Lecture 25
Shape Deformation

Last Time: Graphics Performance

- Early Z
- Deferred Shading
- Using Environment Maps for Complex Lighting
- Texture Use and Re-Use (Atlases)
- Avoiding State Changes (big objects)
    - Matrix Palettes

Animation by Transformation

Translate or rotate...

1. Change each vertex
    - compute N vertices
    - transmit N vertices between CPU and GPU
2. Change a transformation
    - change 1 number (maybe 12 for a matrix)
    - send 1 matrix to GPU

Downside: limited things we can do (with simple transformations)

More Generally...

Make one shape

Deform it to other shapes

- easier to animate
- easier to model/control

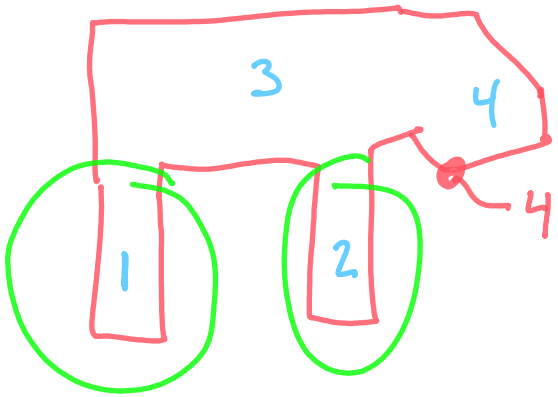
Animation by Deformation

Advantages:

1. performance (don't need to compute every vertex)
    - no need to send mesh to graphics hardware each frame
    - per-vertex computation with limited data
2. authoring (artists don't have to sculpt every vertex)
    - design base shape and make coarse adjustments
3. storage (don't need to remember every vertex in every pose)
4. re-use (apply deformations to different base shapes)

Matrix Palette

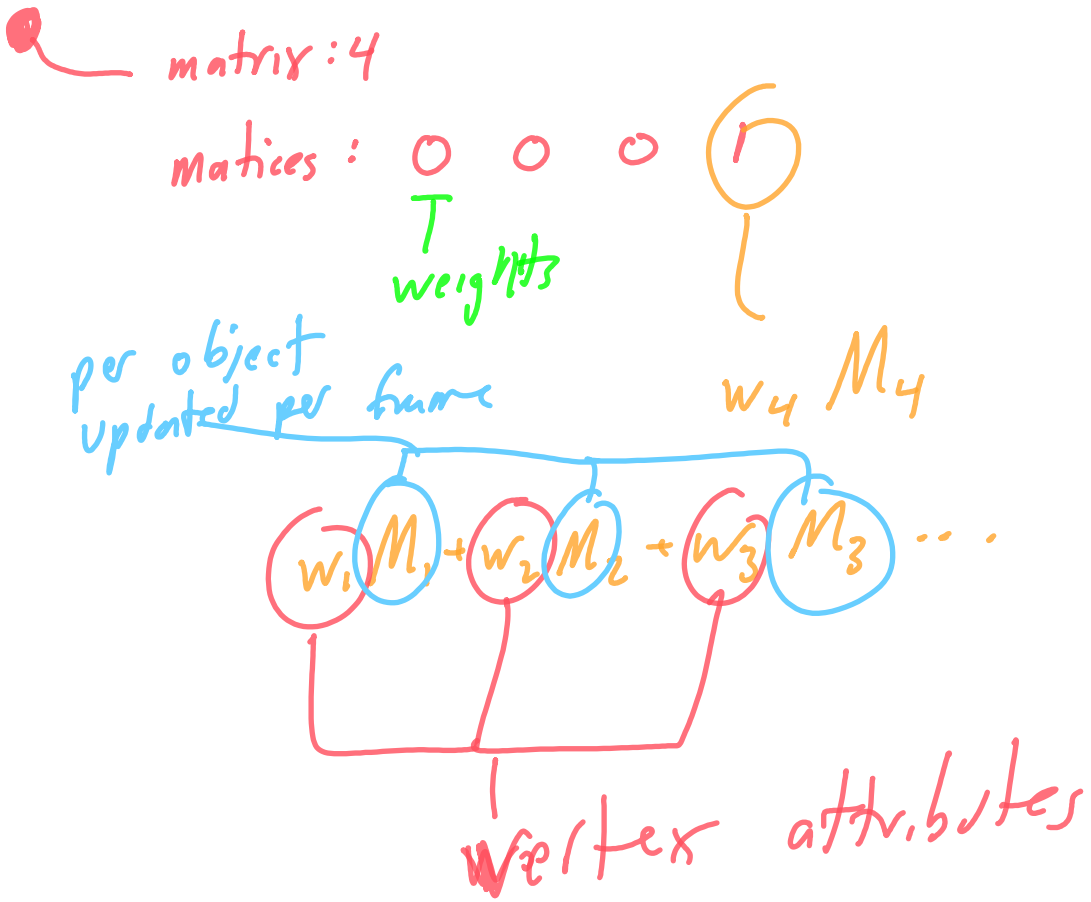
1. Pass multiple matrices (an array of them)
2. Each vertex specifies which matrix it is part of
    - attribute



