# Implementation Report

## Introduction

Shaders are computer programs for rendering graphic data. They run on GPU and calculate visual elements and control aspects such as texture, color, lightening and others. This project implements a DirectX 11 program that displays various 3D models using HLSL shaders.

## Shaders Overview

There are many different types of shaders. The most common are the vertex shader and the pixel shader. Vertex shaders operate on vertices, producing one vertex as its output from a set of vertices provided as input. Pixel shaders (or fragment shaders) produce the color of the pixel as output from color, texture coordinates and other data as input (Fosner, 2003). In the modern graphics pipeline the vertex shader stage goes after assembling data to primitives. After that, tessellator computes detailed geometric surfaces. Next, geometry shader generates additional vertices. Then vector information is converted into a raster image. Finally, pixel shader calculates shading and outputs a final pixel color (Krasnoproshin *et al.*, 2020).

This project implements mostly vertex shaders and pixel shaders. Typically, vertex shader transforms each vertex position into screen space. The output of the vertex shader is passed to the pixel shader. The pixel shader calculates the color of the pixel and outputs the final pixel color (Zentai *et al.*, 2013). Combining different vertex and pixel shaders provides different model effects.

## Implementation Techniques in the Shaders

* The shadow mapping was implemented for all the models.
* Teapot is controlled by Key\_I, Key\_K, Key\_J, Key\_L, Key\_U, Key\_O, Key\_Period, Key\_Comma.
* Per-pixel lighting calculations.
* Wiggling effect was implemented by recalculation the wiggle variable in the UpdateScene() function. Wiggling\_vs.hsls is a shader that wiggles the sphere.
* Scrolling\_ps.hsls is a pixel shader for scrolling the texture up.
* The Cube changing the texture from stone to wood and back (MixingTextures\_ps)
* Cartoon style of the Troll: CellShading\_vs.hsls, CellShading\_ps.hsls, CellShadingOutline\_ps.hsls, CellShadingOutline\_vs.hsls. Also, the Troll casts a shadow.

## Potential Improvements

* Lightning for the wiggling sphere.
* Adding point lights.
* Adding bumping to the floor.
* Adding more controls.

## References

Fosner, R. (2003) *Real-Time Shader Programming*. San Francisco: Morgan Kaufmann Publishers.

Krasnoproshin, V., Mazouka, D. (2020) 'Graphics Pipeline Evolution Based on Object Shaders', *Pattern Recognit and Image Analysis.* 30, pp. 192–202. Available at: <https://doi.org/10.1134/S105466182002008X> (Accessed: 3 April 2025).

Zentai, N. Zs., Kaczur, N.S. (2013) 'What are the shaders and how they work?', *Informatika*, 15(1), pp. 4-7. Available at: <https://web.archive.org/web/20180421003516id_/http://informatika.gdf.hu/wp-content/uploads/sites/25/2017/04/informatika_41_02.pdf> (Accessed: 3 April 2025).