

**UNIVERSITY OF NEBRASKA AT OMAHA  
COURSE SYLLABUS/DESCRIPTION**

Department and Course Number	CSCI 4620
Course Title	Computer Graphics
Course Coordinator	Hassan Farhat
Total Credits	3
Date of Last Revision	February 28, 2014

**1.0 Course Description:**

- 1.1 Overview of content and purpose of the course (Catalog description).  
The course covers topics that deal with computer graphics rendering of primitive objects, polygon clipping algorithms, polygon fill algorithms, two-dimensional transformations, fractal drawing, three-dimensional transformations, viewing camera rendering and projections, object representations, three-dimensional curve and surface rendering algorithms, and hidden line and surface removal algorithms.
- 1.2 For whom course is intended.  
The course is designed primarily for third or fourth year majors in Computer Science.
- 1.3 Prerequisites of the course (Courses).  
CSCI 3320
- 1.4 Prerequisites of the course (Topics).
  - 1.4.1 Algorithmic problem-solving in the context of a modern programming language:
    - 1.4.1.1 Abstract Data Types & Objects
    - 1.4.1.2 Dynamic Memory Management
    - 1.4.1.3 Recursion
    - 1.4.1.4 Object Oriented Techniques
- Primary data structures: Stacks & Queues, Linked Lists, Trees
- 1.5 Unusual circumstances of the course.  
None

**2.0 Objectives:**

- 2.1 Study the underlining algorithms found in graphics 2-D packages.
- 2.2 Study the geometrical transformation applied to objects.
- 2.3 Study 3-D curve and surface algorithms, in rendering, surface and hidden line removal algorithms
- 2.4 Study fractals and iterated function system

**3.0 Content and Organization:**

Contact hours

3.1	Introduction to Computer Graphics	3
	What is Computer Graphics?	
	Elements of Pictures created in Computer Graphics	
	▪ Polylines	
	▪ Text	
	▪ Filled Regions	
	▪ Raster Images	
3.2	Representation of Gray Shades and Color for Raster Graphics	3
	Graphics Display Devices	
	▪ Line Drawing Displays	
	▪ Raster displays	
	▪ Indexed Color and the LUT	
	▪ Other raster Display Devices	
	▪ Hard Copy Raster Devices	
	Graphics Input Primitives and Devices	
	▪ Types of Input Graphics Primitives	
	▪ Types of Physical Input Devices	
3.3	Drawing Figures	5
	Getting Started Making Pictures	
	▪ Device-independence Programming	
	▪ Windows-based Programming	
	▪ Opening a Window for Drawing	
	Drawing Basic Graphics Primitives	
	Making Line Drawings	
	▪ Drawing Polylines and Polygons	
	▪ Line Drawing using “moveto()” and “lineto()”	
	▪ Drawing Aligned Rectangles	
	▪ Aspect Ratio of an Aligned Rectangle	
	▪ Filling Polygons	
	▪ Other Graphics Primitives	
3.4	More Drawing Tools	5
	World Windows and Viewpoints	
	▪ The Mapping from the Window to the Viewport	
	▪ Setting the Windows and Viewport Automatically	
	Clipping Lines	
	▪ Clipping a Line	
	▪ The Cohen-Sutherland Clipping Algorithm	
	Drawing Circles and Arcs	
	▪ Drawing arcs	
	Using the Parametric Form of a Curve	
	▪ Parametric Forms for Curves	
	▪ Drawing Curves Represented Parametrically	
	▪ Superellipses	
	▪ Polar Coordinates Shapes	
	▪ 3D Curves	
3.5	Vector Tools for Graphics	4