Specifying which matrix

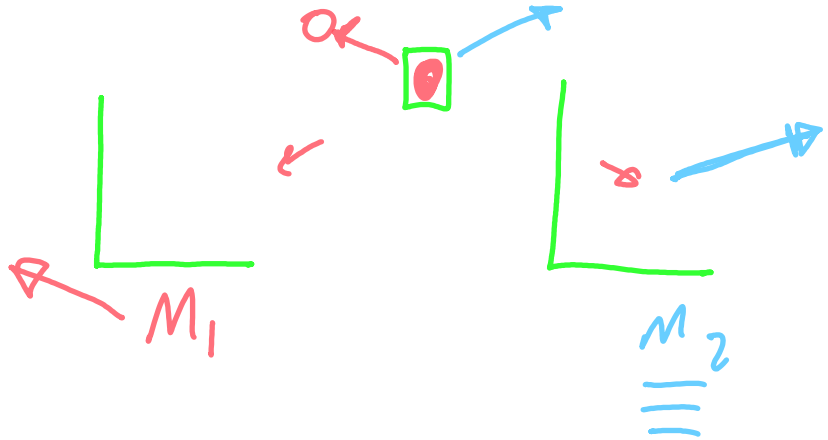
1. Give the number of the matrix
2. Give a vector of weights
    - allows for blending

This is really a setup for skinning...



Skinning

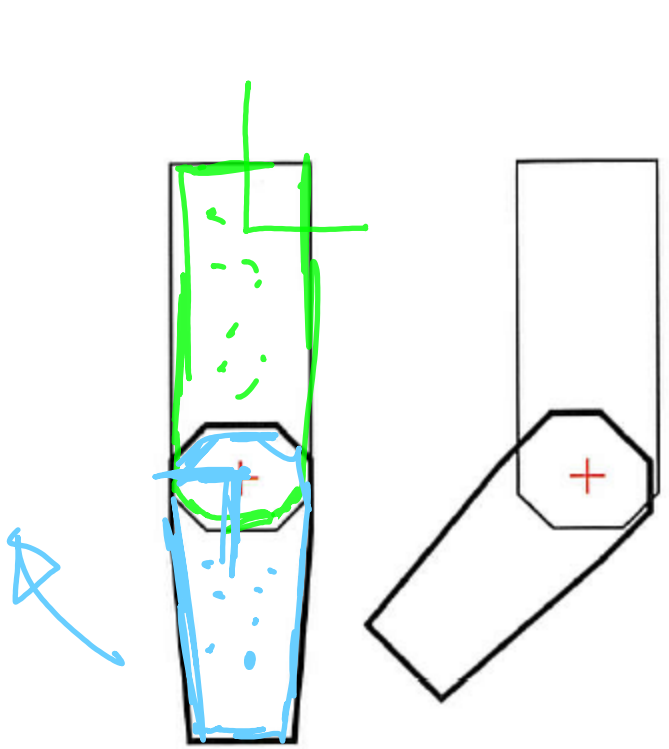
1. Each point in one transformation
2. A point in multiple transformations
    - the point has a different "initial" position
    - the transformations are relative



