Beyond 314: Other Graphics Courses

- 426: Computer Animation
  - will be offered next year (2016/2017)
- 424: Geometric Modelling
  - will be offered in two years (2017/2018)

- 526: Algorithmic Animation - van de Panne
- 530P: Sensorimotor Computation - Pai
- 533A: Digital Geometry – Sheffer
- 547: Information Visualization - Munzner
Final

• exam notes: noon Thu Apr 14 SWNG 122
  • exam will be timed for 2.5 hours, but reserve entire 3-hour block of time just in case
  • closed book, closed notes
  • except for 2-sided 8.5”x11” sheet of handwritten notes
    • ok to staple midterm sheet + new one back to back
  • calculator: a good idea, but not required
    • graphical OK, smartphones etc not ok
  • IDs out and face up
Final Emphasis

- covers entire course
- includes some material from before midterm
  - transformations, viewing
  - H1/H2, P1/P2
- but much heavier weighting for material after midterm
  - H3/H4, P3/P4
- post-midterm topics:
  - shaders
  - lighting/shading
  - raytracing
  - collision
  - rasterization / clipping
  - hidden surfaces / blending / picking
  - textures / procedural
  - color
- light coverage
  - animation, visualization
Sample Final

• final+solutions now posted
  • Jan 2007

• note some material not covered this time
  • projection types like cavalier/cabinet: Q1b, Q1c,
  • antialiasing/sampling: Q1d, Q1l, Q12
  • image-based rendering: Q1g
  • clipping algorithms: Q8, Q9
  • scientific visualization: Q14
  • curves/splines: Q18, Q19

• missing some new material
  • shaders
Studying Advice

- do problems!
  - work through old homeworks, exams
    - especially from years where I taught
Review – Fast!!
Review: 2D Rotation

\[ x' = x \cos(\theta) - y \sin(\theta) \]
\[ y' = x \sin(\theta) + y \cos(\theta) \]

\[
\begin{bmatrix}
x' \\
y'
\end{bmatrix} =
\begin{bmatrix}
\cos(\theta) & -\sin(\theta) \\
\sin(\theta) & \cos(\theta)
\end{bmatrix}
\begin{bmatrix}
x \\
y
\end{bmatrix}
\]

counterclockwise, RHS
Review: Shear, Reflection

- shear along x axis
  - push points to right in proportion to height

\[
\begin{bmatrix}
  x' \\
  y'
\end{bmatrix} = \begin{bmatrix}
  1 & sh_x \\
  0 & 1
\end{bmatrix} \begin{bmatrix}
  x \\
  y
\end{bmatrix} + \begin{bmatrix}
  0 \\
  0
\end{bmatrix}
\]

- reflect across x axis
  - mirror

\[
\begin{bmatrix}
  x' \\
  y'
\end{bmatrix} = \begin{bmatrix}
  1 & 0 \\
  0 & -1
\end{bmatrix} \begin{bmatrix}
  x \\
  y
\end{bmatrix} + \begin{bmatrix}
  0 \\
  0
\end{bmatrix}
\]
Review: 2D Transformations

**matrix multiplication**

\[
\begin{bmatrix}
x' \\
y'
\end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]

**scaling matrix**

\[
\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]

**rotation matrix**

**vector addition**

\[
\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} x + a \\ y + b \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix}
\]

**translation multiplication matrix??**

\[
\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix}
\]
Review: Linear Transformations

- Linear transformations are combinations of:
  - shear
  - scale
  - rotate
  - reflect

- Properties of linear transformations:
  - Satisfies $T(sx+ty) = s \ T(x) + t \ T(y)$
  - Origin maps to origin
  - Lines map to lines
  - Parallel lines remain parallel
  - Ratios are preserved
  - Closed under composition

\[
\begin{bmatrix}
  x' \\
  y'
\end{bmatrix}
= \begin{bmatrix}
  a & b \\
  c & d
\end{bmatrix}
\begin{bmatrix}
  x \\
  y
\end{bmatrix}
\]
\[
x' = ax + by
\]
\[
y' = cx + dy
\]
Review: Affine Transformations

- affine transforms are combinations of
  - linear transformations
  - translations