	Vectors	
	▪ Operations with Vectors	
	▪ Linear Combinations of Vectors	
	▪ The Magnitude of a Vector, and Unit Vectors	
	The Dot Product	
	The Cross Product of Two Vectors	
	▪ Geometric Interpretations of the Cross Product	
	▪ Finding the Normal to a Plane	
	Representations of Key Geometric Objects	
	▪ Coordinate Systems and Coordinate Frames	
	▪ Affine Combinations of Points	
	▪ Linear Interpretation of Two Points	
	▪ “Tweening” for Art and Animation	
	▪ Preview: Quadratic and Cubic Tweening, and Bezier Curves	
	▪ Representing Lines and Planes	
	Finding the Intersection to Two Line Segments	
	Intersections of Lines with Planes, and Clipping	
3.6	Transformations of Objects	5
	Introduction to Transformations	
	▪ Transforming Points and Objects	
	▪ The Affine Transformation	
	▪ Geometric effects of elementary 2D Affine Transformations	
	▪ The Inverse of an Affine Transformation	
	▪ Composing Affine Transformations	
	▪ Examples of Composing 2D Transformations	
	▪ Some Useful Properties of Affine Transformations	
	3D Affine Transformations	
	▪ The Elementary 3D Transformations	
	▪ Composing Rotations	
	▪ Combining Rotations	
	▪ Summary of Properties of Affine Transformations	
	Changing Coordinate Systems	
	Using Affine Transformations in a Program	
	▪ Saving the CT for Later Use	
3.7	Modeling Shapes with Polygonal Meshes	7
	Introduction to Solid Modeling with Polygonal Meshes	
	▪ Defining a Polygonal Mesh	
	▪ Finding the Normal Vectors	
	▪ Properties of Meshes	
	▪ Mesh Models for Nonsolid Objects	
	▪ Working with Meshes in a Program	
	Polyhedra	
	▪ Prisms and Antiprisms	
	▪ The Platonic Solids	
	▪ Other Interesting Polyhedra	
	Extruded Shapes	

	<ul style="list-style-type: none"> <li>▪ Creating Prisms</li> <li>▪ Arrays of Extruded Prisms: “Bricklaying”</li> <li>▪ Extrusions with a “Twist”</li> <li>▪ Building Segmental Extrusions: Tubes and Snakes</li> <li>▪ “Discretely” Swept Surfaces of Revolution</li> </ul>	
	Mesh Approximations to Smooth Objects	
	<ul style="list-style-type: none"> <li>▪ Representations for Surfaces</li> <li>▪ The Normal Vector to a Surface</li> <li>▪ The Effect of an Affine Transformation</li> <li>▪ Three “Generic” Shapes: Sphere, Cylinder, and Cone</li> <li>▪ Forming a Polygonal Mesh for a Curved Surface</li> <li>▪ Ruled Surface</li> <li>▪ Surfaces of Revolution</li> <li>▪ The Quadric Surfaces</li> <li>▪ The Superquadrics</li> <li>▪ Tubes Based on 3D Curves</li> <li>▪ Surfaces Based on Explicit Functions of Two Variables</li> </ul>	
3.8	Three-Dimensional Viewing	4
	Camera coordinates	
	<ul style="list-style-type: none"> <li>▪ Setting the View Volume</li> <li>▪ Positioning and Pointing the Camera</li> </ul>	
	Building a Camera in a Program	
	<ul style="list-style-type: none"> <li>▪ “Flying” the Camera</li> </ul>	
	Perspective Projections of 3D Objects	
	<ul style="list-style-type: none"> <li>▪ Perspective Projection of a Point</li> <li>▪ Perspective Projection of a Line</li> <li>▪ Incorporating Perspective in the Graphics Pipeline</li> </ul>	
	Producing Stereo views	
	Taxonomy of Projections	
	<ul style="list-style-type: none"> <li>▪ One-, Two-, Three-Point Perspective</li> <li>▪ Types of Parallel Projections</li> </ul>	
3.9	Rendering Faces for Visual Realism	5
	Introduction to Shading Models	
	<ul style="list-style-type: none"> <li>▪ Geometric Ingredients for Finding Reflected Light</li> <li>▪ Computing the Diffuse Component</li> <li>▪ Specular Reflection</li> <li>▪ The Role of Ambient Light</li> <li>▪ Combining Light Contributions</li> <li>▪ Adding Color</li> <li>▪ Shading and the Graphics Pipeline</li> <li>▪ Using Light Sources in OpenGL</li> <li>▪ Working with Material Properties in OpenGL</li> <li>▪ Shading of Scenes Specified by SDL</li> </ul>	
	Flat Shading and Smooth Shading	
	<ul style="list-style-type: none"> <li>▪ Flat Shading</li> <li>▪ Smooth Shading</li> </ul>	