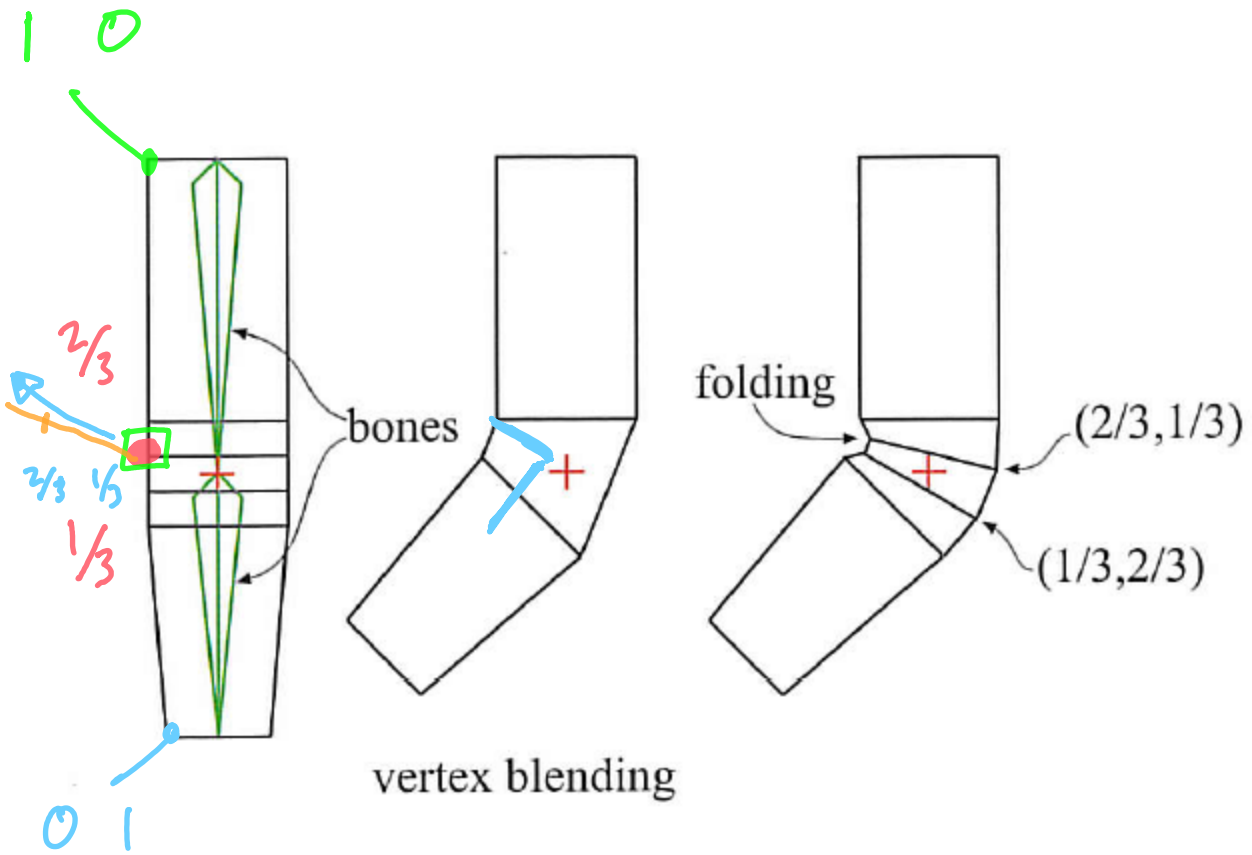
Skinning

Linear Blend Skinning

- Each vertex in multiple coordinate systems (weighting)



rigid-body



vertex blending

$$w_1 M_1 + w_2 M_2$$

Blend Matrices?

$$p_s = \alpha M_1 p_0 + \beta M_2 p_0$$

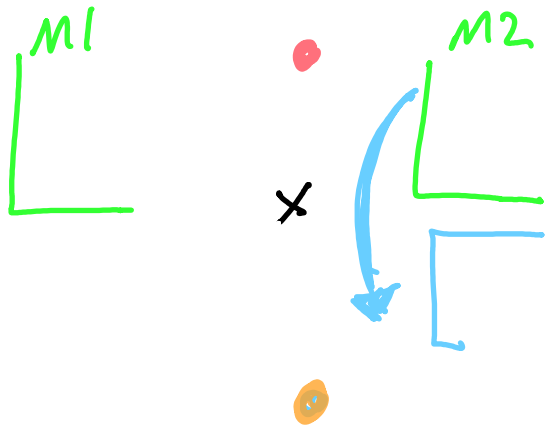
is the same as

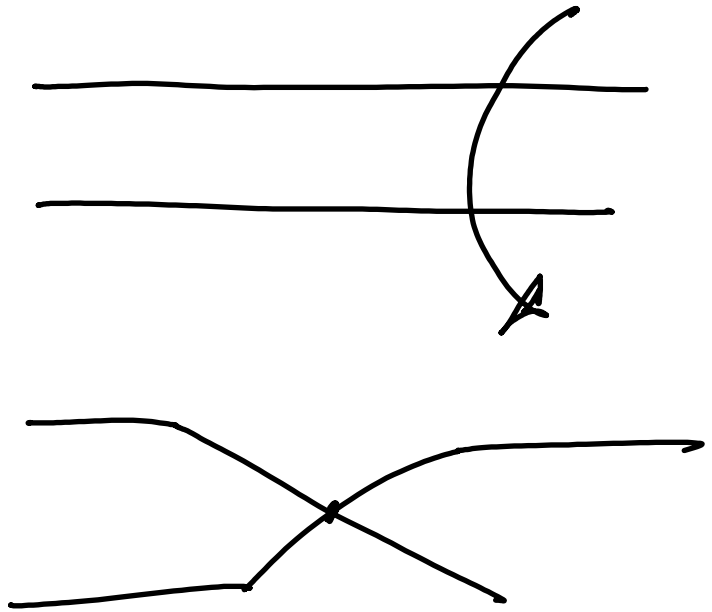
$$p_s = (\alpha M_0 + \beta M_2) p_0$$

If M_1 and M_2 are rotations, scaling/adding doesn't give a rotation!

