



COURSE SPECIFICATION			
NAME OF COURSE: Introduction to programming in C and OpenGL		COURSE CODE:	
STATUS: (main,optional, Free Choice) Preparatory Course	LEVEL: (F,A,P,1,2,3,M)	UNIT VALUE:	TERMS TAUGHT:
Department offering course:	Course Co-ordinator: Silvester Czanner Martin Samuelcik	Date of course commencement:	
Degree Programmes in which to be offered:			
Pre-requisites:	Indicate whether a new course or name of course being replaced: new	Total Contact Hours: 40	
AIMS OF THE COURSE: This course aims to provide the student with an introductory knowledge of programming in C language using OpenGL functionality.			

INTENDED LEARNING OUTCOMES

1. Basic knowledge of programming in ANSI C language
2. Basic knowledge of programming computer graphics routines using OpenGL functionality
- 3.
- 4.



LEARNING AND TEACHING STRATEGIES TO BE USED:

1. 3 Lectures to explain and to illustrate the concepts
2. 1 tutorial to explain writing, compiling and executing programs written in C language using OpenGL functionality
1. 11 lectures to explain and to illustrate the concepts
2. 5 tutorials to explain writing, compiling and executing programs written in C and OpenGL.
3. Laboratories and example case studies
- 3.
- 4.



ASSESSMENT CRITERIA (SHOULD LINK EXPLICITLY TO INTENDED LEARNING OUTCOMES):

1. 20% results from the tutorials
2. 30% results from the assignments 1-4
3. 50% individual project (could be based on group projects, but everybody must clearly demonstrate the contribution)???
- 4.
- 5.
- 6.

TRANSFERABLE SKILLS AND OTHER ATTRIBUTES

1. Ability to design efficient individual graphics programs using OpenGL
2. Creativity of environments
3. Organizational skills



LEARNING AND TEACHING STRATEGIES USED:

1. Individual practical work
2. Lectures and tutorials
3. Project and time management advice



ASSESSMENT CRITERIA (SHOULD LINK EXPLICITLY TO INTENDED LEARNING OUTCOMES):

1. Practical exercises
- 2.
- 3.

Comment [U1]: How about we grade them on the basis of one single project? They've seen bits and pieces throughout the tutorials; they now have to come up with a setting that allows them to use as much of it as they can think of, in a coherent way, and of course adding whatever they might come up with.

That way we grade 1. The assimilation of techniques taught in class. 2. Their capacity to design their own program, evaluating available resources (time and knowledge). 3. Their capacity to create new things with the building blocks taught in class.

COURSE OUTLINE/SYLLABUS:

Lectures

1. day **Introduction to C language**
 1. lecture (1 hour)
 - a. Program Structure
 - b. Program Control
 - c. Assignment & Logical Compare
 - d. Functions, Variables, & Prototyping
 2. lecture (1 hour)
 - a. The C Pre-processor
 - b. Strings and Arrays
 - c. Pointers
 3. lecture (1 hour)
 - a. Standard Input/Output
 - b. File Input/Output
 - c. Structures
 - d. Dynamic Allocation
 4. tutorial (2 hours)
 - a. writing a program (source code) in C language
 - b. using C compiler, compile the source code
 - c. linking libraries to the C program and create an executable
 - d. running the executable
 5. exercises and group projects (3 hours)
 - a. Writing C programs according the assignments

Comment [U2]: This is the part that worries me the most. I have no experience teaching OpenGL to students who don't know C, and I doubt you can teach enough material in a few hours so that they can go ahead and build small programs. I guess this depends a lot on the Entry Requisites that we define...

2. day **Introduction to OpenGL**

1. lecture (1 hour)
 - a. Introduction
2. lecture (1 hour)
 - a. Colors
 - b. Viewing
 - c. Primitives
3. lecture (1 hour)
 - a. Interaction and GLUT
4. Tutorial (2 hours)
 - a. writing C program using OpenGL and GLUT
 - b. using C compiler, compile the source code
 - c. linking OpenGL libraries to the C program and create an executable
 - d. running the executable
5. exercises and group projects (3 hours)
 - a. Writing OpenGL programs according the assignments
 - b. Using the source code from tutorial

3. day **2D images and bitmaps in OpenGL**

1. lecture (1 hour)
 - a. Images and Bitmaps I
2. lecture (1 hour)
 1. Images and Bitmaps II
3. tutorial (1 hour)
 1. Drawing digital images
 2. Text in OpenGL
4. tutorial (2hours)