- Removing Hidden Surfaces
    - The Depth Buffer approach
- 3.10 Approaches to Infinity 4
  - Fractals and Self-Similarity
    - Successive Refinement of Curves
    - Drawing Koch Curves and Snowflakes
    - Fractional Dimension
  - String Production and Peano Curves
    - Producing Recursively and Drawing in a Program
    - Allowing Branching
  - Tiling the Plane
    - Monohedral Tilings
    - Dihedral Tilings
    - Drawing Tilings
    - Reptiles
  - Creating an Image by Means of Iterated Functions Systems
    - An Experimental Copier
    - Some Underlying theory of the Copying Process
    - Drawing the k-th Iterate
    - The Chaos Game
    - Finding the IFS, Fractal Image Compression
  - The Mandelbrot Set
    - Mandelbrot Sets And Iterated Function Systems
    - Defining the Mandelbrot Set
    - Computing whether the point  $c$  is in the Mandelbrot Set
    - Drawing the Mandelbrot Set
    - Some Notes on the Mandelbrot Set
  - Julia Sets
    - The Filled-in Julia Set  $K_c$
    - Drawing Filled-in Julia Sets
    - Some Notes on the Mandelbrot Set
    - The Julia Set  $J_c$
  - Random Fractals
    - Fractalizing a Segment
    - Controlling the Spectral Density of the Fractal Curve

#### 4.0 Teaching Methodology:

- 4.1 Methods to be used.
 

The coverage will be mainly through lecture homework and program assignments. The program assignments will use any programming language that includes primitive calls to pixel drawing. The assignments are designed to build a simple graphics package.
- 4.2 Student role in the course.
 

As part of the evaluation process, the student role may include a detailed presentation of solutions to program assignments.

- 4.3 Contact hours.  
3 contact hours.

**5.0 Evaluation:**

- 5.1 Type of student projects that will be the basis for evaluating student performance, specifying distinction between undergraduate and graduate, if applicable. For Laboratory projects, specify the number of weeks spent on each project).

The grade will be based on a combination of examinations, student homework, and lab assignments.

- 5.2 Basis for determining the final grade (Course requirements and grading standards) specifying distinction between undergraduate and graduate, if applicable.

The weights used in determining the final grade may vary, but are typically similar to the following.

Component	Grading
Homework	25%
exams	35%
Program assignments	40%.

- 5.3 Grading scale and criteria.

Points	Grade
97-100%	A+
90-96%	A
87-89%	B+
80-86%	B
77-79%	C+
70-76%	C
67-69%	D+
60-66%	D
0-59%	F

**6.0 Resource Material**

- 6.1 Textbooks and/or other required readings used in course.  
D. Hearn, P. Baker, “Computer Graphics, second edition,” Prentice Hall, 2011.
- 6.2 Other suggested reading materials, if any.  
None
- 6.3 Other sources of information.

None

- 6.4 Current bibliography of resource for student's information.
  - 6.4.1 P. Watt, "The Computer Image," Addison Wesley.
  - 6.4.2 Pavilidis, "Interactive Computer in X," PWS.
  - 6.4.3 Burger, "Interactive Computer," Addison Wesley
  - 6.4.4 Foley, Van Dam, Feiner, Hughes, "Computer Graphics Principle and Practice," Addison Wesley.
  - 6.4.5 E. Angel. "Interactive Computer Graphics, A Top-Down Approach Using OpenGL, 6th edition". Addison Wesley, 2012
  - 6.4.6 E. Angel, "OpenGL, A Primer, 2nd edition", Addison Wesley, 2005.
  - 6.4.7 D. Hearn, M. Baker, W. Carithers, "Computer Graphics with OpenGL, 4th edition", Prentice Hall, 2011.
  - 6.4.8 F. Hill, S. Kelly, "Computer Graphics using OpenGL, 3rd edition", Prentice Hall, 2007.
  - 6.4.9 D. Rogers, J. Adams, "Mathematical Elements for Computer Graphics, 2nd edition", McGraw Hill, 1990.

