DeVry University
Curriculum Guide

For all locations and online:

GSP-380 Multimedia Programming with Lab

Contact Hours: 5 Credit Hours: 4

Course Description:

This course introduces programming principles and techniques used to integrate graphics, animation and sound assets into reactive environments that encourage interactivity and engage the end-user or player. Principles of asset management and asset conversion are also covered. Prerequisites: GSP-260 and GSP-340 / 5-4

Author:	Rick Blunt
Date:	November 27, 2006

Revised By:
Rev. Date:

Course Strategy:

SCOPE

This course is designed as the students' first exposure to multimedia programming and is the only dedicated course that provides such. This course builds on the structured programming concepts of the previous courses, CIS-170 (A, B, or C) Programming with Lab, GSP-260 Video and Audio Design Fundamentals with Lab, and GSP-340 Modification and Level Design with Lab. Students should be comfortable programming in C++ and be able to make a basic program on their platform of choice. A basic understanding of 3D math (primarily vectors and matrices) will also be helpful.

The course uses two textbooks that cover two different multimedia authoring technologies, DirectX and OpenGL. Since the majority of the market uses one, or both, of the two tools, this will ensure students are familiar with, and prepared for future work environments regardless where they go. Although teaching with two books and two systems can be challenging, it will eventually ensure the marketability of the students.

LEVEL

In this course, it is assumed that students never programmed multimedia before. The emphasis is on robust development and good programming skills regardless of which authoring system, DirectX or OpenGL, is used. The students will understand basic methods of image processing and sound processing. They will be acquainted with compression and decompression methods for image and sound data in PC environment. They will also know how to exploit programmers interfaces' for image and sound and get acquainted with data formats for image and sound representation and with basic features of digital signal processors. They will practice raster operations in Windows API, DirectX, and OpenGL and with sound interfaces of Windows API and DirectX, too. Finally, they will be acquainted with the anticipated development of multimedia data processing.

RATIONALE

Graphics APIs such as OpenGL and DirectX have become commonplace throughout the graphics and game worlds, and understanding how they work, their limitations, and ways to work around these limitations is important when writing and extending complex applications. Most PC game engines will use either DirectX or OpenGL as an actual low level rendering technology.

DirectX is a collection of APIs for handling tasks related to multimedia, especially game programming, on Microsoft platforms. It is widely used in the development of computer games for Microsoft Windows, the Xbox and Xbox 360. DirectX has also become more widely used among other software production industries such as the engineering and simulation sectors because of its ability to quickly render high-quality 3D graphics using the latest 3D graphics hardware.

OpenGL (**Open G**raphics **L**ibrary) is a standard specification defining a cross-language cross-platform API for writing applications that produce 3D computer graphics (and 2D computer graphics as well). The interface consists of over 250 different function calls which can be used to draw complex three-dimensional scenes from simple primitives. OpenGL was developed by Silicon Graphics and is popular in the video games industry where it competes with Direct3D on Microsoft Windows platforms. OpenGL is widely used in CAD, virtual reality, scientific visualization, information visualization, flight simulation and video game development.

TEACHING SUGGESTIONS

Methods of Instruction and Evaluation: To ensure students are involved as much as possible in the learning process, a variety of teaching and assessment methods will be used.

- Lectures with Classroom Discussion: Interactive presentation of course materials to include question and answer sessions and small group responses.
- Homework: Reading and written assignments to be assigned to each week. All assignments must be typed to be turned in. Please make sure students keep a copy of their work for themselves as assignments will be reviewed for completeness and will not be returned.
- Game Journal: Each week while in the GSP program, students are required to keep a journal of their game play. In this course they are required to play at least four hours of games each week. In their journal they will analyze, reflect, and detail their game play. Each week's journal entry must be at least 250 words long.
- Quizzes: Weekly quizzes may be given as a method to stimulate class discussion and to review the reading assignments to be discussed during the class period.
- iOptimize Threaded Discussions: The iOptimize shell will come with a number of suggested discussion questions for each week. Instructors may also create their own questions to customize the course to their areas of expertise.
- Exams: 1 Midterm and Final Exam
- Course Project: Each student is required to make a short multimedia game for grading. The following characteristics/attributes will be considered during grading:
  - Title
  - Description/feature set
  - Purpose/application
  - Premise/high concept
  - Genre
  - Platform
  - Audience/market
  - Rating
  - Player mode
  - Time interval
  - Backstory/synopsis
  - Character descriptions
  - Competitive analysis
  - Rules
  - Challenges
  - Strategies
  - Theory/game balance
  - Perspective/game view
  - User interface
  - Audio
- Team Presentations: Develop professional business presentations skills while fulfilling stated learning objectives. This written assignment will be uploaded to the Team Presentations folder of the Doc Sharing section of the DeVry eLearning Platform (DEP) the day before the class in which it is due:
  - Each presentation will be 10-15 minutes with 5-10 minutes for audience questions
  - Presentations will be in PowerPoint
  - Minimum 5 slides, maximum 20 slides
  - **All team members will participate in the oral presentation**