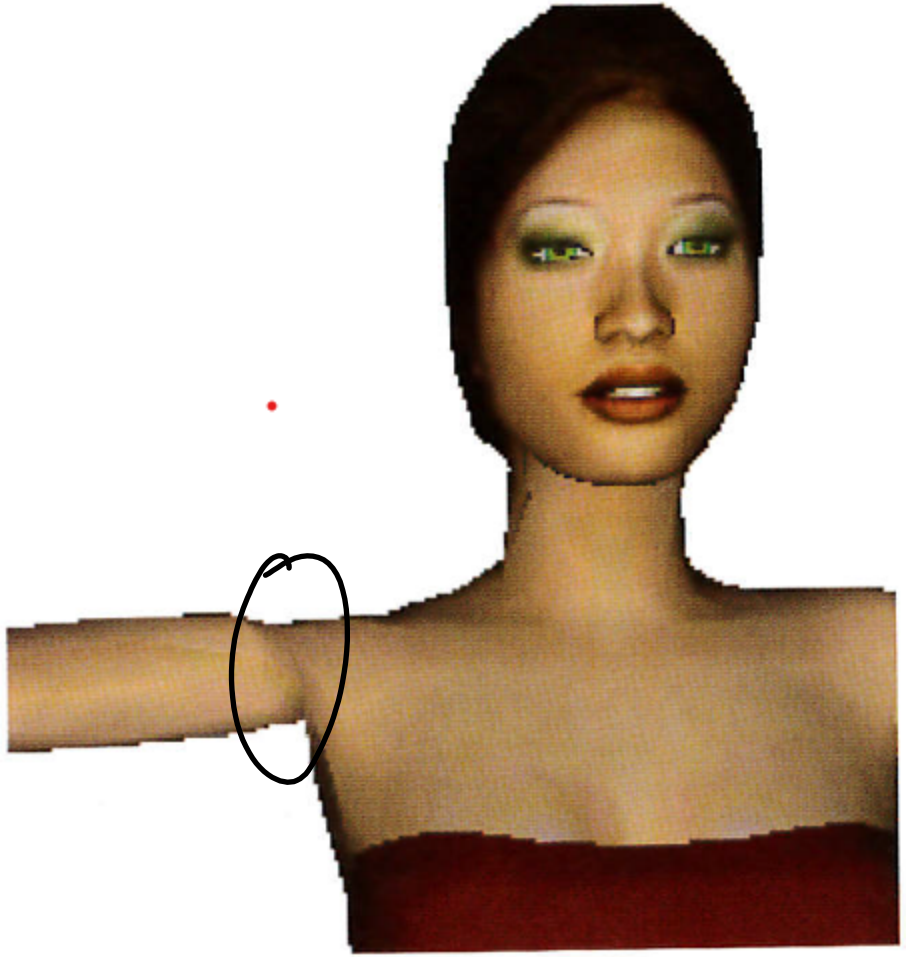
This leads to weird artifacts...

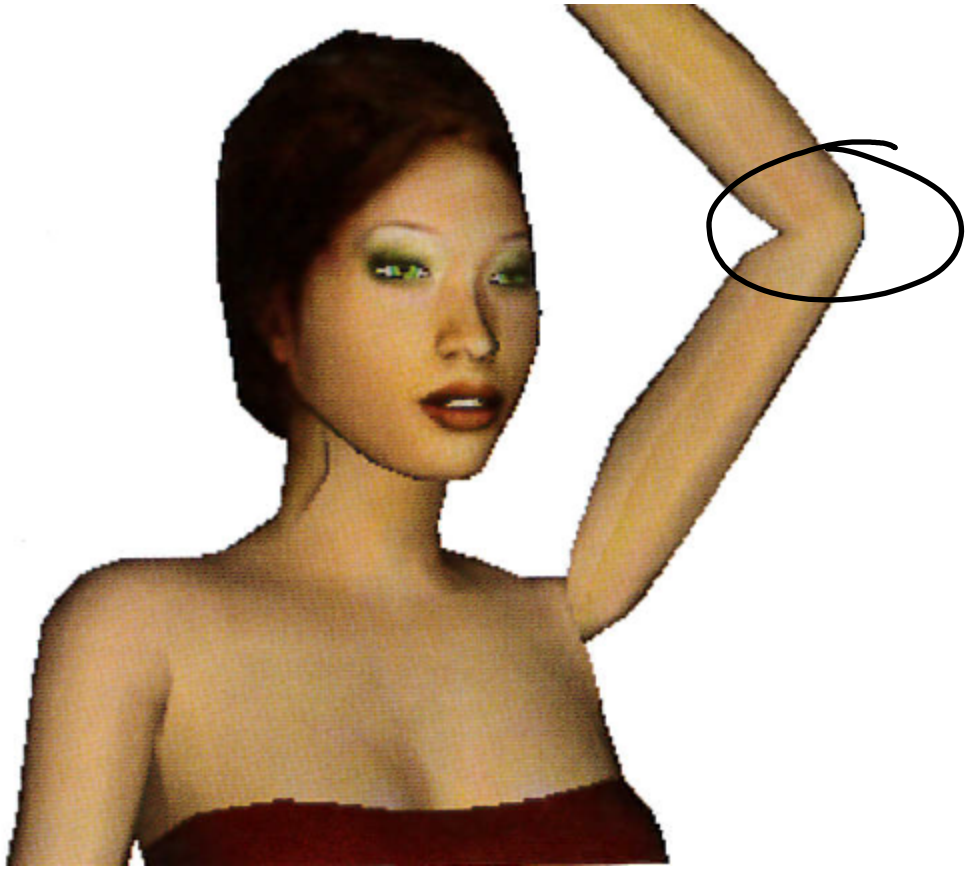
Blend Artifacts





Blend artifacts





Skeletons

connected coordinate systems

Bones as coordinate systems

- how to specify movements?
  - inverse kinematics
  - motion capture
- how to draw the character

Linear Blend Skinning?

- Popular because it is simple
- Artists work around artifacts
- More complex alternatives exist - trade complexity for quality

More Generally...