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

- properties of affine transformations
  - origin does not necessarily map to origin
  - lines map to lines
  - parallel lines remain parallel
  - ratios are preserved
  - closed under composition
Review: Homogeneous Coordinates

- **homogenize** to convert homog. 3D point to cartesian 2D point:
  - divide by w to get \((x/w, y/w, 1)\)
  - projects line to point onto \(w=1\) plane
  - like normalizing, one dimension up

- when \(w=0\), consider it as direction
  - points at infinity
  - these points cannot be homogenized
  - lies on x-y plane

- \((0,0,0)\) is undefined
Review: 3D Homog Transformations

- use 4x4 matrices for 3D transformations

\[
\begin{align*}
\text{translate}(a,b,c) & : \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & a & 1 & 0 \\ 1 & b & 1 & 0 \\ 1 & c & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \\
\text{scale}(a,b,c) & : \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}
\end{align*}
\]

\[
\begin{align*}
\text{Rotate}(x, \theta) & : \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \\
\text{Rotate}(y, \theta) & : \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \\
\text{Rotate}(z, \theta) & : \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & 0 & -\sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ \sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}
\end{align*}
\]
**Review: 3D Shear**

- **general shear**
  \[
  \text{shear}(h_{xy}, h_{xz}, h_{yx}, h_{yz}, h_{zx}, h_{zy}) = \begin{bmatrix}
  1 & h_{xy} & h_{zx} & 0 \\
  h_{xy} & 1 & h_{zy} & 0 \\
  h_{xz} & h_{yz} & 1 & 0 \\
  0 & 0 & 0 & 1 
\end{bmatrix}
  \]

- "x-shear" usually means shear along x in direction of some other axis
  - **correction:** not shear along some axis in direction of x
  - to avoid ambiguity, always say "shear along <axis> in direction of <axis>"

\[
\text{shearAlongXinDirectionOfY}(h) = \begin{bmatrix}
  1 & h & 0 & 0 \\
  0 & 1 & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1 
\end{bmatrix}
\]

\[
\text{shearAlongXinDirectionOfZ}(h) = \begin{bmatrix}
  1 & 0 & h & 0 \\
  0 & 1 & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1 
\end{bmatrix}
\]

\[
\text{shearAlongYinDirectionOfX}(h) = \begin{bmatrix}
  1 & 0 & 0 & 0 \\
  h & 1 & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1 
\end{bmatrix}
\]

\[
\text{shearAlongYinDirectionOfZ}(h) = \begin{bmatrix}
  1 & 0 & 0 & 0 \\
  0 & 1 & h & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1 
\end{bmatrix}
\]

\[
\text{shearAlongZinDirectionOfX}(h) = \begin{bmatrix}
  1 & 0 & 0 & 0 \\
  0 & 1 & 0 & 0 \\
  h & 0 & 1 & 0 \\
  0 & 0 & 0 & 1 
\end{bmatrix}
\]

\[
\text{shearAlongZinDirectionOfY}(h) = \begin{bmatrix}
  1 & 0 & 0 & 0 \\
  0 & 1 & 0 & 0 \\
  0 & h & 1 & 0 \\
  0 & 0 & 0 & 1 
\end{bmatrix}
\]
Review: Composing Transformations

ORDER MATTERS!

Ta Tb = Tb Ta, but Ra Rb != Rb Ra and Ta Rb != Rb Ta

- translations commute
- rotations around same axis commute
- rotations around different axes do not commute
- rotations and translations do not commute
Review: Composing Transformations

\[ p' = TRp \]

- which direction to read?
  - right to left
    - interpret operations wrt fixed coordinates
    - moving object
  - left to right
    - OpenGL pipeline ordering!
    - interpret operations wrt local coordinates
    - changing coordinate system
    - OpenGL updates current matrix with postmultiply
      - `glTranslatef(2,3,0);`
      - `glRotatef(-90,0,0,1);`
      - `glVertexf(1,1,1);`
- specify vector last, in final coordinate system
- first matrix to affect it is specified second-to-last
Review: Interpreting Transformations

\[ p' = TRp \]

- translate by \((-1,0)\)
- left to right: changing coordinate system
- right to left: moving object

