**3D Interactive Models in Physics**

**1. Introduction**

This project is born out of a passion for combining the power of coding with the beauty of physics. Inspired by the enlightening approach of 3Blue1Brown [[1]](https://www.youtube.com/@3blue1brown) and the capabilities of p5.js, this endeavor seeks to bridge the gap between static 3D images and a deep understanding of complex structures.

In physics, the power of visualization cannot be overstated. Traditional static images often fall short in conveying the intricacies of crystal structures, vector fields, 3D functions and other concepts. Recognizing this limitation during a solid-state physics lecture, I embarked on a coding journey initially for the sheer joy of learning and exploring.

Having previously created a simple cubic model using Processing, the project gradually evolved as I discovered the versatility of p5.js, a JavaScript library designed for drawing and animations. What made it even more appealing was its compatibility with any web browser, eliminating the need for specific software installations.

The incorporation of WEBGL in p5.js opened up exciting possibilities for creating intricate 3D crystal structures. The project offers a platform for interactive 3D models with a dynamic experience, allowing users to explore and understand complex physical phenomena effortlessly.

The project currently encompasses an array of interactive models, Solid State Models, Fourier Series, Vector Fields Plotter and Lissajous Figures. Solid State Models are coded in Processing Software while the others uses WEBGL incorporated in p5js.

The project is open to all and one can find it on my Git Hub Repository <https://vyasmokalzz.github.io/Physics-Models/>. Overall User Interface is kept very convenient and easy to handle. The site currently only works on desktop browsers and mobile support is not incorporated. The aim is to visualize these concepts and provide an engaging platform for learning and exploration.

**2. Theoretical Background**

**2.1 What is p5js and WEBGL**

p5.js [[2]](https://p5js.org/) is a free and open-source JavaScript library for creative coding, with a focus on making coding accessible and inclusive for artists, designers, educators, beginners, and anyone else!

WebGL [[3]](https://p5js.org/learn/getting-started-in-webgl-coords-and-transform.html), or Web Graphics Library, is a JavaScript API (Application Programming Interface) that enables the rendering of 3D and 2D graphics within web browsers. It provides a bridge between the browser and the computer's GPU (Graphics Processing Unit), allowing for hardware-accelerated graphics rendering.

A basic p5.js program incorporating WEBGL looks like this:

function setup() {

    createCanvas(windowWidth, windowHeight, WEBGL);

    describe('a red box on a white background');

  }

  function draw(){

    background(255);

    fill(255,0,0);

    box();