A	= Minimum use slides, good command of content, good conversational tone
B	= Occasional use of slides, some knowledge of the content, good conversational tone **(or)** absent from presenting but provided ample content/slides and full share of the project
C	= Frequent use of the slides, minimal knowledge of content, non-smooth speech

- Each presentation will be orally evaluated by the audience and Professor
- Use the template found in Doc Sharing

D = Reading the slides like it's the first time you've seen them. It's pretty obvious all you did was let the others do the work, you just showed up and are trying to get credit

F = Not presenting in class, no out-of-class-work.

- Software: *DirectX 9.0c* by Microsoft downloadable for free from
  - http://www.microsoft.com/downloads/details.aspx?FamilyId=2DA43D38-DB71-4C1B-BC6A-9B6652CD92A3&displaylang=en

Texts/References:

Beginning OpenGL Game Programming
David Astle & Kevin Hawkins
Premier/Thomson-Course, 2004
ISBN: 1592003699

Beginning DirectX9
Wendy Jones
Premier/Thomson-Course, 2004
ISBN: 1592003494

Course Terminal Objectives:

1. Given the appropriate software tools, create a demonstration game, simulation, or world environment that uses color and shading, models lighting with controls, uses blending, and obscures objects in the distance (fog).

   Suggested Enabling Objectives:

   A. Color
      1. Describe color in terms of emissions from a standard computer monitor and how the cone cells in the eye respond to color.
      2. Identify which color specification method should be used in various effects.
      3. Identify which color specification method should be used with each of the following: texture mapping, lighting, smooth shading, fog, antialiasing and blending.
      4. Describe how the color of a polygon is determined under flat shading.
      5. Describe how the color of a polygon is determined under *smooth shading*.

   B. Lighting
      1. Identify factors that determine the intensity of the ambient light component for a particular vertex.
      2. Identify factors that determine the intensity of the diffuse light component for a particular vertex.
      3. Identify factors that determine the intensity of the specular light component for a particular vertex.
      4. Describe how the specification or a local or infinite viewpoint affects specular highlight calculations. State, which type of viewpoint produces more efficient calculations.
      5. 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.
      6. State what transformations should be on the model view stack when positioning a light source that is tied to the viewpoint.
      7. State what transformations should be on the model view stack when positioning a stationary light source.
      8. Describe the purpose of an attenuation factor.
      9. State what is determined by the shininess exponent.
      10. Correctly assign normal vectors to the vertices of an object.

2. Given descriptions of three dimensional objects, render images from such models, that includes geometry, viewpoint, texture and lighting information, while applying filtering and resolution functions.

   Suggested Enabling Objectives:

   A. Basic Rendering
      1. Define culling and describe how polygon face culling can affect the manner in which the front and back faces of polygons are rendered.
      2. Explain the default widths of points and lines and how this will affect their apparent sizes under varying screen resolutions.
      3. Give a definition for stipple patterns and state their intended purpose.

4. Describe the purpose of normal vectors and the required length of a normal vector for use in lighting calculations.
5. 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.
6. State the reason for making use of vertex arrays and list five types of data that may be contained in a vertex array.

3. Given existing textures in 2D, 3D, and cubes with data arrays, map them onto primitives, modifying initial state values to select textures and using transformations to manipulate objects and projections, buffers to modify surfaces, and meshes for positioning.

Suggested Enabling Objectives:

A. Texture Mapping
1. State the primary purpose of texture mapping.
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.

B. 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.

4. Given a prerendered video track (video data list) and a prerendered audio track (audio data list), interleave the two data lists in such a way that when played, allow simultaneous playback throughout the game, animation, or movie.

Suggested Enabling Objectives:

1. Interleave multiple audio and video streams into a single stream
2. Demonstrate how to manage the buffering at the decoder
3. Demonstrate how to synchronize the streams on play back, and time identification for each of the streams.

5. Given existing content libraries, use 3-D modeling and animation software in developing 3-D computer models and creating presentations of those models in story-telling motion-laden scenes.

Suggested Enabling Objectives:

A. Transformations
1. List the transformations that must be carried out to produce world coordinates, eye coordinates, clip coordinates, and window coordinates.

