SOM Problem Solution & Vibration simulation

ME-735 CGPM

Project Report

Group Number: 3

Saurabh Bajaj	**13D100008**
Nihar Mehta	**13D100011**
Mangesh Atpadikar	**13D100020**
Durgesh Ahirwar	**13D100037**

Faculty Advisor : S.S. Pande

Indian Institute of Technology Bombay,
Powai-400076

Contents

1.Objective:

To build software that simulates the problems of strength of materials like bending of beams,etc. and displays the graphical results.
Show animated graphical output for Vibrational Loading in 2D.

Features

Differential Equation of the problem is given as output.
Simulated Animation of Vibrations in the beam as expected to be seen in real life.

Languages used:

C++
Libraries: OpenGl, Win32 API

Platforms used:

CodeBlocks, GitHub

2. Algorithm

User Interface:

UI is developed in ***win32 api*** which offer incredible options for making it very user friendly and intuitive to use.

The Windows API, informally WinAPI, is Microsoft's core set of application programming interfaces (APIs) available in the Microsoft Windows operating systems. The name Windows API collectively refers to several different platform implementations that are often referred to by their own names (for example, Win32 API); see the versions section.The Windows API (Win32) is a very focused mainly on the programming language C in that its exposed functions and data structures are described in that language in recent versions of its documentation.

In win32 api everything like text, textbox, button all of them are defined as window.There is one main parent window named as “hwnd” and all other are child window of it.

Header files we need

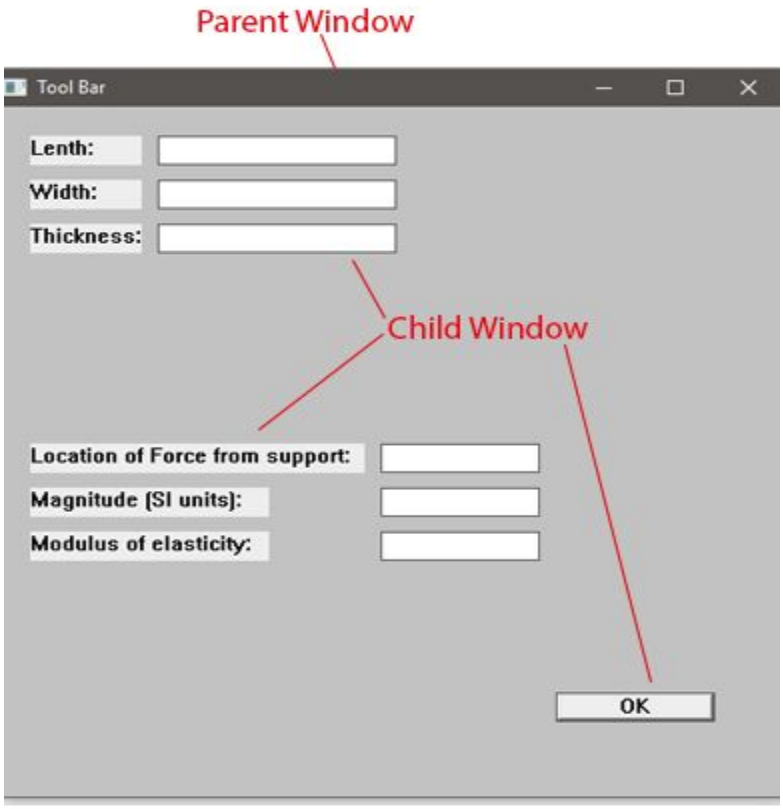
```
#include <windows.h>
#include <typeinfo>
```

How to define Parent Window

```
hwnd = CreateWindowEx (
   0,                          // Extended possibilites for variation
szClassName,                   // Classname
"Tool Bar",                    // Here we Title Text of the window
WS_MINIMIZEBOX | WS_MAXIMIZEBOX ,
                                 //WS_MAXIMIZE- allows to maximise window similarly
                                // WS_MINIMIZEBOX - The window has a maximize button
 CW_USEDEFAULT,                //Windows decides the position
 CW_USEDEFAULT,                //where the window ends up on the screen
 500,                           // The programs width
 500,                           //and height in pixels
HWND_DESKTOP,                  // The window is a child-window to desktop
 NULL,                         // No menu
hThisInstance,                 // Program Instance handler
 NULL                          // No Window Creation data
                              );
```

