exponential form, you can build splines:

- everything is additive, so you can use the same polynomial bases
- $\log(q(t)) = B_1(t)\log(q_1) + B_2(t)\log(q_2) + B_3(t)\log(q_3) + B_4(t)\log(q_4)$
- $q(t) = \exp(\log(q(t)))$

- potential problems: singularities of the exponential map

Alternative: de Casteljau in quaternion space



Dividing a Bézier curve into segments

Slide 30 6.837 Fall 2006

## What we know about quaternions

2 constructions:

- $(\cos \theta/2; \sin \theta/2 \mathbf{v})$  represents rotation of  $\theta$  around  $\mathbf{v}$
- Extension of complex numbers:  $d+ia+jb+kc$  with 3 roots of -1

Addition, multiplication by scalar

Multiplication:  $(d; \mathbf{u}) (d'; \mathbf{u}') = (dd' - \mathbf{u} \cdot \mathbf{u}'; d\mathbf{u}' + d'\mathbf{u} + \mathbf{u} \times \mathbf{u}')$

- Corresponds to rotation composition
- Non commutative, same as rotations

Identity quaternion is  $(1; \mathbf{0})$

The conjugate  $(d; -\mathbf{u})$  is the inverse rotation

- $q^*q = (1; \mathbf{0})$  for unit quaternions

To apply a rotation to  $\mathbf{p}$ , compute  $q (0; \mathbf{p}) q^{-1}$

Exponential/Log form

- and application to slerp/splines

Slide 31

6.837 Fall 2006

## Questions?

Slide 32

6.837 Fall 2006

## Pros and cons of quaternions

Advantages

- Compact – only 4 numbers (vs. 9 for a matrix)
- Fast computation (16 multiplication vs. 27 for matrices)
- Excellent for interpolation (slerp)
- Good numerical behavior, no numerical drift

Disadvantages

- Slower to apply rotation (24 multiplications vs. 9 for matrix)
- Hard to compose with other transforms
- Not intuitive at first sight (but you should be over it by now :-)

Slide 33

6.837 Fall 2006

## Extensions of quaternion splines

Better interpolation

- E.g. minimize acceleration, velocity constraint
- [http://www.gg.caltech.edu/STC/rr\\_sig97.html](http://www.gg.caltech.edu/STC/rr_sig97.html)
- <http://portal.acm.org/citation.cfm?id=218486&dl=ACM&coll=portal&CFID=1729050&CFTOKEN=74418864>
- <http://portal.acm.org/citation.cfm?id=134086&dl=ACM&coll=portal&CFID=1729050&CFTOKEN=74418864>



From Kim et al. 1995

Slide 34

6.837 Fall 2006

## Buzzword

Quaternions are a Lie group:  
manifold+group

Slide 35

6.837 Fall 2006

## Key references

<http://portal.acm.org/citation.cfm?id=325242>

[http://www.gg.caltech.edu/STC/rr\\_sig97.html](http://www.gg.caltech.edu/STC/rr_sig97.html)

for pdf:

<http://portal.acm.org/citation.cfm?id=258870&coll=portal&dl=ACM&CFID=1729050&CFTOKEN=74418864>

<http://portal.acm.org/citation.cfm?id=134086&dl=ACM&coll=portal&CFID=1729050&CFTOKEN=74418864>

<http://portal.acm.org/citation.cfm?id=218486&dl=ACM&coll=portal&CFID=1729050&CFTOKEN=74418864>

Slide 36

6.837 Fall 2006

## Links

<http://www.euclideanspace.com/maths/geometry/rotations/conversions/eulerToQuaternion/index.htm>  
<http://history.hyperjeff.net/hypercomplex>  
<http://math.hyperjeff.net/hypercomplex/>  
<http://www.gamedev.net/reference/programming/features/qpowers/default.asp>  
[http://www.ogre3d.org/wiki/index.php/Quaternion\\_and\\_Rotation\\_Primer](http://www.ogre3d.org/wiki/index.php/Quaternion_and_Rotation_Primer)  
<http://www.sjbrown.co.uk/?article=quaternions>  
[http://www.isner.com/tutorials/quatSpells/quatSpells\\_14.htm](http://www.isner.com/tutorials/quatSpells/quatSpells_14.htm)  
<http://www.geometrictools.com/Documentation/KeyframeAnimation.pdf>  
<http://en.wikipedia.org/wiki/Quaternion>  
<http://www.euclideanspace.com/maths/algebra/realNormedAlgebra/quaternions/>  
<http://local.wasp.uwa.edu.au/~pbourke/fractals/quatJulia/>  
<http://portal.acm.org/citation.cfm?id=325242>  
<http://books.elsevier.com/companions/0120884003/vq/Quaternion-Maps/index.html>  
<http://www.maths.tcd.ie/pub/HistMath/People/Hamilton/Letters/BroomeBridge.html>  
<http://www.akpeters.com/product.asp?ProdCode=1349>  
[http://en.wikipedia.org/wiki/Quaternions\\_and\\_spatial\\_rotation](http://en.wikipedia.org/wiki/Quaternions_and_spatial_rotation)  
<http://ftp.cis.upenn.edu/pub/graphics/shoemake/quatut.ps.Z>  
<http://books.elsevier.com/companions/0120884003/vq/index.html>  
<http://www.unpronounceable.com/julia/>  
<http://number-none.com/product/Understanding%20Slerp.%20Then%20Not%20Using%20It/>  
<http://www.gamedev.net/reference/programming/features/qpowers/page7.asp>  
<http://graphics.stanford.edu/courses/cs348c-95-fall/software/quatdemo/>  
<http://www.gamedev.net/reference/articles/article1199.asp>

