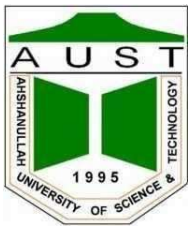
*Ahsanullah University of Science & Technology*
Department of Computer Science & Engineering




# A Helicopter

## Computer Graphics Lab (CSE 4204)

### Project Final Report



| Submitted By: | |
|---|---|
| Alimul Islam Eram Khan | 20200104110 |
| Md. Fahim Shakil Chowdhury | 20200104128 |
| Group no | 6 |

# Project Requirements:

The project, titled "3D Helicopter Visualization," was developed to fulfill the requirements of the CSE4204 Computer Graphics Lab final project. The specified requirements are as follows:

1. **3D Objects**:
    - Helicopter body with texture.
    - Helicopter wings (main and tail rotors) with texture.
2. **Keyboard Interaction**: Camera movement around the helicopter using WASD keys for horizontal navigation and arrow keys for vertical adjustment.
3. **Mouse Interaction**: Light position rotation around the helicopter controlled by mouse movement, with mouse wheel adjusting camera zoom and mouse click initiating a flight loop.
4. **Animation**: Rotation of the helicopter's main and tail rotor blades, with additional animations for hovering and a circular flight path.
5. **Core Technical Requirements**:
    - Implementation of custom shaders (vertex and fragment shaders) for rendering objects.
    - Lighting model with dynamic light positioning and ambient effects.
    - Perspective projection for realistic 3D visualization.
    - Textures applied to all objects for enhanced visual realism.
    - Animation for dynamic scene elements.
    - Mouse and keyboard interaction for user control.
6. **Additional Requirements**: The project must be visually appealing, unique, and free from plagiarism, with an emphasis on aesthetic quality.

The project was implemented using **Three.js**, incorporating custom geometries for the helicopter and environment, custom shaders for realistic rendering, and a variety of interactive and animated features to create a visually stunning and immersive experience.

# Software Platform:

The following tools and technologies were utilized in the development of the project:

- **Three.js**: A JavaScript library for creating and rendering 3D scenes in WebGL (version referenced in three.min.js).
- **WebGL**: For rendering the 3D graphics in the browser.
- **GLSL (OpenGL Shading Language)**: For authoring custom vertex and fragment shaders to manage object appearance and lighting (shaders.js).
- **HTML5/CSS3**: For structuring the user interface, including the Heads-Up Display (HUD), control instructions, and styling (index.html, styles.css).
- **JavaScript**: For implementing interactivity, animations, and scene management across files such as main.js, controls.js, animations.js, and others.

- **Browser Environment**: The project was tested on modern web browsers (Google Chrome) which support WebGL.

## Project Features:

The project encompasses a comprehensive set of features to deliver an immersive and visually appealing 3D helicopter visualization. Each feature is described below, including its functionality and implementation details:

