

MV4202: Learning Objectives Summer 2000

The following is a list of instructional objectives for MV4202. The lectures, labs and tests are based upon these objectives. Each describes some action or task, which the student must be able to perform in order to successfully complete the course. Several include the conditions under which the action or task should be performed.

The purpose of these objectives is threefold. First, they provide a framework for the design of the course. Second, they allow both the student and the instructor to be able to determine whether the objectives of the course are being met. In this sense they should eliminate much of the guesswork for students when preparing for tests. Third, they should provide the students with a way to organize their efforts toward accomplishment of a known goal.

OpenGL Basics

1. Write a one-sentence description of the OpenGL API.
2. Identify the purpose of the following OpenGL associated libraries and identify the windowing system under which each would be used: GLU, GLX, WGL, GLUT, Motif and MFC.
3. Given a command from the OpenGL, GLU, GLUT, GLW, WGL or MFC libraries, identify the library to which it belongs.
4. Given a full name of a general OpenGL function (command), based on that name describe the number and type of arguments to be passed to that function.
5. Given the normal and vector forms of an OpenGL command, identify which is more efficient.
6. List five modes or variables, which are part of the OpenGL state machine.
7. Describe in one or two sentences the difference between a mode and a variable.

Basic Rendering

1. Draw a picture of the standard graphics coordinate system used by OpenGL and most graphics APIs'.
2. List the three graphics primitives used to model objects in OpenGL.
3. List three characteristics a polygon should have to insure proper rendering with OpenGL.
4. State how OpenGL differentiates between the front and back faces of polygons.
5. Define *culling* and describe how polygon face culling can affect the manner in which the front and back faces of polygons are rendered.
6. Given a description of a simple object, write a method to draw that object.
7. Identify OpenGL commands in a list that can and can not appear between a *glBegin/glEnd* pair.
8. State the default widths of points and lines. Describe how this will affect their apparent sizes under varying screen resolutions.
9. Give a definition for *stipple patterns* and state their intended purpose.
10. State the purpose of *normal vectors*.
11. State the required length of a normal vector for use in lighting calculations.
12. Given the description of a model, state whether the normal vectors of two co-located vertices, which are common to two adjacent polygons, should be distinct or averaged.
13. State the reason for making use of vertex arrays
14. List five types of data that may be contained in a *vertex array*.

Color

1. Name two modes, which may be used to specify color under OpenGL.
2. List the colors emitted by a standard computer monitor.
3. List the colors of light which the different types of cone cells in the eye respond to best
4. Identify which color specification method should be used with each of the following: texture mapping, lighting, smooth shading, fog, antialiasing and blending.

5. List the two types of shading available in OpenGL. Identify which should be used when trying to achieve lighting effects.
6. Describe how the color of a polygon is determined under *flat shading*.
7. Describe how the color of a polygon is determined under *smooth shading*.

Transformations

1. Write the name of the rule used to determine the sign of a rotation in OpenGL.
2. List the transformations that must be carried out to produce *world coordinates*, *eye coordinates*, *clip coordinates*, and *window coordinates*.
3. List four transformations, which are carried out on each vertex during the rendering process.
4. In terms of OpenGL statements in a graphics application, give the order in which the following transformations should be specified: modeling, viewing, and projection.
5. In terms of matrix operations state the order in which the following transformations are actually applied to each vertex: modeling, viewing, and projection.
6. Given a description of a position and orientation and several OpenGL transformation commands (*glTranslate*, *glRotate*, and *glScale*), put the commands in the order required to achieve the desired position and orientation.
7. Describe two or more ways to achieve the same orientation and location using different axes and angles of rotation.
8. Given multiple lists of *glTranslate*, *glRotate*, and *glScale* commands, identify which describe the same orientations and locations.
9. Given a description of a position and orientation and the matrices created by several OpenGL transformation commands (*gluPickmatrix*, *glPerspective*, *glOrtho*, *glTranslate*, *glRotate*, and *glScale*), write an equation which puts the matrices in the order required to transform a vertex to the desired position and orientation.
10. Given a sequence of transformations and stack operations, draw a picture that illustrates the state of the stack as the sequence progresses.
11. List the two basic types of Projection transformations.
12. Identify the principle characteristics of an orthogonal projection.
13. Identify the principle characteristics of a perspective projection.
14. Describe the shape of the view volume associated with a perspective projection.
15. Describe the shape of the view volume associated with an orthogonal projection.
16. Define *clipping* and state the purpose of a clipping plane.
17. Write an equation, which defines *aspect ratio*.
18. Describe what relation is required between the aspect ratios associated with the *projection* and *viewport transformations* in order to produce an undistorted view of a "scene".
19. State the type of units in which the size of the viewport is specified.