2. In terms of matrix operations, state the order in which the following transformations are actually applied to each vertex: modeling, viewing, and projection.
3. Describe two or more ways to achieve the same orientation and location using different axes and angles of rotation.
4. Given multiple lists of commands, identify which describe the same orientations and locations.
5. Given a sequence of transformations and stack operations, draw a picture that illustrates the state of the stack as the sequence progresses.
6. List the two basic types of Projection transformations. An ability to apply knowledge of basic 3-D modeling and scene creation principles
7. An ability to design 3-D scenes to meet desired needs and tell an effective story
8. An ability to identify, formulate and solve issues and problems involved in creating effective 3-D motion, models and scenes
9. An ability to identify, formulate and solve issues and problems involved in creating effective 3-D models, motion, and scenes

6. Given the appropriate software tools and the need to display 2D text, such as for captions for characters, display and position 2D text in a multimedia environment such as a simulation or game.

Suggested Enabling Objectives:

A. Font Rendering
   1. List the four basic steps needed to generate individual display lists for each glyph of a font in the win 32 environment.
   2. Identify the advantages and disadvantages of bitmap and true type fonts.
   3. Describe when a bitmap font will and will not appear on the screen based on the raster position.

7. Given an existing 3D animation, add audio appropriately tied to the animation’s story, gameplay structure, and user interface.

Suggested Enabling Objectives:

A. Compare digital and MIDI sound.
B. Describe sound hardware.
C. Work with primary and secondary sound buffers.
D. Identify features associated with the game’s story, gameplay structure, interface, and use of audio.
E. Develop a general understanding of audio formats and hardware structure.
F. Explain how to modify a musical script or a sound effect (such as an engine revving up or down) in response to game events in real time, i.e.: the beat of the music could quicken as the action heats up.

8. Given input devices such as a mouse, a keyboard, and a joystick, use the input devices with a stock game engine to control the connected animated environment.

Suggested Enabling Objectives:

A. Differentiate between keyboard/button type input (on/off), mouse input which is relative to screen coordinates and joystick type input that have a range of motion.
B. Demonstrate how action mapping allows for assigning specific actions in a game to the buttons and axis on your input devices.
C. Differentiate between DirectX and Direct Input.
D. Evaluate the use of game engines vs. DirectX vs. front-end visual game design tools in the design of game user inputs.
E. Design effective inputs, outputs and interfaces
F. Input and output design
   1. The design boundary
   2. Output design and technology
   3. Input design and technology
   4. Selection of appropriate output and input technologies

9. Given a 3D environment describe a 3D world and how to represent a virtual camera that mathematically describes the perspective from which the world is viewed in terms of the volumes of space the camera sees.

Suggested Enabling Objectives:

A. Demonstrate field of view expertise
B. Demonstrate aspect ratio
C. Demonstrate position, look or view direction, up or roll
D. Demonstrate near and far clip planes
E. Demonstrate first vs. 3rd person camera issues
F. Describe the shape of the view volume associated with a perspective projection.
G. Describe the shape of the view volume associated with an orthogonal projection.
H. Define clipping and state the purpose of a clipping plane.
I. 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."
J. learn the steps necessary to take a 2D "picture" of the 3D world based on what the camera "sees", these steps as a whole are referred to as the *rendering pipeline*.
K. The Rendering Pipeline
   The Rendering Pipeline
   1. Chapter 12 Building a Flexible Camera Class
   2. Camera Design
   3. Implementation Details
   4. Camera Sample

10. Given an existing 3D environment, create a 3D character who exhibits ranges of motion for body joints, key frame animation, implementation of such as the ability to perform a complex sequence of activities that have been built up from the basic motions. The character animation should include smooth interpolation of body joint regions when characters are in motion, blending of different motion sequences, footstep locomotion, motion capture capabilities (using actors outfitted with special hardware), modeling of the movement of cloth and hair, physics of character interaction (bumping into each other or falling down), manipulation of various props and objects.