**7.0 Computer Science Accreditation Board (CSAB) Category Content (class time in hours):**

<i>CSAB Category</i>	<i>Core</i>	<i>Advanced</i>
Data structures		
Computer organization and architecture		
Algorithms and software design	40	
Concepts of programming languages		

**8.0 Oral and Written Communications:**

Every student is required to submit at least   0   written reports (not including exams, tests, quizzes, or commented programs) to typically        pages and to make   0   oral presentations of typically        minutes duration. Include only material that is graded for grammar, spelling, style, and so forth, as well as for technical content, completeness, and accuracy.

**9.0 Social and Ethical Issues:**

No coverage

**10.0 Theoretical content**

	Contact hours
10.1 Introduction to Computer Graphics and rendering of primitives	11.0
10.2 Clipping algorithms and parametric curve drawing	5.0
10.3 Transformations of Objects	5.0

10.4	Three-Dimensional Viewing	5.0
10.5	Modeling Shapes with Polygonal Meshes	7.0
10.6	Vector Tools for Graphics	4.0
10.7	Rendering Faces for Visual Realism	5.0
10.8	Approaches to Infinity	4.0

### 11.0 Problem analysis:

Students learn to solve graphics rendering problems. The analysis part deals with vector and matrix algebra.

### 12.0 Solution design:

Students will learn that producing efficient algorithms is an important part of computer graphics.

### CHANGE HISTORY

<i>Date</i>	<i>Change</i>	<i>By whom</i>	<i>Comments</i>
10/30/2002	Initial ABET version	Farhat	
06/13/2003	Cleanup	Wileman	
10/13/2008	Insertion of table mapping course objectives to program outcomes	Qiuming Zhu	
2/16/2014	Updated Resource Material	Farhat	
2/28/2014	Updated course description and contents and	Qiuming Zhu	

**UNIVERSITY OF NEBRASKA AT OMAHA**  
**Mapping of CS Program Outcomes vs. course objectives**

Department and Course Number	CSCI 4620
Course Title	Computer Graphics
Course Coordinator	Hassan Farhat / Qiuming zhu
Total Credits	3
Date of Last Revision	October 13, 2008

**Instructions:** Paste or type the course objectives in the left-hand column. Indicate the relationship between course objective and program outcome by placing one of the two following marks in the appropriate cell:

- S – Strong relationship**  
**X – Contributing relationship**

Course objective	CS Program Outcomes										
	(a) knowledge of discipline	(b) analyze problem, define	(c) design and implement solution	(d) function on a team	(e) ethical issues	(f) communicate effectively	(g) analyze impact of computing	(h) continued professional development	(i) Current techniques and tools	(j) apply foundations	(k) apply design and development
1. Study the underlining algorithms found in graphics 2-D packages.	S	X	X						X	X	
2. Study the transformation applied to objects.	S	X	X						X	X	
3. Study 3-D curve and surface algorithms, in rendering, surface and line removal algorithms	X	S	S				X		X	S	X
4. Study fractals and iterated function system	X	S	S				X	X	X	S	X
5.											

**CS Program Outcomes (2008)**

- (a) An ability to apply knowledge of computing and mathematics appropriate to the discipline;
- (b) An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution;
- (c) An ability to design, implement and evaluate a computer-based system, process, component, or program to meet desired needs;
- (d) An ability to function effectively on teams to accomplish a common goal;
- (e) An understanding of professional, ethical, legal, security, and social issues and responsibilities
- (f) An ability to communicate effectively with a range of audiences
- (g) An ability to analyze the local and global impact of computing on individuals, organizations and society
- (h) Recognition of the need for, and an ability to engage in, continuing professional development
- (i) An ability to use current techniques, skills, and tools necessary for computing practices
- (j) An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices
- (k) An ability to apply design and development principles in the construction of software systems of varying complexity.