CHAPTER-1

INTRODUCTION

1.1 Aim

The aim of this project is to develop a 2-D atom simulator, which contains options like selecting the user desired element, simulating the selected element. And also stopping the simulation when user wants. The interface should be user friendly and should use mouse and keyboard interface for the interaction with the user. The main goal is to show the users how an element structure is and how the electrons revolves around the nucleus so that one can easily get the knowledge of atom.

1.2 History

The phrase “Computer Graphics” was coined in 1960 by William Fetter, a graphic designer for Boeing. The field of computer graphics developed with the emergence of graphics hardware. The first major advance in computer graphics was the development of the Sketchpad by Ivan Sutherland. Further advances in computing led to greater advancement in interactive computer graphics. In 1959, the TX2 computer was developed at MIT’s Lincoln Laboratory.

1.3 Application of Computer Graphics

- Computational biology
- Computational physics
- Computer-aided design
- Computer simulation
- Digital art

1.4 Introduction to OpenGL

The Computer Graphics is one of the most effective and commonly used methods to communicate the processed information to the user. It displays the information in the form of graphics objects such as pictures, charts, graphs and diagram instead of simple text.

Figure 1.4: Open GL Logo

OpenGL is a low-level graphics library specification. It makes available to the Programmer a small set of geometric primitive– points, lines, polygons, images, and bitmaps. OpenGL provides a set of commands that allow the specification of geometric objects in two or three dimensions, using the provided primitives, together with commands that control how these objects are rendered. Since OpenGL drawing commands are limited to those that generate simple geometric primitive (points, lines and polygons), the OpenGL utility (GLUT)has been created to aid in the development of more complicated three-dimensional objects such as a sphere, a torus, and even a teapot. GLUT may not be satisfactory for full-featured OpenGL application, but it is a useful starting points for learning OpenGL. OpenGL API-based application can run on systems ranging from consumer electronics to PCs, workstations and supercomputer, as a Result, application can scale to any class of machine that are minimum.

GLUT

GLUT, short for OpenGL Utility Toolkit, is a set of support libraries available on every major platform. OpenGL does not directly support any form of windowing, menus, or input. That's where GLUT comes in. It provides basic functionality in all of those areas, while remaining platform independent, so that you can easily move GLUT-based applications from, for example, Windows to UNIX with few, if any, changes.

1.5 Project Related Concepts

The objective is to build an atom simulator which can convince the audience about the structure of an element. The coding is implemented for the atoms from Hydrogen to Neon, that is for 10 elements. In this simulation importance is given on a structure of an element and how the electrons revolve around the nucleus.

The basic requirements of the atom simulator are analyzed to be:

1. **User Interface:** User should be able to select an element and start the simulation on their own. They can start, stop and change the elements of their choice and after this they can exit the simulation.
2. **Element Selection:** User can select the element from Hydrogen to Neon for the simulation, that is from atomic number 1 to atomic number 10.
3. **Start/ Stop Simulation:** User after selecting an element from the mentioned list he/she can start the simulation. As soon as they select start, the electrons around the nucleus

starts revolving around the nucleus within their orbit. If they select the stop simulation option, the simulation will be stopped.

1.6 INTERFACES

1.6.1 Mouse Interface

- **Select element:** When the user clicks the right click button on the mouse, the screen will be prompted with the list of options. First option is to select the user desired element from the list. The list contains elements from Hydrogen to Neon for the simulation, that is from atomic number 1 to atomic number 10.
- **Simulate:** After user selecting an element, when he/she clicks on the simulate option, the electrons around the nucleus starts revolving around the nucleus within their orbit.
- **Stop simulation:** If a user selects this option, the simulation will be paused.
- **Exit:** The program execution will be terminated and the window will be destroyed after selecting this option.

1.6.2 Keyboard Interface

Three functionalities are implemented using the keyboard function.

- After selecting an element, if a user presses spacebar the simulation will be started.
- After starting the simulation if the user clicks on ‘S’ key, simulation will be paused.
- If the user clicks on the ‘Q’ key, program execution will be terminated and the window will be destroyed.

CHAPTER-2

SYSTEM REQUIREMENTS SPECIFICATION

Visual Studio 2005 delivers on Microsoft's vision of smart client applications by letting developers quickly create connected applications that deliver the highest quality rich user experiences. This new version lets any size organization create more secure, more manageable, and more reliable applications that take advantage of Windows Vista, windows7, 2007 Office System and the Web. By building these new types of applications, organizations will find it easier than ever to capture and analyze information so that they can make effective business decisions.