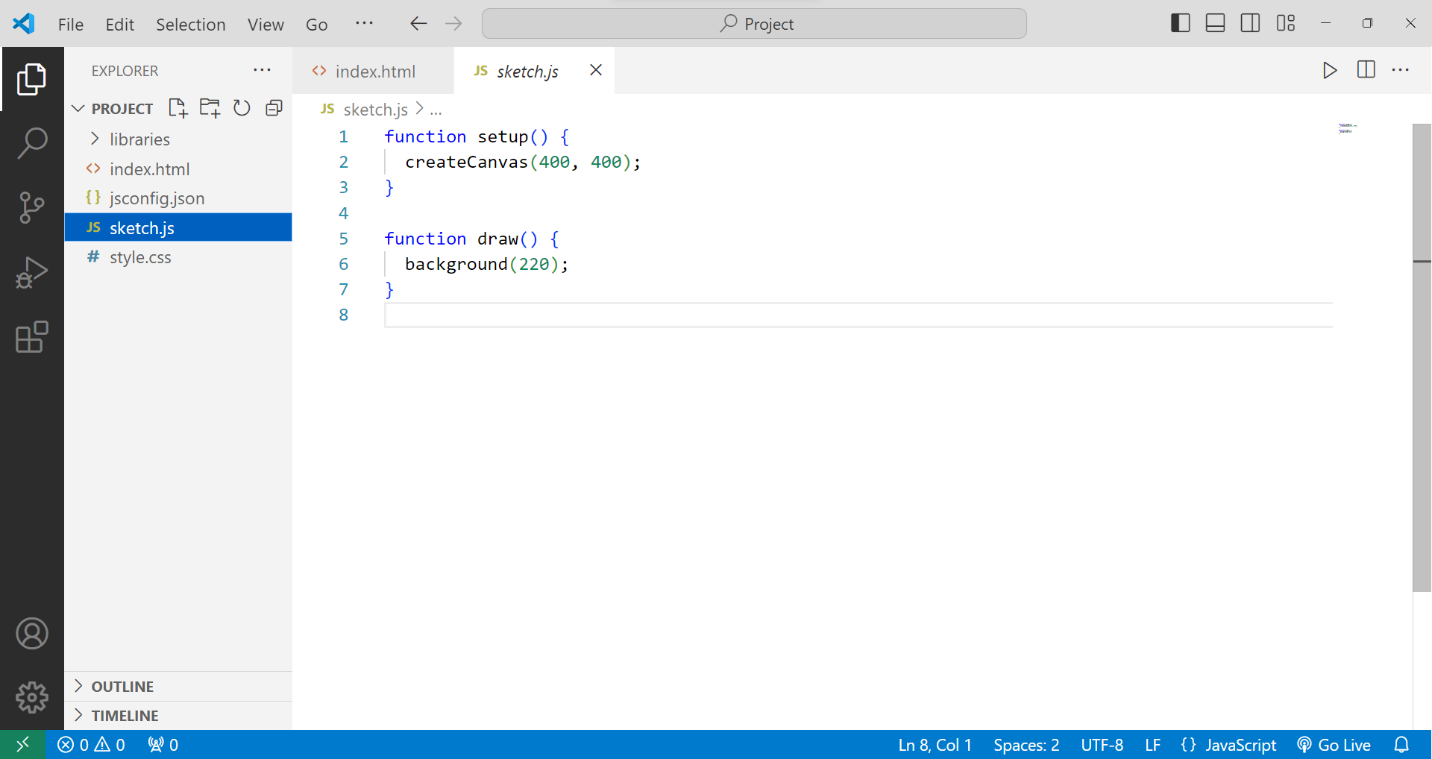
  }

One of the most fundamental differences between working in 2D and working in 3D is the most obvious: there is one more dimension to work with. In addition to the horizontal and vertical position (x and y axes) of an element in our drawing, 3D adds depth, the z-axis.

When drawing in 2D, the point (0,0) is located at the top left corner of the screen. In WebGL mode, the origin of the sketch (0,0,0) is located in the middle of the screen. By default, the x-axis goes left-to-right, y-axis goes up-to-down, and the z-axis goes from further-to-closer.

**2.2 Creating working Environment**

Throughout the project I have used Visual Studio Code [[4]](https://code.visualstudio.com/) (VS Code) as code editor. To incorporate p5js into webpage I used VS Code extension p5.vscode [[5]](https://marketplace.visualstudio.com/items?itemName=samplavigne.p5-vscode). It sets up the entire coding environment using Command Pallet, giving libraries, CSS, JavaScript and HTML files. The sketch can be edited in sketch.js file as shown below.



**2.3 How to make Basic Objects in p5js**

In p5js we can create variety of shapes which includes arc, ellipse, circle, line, point, quadrilateral, rectangle, square, triangle and their corresponding functions are arc(), ellipse(), circle(), line(), point(), quad(), rect(), square(), triangle() respectively. Consider the following example

function setup() {

  createCanvas(400, 300);

}

function draw() {

  background(220);

  stroke(0);

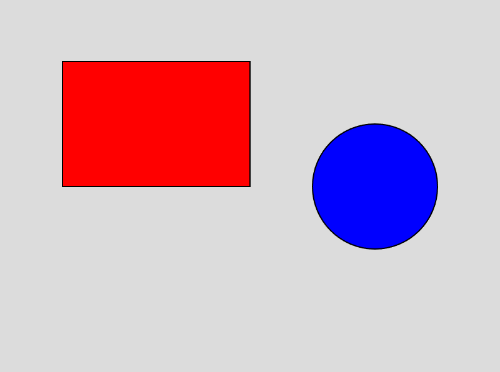
  fill(255,0,0);  //specify color in RGB

  rect(50,50,150,100);  // specify (x1,y1) and (x2,y2) corner coordinates

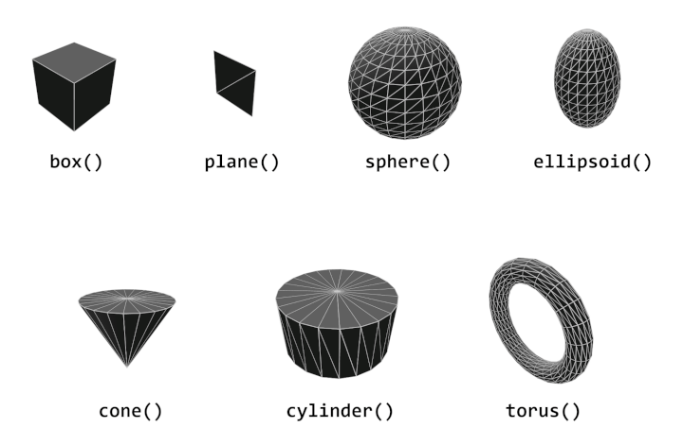
  fill(0,0,255);