Slide 37

6.837 Fall 2006

## Skinning

Slide 38

6.837 Fall 2006

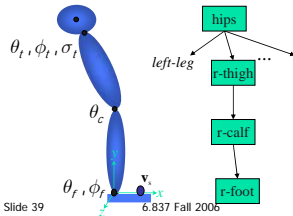
## Back to hierarchical modeling

Remember forward kinematics?

- Rotation at each joint

Great to model robots

What about smooth surfaces such as human outer skin?



Slide 39

6.837 Fall 2006

## Skinning

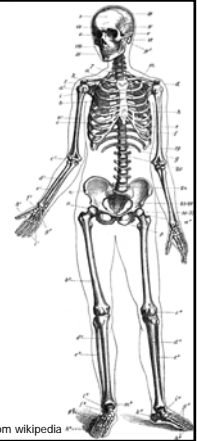
We know how to deform bones, now we need to infer how skin deforms

- This is called skinning

Most popular technique:

Skeletal Subspace Deformation (SSD)

- Each bone yields a deformation of the space around it (rotation)
- In the middle of a limb, the skin points follow the bone rotation
- At a joint, skin is deformed according to a weighted combination of the bones

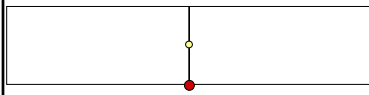


Slide 40

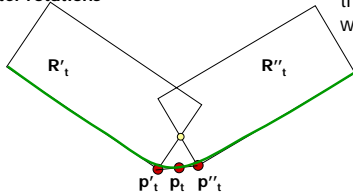
From wikipedia

## Skeletal Subspace Deformation (SSD)

Rest pose



After rotations



Slide 41

6.837 Fall 2006

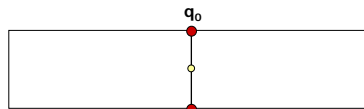
Bone rotations  $R'$  &  $R''$

Vertex  $p$  has weights 0.5 0.5

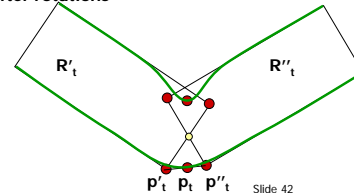
Transform according to  $R'_t$  and  $R''_t$  yields  $p'_t$  and  $p''_t$

the new position is the weighted average

## Not perfect



After rotations



Slide 42

6.837 Fall 2006

## Pseudocode

Do the usual forward kinematics:

- get a matrix  $M_i(t)$  per bone

For each skin vertex  $v_j$

- $v_j(t) = \sum w_{ij} M_i(t) v_j$
- normal:  $n_j(t) = \sum w_{ij} M_i^{-T}(t) n_j$

- where  $w_{ij}$  is the weight map.
- Weights for a vertex, usually sum to one.

Note that the weights are constant over time

- Only a small number of matrices change
- → enable implementation in graphics hardware (little information to update for each frame)

Slide 43

6.837 Fall 2006

## SSD

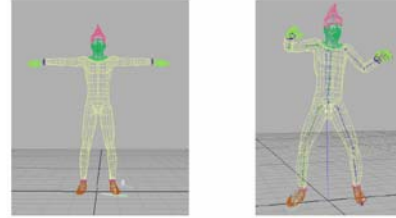


Image: Nvidia

Slide 44

6.837 Fall 2006

## SSD limitations

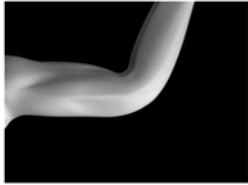


Figure 2: The 'collapsing elbow' in action, c.f. Figure 1.



Figure 3: The forearm in the 'twist' pose, as in turning a door handle, computed by SSD. As the twist approaches 180° the arm collapses.

From: Pose Space Deformation: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation  
J. P. Lewis, Matt Cordner, Nickson Fong

Slide 45

6.837 Fall 2006

## SSD eye candy

from cgcharacter



Slide 46

6.837 Fall 2006

## The secret of SSD

Choose the appropriate weights

- Users can paint weight maps
- Weights can be optimized to match a set of example poses

Slide 47

6.837 Fall 2006

## Limitation of SSD

It is a **linear** combination of transformation  
Rotations beg to be combined differently (quaternions!!!!)

Slide 48

6.837 Fall 2006

## Cool demo

### Skinning Mesh Animations

Doug L. James      Christopher D. Twigg  
Carnegie Mellon University



fig 1. Stampede. Ten thousand skinned mesh animations (SMAs) synthesized in graphics hardware at interactive rates. All SMAs are formed using only traditional matrix palette skinning with well-known nonrigid bone transforms. Distant SMAs are simplified.

Slide 49

6.837 Fall 2006