2.1 Software Requirements

• Operating system :	Ubuntu,MAC,Windows .
• Language Tool :	OpenGL
• Compiler :	GNU GCC Compiler / C++ Compiler.
• Libraries :	GL/gl.h, GL/glut.h, GL/glu.h.
• Documentation Tools :	vi-editor, geditor, visual studio.

2.2 Hardware Requirements

• Processor :	Intel CORE i3,i5
• RAM :	16Mb RAM.
• Monitor :	Graphics Compatible.
• Keyboard :	Normal keyboard(QWERTY).
• Backup Media :	Hard disk.

CHAPTER 3

SYSTEM DESIGN

3.1 Window design

Atom simulation uses only one window. That is

- **Atom simulation (Main window):** This window contains all the contents that is menu bar and simulation display. This is window used for all the events and functions in this project. In this window we display simulation of first 10 atoms in the periodic table. And all mouse and keyboard events triggered in this window. All the labels and Information about the model will be displayed on this window.

3.2 Menu bar

Menu bar is designed so that one can easily access the various options like selecting the elements or starting the simulation or stopping the simulation etc.

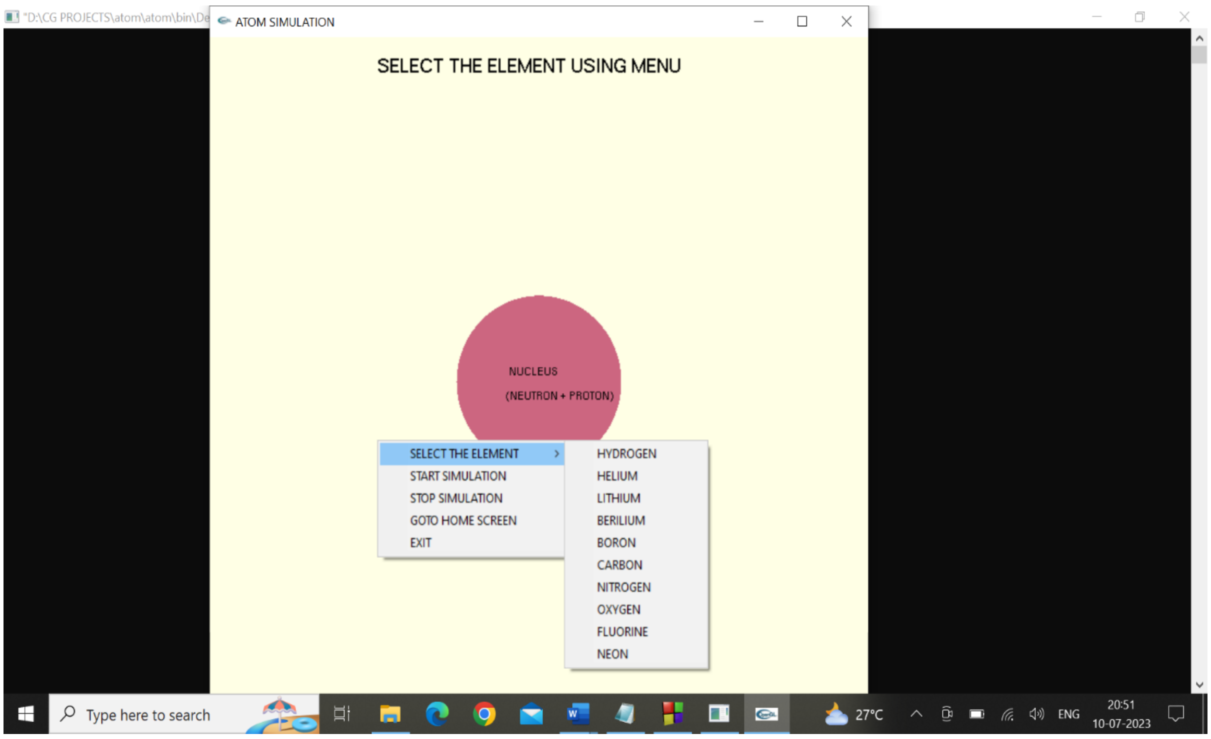


Figure 3.2: Menu Bar

3.3 Simulation display

As soon as user selects the element and click on the simulate option, the electrons around the nucleus starts revolving around the nucleus within their orbit.