1. **Helicopter Model with Textures**:
    - **Description**: The helicopter is a custom 3D model constructed using Three.js geometries, comprising a cylindrical body, spherical ends, a semi-spherical cockpit, a tail boom, landing skids, and rotating main and tail rotors. All components incorporate custom shader materials with procedural textures.
    - **Implementation**: Defined in helicopter.js, the model employs geometries such as CylinderGeometry, SphereGeometry, and BoxGeometry. Custom shaders from shaders.js (vertexShader and fragmentShader) apply metallic and roughness effects with base colors (e.g., body: 0x2980b9; cockpit: 0x3498db, semi-transparent; rotors: 0x34495e). Noise functions in the fragment shader generate texture variations, integrated with dynamic lighting.
    - **Visual Appeal**: The model's design, enhanced by metallic reflections and noise-based texturing, provides a realistic and aesthetically refined appearance.
2. **Dynamic Lighting Model**:
    - **Description**: A directional light source orbits the helicopter, simulating a sun or moon, with adjustable colors for day (0xffffff) and night (0xaaaaFF). Supplementary lights include a point light for highlights, ambient light, fill light, and a spotlight beneath the helicopter for nighttime illumination.
    - **Implementation**: Light positioning is managed in controls.js via mouse input, with orbital movement (radius: 15 units). Day/night transitions in animations.js and main.js update light colors, fog, and visibility of sun/moon meshes. Custom shaders across files (e.g., helicopter.js, environment.js) compute diffuse, specular, and ambient lighting using the light's position. The spotlight in spotlight.js uses THREE.SpotLight with intensity adjustable via mouse wheel (range: 0.5–2.0) and toggled via 'O' key (night mode only).
    - **Visual Appeal**: The lighting system ensures realistic shadows and transitions, enhancing depth and immersion, particularly with the spotlight's dramatic nighttime effects.
3. **Perspective Projection**:
    - **Description**: A perspective camera provides a realistic 3D view, allowing orbital movement around the helicopter with zoom functionality.
    - **Implementation**: Initialized in main.js as THREE.PerspectiveCamera with a 75-degree field of view, dynamic aspect ratio, and clipping planes (0.1/1000). Position updates occur in controls.js based on keyboard and mouse inputs, with optional follow mode (cameraFollow toggled via 'C' key).

- **Visual Appeal**: This projection creates natural depth perception, contributing to the scene's lifelike quality.

4. **Environment with Textured Objects**:
   - **Description**: The scene features a ground plane (or procedural terrain), buildings, trees, skybox, stars, clouds, and a helipad, all enhanced with custom textures and shaders.
   - **Implementation**:
     - **Ground/Terrain**: Plane geometry with green texture (0x27ae60) and noise shader in environment.js. Terrain in terrain.js uses Perlin noise for heightmaps (up to 5 units), toggled via 'T' key.
     - **Buildings**: Box geometries with gray texture (0x7f8c8d) and emissive windows (0xffff88) in environment.js.
     - **Trees**: Cylinder trunks (0x8B4513) and conical foliage (0x228B22) with noise shaders in environment.js.
     - **Skybox**: Spherical geometry with day/night shader in environment.js and shaders.js.
     - **Stars**: Point cloud visible at night in environment.js.
     - **Clouds**: Billboard planes with noise-based density and transparency shaders, influenced by wind in clouds.js.
     - **Helipad**: Circular geometry with procedural canvas texture (white "H" and ring), emissive at night in helipad.js.
   - **Visual Appeal**: The diverse, dynamically textured environment creates a unified and captivating backdrop, responsive to lighting and environmental changes.
5. **Animations**:
   - **Description**: Includes rotor rotation, hovering, circular flight path, cloud movement, day/night transitions, wind-induced tree sway, and rain effects.
   - **Implementation**:
     - **Rotors**: Rotated in animations.js (main: 0.3 rad/frame; tail: 0.5 rad/frame), with audio toggled via spacebar.
     - **Hovering**: Y-position oscillation using sine functions in animations.js.
     - **Flight Path**: Circular trajectory (radius: 10 units) in flightpath.js and animations.js, activated by mouse click.
     - **Clouds**: Wind-driven movement with boundary wrapping in clouds.js.
     - **Day/Night**: Updates lighting, fog, and visibility in animations.js and main.js, toggled via 'N' key.
     - **Wind**: Tree rotation via sine/cosine in wind.js, toggled via 'W' key, with day/night strength variation.
     - **Weather (Rain)**: 500-particle system with velocity updates in weather.js, toggled via 'R' key (night only), influenced by wind.
   - **Visual Appeal**: Smooth, realistic animations add dynamism and environmental responsiveness.
6. **Mouse and Keyboard Interaction**:
   - **Description**: Comprehensive controls for camera, lighting, animations, and scene toggles, including object selection.
   - **Implementation**:

        - **Mouse**: Light rotation in controls.js; wheel for zoom or spotlight intensity; click for flight or selection in animations.js and interactions.js.
        - **Keyboard**: Detailed mappings in controls.js, hud.js, helipad.js, clouds.js, flightpath.js, terrain.js, interactions.js, spotlight.js, wind.js, weather.js (e.g., WASD for camera, 'H' for HUD).
    - **Visual Appeal**: User-friendly interactions with real-time UI feedback enhance engagement.
7. **Heads-Up Display (HUD)**:
    - **Description**: Displays speed, altitude, fuel, and a mini-map.
    - **Implementation**: HTML elements in hud.js with orthographic mini-map rendering, toggled via 'H' key.
    - **Visual Appeal**: Semi-transparent design integrates seamlessly.
8. **Additional Features**:
    - **Exhaust Particles**: Animated system in helicopter.js.
    - **Sound Effects**: Rotor audio in animations.js.
    - **UI Elements**: Loading screen and styling in main.js, styles.css.
    - **Aesthetic Enhancements**: Fog, sun/moon, and procedural elements.

# Features:

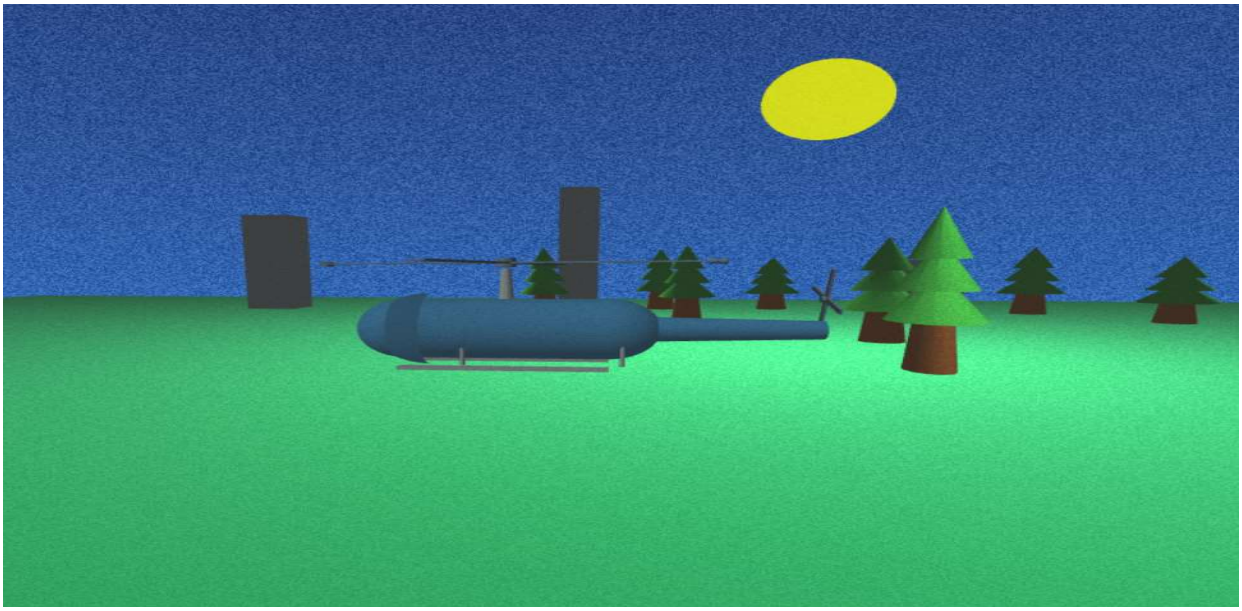
| SL No | Features | Status |
|---|---|---|
| 1 | Helicopter Model with Textures | Implemented |
| 2 | Dynamic Lighting Model (incl. Spotlight) | Implemented |
| 3 | Perspective Projection | Implemented |
| 4 | Environment with Textured Objects (incl. Terrain) | Implemented |
| 5 | Rotor and Flight Animations | Implemented |
| 6 | Mouse and Keyboard Interaction (incl. Object Selection) | Implemented |
| 7 | Heads-Up Display (HUD) with Mini-Map | Implemented |
| 8 | Day/Night Cycle | Implemented |
| 9 | Cloud System with Wind Animation | Implemented |
| 10 | Helipad with Procedural Texture | Implemented |