1. Bitmaps
2. Mixing bitmaps and geometry
3. Manipulating pixels, writing, reading, zooming
4. Import images from different formats
5. exercises and group projects (3 hours)
 1. Writing OpenGL programs according the assignments
 2. Using the source code from tutorials

4. day **3D modeling in OpenGL**

1. lecture (1 hour)
 1. 3D modeling
 2. Transformations
2. lecture (1hour)
 1. Viewing
 2. Projections
3. lecture (1 hour)
 1. GLUT objects
 2. Hierarchical modelling
4. tutorial (2 hours)
 1. 3D object modelling
 2. applying different types of transformation
 3. applying different types of projections
 4. hierarchical modeling
5. exercises and group projects (3 hours)
 1. Writing OpenGL programs according the assignments
 2. Using the source code from tutorials

5. day **Shading, Reflection models and Textures in OpenGL**

1. lecture (2 hours)
 1. Light, Shading, Reflection Models, Transparency
2. lecture (2 hours)
 1. Textures
3. tutorial (1 hour)
 1. Light sources
 2. Phong reflection model
 3. Texture mapping
4. exercises and group projects (3 hours)
 1. Writing OpenGL programs according the assignments
 2. Using the source code from previous tutorials

Exercises and Group projects

Exercise 1:

This exercise is designed as an introduction to programming with OpenGL. In it students will create a very simple OpenGL application. By the time they complete it, they should understand the basic framework of an OpenGL/GLUT program. They can use parts of the source codes from programs released from tutorials. The code they write will also serve as a useful starting point for future exercises.

Student's task:

Create an OpenGL program using GLUT that does the following:

1. Displays an OpenGL window with size 500x500 pixels.
2. At the beginning it draws a smiling face on yellow background (see Picture 1).



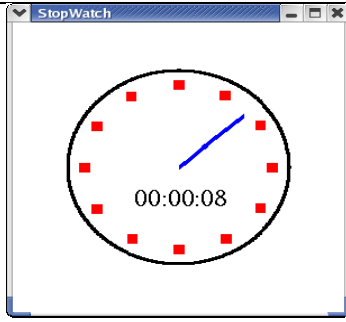
3. Clicking the right mouse button the face will switch to angry face (see Picture 2).
4. Clicking the left mouse button the face will switch to smiling face.
5. Pressing "B" or "b" key the face will switch to angry face.
6. Pressing "G" or "g" key the face will switch to smiling face.
7. Pressing Escape closes the window and end program execution.

Exercise 2:

This exercise is designed as an introduction to 2D animations using OpenGL. Students will create a very simple OpenGL animation. By the time they complete it, they should understand the callback functions, 2D modeling, bitmap fonts and the usage of the timer function in OpenGL/GLUT program. You can use parts of the source codes from tutorials.

Student's task:

Create an OpenGL program using GLUT that does the following:



1. Displays an OpenGL window with size 500x500 pixels.
2. At the beginning it draws a 2D stopwatch on white background (see Picture).
3. Clicking the left mouse button the pointer will start to move in clockwise direction and the stopwatch will start to count time in seconds. (function START)
4. Clicking again the left mouse button, the stopwatch will stop counting the time (function PAUSE).
5. Clicking the right mouse button, the time on the stopwatch will reset to 00:00:00, the pointer returns to starting vertical position. (function RESET)
6. Pressing "S" or "s" key do the function START
7. Pressing "P" or "p" key do the function PAUSE
8. Pressing "R" or "r" key do the function RESET
9. Pressing Escape closes the window and end program execution.

Exercise 3:

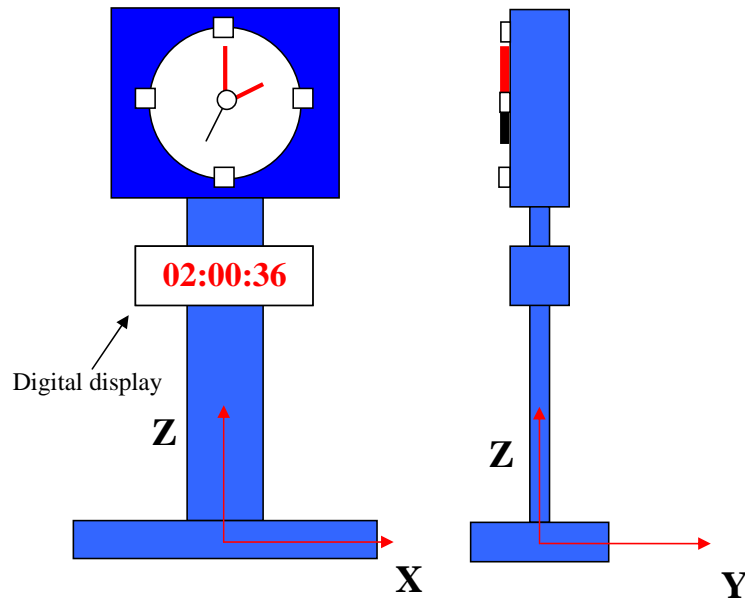
This exercise is designed as an introduction to 3D modeling using OpenGL. Students will create an OpenGL interactive program. By the time they complete it, they should understand the mouse motion callback functions, 3D modeling, bitmap fonts and the usage of the timer function in OpenGL/GLUT program. They can use parts of the source codes from tutorials.