Display Lists

1. State the purpose of an OpenGL display list.
2. Identify what type of modifications can be made on a display list once it has been constructed.
3. Name and describe the two modes under which the commands of a display list can be executed when it is created.
4. Describe the difference between *glCallList* and *glCallLists* in one or two sentences.
5. Write a short segment of code to create and execute a hierarchical display list.

Font Rendering

1. List the font rendering capabilities provided by the OpenGL and GLU libraries.
2. List the four basic steps needed to generate individual display lists for each glyph of a font in the win 32 environment.
3. Identify the advantages and disadvantages of bitmap and true type fonts.
4. Describe when a bitmap font will and will not appear on the screen based on the raster position.

Selection

1. List the three OpenGL *rendering modes* and describe in one or two short statements what occurs in each mode.
2. State the purpose of a *picking matrix* and how it is specified and used.
3. List the elements of a *hit record* and describe each.
4. Describe in general terms how hit records are generated.
5. State whether or not the contents of the color buffer are altered while in the *selection* rendering mode.

Lighting

1. List the four OpenGL material properties
2. List the three types of light “emitted” by an OpenGL light source.
3. Write a correct definition of ambient light.
4. Write a correct definition of diffuse light.
5. Write a correct definition of specular light.
6. Identify factors that determine the intensity of the ambient light component for a particular vertex.
7. Identify factors that determine the intensity of the diffuse light component for a particular vertex.
8. Identify factors that determine the intensity of the specular light component for a particular vertex.
9. List the three factors with which the OpenGL lighting model is concerned.
10. Describe how the specification of a *local* or *infinite viewpoint* affects specular highlight calculations. State, which type of viewpoint produces more efficient calculations.
11. Correctly answer questions concerning the *emission* material property regarding when objects with an “emissive” color are visible, when the property is commonly used, under what circumstances objects with an emissive color become a light source.
12. Write the necessary code to correctly set the direction of a directional light source.
13. Write the necessary code to correctly position a positional light source.
14. Write the necessary code to correctly position and direct a “spotlight”. Correctly set the *cutoff angle*.
15. State what transformations should be on the modelview stack when positioning a light source that is tied to the viewpoint.
16. State what transformations should be on the modelview stack when positioning a stationary light source.
17. State what transformations should be on the modelview stack when positioning a light source that is moving or has a location, which is relative to objects in the scene.
18. Describe the purpose of an attenuation factor.
19. State what is determined by the shininess exponent.
20. Correctly assign normal vectors to the vertices of an object.

Frame Buffer

1. List four buffers, which may be contained in the *frame buffer*.
2. Name and describe the primary component used to “construct” a frame buffer.
3. State how *pixel depth* is determined.
4. State how *color depth* is determined.

Hidden Surface Removal

1. List two algorithms used for hidden surface elimination.
2. Write the three basic steps of the *Z buffer algorithm*.
3. Describe the primary limitation of the Z buffer algorithm in one or two sentences

Texture Mapping

1. State the primary purpose of texture mapping using one of two sentences.
2. Name the discrete elements of which a texture map is composed.
3. State in what coordinate space the rectangular array of *texels* is said to lie.
4. List the coordinate components of *texture coordinate space*.
5. State the purpose of a *texture border*.
6. Describe how *mipmaps* are used.
7. State the difference between *minification* and *magnification*.
8. State what is determined by the *texturing mode*.
9. Given a description of a simple texturing problem, correctly set up the texture environment and assign the texture coordinates.