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

Final Review I
University of British Columbia
CPSC 314 Computer Graphics
Jan-Apr 2016
Tamara Munzner

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
- use 4x4 matrices for 3D transformations

Review: 2D Transformations
- shear along x axis
  - push points to right in proportion to height
  - reflect across x axis
  - minor
- reflection matrix

Review: 3D Shear
- general shear
- "x-shear" usually means shear along x in direction of some other axis
- correction: not shear along some axis in direction of a
- to avoid antiguidy, always use "shear along react x direction of (2-x)"

Review: 3D Homog Transformations
- use 4x4 matrices for 3D transformations

Review: Affine Transformations
- affine transforms are combinations of
  - linear transformations
  - translations
  - 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: Linear Transformations
- linear transformations are combinations of
  - shear
  - scale
  - rotate
  - reflect
  - properties of linear transformations
  - satisfies T(ax, ay) = a T(x) + a T(y)
  - origin maps to origin
  - lines map to lines
  - parallel lines remain parallel
  - ratios are preserved
  - closed under composition

Review: 2D Rotation
- (x', y') = (x, y) + (0, 0)
  - counter clockwise, RHS
- rotation matrix

Review: 3D Transformations
- ROTATION: trivia:
  - x,y,z
  - planes
  - lines

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

Sample Final
- Jan 2007
- note some material not covered this time
- projection types like cavalier/cabinet: Q1b, Q1c, Q2, Q3
- antialiasing/sampling: Q1d, Q1f, Q12
- image-based rendering: Q1g
- clipping algorithms: Q6, 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: Composing Transformations
- which direction to read?
  - right to left
  - interpret operations wrt fixed coordinates
  - moving object
  - left to right
  - interpret operations wrt local coordinates
  - changing coordinate system
  - OpenGL pipeline ordering:
    - glTranslate(0,0,1);
    - gluPerspective(45,0.1,1);
  - specify vector last, in final coordinate system
  - first matrix to affect it is specified second-to-last

OpenGL pipeline ordering!

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

Review: Transformation Hierarchies
- transforms apply to graph nodes beneath them

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
- \( \mathbf{M}^{\text{V2W}} = \mathbf{R}^{\text{TR}} \)
- we derived position of camera as object in world
  - invert for gluLookAt: go from world to camera!
- \( \mathbf{M}^{\text{W2V}} = (\mathbf{M}^{\text{V2W}})^{-1} \mathbf{R}^{\text{TR}}^{-1} \)

Review: Graphics Cameras
- real pinhole camera: image inverted
- computer graphics camera: convenient equivalent

Review: Arbitrary Rotation
- arbitrary rotation: change of basis
  - given two orthonormal coordinate systems XYZ and ABC
  - if location in the XYZ coordinate system is \((x_0,y_0,z_0,1)\), ... 
    - transformation from one to the other is matrix \( \mathbf{R} \) whose
      - columns are \( \mathbf{R}(1) \), \( \mathbf{R}(2) \), \( \mathbf{R}(3) \)

Review: Field-of-View Formulation
- FOV in one direction + aspect ratio (w/h)
- determines FOV in other direction
- also set near, far (reasonably intuitive)