**Department of Computing**

**CS-361: Computer Graphics  
  
Class: BSCS-12ABC & SE12AB**

**Lab 09: Implementing Texture Mapping**

**CLO 2 -** Apply mathematical and algorithmic principles to implement basic computer graphics techniques, such as line drawing and shading.

**CLO 3-** Develop interactive graphics applications using modern graphics APIs such as OpenGL or DirectX.

**CLO 4 -** Design and implement 2D and 3D graphical solutions for real-world problems.

**Date: 08th April 2025**

**Time: 2:00 PM – 4:50 PM**

# Instructor: Dr. Sidra Sutana

# Lab Engineer: Mr. Aftab Farooq

**Lab 09: Implementing Texture Mapping**

**Introduction:**Texture mapping is a technique used in computer graphics to enhance the visual realism of 2D and 3D objects by applying images or patterns (textures) to their surfaces. It involves mapping texture coordinates onto geometric shapes to create detailed visual effects without increasing geometric complexity.

### **Lab Objective:**

The aim of this lab is to introduce students to the concept of texture mapping in computer graphics. Students will learn how to apply textures to geometric shapes and implement texture mapping using a graphics programming environment.

## Tools/Software Requirement:

* **Operating System:**
  + Windows / macOS / Linux (Ubuntu recommended)
* **Development Environment:**
  + **Windows:** [Code::Blocks](http://www.codeblocks.org/) or [Visual Studio](https://visualstudio.microsoft.com/)
  + **macOS:** [Xcode](https://developer.apple.com/xcode/)
  + **Linux:** GCC and g++ compilers
* **Graphics Libraries:**
  + **OpenGL** (built-in on macOS and Linux, available in Windows IDEs)
  + **GLUT** (OpenGL Utility Toolkit)
  + **GLEW** (OpenGL Extension Wrangler Library)
* **Package Manager (for macOS/Linux):**
  + **Homebrew** (macOS): brew install freeglut glew
  + **APT** (Linux): sudo apt-get install freeglut3-dev glew-utils
* **Compilers:**
  + **Windows:** MinGW (for Code::Blocks) or Microsoft C++ Compiler (for Visual Studio)
  + **macOS/Linux:** GCC/G++
* A programming environment (e.g., Visual Studio, PyCharm, or any IDE of your choice).
* A graphics library (optional, e.g., OpenGL, SDL, or a simple image library for saving images).
* Basic knowledge of Texture Mapping.

## Prerequisites :

 Familiarity with 2D and 3D coordinate system.   
  Knowledge of geometric transformations (translation, rotation, scaling).  
  Ability to work with image files and manipulate pixels

**Lab Tasks :**

## Task 1: Load and Display a Texture on a 2D Shape

Write a program using any language to :

* Load an external image file as a texture.
* Map the texture to a 2D rectangle (quad).
* Display the textured quad on the screen.

## Task 2: Texture Mapping on 3D Objects

1. Extend your program to map textures on basic 3D objects (e.g., cube or sphere).
2. Ensure proper texture coordinates for each face of the object.
3. Allow user interaction to rotate the 3D object using keyboard or mouse.

## Task 3: Advanced Texture Mapping Techniques.

1. Implement one or more of the following:
   * Procedural textures (e.g., checkerboard or gradient patterns)
   * Multi-texturing (applying more than one texture)
   * Dynamic texture updates (e.g., video or animation as texture)
2. Discuss the advantages and use cases of these techniques in your report.

### **Deliverables:**

 Compile a single word document by filling in the solution part and submit this Word file on LMS

 Include screenshots of the program outputs.

 Submit your Lab Word File and code files seperately on submission link.

# Lab Rubrics

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Lab Rubrics for (Lab-09:Implementing Texture Mapping) | | | | | |
|  | | | | | |
| **Sr.**  **No.** | **Assessment** | **Unacceptable (0 Marks)** | **Does Not Meet Expectations (1/2 Marks)** | **Meets Expectations (3/4 Marks)** | **Exceeds Expectations (5 Marks)** |
| **1** | **Illustrating the basic understanding of semantics and syntax**  **(CLO3, PLO5)** | The student did not submit any work.  OR  The student plagiarized the solution and/or used unfair means. | The student is unable to demonstrate the understanding of syntax of C language and is unable to write an executable code.  The student is not able to understand the structure of a program at all. | The student demonstrates some understanding of syntax of C language and is able to write a code with few errors.  The student is able to understand the structure but still learning the syntax. | The student demonstrates good understanding of syntax of C language and is able to write executable code without help  The student is able to understand the structure and is able to identify problems in the code  when introduced |
| **2** | **Software Tool Usage**  **(CLO4-PLO3)** | The student demonstrates a lack of understanding of tool usage.  Implementation has syntax/semantic/runtime errors, and the student is unable to debug and correct the errors.  The code has inadequate comments and variable names and does not adhere to the coding standards.  No Error handling has been performed.  Documentation is poorly structured. | The student demonstrates some understanding of tool usage.  The codes are correct in terms of their syntax, however, the program output is not always correct in all test cases.  The code has limited comments and inconsistent variable names and may not adhere to the coding standards.  Some Error handling has been performed.  Documentation is adequately structured. | The student demonstrates a good understanding of tool usage.  Furthermore, his/her coding is complete and functional, and the program output is correct in all test cases.  The code has sufficient comments and consistent variable names and reasonably adhere to the coding standards.  Adequate Error handling has been performed.  Documentation is well structured. |