Project 2: Non-Photorealistic Rendering Shaders in WebGL

1. Project Overview

In this term project, students will **design and implement a collection of Non-Photorealistic Rendering (NPR) shaders** using **WebGL**. The goal is to explore how artistic styles—particularly those inspired by **oil painting brushstrokes, ink drawings, and hand-crafted illustration techniques**—can be simulated through programmable shading. Students are expected to combine technical understanding of GPU shader programming with creative experimentation in visual style.

The final system will be a **custom rendering pipeline** capable of applying multiple NPR effects to 3D models or scenes in real time. Students should provide a clear way to compare different NPR algorithms, both visually and technically.

2. Learning Objectives

By completing this project, students will:

- Gain hands-on experience with **WebGL**, GLSL shader programming, and GPU pipelines.
- Understand core NPR techniques and their mathematical/algorithmic foundations.
- Explore how traditional artistic styles (e.g., oil brushstrokes, hatching, contouring) can be transformed into digital rendering strategies.
- Build an extensible rendering pipeline that combines **vertex shaders**, **fragment shaders**, **post-processing passes**, and optional multi-pass rendering.
- Demonstrate algorithmic creativity by designing at least one **custom** NPR effect inspired by an artistic medium.

3. NPR Techniques to Explore

Students must implement **at least two** of the NPR techniques below, including at least one student-designed or significantly extended method.

Core Techniques:

- **Toon Shading (Cel Shading)**

  Simplified lighting model with quantized shading bands and stylized highlights.

- **Edge Detection (Silhouette or Screen-Space Edges)**

  Detect feature edges using normal discontinuities, depth differences

- **Hatching / Cross-Hatching (difficult!)**

  Tone-mapping via line textures or procedural strokes, dynamically aligned with surface curvature or lighting gradients.

- **Oil-Painting Brushstroke Simulation (easy to implement, but requires some planning)**

Using noise functions, stroke textures, or tangent-space orientation to mimic the layered textured look of oil paint.

4. Requirements & Deliverables

A. Functional Requirements

1. A WebGL-based renderer capable of loading and displaying at least one 3D model or scene.
2. Multiple NPR shaders selectable at runtime (minimum: **two distinct techniques**).
3. A comparison interface (toggle or side-by-side view) to evaluate stylistic differences across techniques.
4. Real-time rendering ($\geq$ 30 FPS on a typical laptop GPU).

B. Technical Requirements

- Code must use **WebGL** or **WebGL2** directly (no Three.js).
- GLSL shaders must be written by the student.
- Proper use of uniforms, attributes, buffers, textures, and shader programs.
- Clean modular architecture (e.g., shader loader, model loader, render loop).

C. Documentation

A written report (4–6 pages) including:

1. **Description of each NPR algorithm** implemented, with formulas or pseudocode.
2. **Artistic inspiration**, including at least one oil painting or brushstroke technique.
3. **Comparative analysis** of the shaders: performance, visual expressiveness, limitations.
4. **Implementation details**: design decisions, pipeline diagrams, optimization steps.
5. **Screenshots** of each method applied to the same object/scene.

5. Suggested Tools & Resources

- WebGL / WebGL2 API
- glMatrix or lightweight math libraries (optional)
- OBJ/GLTF model loaders (student-implemented or minimal adapted code)
- Online shader tools (ShaderToy, GLSL Sandbox) for prototyping
- Reference literature on NPR (Strothotte & Schlechtweg, Gooch et al.)