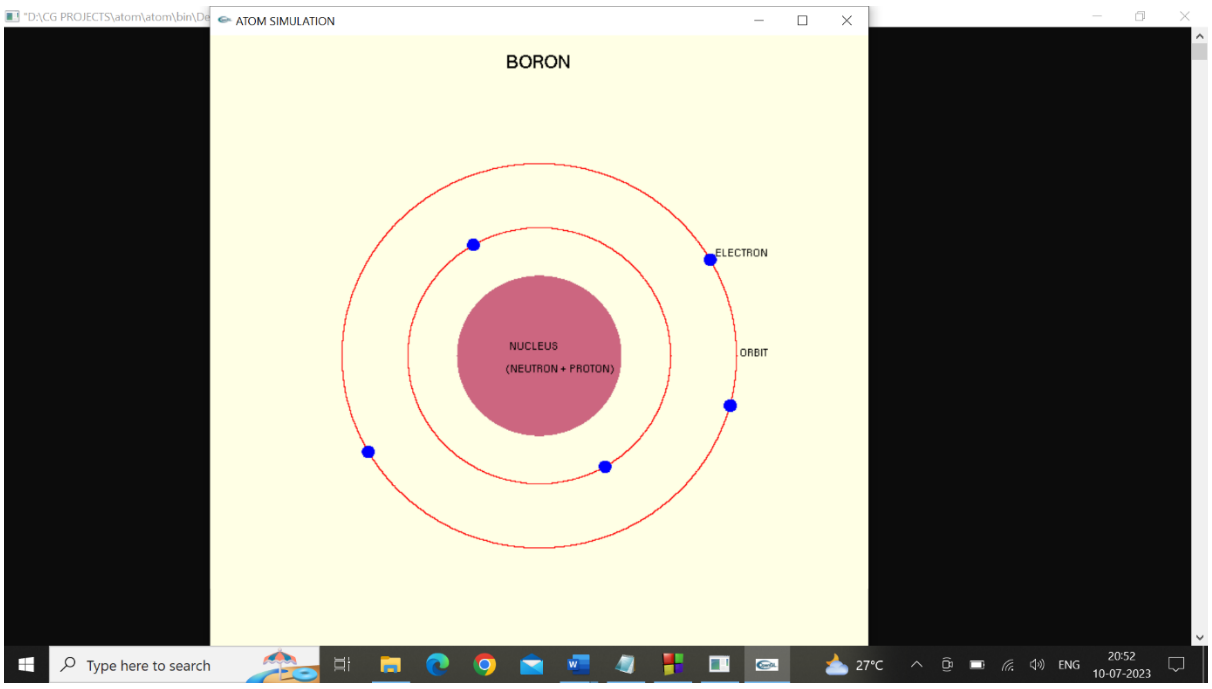


Figure 3.3: Simulation display

3.4 Flow Diagram

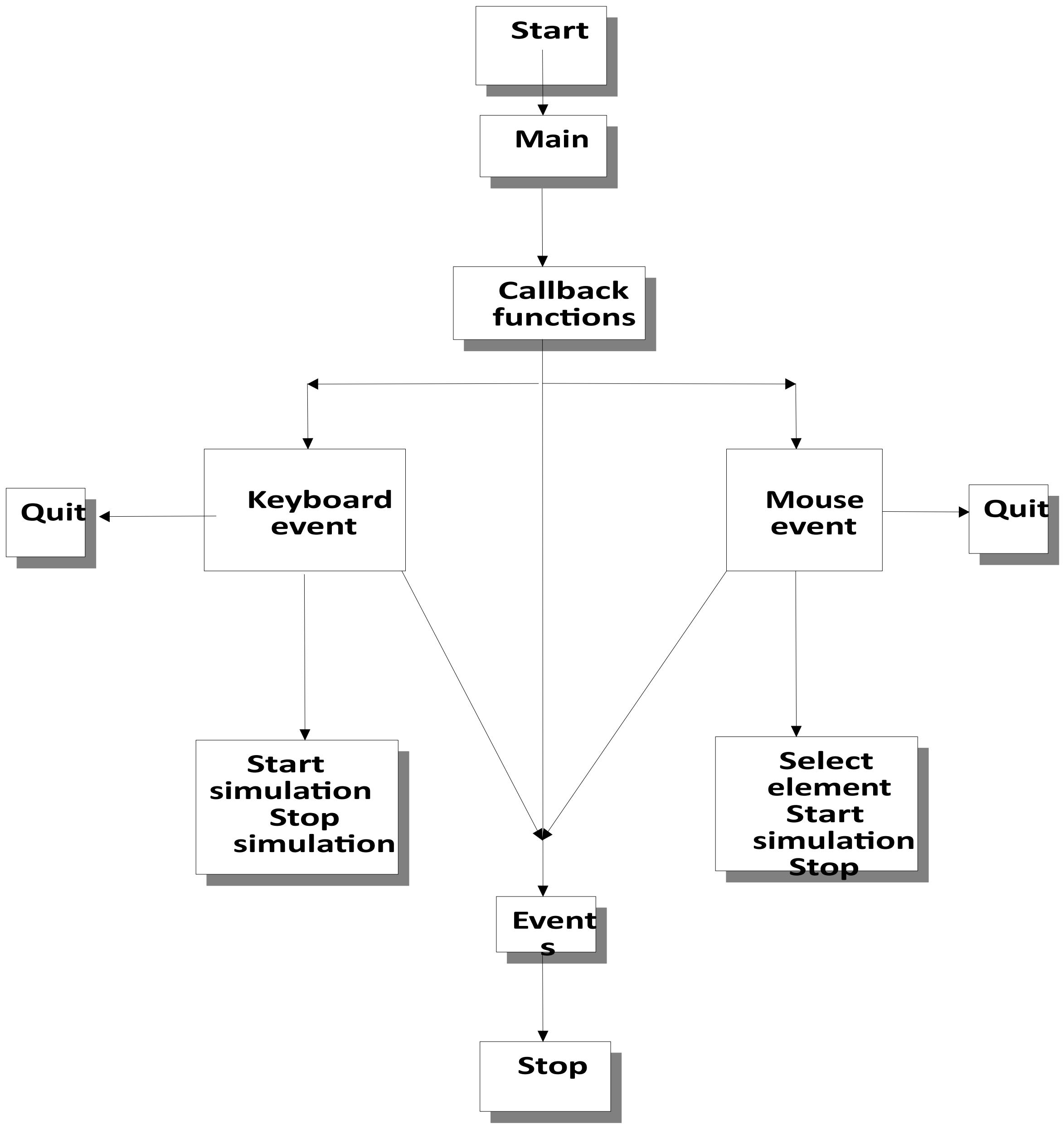


Figure3.4 Flow Diagram of Atom Simulation

CHAPTER-4

IMPLEMENTATION

4.1 Functions used

glRasterPos3f(x, y, z): OpenGL maintains a 3-D position in window coordinates. This position, called the raster position, is maintained with subpixel accuracy. The current raster position consists of three window coordinates (x, y, z), a clip coordinate w value, an eye coordinate distance, a valid bit, and associated color data and texture coordinates.

glutCreateMenu(menu): glutCreateMenu creates a new pop-up menu and returns a unique small integer identifier. The range of allocated identifiers starts at one. The menu identifier range is separate from the window identifier range. Implicitly, the current menu is set to the newly created menu. This menu identifier can be used when calling glutSetMenu.

glutAddMenuEntry(args): glutAddMenuEntry adds a menu entry to the bottom of the current menu. The string name will be displayed for the newly added menu entry. If the menu entry is selected by the user, the menu's callback will be called passing value as the callback's parameter.

glutAttachMenu(button): glutAttachMenu attaches a mouse button for the current window to the identifier of the current menu; glutDetachMenu detaches an attached mouse button from the current window. By attaching a menu identifier to a button, the named menu will be popped up when the user presses the specified button. button should be one of GLUT_LEFT_BUTTON, GLUT_MIDDLE_BUTTON, and GLUT_RIGHT_BUTTON.