Make one shape

Deform it to other shapes

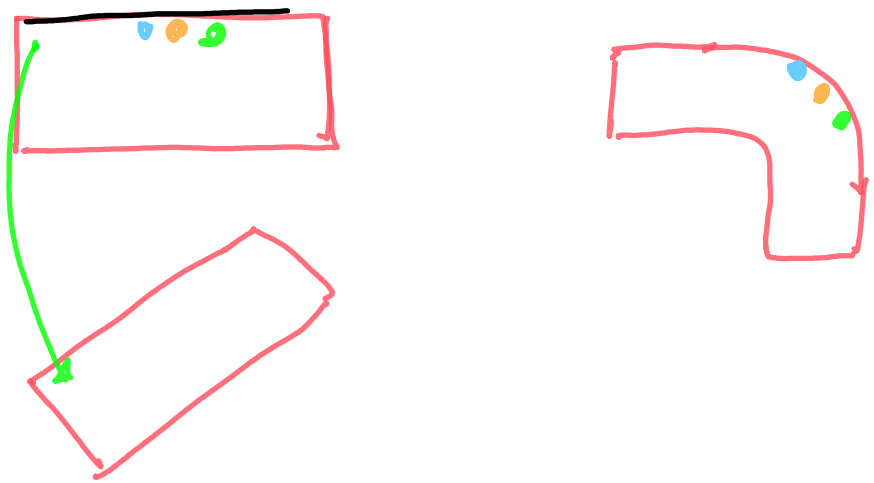
- easier to animate
- easier to model/control

Non-Linear Deformations

- Bend/Twist/Other
- Lattice / Free-Form Deformation
- Cages

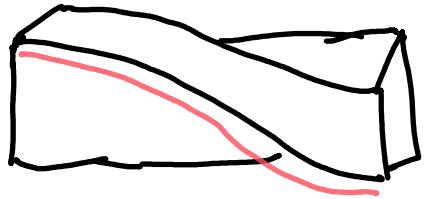
Deformations as space transformations

$$x', y', z' = f(x, y, z)$$



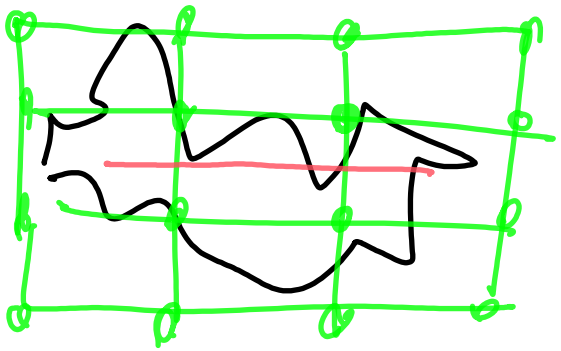
Bend

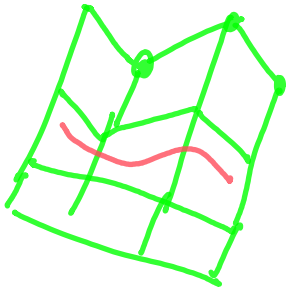
Twist

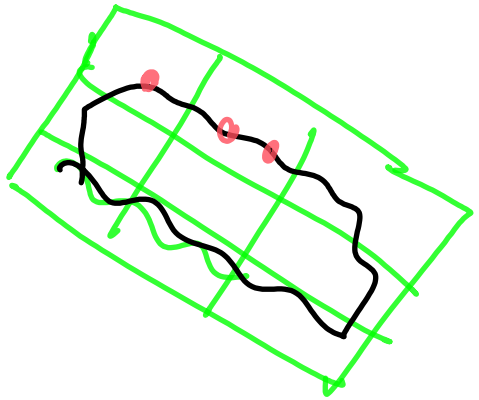


Grid Deformers

Free-Form Deformations (FFD)