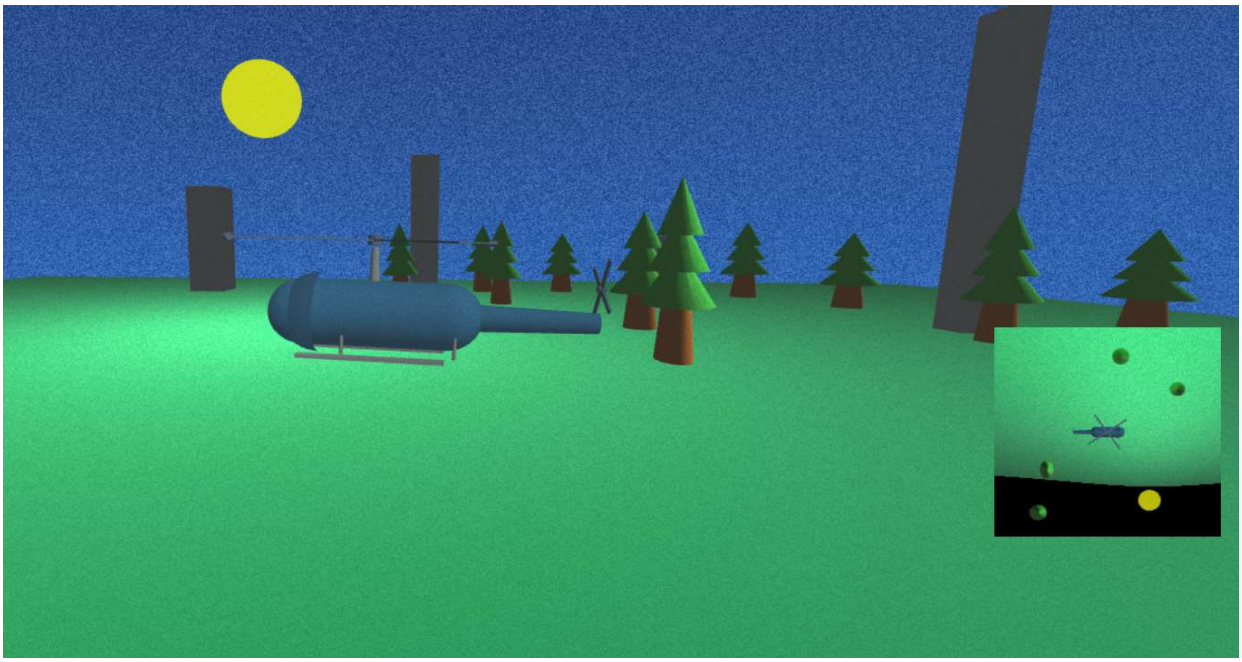
| | | |
|---|---|---|
| 11 | Exhaust Particle System | Implemented |
| 12 | Rotor Sound Effects | Implemented |
| 13 | Flight Path Visualization | Implemented |
| 14 | Dynamic Fog and Skybox | Implemented |
| 15 | Shadow Casting | Implemented |
| 16 | Wind Effects (Tree Sway, Rain Influence) | Implemented |
| 17 | Weather System (Rain Particles) | Implemented |
| 18 | Custom Shaders for All Materials | Implemented |
| 19 | UI Styling and Loading Screen | Implemented |