Note that the menu is attached to the button by identifier, not by reference. **glutMouseFunc(args):** glutMouseFunc sets the mouse callback for the current window. When a user presses and releases mouse buttons in the window, each press and each release generates a mouse callback. The button parameter is one of GLUT_LEFT_BUTTON, GLUT_MIDDLE_BUTTON, or GLUT_RIGHT_BUTTON. For systems with only two mouse buttons, it may not be possible to generate GLUT_MIDDLE_BUTTON callback. For systems with a single mouse button, it may be possible to generate only a GLUT_LEFT_BUTTON callback. The state parameter is either GLUT_UP or GLUT_DOWN indicating whether the callback was due to a release or press respectively. The x and y callback parameters indicate the window relative coordinates when the mouse button state changed. If a GLUT_DOWN callback for a specific button is triggered, the program can assume a GLUT_UP callback for the same button will be generated (assuming the window still has a mouse callback registered) when the mouse button is released even if the mouse has moved outside the window.

glutKeyboardFunc(args): glutKeyboardFunc sets the keyboard callback for the current window. When a user types into the window, each key press generating an ASCII character will generate a keyboard callback. The key callback parameter is the generated ASCII character. The state of modifier keys such as Shift cannot be determined directly; their only effect will be on the returned ASCII data. The x and y callback parameters indicate the mouse location in window relative coordinates when the key was pressed. When a new window is created, no keyboard callback is initially registered, and ASCII key strokes in the window are ignored. Passing NULL to glutKeyboardFunc disables the generation of keyboard callbacks. During a keyboard callback, glutGetModifiers may be called to determine the state of modifier keys when the keystroke generating the callback occurred.