• same relative position between object and basis vectors
Review: General Transform Composition

• transformation of geometry into coordinate system where operation becomes simpler
  • typically translate to origin

• perform operation

• transform geometry back to original coordinate system
Review: Arbitrary Rotation

- arbitrary rotation: change of basis
  - given two orthonormal coordinate systems XYZ and ABC
    - A’s location in the XYZ coordinate system is \((a_x, a_y, a_z, 1)\), ...
  - transformation from one to the other is matrix \(R\) whose columns are \(A, B, C\):

\[
R(X) = \begin{bmatrix}
  a_x & b_x & c_x & 0 \\
  a_y & b_y & c_y & 0 \\
  a_z & b_z & c_z & 0 \\
  0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

\(R(X) = (a_x, a_y, a_z, 1) = A\)
Review: Transformation Hierarchies

- transforms apply to graph nodes beneath them
Review: Normals

• polygon:
  
  \[ N = (P_2 - P_1) \times (P_3 - P_1) \]

• assume vertices ordered CCW when viewed from visible side of polygon

• normal for a vertex
  
  • specify polygon orientation
  • used for lighting
  • supplied by model (i.e., sphere), or computed from neighboring polygons
Review: Transforming Normals

• cannot transform normals using same matrix as points
  • nonuniform scaling would cause to be not perpendicular to desired plane!

\[
P \quad P' = MP \\
N \quad N' = QN
\]

given M, what should Q be?

\[
Q = (M^{-1})^T
\]

inverse transpose of the modelling transformation
Review: Camera Motion

- rotate/translate/scale difficult to control
- arbitrary viewing position
  - eye point, gaze/lookat direction, up vector
Review: Constructing Lookat

- translate from origin to **eye**
- rotate **view** vector (lookat – eye) to **w** axis
- rotate around **w** to bring **up** into **vw**-plane
Review: V2W vs. W2V

- $M_{V2W} = TR$
- we derived position of camera as object in world
  - invert for `gluLookAt`: go from world to camera!
- $M_{W2V} = (M_{V2W})^{-1} = R^{-1}T^{-1}$

\[
M_{W2V} = \begin{bmatrix}
  u_x & u_y & u_z & -e_x \\
  v_x & v_y & v_z & -e_y \\
  w_x & w_y & w_z & -e_z \\
  0 & 0 & 0 & 1
\end{bmatrix} = \begin{bmatrix}
u_x & v_y & v_z \\\n  w_x & w_y & w_z \\\n  0 & 0 & 0 \\\n   0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
M_{W2V}^{-1} = \begin{bmatrix}
  u_x & u_y & u_z \\\n  v_x & v_y & v_z \\\n  w_x & w_y & w_z \\\n  0 & 0 & 0 \\\n  0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
T = \begin{bmatrix}
  1 & 0 & 0 & e_x \\
  0 & 1 & 0 & e_y \\
  0 & 0 & 1 & e_z \\
  0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
R = \begin{bmatrix}
  u_x & v_x & w_x & 0 \\
  u_y & v_y & w_y & 0 \\
  u_z & v_z & w_z & 0 \\
  0 & 0 & 0 & 1
\end{bmatrix}
\]
Review: Graphics Cameras

• real pinhole camera: image inverted

- computer graphics camera: convenient equivalent
Review: Basic Perspective Projection

\[ \frac{y'}{d} = \frac{y}{z} \Rightarrow y' = \frac{y \cdot d}{z} \]
\[ x' = \frac{x \cdot d}{z} \quad z' = d \]

\[
\begin{bmatrix}
x \\
y \\
z \\
d
\end{bmatrix}
\]

homogeneous coords

\[
\begin{bmatrix}
x \\
y \\
z
\end{bmatrix}
\]

\[
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1/d & 0 & 0
\end{bmatrix}
\]
Review: Asymmetric Frusta

- our formulation allows asymmetry
- why bother? binocular stereo
  - view vector not perpendicular to view plane

Diagrams showing asymmetry in frusta for both right and left eyes.
Review: Field-of-View Formulation

- FOV in one direction + aspect ratio (w/h)
  - determines FOV in other direction
  - also set near, far (reasonably intuitive)