Student's task:

Create an OpenGL program using GLUT that does the following:

1. Displays an OpenGL window with size 500x500 pixels.
2. At the beginning it draws a 3D street orientation clock on white background (see Picture). The clock model must contain minimum 4 parts (objects).

3. Movements with clock model are based on implementation of a simple version of virtual trackball. The user should be able to rotate the clock model in Z and Y directions using mouse motion callback.
4. Clicking the left mouse button the clock will start to count a time. (function START)
5. Clicking the right mouse button, the time on the clock will reset to 00:00:00. All three pointers return to starting vertical position. (function RESET)
6. Pressing "S" or "s" key do the function START
7. Pressing "R" or "r" key do the function RESET
8. Pressing Escape closes the window and end program execution.
9. Extra points: Display and run the digital clock display.
10. Extra points: Setup the time on the clock display with clicking on the pointers and move them with mouse motion.



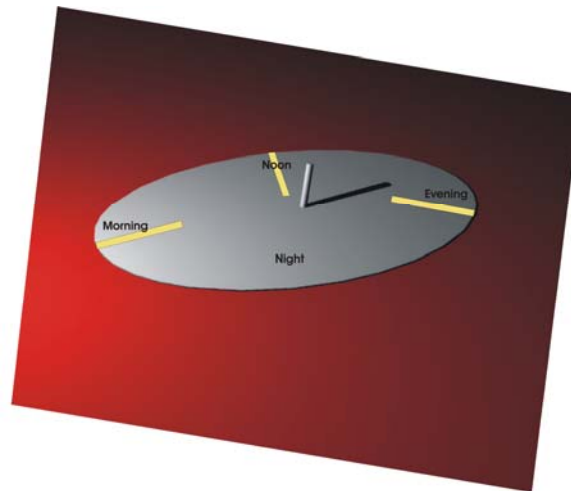
Exercise 4:

This exercise is designed as an introduction to 3D modeling using OpenGL. In it students will create an OpenGL interactive program. By the time they complete it, they should understand the mouse motion callback functions, 3D modeling, bitmap fonts and PHONG reflection model in OpenGL/GLUT program. Students can use parts of the source codes from tutorials.

Student's task:

Create an OpenGL program using GLUT that does the following:

1. Displays an OpenGL window with size 800x600 pixels.
2. At the beginning it draws a 3D street sun clock (see Picture for illustration).
3. The functionality of the sun clock is based on Phong reflection model.
4. Movements with sun are based on implementation of a simple version of virtual trackball. The user should be able to rotate the sun around the clock using elliptic trajectory.
5. The clock will show the approximate time by pointing the shadow of the clock's hand at some labels (morning, noon, evening).
6. Pressing Escape closes the window and end program execution.
7. Extra points: add into the sun clock scene a background image with some scenery that the clock will look as a part of the scenery.



Comment [U3]: I'm very happy in general with these tutorials. This is, I think, the right level that we can realistically achieve. That's why I believe other stuff like GPGPU, multithreading... can make great side talks, at high level, just to give the students an idea of other things that can be done. But focusing on these tutorials right here.

KEY TEXTS AND/OR OTHER LEARNING MATERIALS:

The Red Book: OpenGL Programming Guide

It's online at <http://www.opengl-redbook.com/code/> but we should have a hard copy or two.

The Blue Book: The OpenGL Reference Manual

Edward Angel, Interactive Computer Graphics (a top-down approach with OpenGL), Third Edition, Addison-Wesley, 2002

Edward Angel, OpenGL A Primer, Addison-Wesley, 2002

Kernighan, Ritchie, The C programming language, second edition

Wright, Lipchak, Haemel, OpenGL Superbible, fourth edition, 2007