Notes

- Additional assignment: assignment 2.5
- Assignment 3 questions?
- Final project questions?

Grid warping

- Define warp function as a two-variable spline
  - That is, divide up region into squares, warp function on each square is polynomial in x and y
  - Enforce smoothness between neighbouring polynomials
- User can directly specify where each grid node is mapped
  - But those may not coincide with features…

Matching features with grids

- Solve it as an optimization problem: find spline coefficients to
  - Minimize error in matching features
  - And minimize "roughness" (e.g. avoid unnecessary stretching)

Computer vision to the rescue

- Perhaps start with line segment or sketched curve from user
- Adjust to fit image contours
  - "Snakes: active contours" - Kass, Witkin, and Terzopoulos
- Also: if we're matching video, let vision track image features instead of artist specifying correspondences every frame

Morphing limitations

- Hard to convey rigid 3d transformations accurately
  - But look for Seitz, Dyer, "View morphing", SIGGRAPH'96
- Warps are continuous: can't destroy pixels, introduce new ones
  - So sizeable rotations impossible - features are obscured or become visible
  - Note also: need a uniform background

Free Form Deformations

- Apply the idea of warping in 3D
- Have well-defined geometry in a natural pose
- Construct a map that can deform space (and the geometry contained within it)
- Animating the map = animating the object
  - Note: map may have many less DOF, and be much more regular, than object (so it's much easier to control)
- Called Free Form Deformations (FFD's)
  - Section 3.7 in text, Sederberg SIGGRAPH'86

Defining FFD's

- Usually a 3D grid that gets distorted, defining either a 3D B-spline or 3D analogue of Catmull-Rom spline
  - Can use simpler trilinear map too, but it's only C^0 smooth
- For special applications, use a cylindrical grid
- Also can define with an arbitrary tetrahedral mesh

Character Skinning

- We have talked about forward kinematics, inverse kinematics, and Lagrangian dynamics for animating skeletons
  - But it's still just a skeleton
  - Rendering each link as a rigid piece of geometry usually looks very strange (maybe OK for robots)
- Need to wrap skin (and flesh, muscle, organs, fat, clothes, accessories, …) around the bones

Linear Blending (SSD's)

- SSD=skeletal subspace deformation
- Idea:
  - Define skin geometry around skeleton in rest pose
  - For each vertex in skin mesh, figure out local coordinates relative to nearby bones
    - Local coordinates for vertex i relative to bone j with 4x4 transformation matrix M_j are $p_{i:j} = M_j^{-1} p_{i:world}$
  - For each vertex, figure out weights $w_{i:j}$ for each bone (e.g. based on distance, or painted on by artist). Each vertex's weights sum to 1
  - When skeleton moves, new position is weighted average of where the nearby bones want to put it

$$x_i = \sum_j w_{i:j} M_j p_{i:j}$$

Controlling SSD's

- Note: for rigid parts of the anatomy, will want to have all but one weight equal to zero
  - So "skin" moves rigidly with bone
- For flexible parts near joints, smoothly change weights from emphasizing one bone to the other
  - Then skin will stay smooth as skeleton moves

SSD problems

- Joint pinching:
  - For large rotations at a joint, weird stuff happens with the skin deflating, folding over itself, …
- Missing the effect of underlying anatomy: muscles (or fat) bulging, wrinkles, …
- Advanced solutions: interpolate from several example poses, simulate volumetric deformation (e.g. masses and springs), …
- One simple solution (needs more artist effort): replace SSD with FFD
  - Put a 3D FFD grid around joint (or over muscle), parameterize its motion according to joint angle

Motion Capture

- We now have some basic tools to build elaborate characters
  - FK/IK/dynamics to move skeleton
  - Skinning around skeleton
  - Particle systems or rigid bodies for passive secondary motion (e.g. clothes, chains, the environment…)
- Still difficult to animate human motion
  - So many DOF even with simplified skeleton
- So record real motion instead: motion capture (mocap)

Mocap basics

- Film an actor
- Estimate skeleton pose from video at every frame: gives motion curves for joint angles etc.
- Replay motion curves applied to CG character

- Big issues:
  - How to do the pose estimation
  - How to "clean up" the output
  - How to use the cleaned up output for a particular task

Estimation

- Several techniques possible:
  - "Markerless mocap": use computer vision to estimate what's going on (hard to do accurately!)
  - Mechanical measurement: strap a frame to actor that directly measures rotations
    - Really annoying to wear frame
  - "Marker-based mocap": stick several markers on tight-fitting suit, figure out where they are, estimate skeleton from that
    - Marker system could be electromagnetic
    - More common: retroreflective