Notes

- Movie today (compositing etc.)

Example

Camera Control

- Text: 3.4, 4.1
- Three situations to think about:
  - CG film
    - Everything is synthetic, so control the camera however you want (artistic control or algorithms + artistic tweaking)
  - Video games
    - Fully automatic control, algorithm has to be smart enough to (almost) always be useful
  - Compositing vfx on real footage
    - Have to make sure CG camera matches the parameters and motion of real camera

CG film camera

- Simplest case: fix it once for shot
  - Too many moving shots are distracting anyhow… Alexander Nevsky vs. The Constant Gardener
- More generally: just more motion curves
  - But how to parameterize?
  - Can view camera as simply a 4x4 matrix transformation - but using a separate spline for each matrix entry is horrid

5

Peeling off camera parameters

- Essentially always want to separate out amount of perspective = field-of-view = lens diameter = …
  - Should only change for a specific cinematographic reason (zooms, etc.)
  - I.e. perspective projection is a separate part of the parameterization
- Similarly depth-of-field and focal plane is usually fixed except for some cinematographic effects
- Separate out skews (avoid them, except for a few cheesy effects)

6

More peeling

- Usually separate out selection of image plane
  - Scaling actually already handled by perspective, but no harm allowing redundant control
  - Normally image is centred
    - For some architectural shots, can keep vertical lines vertical…
    - Zooms
- Left with specifying position and orientation of camera

7

Possible parameterizations

- Could just do motion curves for position (x,y,z) and Euler angles (heading, pitch, roll)
  - Intensely annoying to keep an object of interest in centre of view
- Could instead do LOOKFROM (x,y,z) LOOKAT (x,y,z) and VUP (x,y,z)
  - Position is at LOOKFROM
  - Camera z-direction is parallel to LOOKAT-LOOKFROM
  - Camera x-direction is VUP × CAMZ   (careful!)
  - Camera y-direction is CAMZ × CAMX
  - Usually VUP is (0,1,0) but can control roll by moving it around --- or a separate roll angle

8

Special Cases

- Circling around target
  - Parameterize target position (x,y,z), distance from target, and angles (heading, pitch, roll)
    - Very useful for interactive modeling also
- Over the shoulder
- Following path of object
- All of these especially useful for video games

Over the shoulder

- Glue the camera to the object’s position motion curve or a displacement from it
- Either use object’s orientation for camera
- Or get CAMZ from motion curve tangent and use VUP to define CAMX, CAMY
  - Maybe additional roll angle for feeling of inertia or G-forces

Path-following camera

- Simplest approach: reuse object trajectory for camera - just lag
  - Camera position t frames behind object
    - Doesn’t handle acceleration/deceleration well!
  - Camera position distance d behind object
    - Need to do arc-length parameterization
- But could be too jerky for comfort
  - Need to smooth out trajectory
  - Replace control points with weighted averages

Path-following camera orientation

- CAMZ should be pointing forwards along trajectory
- Simplest approach: use trajectory tangent (first derivative of curve)
  - But doesn’t really point at object
- So use LOOKAT/VUP approach
  - Again, additional roll angle useful for controlling feeling of inertia, G-forces

Image Space Constraints

- Another approach for camera specification
- Specify some world space points that must stay at some image space points
  - From there calculate camera parameters
- Special cases are tedious to derive
- Better approach: Gleicher and Witkin, “Through-the-Lens Camera Control”, SIGGRAPH’92

Through-the-Lens

- Problem is that image-space constraints on world-space points are highly nonlinear
  - Directly solving is painful
- Instead differentiate constraints
  - Constrain time derivative of camera parameters to follow time derivative of constraints
  - Constraint on derivatives is linear: f(p)=0 becomes df/fp(p) * dp/dt = 0
  - If not enough constraints, minimize camera change in some way - constrained optimization