CHAPTER-5

ALGORITHM

5.1 Algorithm of Atom Simulation

Step 1: Initialize the simulation:

- Set up the simulation parameters (time step, simulation duration, etc.).
- Define the initial positions and velocities of the atoms.

Step 2: While the simulation time is less than the desired duration:

Compute forces on each atom:

- For each atom pair, calculate the interatomic forces based on the chosen potential energy function (e.g., Lennard-Jones potential or Coulombic interactions).
- The forces can be computed by taking the gradient of the potential energy function.
- Update atom positions and velocities:
- For each atom:
- Compute the acceleration based on the forces acting on the atom and its mass.
- Update the velocity by integrating the acceleration over the time step.
- Update the position by integrating the velocity over the time step.
- Apply boundary conditions (if necessary):
- Check if any atom has moved outside the simulation space.
- If so, adjust its position according to the chosen boundary condition (e.g., periodic boundary conditions).
- Increment the simulation time by the time step.

Step 3: Visualize the simulation:

- Render the atoms in 3D space using computer graphics techniques.
- Assign colors and sizes to represent atom types or properties.
- Apply shading and lighting models to enhance the visual representation.

CHAPTER-6

TESTING

6.1 TEST CASES

Table 6.1: Test cases for Mouse interface

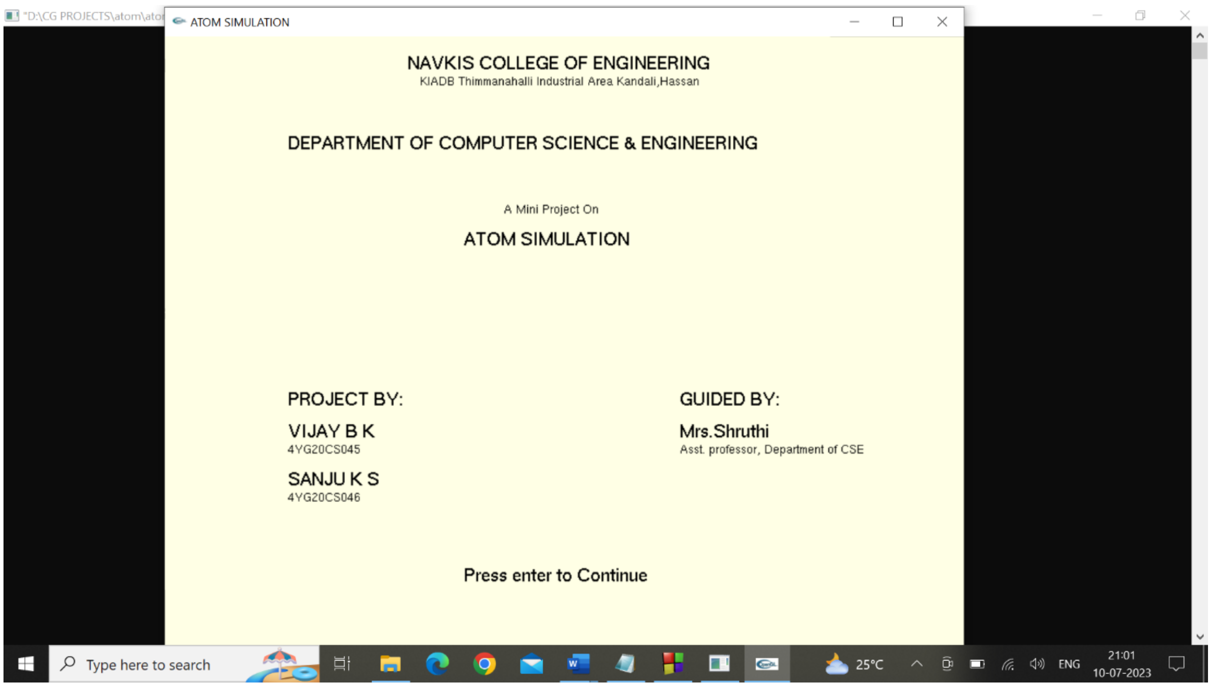
Sl. No	Functionality	Comments	Remarks
1.	Mouse right click	It shows menu bar to user.	Pass
2.	Selecting the options 1. Select element 2. Simulate 3. Stop simulation 4. Exit 5. Goto Home	It shows the list of option to user from which they can select an option. Selects the element in the given list. Starts the simulation. Stops the simulation. Exits from the window. Display starting window	Pass