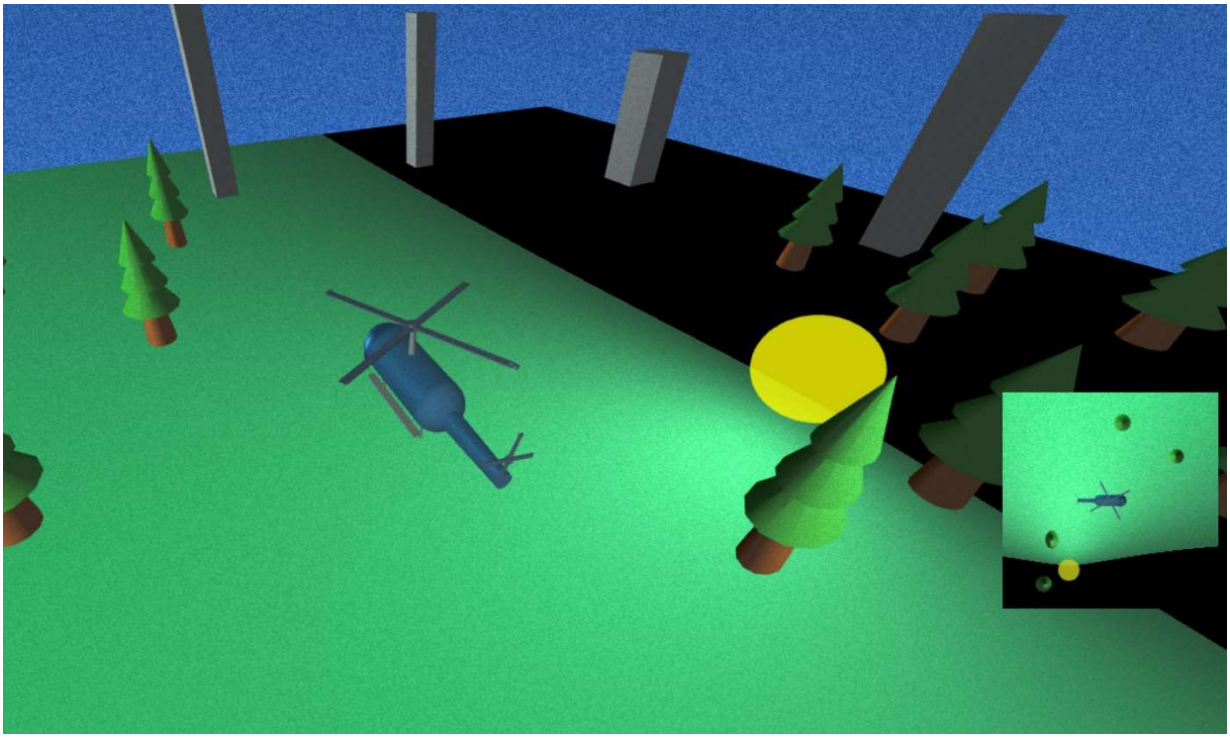
Table 01: Project Feature Table

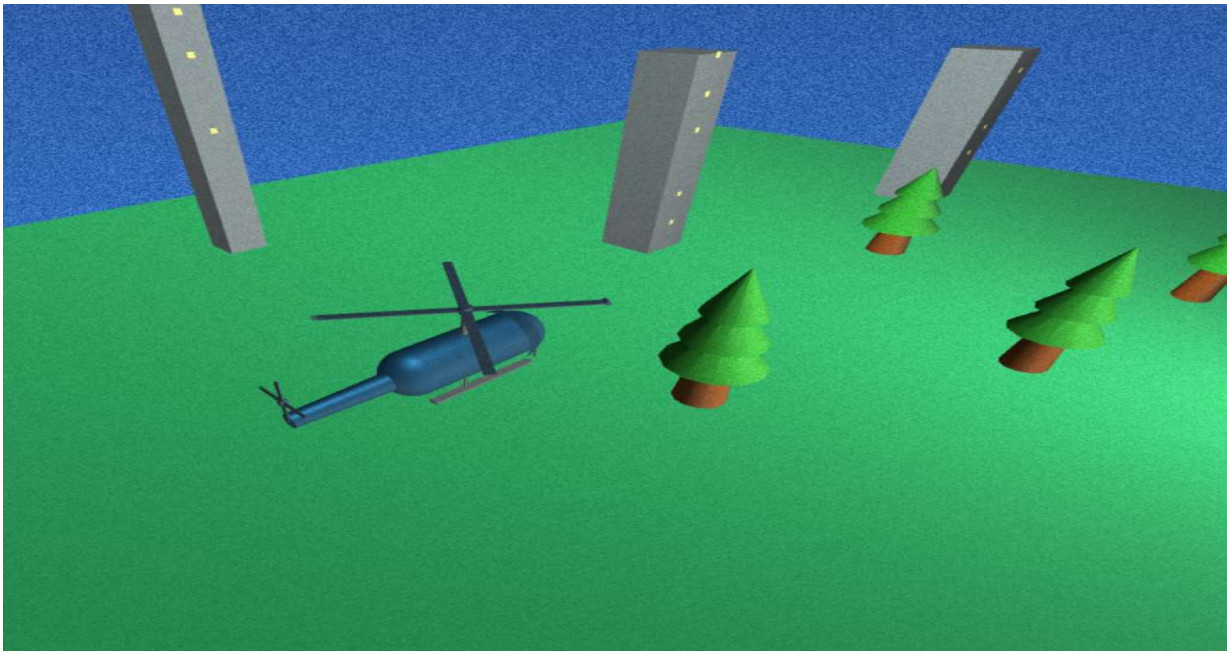
# Snapshots:

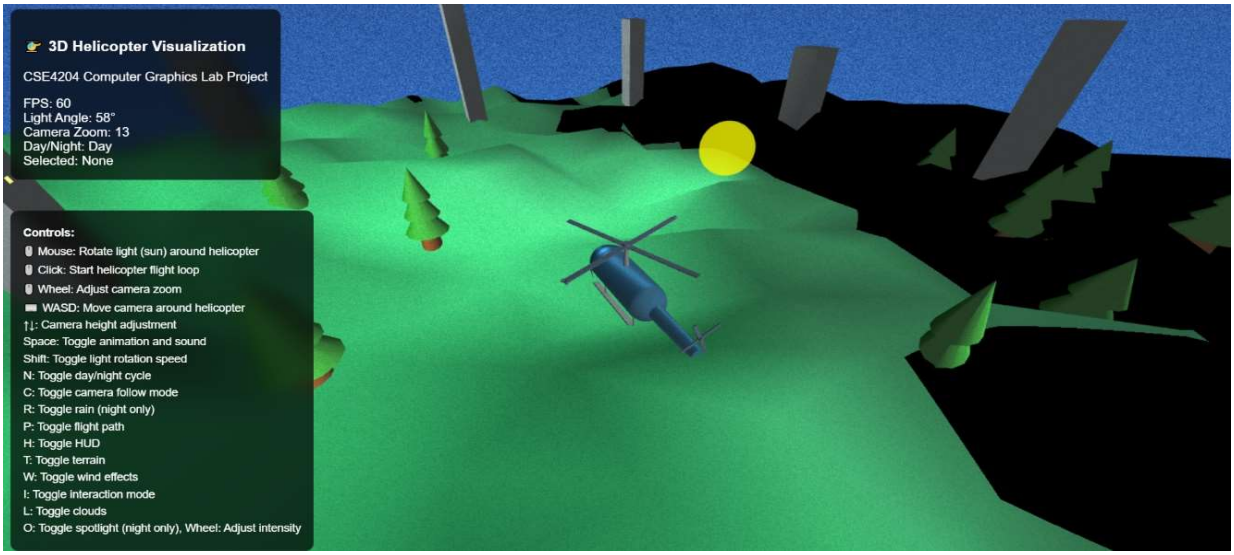
**Snapshots of our project features like -**Day Mode Overview: Helicopter in default position with detailed environment ,Night Mode: Spotlight effect with illuminated helipad and stars ,Flight Path: Helicopter following circular trajectory with path visualization ,HUD Display: Information overlay with mini-map visible ,Weather Effects: Rain particles during night mode ,Interactive Selection are given below:

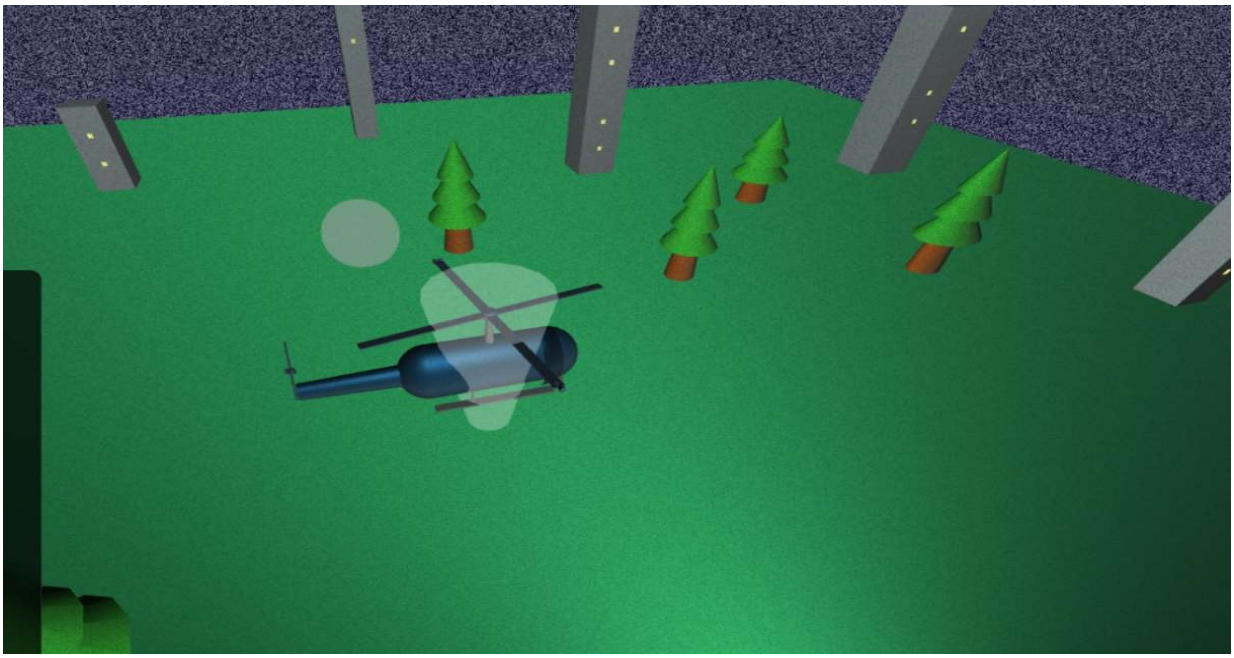










# Contribution:

- **Alimul Islam Eram Khan (20200104110)-50%**: Helicopter model (helicopter.js), animations (animations.js), shaders (shaders.js), lighting (main.js, spotlight.js), wind/weather (wind.js, weather.js), terrain/interactions (terrain.js, interactions.js), UI (index.html, main.js).

- **Md. Fahim Shakil Chowdhury (20200104128)-50%**: Environment (environment.js), clouds and helipad (clouds.js, helipad.js), HUD and controls (hud.js, controls.js), flight path (flightpath.js), UI( index.html ,styles.css)

## Future Work:

- **Model Import**: Use external 3D models for enhanced detail.
- **Weather Expansion**: Include snow or fog.
- **Direct Controls**: User-controlled helicopter movement.
- **Advanced Terrain**: Biomes and deformation.
- **Multiplayer**: Networked helicopters.
- **VR Integration**: WebXR support.
- **Optimization**: Improve performance for diverse devices.
- **AI Elements**: Automated scene behaviors.