How to define Child Window

```
Child_name = CreateWindow("TYPE",                    //STATIC- For non-editable text
                                                     //EDIT- For textbox for taking input
                                                     //BUTTON- For button
                          "Text that will show",
                           WS_VISIBLE | WS_CHILD ,   //it is a child window
                           20,                        //Windows decides the position
                           20,                              //where the window ends up on the
screen
                           70,                        //width
                           20,                        //and height in pixels
                          hwnd ,                      //name of the parent window
                          NULL,              // If it is a button this entry help us to define button action
                          NULL,NULL);
```



DestroyWindow(location);

This function used to destroy the previously created window

GetWindowText(textboxX, &position1[0] ,20);

This is used to save input from the textbox

For eliminating invalid inputs " if " and "goto" functions is used with an error message that work in loop.

Integration:

We connected UI and Backend with help of text file. First we save the input from UI and save to the text file using ofstream.After that we open opengl window which read the value from text file using istream function and start solving the problem. All this happen on a click of button.

Graphics:

Platform used for coding is OpenGl with C++.

Following was implemented in order:

a. Taking Input values given by user in UI via a file.
b. Creating a window of size 1366x768 pixels.
c. Drawing initial state of given beam and force at appropriate location depending on whether it is cantilever or simply supported.
d. Drawing labels for x=0, x=L and force locations. Roller support and Fixed support is drawn for Simply supported. Wall is drawn for Cantilever.
e. Printing differential equations and their solutions.

Type of Problem	**Differential Equation**
Simply supported forced	$d^2y/dx^2 = -P(L-a)x/(EIL)$ for 0<= x <= a $d^2y/dx^2 = -P(L-a)x/(EIL) + P(x-a)/EI$ For a < x <=L
Cantilever forced	$d^2y/dx^2 = P(x-a)/EI$

	for 0<= x <= a $d^2y/dx^2 = 0$ For a < x <=L
Simply supported free vibration	$EI\frac{\partial^4 w}{\partial x^4} = -\mu\frac{\partial^2 w}{\partial t^2}$ μ *is mass per unit length*
Cantilever free vibration	

f.

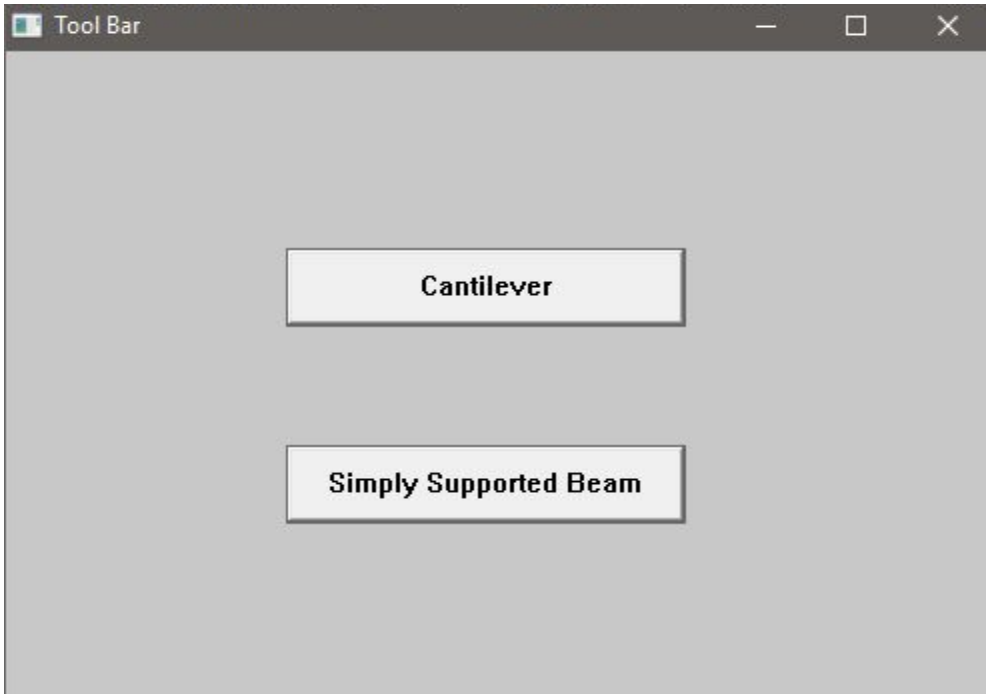
Type of Problem	**Solution**
Simply supported forced	$y = \frac{Pbx}{6lEI}\left(l^2 - x^2 - b^2\right)$ for $0 < x < a$ $y = \frac{Pb}{6lEI}\left[\frac{l}{b}(x-a)^3 + \left(l^2 - b^2\right)x - x^3\right]$ for $a < x < l$
Cantilever forced	$y = \frac{Px^2}{6EI}(3a - x)$ for $0 < x < a$ $y = \frac{Pa^2}{6EI}(3x - a)$ for $a < x < l$
Simply supported free vibration	$\omega(x,t) = sin(n\pi x/L) * sin(n^2\pi^2\sqrt{EI/mL^4})$
Cantilever free	$\omega(x,t) = cos(\alpha^2\sqrt{EI/mL^4}) *$ $[\{cos(\alpha x/L) - cosh(\alpha x/L)\} + A * \{sin(\alpha x/L) - sinh(\alpha x/L)\}]$ $A = (cos(\alpha) + cosh(\alpha)) / (sinh(\alpha) - sin(\alpha))$