Table 6.2: Test cases for Keyboard interface

Sl. No	Functionality	Comments	Remarks
1.	Choosing the options 1. Simulate 2. Stop simulation 3. Exit	It shows the list of option to user from which they can select an option. Starts the simulation. (Space bar) Stops the simulation. (‘S’) Exits from the window. (‘Q’)	Pass

CHAPTER-7

RESULTS



NAVKIS COLLEGE OF ENGINEERING
KIADB Thimmanahalli Industrial Area Kandali,Hassan

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

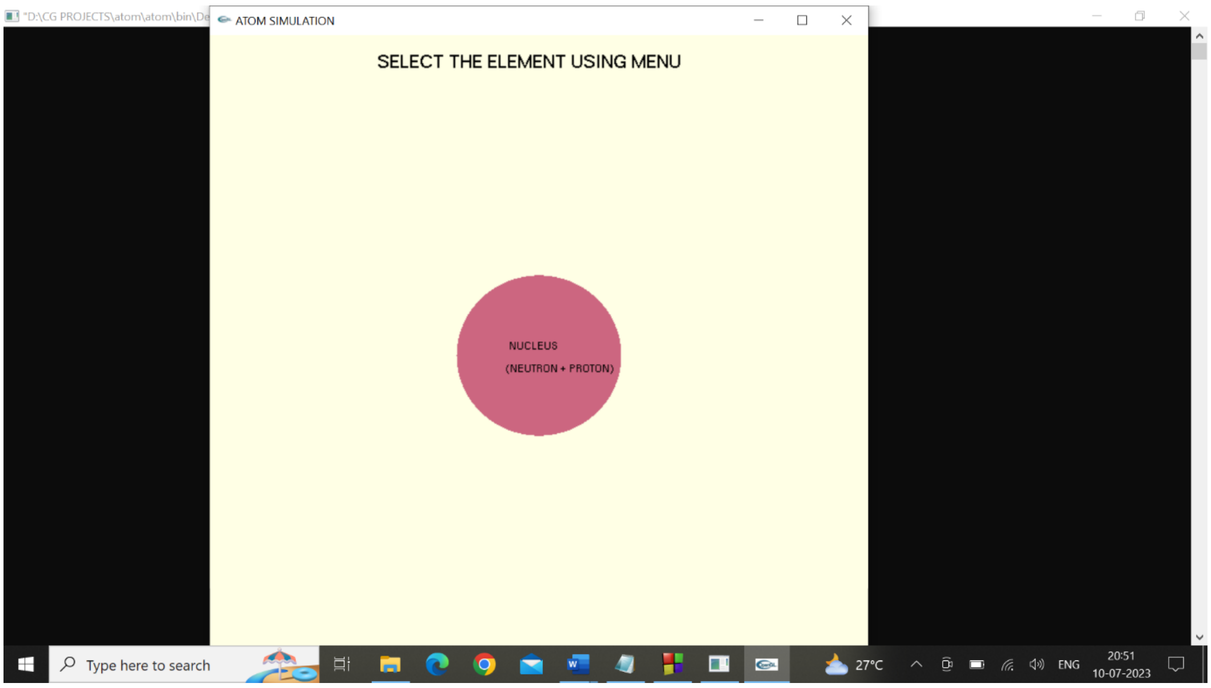
A Mini Project On
ATOM SIMULATION

PROJECT BY:
VIJAY B K
4YG20CS045
SANJU K S
4YG20CS046

GUIDED BY:
Mrs.Shruthi
Asst. professor, Department of CSE

Press enter to Continue

Figure 7.1: Home Screen



SELECT THE ELEMENT USING MENU

Figure 7.2: Starting Screen

