

EECE 478

Animation

Smooth Animation

Basics:
- *Small* frame-to-frame (F2F) motions look smooth
  - Large motions cause "motion blur" (advanced)
- *Consistent* F2F motions look smooth
- Must match time-to-render with perceived motion

Options:
- Fixed frame rate
- Variable frame rate

Fixed Frame Rate

- Screen refresh controlled by timer
- Objects move similar amount in each frame
  - (dx,dy)/dt nearly constant
- Hint: Use glutTimer() or equiv.

Variable Frame Rate

- Screen refreshes as fast as possible
  - As long as something is moving
  - Hint: Use glutPostRedisplay() in idle func.
- Each object/joint etc. holds (dx,dy)/dt
  - Calculate $\Delta t = t_2 - t_1$
  - $x' = x + (dx/dt) * \Delta t$

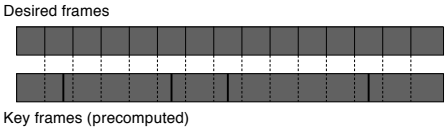
General Animation

Patterned sequences of position+orientation

- Generated by simulation ($p' = p + v\,dt$)
  - Low memory
  - High computation
- Stored and replayed
  - High memory
  - Low computation
- Keyframing
  - Best of both worlds

Keyframes

- A small set of precomputed fixed positions
  - Interpolate between keyframes for actual frames
  - Interpolate position and rotation of all elements



Kinds of Interpolation

Simultaneous blend of shape and texture

- Map a set of points from KF1 to KF2
- Create mesh to interpolate between key points
- Interpolate mesh and blend textures simultaneously
- aka *Morphing*

Kinds of Interpolation

Linear interpolation (LERP)

$P(t_0) = p_0$

$P(t_1) = p_1$

$\alpha(t) = (t-t_0)/(t_1-t_0)$

$P(t) = (1-\alpha(t))\ p_0 + \alpha(t)\ p_1$

Kinds of Interpolation

- Simulating acceleration
  - Use $1\text{-}\cos(kt)$ curve to simulate acceleration from standing start
  - Or $\sin(kt)$ for deceleration to stop
- Smooth interpolation
  - Linear interpolation of rotations (quaternions!)
  - Or, use *splines* with keyframes as control points (smooths acceleration)

Bones and Skin

See http://darwin3d.com/gdm1998.htm#gdm0598

Bones:
- A basic skeleton that defines a hierarchical set of local frames linked by instance tfms

Skin:
- A mesh associated with the "bones" where the positions of mesh elements are derived from the bone positions
- Typically a mesh point is defined in one or two "bone frames" and position is a weighted blend of both positions.

Example: Transition Point

- If $[T_1, R_1]$ is B_0 to B_1 tfm
- We have P in B_0/B_1 transition
  - P_0 is pos in B_0: weight 0.3
  - P_1 is pos in B_1: weight 0.7 = 1.0-0.3
  - $P = 0.3 * P_0 + 0.7 * (R_1T_1P_1)$
    - *In B_0 frame*
- Most P in one frame or other
  - weight = 1.0
  - No blending needed!
- N.B. Normals dynamic too!

Quaternions

- Necessary to interpolate rotations