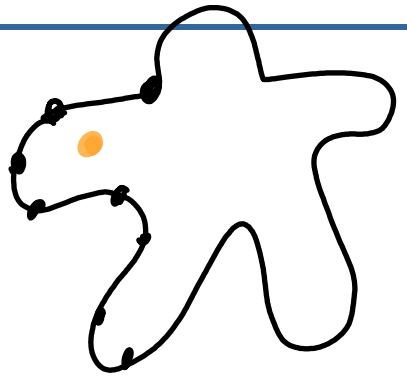


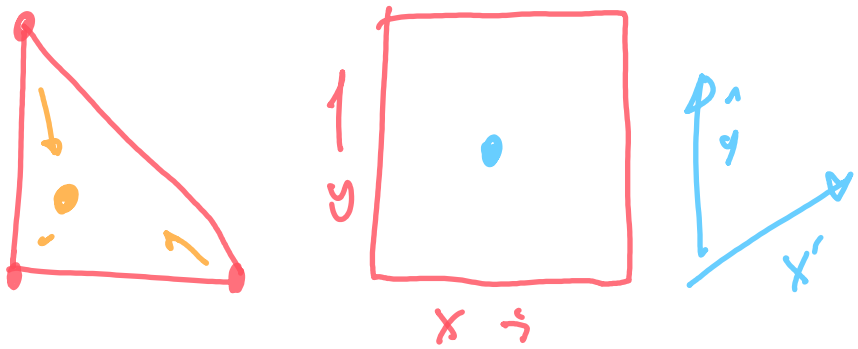




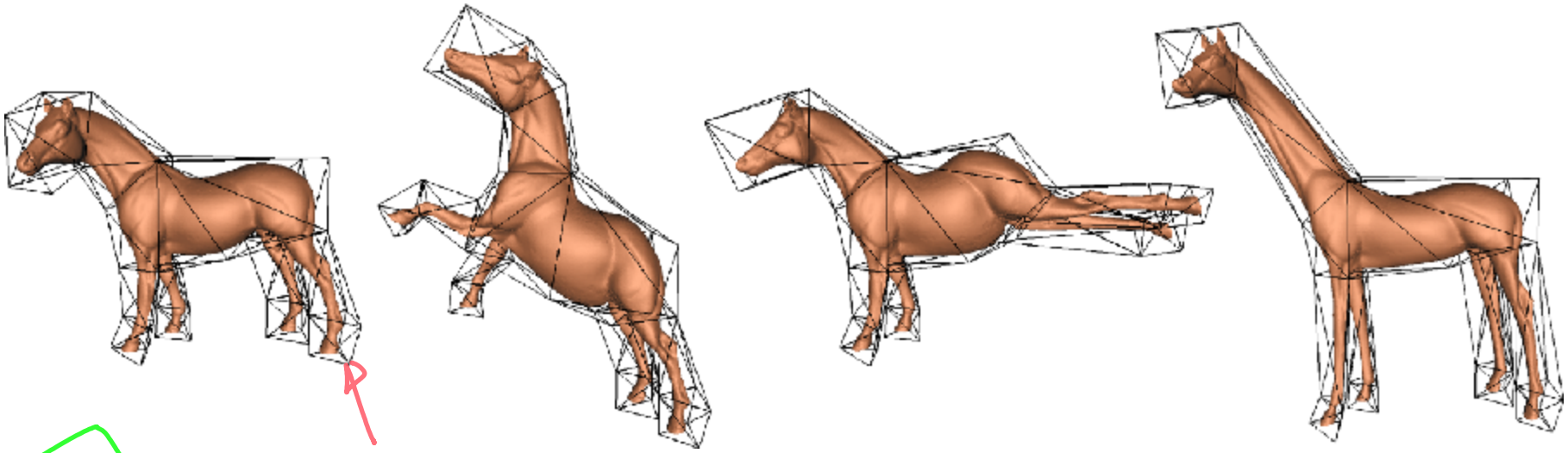
The trick: coordinates in irregular shapes

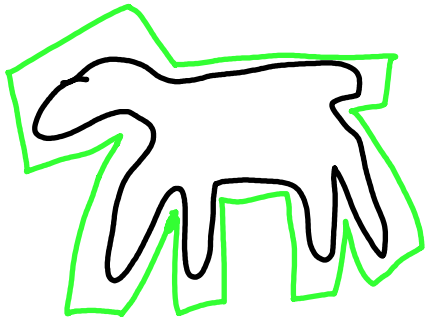
- Triangles (Barycentric)
- Squares (XY)
- Anything else? (generalized barycentric)



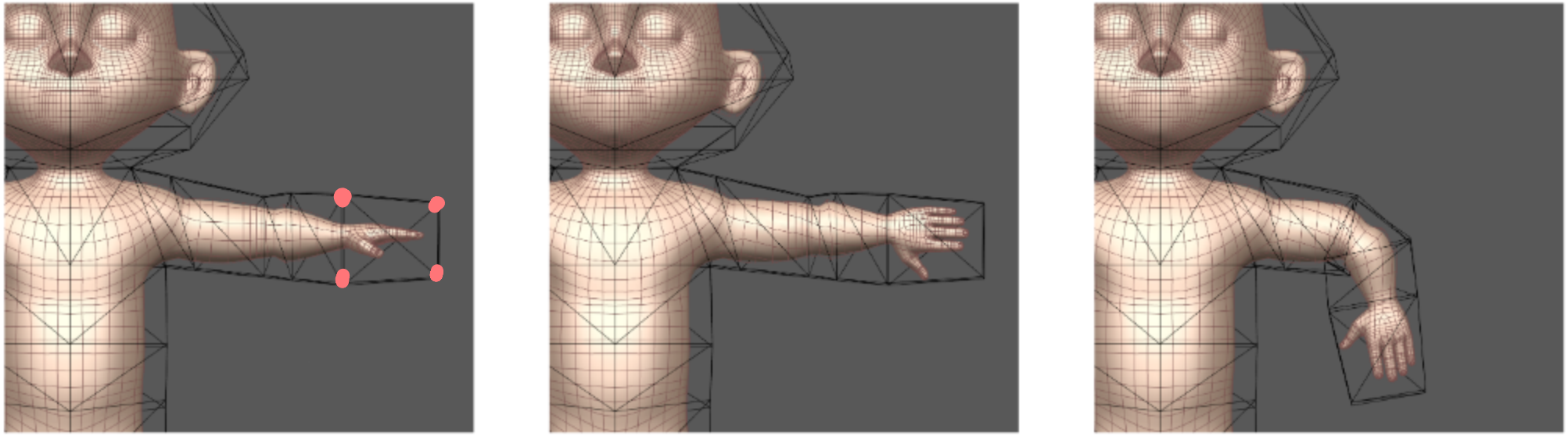


Cages (Coordinate-based Deformations)





Harmonic Coordinates (Pixar 2000s)