Suggested Enabling Objectives:

Course Terminal Objectives (continued):

A. Get the character speed right to avoid “slipping”, surface/terrain collision detection
B. Discussing blending animations & interpolation.
C. Compare the foundations of 3D animation and modeling, programming, production, and artificial intelligence.
D. Visualize, construct, and control a 3D character model figure.
E. Visualize and implement a game design plan that describes the controls, interface and artistic style of an electronic game.
F. Create and port a low-polygon model to a game format.
G. Create a 3D game animation with real-time textures.
H. The terms that are often used to describe these complex actions are *primitive motions*, *skills*, and *behaviors*. A *primitive motion* may simply involve rotating a character’s wrist joint.
I. Other complex actions using the simulator programming extensions. A *skill* may involve a simple sequence of primitive motions, such as reaching for and grasping a tool. A *behavior* may represent a much more complex set of *skills*, such a removing a spark plug from an engine.
J. Visual Fidelity - Visual or image fidelity is a function
   1. Color depth
   2. Size (physical size and resolution)
   3. Shading fills and light focus
   4. Frame rate …(motion)
   5. Depicted detail

Content Outline — Suggested Time Allocations

I. Course Introduction — 5%

A. Welcome, Course Overview, Syllabus, & Course Expectations
B. iOptimize demo, walkthrough and setup
C. Course projects overview
D. Quiz*
E. Lab:
   1. DirectX Software Developer Kit overview and setup
A. Online:
   1. Threaded discussions
   2. Quiz*

II. Week 2: Introduction to Multimedia — 7%

A. Reading and Classroom
   1. Astle & Hawkins Introduction
   2. Astle & Hawkins Chapter 1: The Exploration Begins... Again
   3. Jones Introduction
   4. Jones Chapter 1: The What, Why, and How of DirectX
   5. Quiz*
B. Lab:
   1. Creating Your First DirectX Window
   2. Introduction to OpenGL
C. Online:
   1. Threaded discussions
   2. Quiz*

III. Week 3: Color, shading, and lighting — 7%

A. Reading and Classroom:
   1. Astle & Hawkins Chapter 5: Colors, Lighting, Blending, and Fog
   2. Jones Chapter 6:Vertex Colors, Texture Mapping, and 3D Lighting
   3. Quiz*
B. Lab:
   1. DirectX Lighting
   2. OpenGL Depth Testing and Lighting
C. Online:
   1. Threaded discussions
   2. Quiz*

IV. Week 4: Transformations, primitives and buffers — 7%

A. Reading and Classroom:
   1. Astle & Hawkins Chapter 3: OpenGL States and Primitives
   2. Astle & Hawkins Chapter 4: Transformations and Matrices

Content Outline (continued):	Suggested Time Allocations

3. Astle & Hawkins Chapter 12: OpenGL Buffers
4. Jones Chapter 4: 3D Primer
5. Jones Chapter 5: Matrices, Transforms, and Rotations
6. Quiz*

B. Lab:
1. DirectX Transformations
2. OpenGL Transformations

C. Online:
1. Threaded discussions
2. Quiz*

V. Week 5: Camera and Spatial Volumes — 7%

A.Reading and Classroom:
1. Jones Chapter 5: Matrices, Transforms, and Rotations
2. Jones Chapter 4: 3D Primer
3. Quiz*

B. Lab:
1. OpenGL Using a camera and simple user input

C. Online:
1. Threaded discussions
2. Quiz*

VI. Week 6: Rendering and Textures — 8%

A. Reading and Classroom:
1. Astle & Hawkins Chapter 2: Creating a Simple OpenGL Application
   Astle & Hawkins Chapter 7: Texture Mapping
2. Jones Chapter 6: Vertex Colors, Texture Mapping, and 3D Lighting
3. Quiz*

B. Lab:
1. DirectX Rendering Primitives
2. OpenGL Texture Mapping

C. Online:
1. Threaded discussions
2. Quiz*

VII. Week 7: Fonts — 7%

A. Reading and Classroom:
1. 2D text
2. Astle & Hawkins Chapter 11: Displaying Text

B. Quiz*
1. Lab:
2. 2D and 3D Fonts

C. Online:
1. Threaded discussions
2. Quiz*

Content Outline (continued): | Suggested Time Allocations

VIII. Week 8: Midterm Exam — 4%

A. Reading and Classroom:
   1. Midterm Exam
B. Lab:
   1. Vertex and Index Buffers
C. Online:
   1. Threaded discussions
   2. Quiz*

IX. Week 9: Audio — 7%

A. Reading and Classroom:
   1. Jones Chapter 10: DirectSound
B. Lab:
   1. Animations
C. Online:
   1. Threaded discussions
   2. Quiz*

X. Week 10: Interleaving — 7%

A. Reading and Classroom:
   1. Astle & Hawkins Chapter 10: Up Your Performance
   2. Quiz*
B. Lab:
   1. Timing and Frames Per Second
C. Online:
   1. Threaded discussions
   2. Quiz*