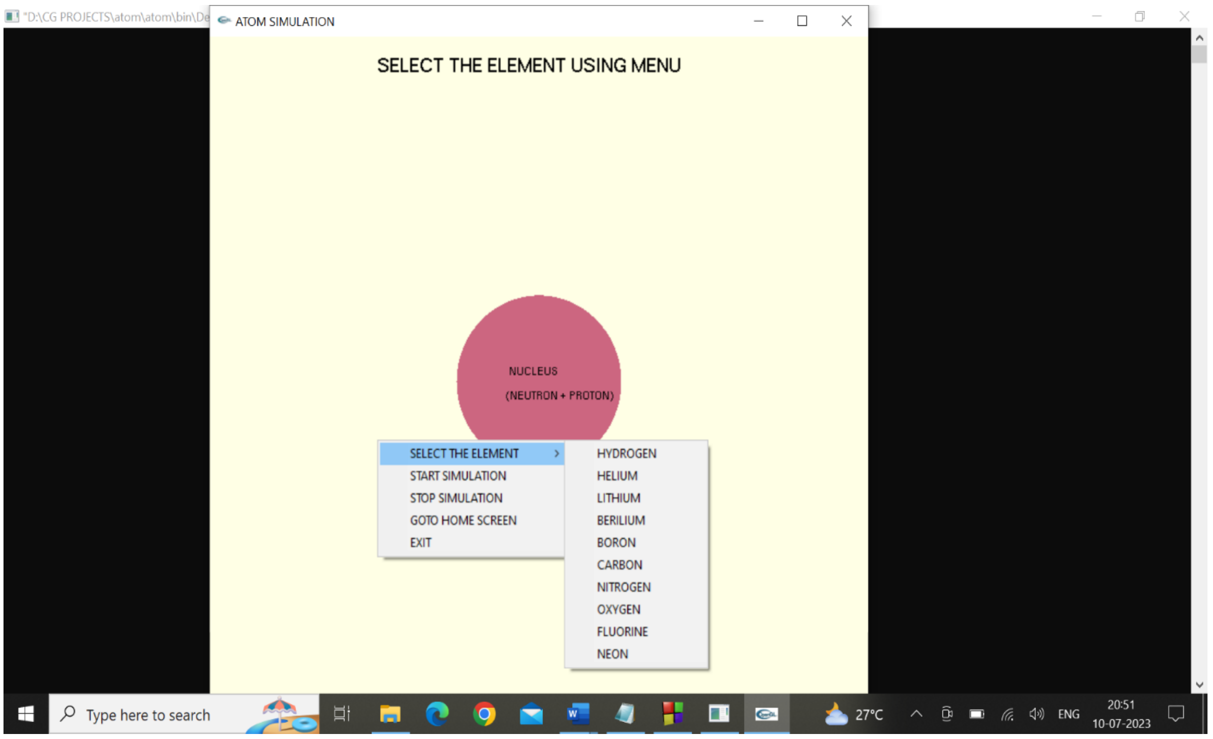


Figure 7.3: Menu Interface

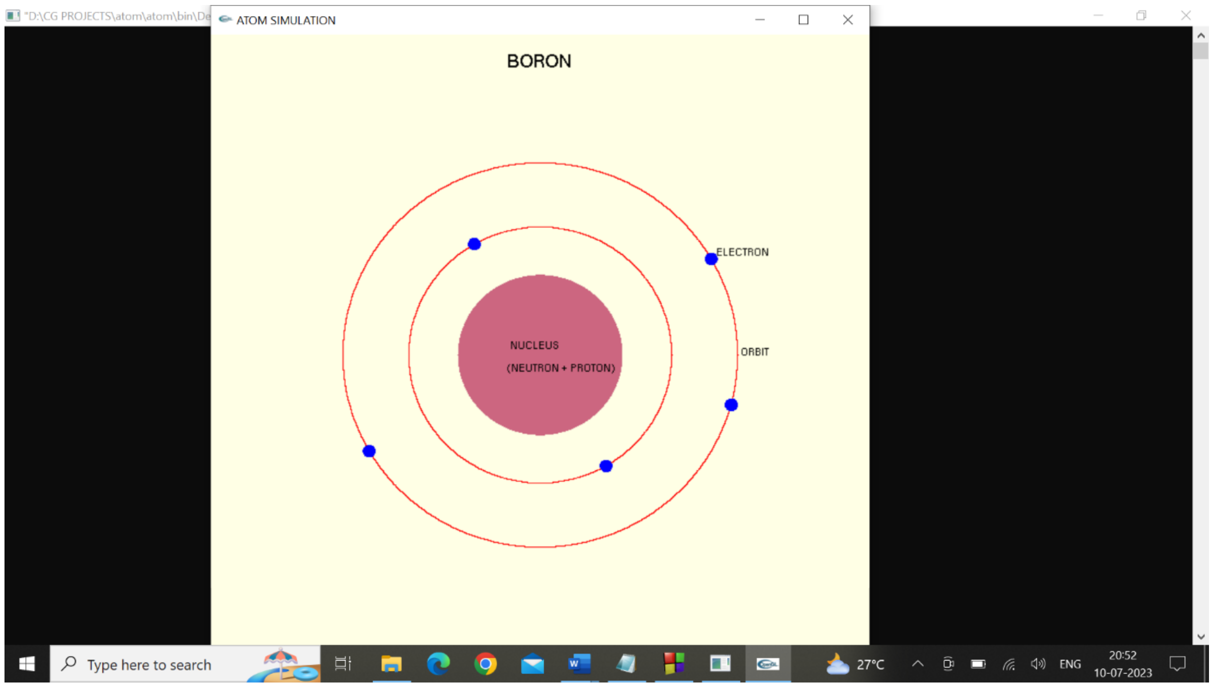


Figure 7.4: Atom Simulation

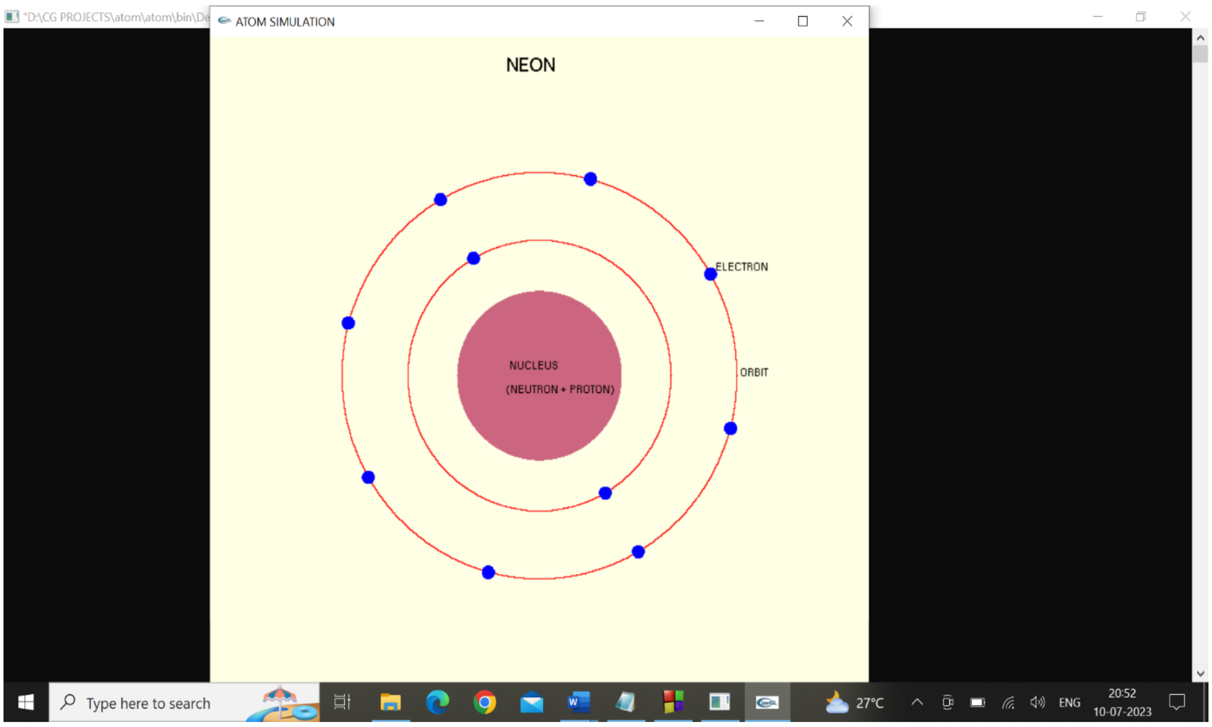


Figure 7.5: Full Screen Atom Simulation

CONCLUSION

This atom simulation is very good project. Users can very easily understand the structure of an element. The interface is mouse driven and the user can select a function by clicking. And also, the interface supports keyboard interface. We have tried our best to make this simulator very realistic, so that user can easily understand the concepts of electrons, orbits, atoms and nucleus etc.

FUTURE ENHANCEMENTS

The following are some of the features that are planned to be supported in the future versions of the atom simulator.

- Adding all the elements from the periodic table.
- Features like showing the simulation with all the important details of an element.
- Adding a search bar for selecting an element from the list of all the elements.
- Making the simulation in 3-D.

REFERENCES

[1]. Interactive Computer Graphics A Top-Down Approach with OpenGL -Edward Angel, 5th Edition, Addison-Wesley, 2008.

[2]. Computer Graphics Using OpenGL – F.S. Hill Jr. 2nd Edition, Pearson Education, 2001.

[3]. Computer Graphics – James D Foley, Andries Van Dam, Steven K Feiner, John F Hughes, Addison-Wesley 1997.

[4]. Computer Graphics - OpenGL Version – Donald Hearn and [5] Pauline Baker, 2nd Edition, Pearson Education, 2003.

[6]. www.google.com.

[7]. www.openglforum.org.