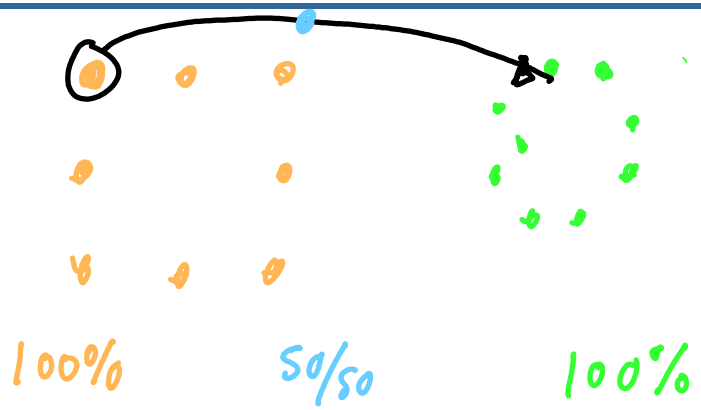
https://graphics.pixar.com/library/HarmonicCoordinatesB/paper.pdf

Multiple Mesh concepts

Shape Interpolation (Morphing)

Create multiple copies of the mesh

- each copy is a **morph target**



Vertices interpolate between targets

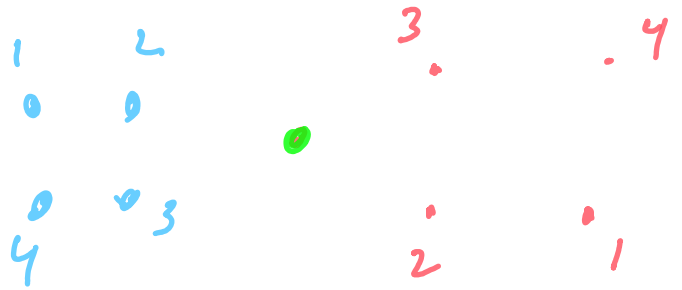
- blend their positions in each target
- $p = w_1 p_1 + w_2 p_2 + w_3 p_3$ for each vertex

Send all meshes to the hardware

Each frame only changes the weights

Downsides

1. Need to make all the meshes
2. Meshes need to correspond
3. In-between values may not be meaningful
4. Control by blending (not always easy)



In THREE

Built in to THREE!

(see that weird blobby thing in the graphics town demo)

OK, But What Can I Do for my Project?

In Graphics Town you can:

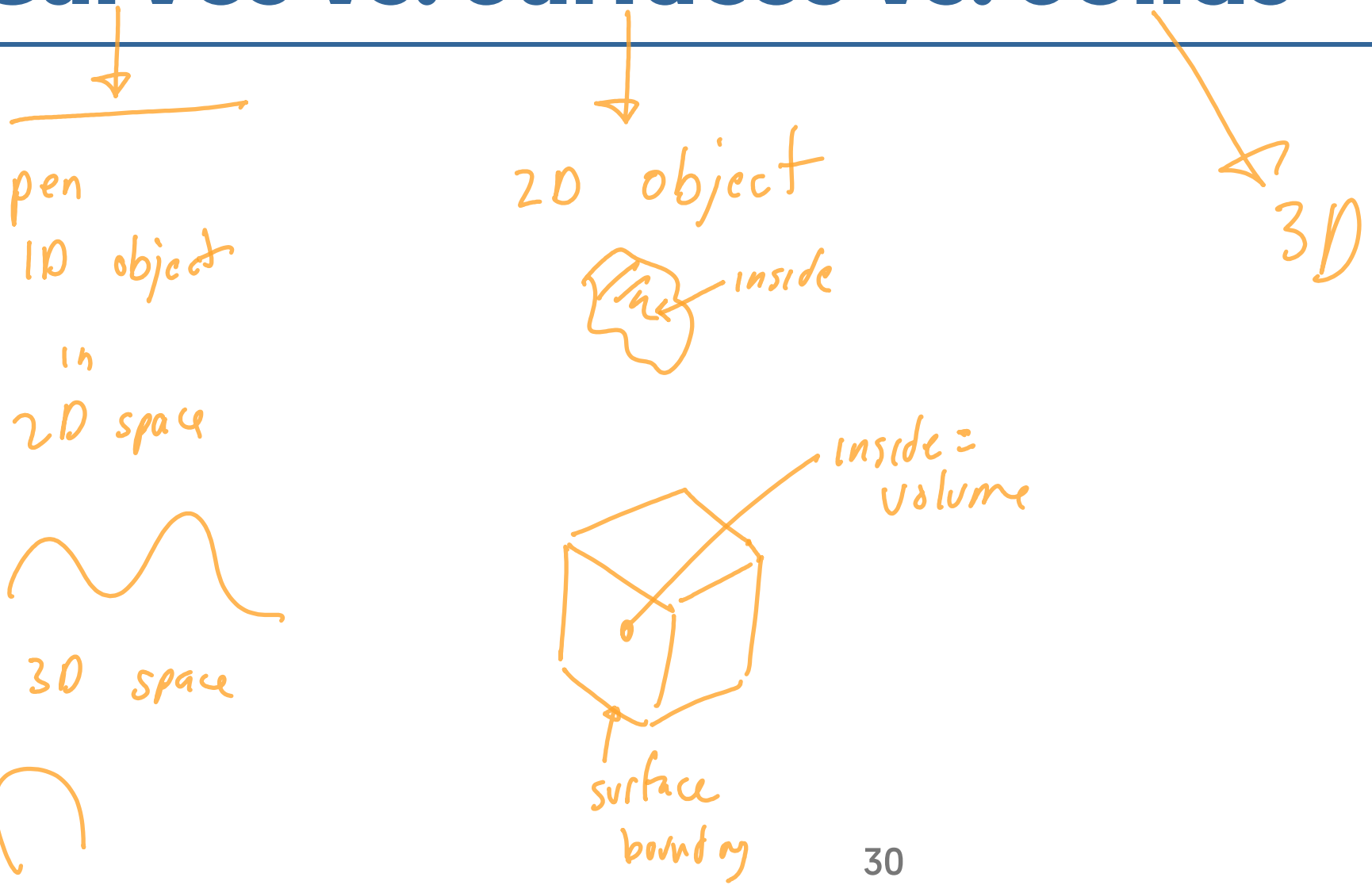
1. Be careful about texture usage (use an Atlas!)
2. Use fewer, bigger objects
3. Use Environment Maps
4. Try Skinning and Morphing (built into THREE)
5. Try implementing complex deformations (good shaders practice)