XI. Week 11: Input Devices — 7%

A. Reading and Classroom:
   1. Jones Chapter 9: Using DirectInput
   2. Quiz*
B. Lab:
   1. DirectX's DirectInput
   2. OpenGL Simple User Input
C. Online:
   1. Threaded discussions
   2. Quiz*

XII. Week 12: 3D Motion — 8%

A. Reading and Classroom:
   1. Jones Chapter 3: Surfaces, Sprites, and Salmon

Content Outline (continued): | Suggested Time Allocations

2. Jones Chapter 4: 3D Primer
3. Jones Chapter 5: Matrices, Transforms, and Rotations
4. Quiz*

B. Lab:
1. DirectX Moving Around in a 3D World

C. Online:
1. Threaded discussions
2. Quiz*

XIII. Week 13: Storytelling — 8%

A. Reading and Classroom:
1. Story-telling presentations
2. Quiz*

B. Lab:
1. Prototype Presentations

C. Online:
1. Threaded discussions
2. Quiz*

XIV. Week 14: Course Review and Prototypes — 7%

A. Reading and Classroom:
1. Course review
2. Quiz*

B. Lab:
1. Project Presentations

C. Online:
1. Threaded discussions
2. Quiz*

XV. Week 15: Final Exam and Course Projects — 4%

A. Reading and Classroom:
1. Final Exam

B. Lab:
1. Game Open House

C. Online:
1. Threaded discussions
2. Quiz*

*Quizzes in class or online at the instructor's preference.

Laboratory

I. Week 1

A. DirectX SDK
   1. The latest version of the DirectX SDK is now available. New features in this release include HLSL shader debugging within PIX, a preview release of Direct3D 10 HLSL shader compilation for Direct3D 9 targets, plus updates to XACT, XInput, and UVAtlas.
   2. This DirectX SDK release contains updates to tools, utilities, samples, documentation, and runtime debug files for x64 and x86 platforms. This release also includes a public pre-release Direct3D 10.

II. Week 2

A. Creating Your First DirectX Window
   1. We can't make cool graphics without a window, now can we? In this tutorial we'll learn how to create a basic window and how to respond to messages sent to the window
B. Introduction to OpenGL
   1. Initialization of OpenGL
   2. Drawing a triangle

III. Week 3

A. DirectX Lighting
   1. Add more realism to your games with lighting! In this tutorial, you'll learn about the lighting system in DirectX and how to incorporate it into your programs.
B. OpenGL Depth Testing and Lighting
   1. Enabling depth testing
   2. Enabling lighting
   3. Specifying the lights

IV. Week 4

A. DirectX Transformations
   1. In this tutorial you'll learn how to animate geometry with the three basic transformations: translation, rotation, and scale.
B. OpenGL Transformations
   2. Applying several transformations to an object
   3. Getting a smooth animation (double buffering)
   4. Combining several transformations
   5. Using the matrix stack

V. Week 5

A. OpenGL Using a camera and simple user input
   1. Writing a class for simulating a camera in OpenGL
   2. Getting some user input with GLUT

VI. Week 6

- A. DirectX Rendering Primitives
    1. In this tutorial, you'll learn about vertex declarations and how to render basic geometry.
- B. OpenGL Texture Mapping

VII. Week 7

- A. 2D and 3D Fonts
    1. Displaying text is vital to every game. With the help of the D3DX library, creating text is easy! In this tutorial you'll learn how to create 2D and 3D text using ID3DXFont and ID3DXMesh.

VIII. Week 8

- A. Vertex and Index Buffers
    1. In this DirectX tutorial, you'll learn how to speed up rendering performance through the use of vertex and index buffers.

IX. Week 9

- A. Animations

X. Week 10

- A. Timing and Frames Per Second
    1. In this DirectX tutorial, we'll create a timing class, which will be crucial in performing all of our animations.

XI. Week 11

- A. DirectX's DirectInput
    1. What would a video game be without user interaction? In this tutorial, you'll learn how to use DirectInput to process mouse and keyboard input.
- B. OpenGL Simple User Input
    1. Getting user input with GLUT
    2. Double buffering
    3. User input

XII. Week 12

- A. DirectX Moving Around in a 3D World
    1. Let's start moving around the game. In this tutorial, you'll learn how to move and rotate a camera using DirectInput and the always helpful D3DX library.

XIII. Week 13

- A. Prototype Presentations
    1. Students present their Project prototype to peer and instructor critique

XIV. Week 14

A. Project Presentations
   1. Students present their final multimedia projects to peer and instructor critique and evaluation

XV. Week 15

A. Game Open House
   1. Students present their final multimedia projects to campus faculty, administration, admissions, and other students (time permitting if all Project presentations completed in Week 14 Lab)