g. Making buttons for Drag and Drop feature and tracking mouse position to determine the location of force.

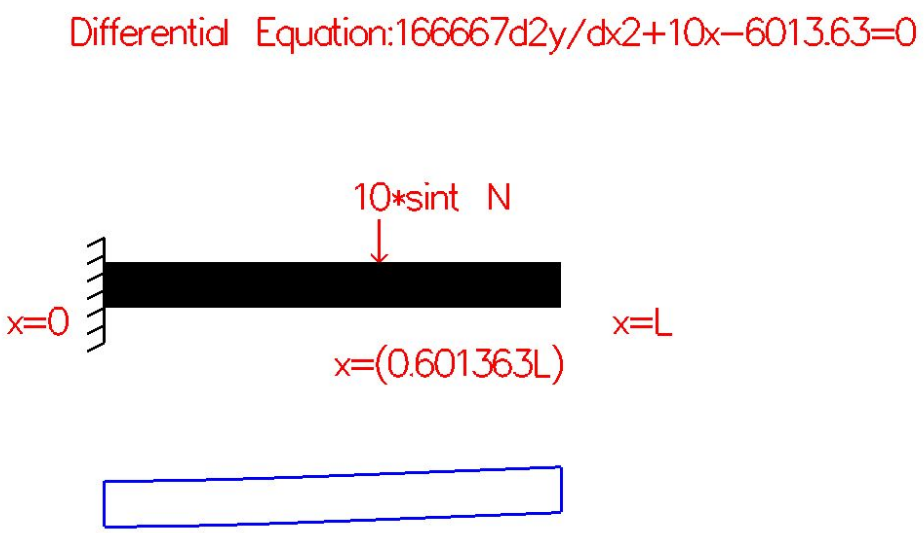
h. Writing functions to be used for plotting displacement corresponding to 4 cases : Cantilever/Fixed Beam and Forced/Free vibrations, using the equations as defined above in solution.
i. Setting up a timer function for animation with 40 frames per second
j. Drawing the state of the beam for a particular time instant.This time instant is increased in steps of 25ms. The plot is chosen depending on which one of four cases is chosen.
k. Updating the Drawing from previous state to next state. This continuous rendering causes the appearance of an animation.

3. Results

A user-friendly UI window with button, textbox, text field which guides the user to operate the program and give input values for problem formulation.



A 1366x768 window is opened. For specified beam dimensions and/or Force magnitude with location we get either one of these after scaled for better representation



a. Graphical displacement of Simply Supported beam along with corresponding differential equation and initial conditions.

b. Graphical displacement of Cantilever beam along with corresponding differential equation and initial conditions.

c. Animation of vibrations for Simply supported beam in vibration.

d. Animation of vibrations for Cantilever beam in vibration.

4. Future Work

Showing the final output in 3-D.

Allowing user to dynamically choose the mode of vibration to be shown.

Allowing to rotate the view in 3-D using mouse drag.

Giving option for beam cross section type : rectangle, circle or I-section.

5. References

1. https://www.opengl.org/discussion_boards/showthread.php/175299-OpenGL-windows-api
2. https://msdn.microsoft.com/en-us/library/aa271855(v=vs.60).aspx
3. https://msdn.microsoft.com/en-us/library/windows/desktop/ms632600(v=vs.85).aspx
4. https://www3.ntu.edu.sg/home/ehchua/programming/opengl/HowTo_OpenGL_C.html
5. https://www3.ntu.edu.sg/home/ehchua/programming/opengl/HowTo_OpenGL_C.html