Avoid doing "stupid" things (things you can't see, redundancy, ...)

More generally...

How do we make smooth shapes?

Curves vs. Surfaces vs. Solids



Curves in 3D

Everything we learned in 2D

just another dimension

dimensions are independent in polynomial curves

$f(v) \Rightarrow x, y, z$

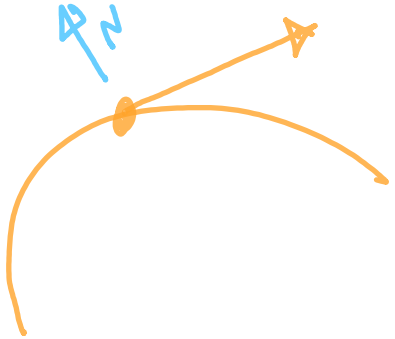
Curves in 3D

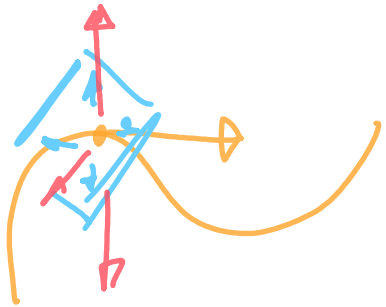
- Tangent Vector
- Normal Plane

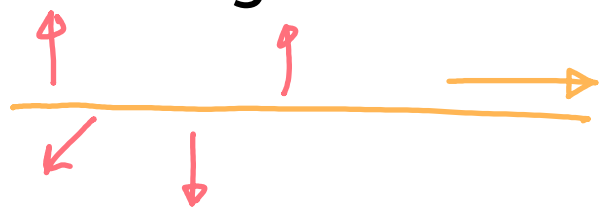
How do we orient an object in the normal plane?

We need a **frame** - a coordinate system that moves along curve

It needs to be consistent



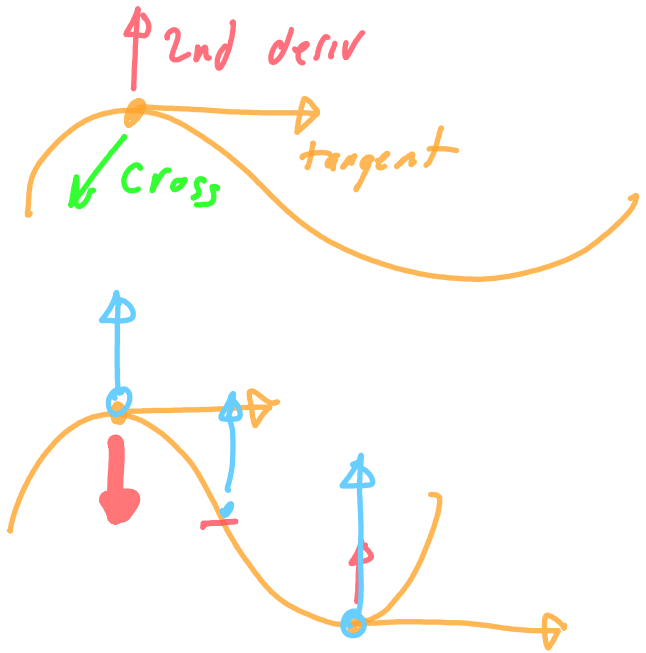




Roller Coasters (Trains in 3D)

The ghost of 559 past...

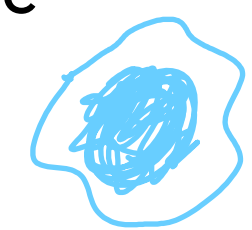
1. Frenet Frame
    - Tangent
    - Normal Vector (direction of 2nd derivative)
    - Bi-Normal (cross product of 1st two)
    - **what if one vanishes?** (straight segment)
2. Interpolate Up Vector



Solid Modeling

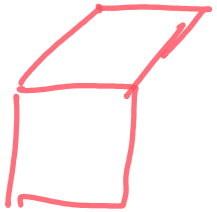
A (non-infinite) surface (area) is bounded by a curve

A curve may (or may not) bound a surface (area)



A (non-infinite) solid is bounded by a surface

A surface may (or may not) bound a solid



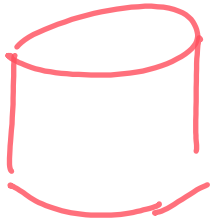
If you want a solid, be careful (that's a different class)

Surface Modeling

Flat surfaces (or piecewise flat)

- polygons
- triangles
- meshes

Standard shapes



- cone
- cylinder
- sphere (ball is volume)
- and many more...