  circle(300,150,100);  // specify center and radius

}



Similarly, we can create 3D Objects incorporating WEBGL into the project. Few supported 3D shapes are box, plane, sphere, ellipsoid, cone, cylinder and torus with their functions being box(), plane(), sphere(), ellipsoid(), cone(), cylinder() and torus() respectively.



One more example of the same:

function setup() {

  createCanvas(400, 400, WEBGL);

}

function draw() {

  background(220);

  stroke(0);

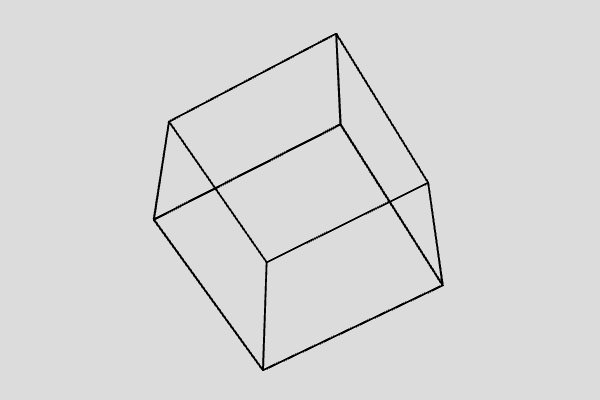
  noFill();

  rotateX(PI/3);

  rotateY(PI/3);

  box(100);

}



**2.4 Events**

A category of functions called events alter the normal flow of a program when an action such as a key press or mouse movement takes place. An event is a polite interruption of the normal flow of a program.

Events like keypresses and mouse press allow us to interact with the computer, hence would facilitate to make our models more interactive to the user, allowing to change the data and orientation according to user’s requirement.

P5js allows us to manipulate a number of mouse and keyboard event.  
Mouse Events

* mousePressed() - Code inside this block is run one time when a mouse button is pressed
* mouseReleased() - Code inside this block is run one time when a mouse button is released
* mouseClicked() - Code inside this block is run once after a mouse button is pressed and released over the element
* doubleClicked() - Code inside this block is run once after a mouse button is pressed and released over the element twice
* mouseMoved() - Code inside this block is run one time when the mouse is moved
* mouseDragged() - Code inside this block is run one time when the mouse is moved while a mouse button is pressed
* mouseOver() - Code inside this block is run once after every time a mouse moves onto the element.
* mouseOut() - Code inside this block is run once after every time a mouse moves off the element

Keyboard Events

* keyPressed() – Code inside this block is run one time when a mouse button is pressed.
* keyTyped() – Code inside this block is run one time when a key is pressed, but action keys such as Ctrl, Shift, and Alt are ignored. The most recent key pressed will be stored in the key variable.
* keyReleased() – Code inside this block is run one time when any key is released.

**2.5 Camera, View Lighting and Material**

Other than just geometry creating 3D models requires to take care of Camera, Lighting and materials to make a visually interesting 3D scene.

**2.5.1 Camera**

**3. Program**

**2.1 What is p5js and WEBGL**

p5.js [[2]](https://p5js.org/) is a free and open-source JavaScript library

**6. References**

1. [**https://www.youtube.com/@3blue1brown**](https://www.youtube.com/@3blue1brown)
2. [**https://p5js.org/**](https://p5js.org/)
3. [**https://p5js.org/learn/getting-started-in-webgl-coords-and-transform.html**](https://p5js.org/learn/getting-started-in-webgl-coords-and-transform.html)
4. [**https://code.visualstudio.com/**](https://code.visualstudio.com/)
5. [**https://marketplace.visualstudio.com/items?itemName=samplavigne.p5-vscode**](https://marketplace.visualstudio.com/items?itemName=samplavigne.p5-vscode)

**Brief Outline**

**1. Introduction**

**2. Theory**

* What is p5js
* Setting up working environment
* How to create objects
* Events in p5js
* What is perspective and camera
* Lights
* Material

**3. Program**

**4. Results and Conclusion**

**5. Future Scope**

**6. References**