UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

FAKULTAS
ILMU
KOMPUTER

CSCE604029 • Computer Graphics
Semester Gasal 2025/2026
Fakultas Ilmu Komputer, Universitas Indonesia

Assignment 2: Mesh Editor

Deadline: xx October 2025 pukul 23.59 WIB

Alignment with CPMK

- Menguasai konsep-konsep dasar grafika komputer secara teoritis seperti model kamera, properti warna dan cahaya, rendering, iluminasi, dan geometri.
- Dapat mendesain solusi grafika komputer untuk pembentukan citra digital menggunakan teknik pemodelan dasar, terutama menggunakan metode berbasis rasterization dan ray tracing.

1 Introduction

This assignment focuses on implementing a **Halfedge Mesh data structure** and several **mesh subdivision algorithms**, which are fundamental techniques in computer graphics for representing and refining polygonal surfaces. Unlike simple vertex-face lists, the halfedge representation explicitly encodes mesh connectivity, making it efficient to traverse adjacency relationships (vertices, edges, and faces).

On top of this structure, students will implement several algorithms such as **Linear Subdivision**, **Triangulation**, **Loop Subdivision**, and **Catmull–Clark Subdivision**, enabling the conversion of coarse polygonal models into smoother and more detailed meshes. Through this assignment, you will gain a deeper understanding of mesh topology, geometry processing, and the foundations of modern surface modeling.

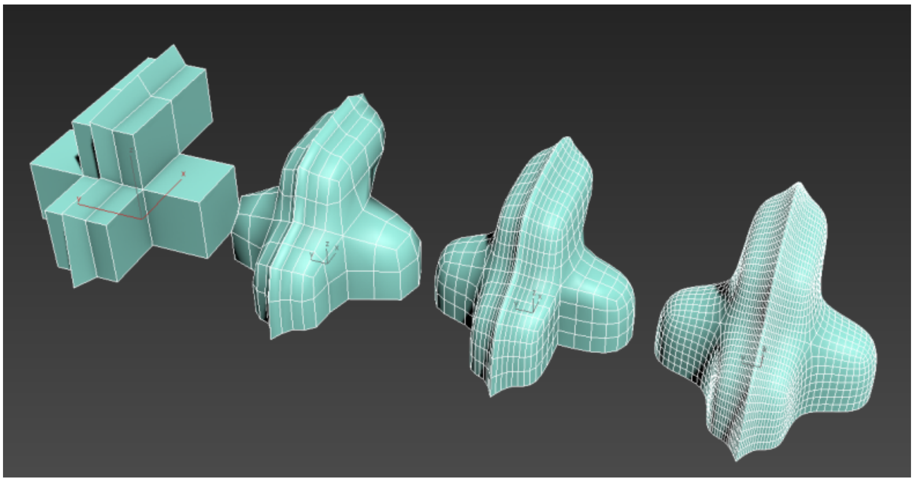

Figure 1: Subdivision example

1.1 What is Halfedge Mesh?

A halfedge mesh is a data structure used to represent polygonal meshes in computer graphics and geometry processing. Unlike simple adjacency lists or winged-edge representations, the halfedge structure explicitly encodes connectivity by splitting each edge into two oppositely directed halfedges.

1.1.1 Motivation for Halfedge Representation

1.1.2 Basic Elements

1.1.3 Mesh Traversal

1.2 Fundamental Mesh Geometry

...

1.2.1 Centroid (Face Point)

1.2.2 Edge Midpoint (Edge Point)

1.2.3 Vertex Valence

1.2.4 Normal & Area

1.3 Subdivision Algorithms

...

1.3.1 Linear Subdivision

1.3.2 Loop Subdivision (Triangular Meshes)

1.3.3 Catmull–Clark Subdivision (General Meshes)

1.4 Topology and Adjacency

...

1.4.1 Edge–to–Faces Adjacency

1.4.2 Vertex-to-Faces, Edges, Neighbors

2 Code Flow and Implementation Tasks

2.1 System Architecture Overview

...

2.2 Detailed Code Execution Flow

...

2.3 Task Breakdown

...

2.4 Task Dependencies and Recommended Order

...

3 Specifications

3.1 Application Usage

Students are expected to implement missing components of a 3D Halfedge Mesh editor, focusing on geometry traversal and subdivision algorithms. The application provides an interactive GUI for loading, editing, and subdividing polygonal meshes.

- **File Loading**: Import OBJ meshes via GUI dialog
- **Camera Navigation**: Orbit, pan, and zoom around the scene
- **Selection**: Click to select mesh objects
- **Operations**: Apply triangulation, linear subdivision, Loop subdivision, or Catmull–Clark subdivision
- **Visualization**: Toggle wireframe, vertex normals, face normals
- **Scene Management**: Insert/delete objects, reset camera

3.2 Testing and Validation

Two meshes are provided for validation:

- **Default Cube** – appears automatically when the program is launched
- `simple_pyramid.obj` – a low-poly pyramid included in the `meshes` folder

Validation Procedure:

1. Load either the default cube or `simple_pyramid.obj`.
2. Apply **Linear Subdivision** and confirm each face is split into smaller faces.
3. Apply **Triangulate** and verify all faces become triangles.

4. Apply **Loop Subdivision** (only valid on fully triangular meshes) and observe the increase in smoothness.

5. Apply **Catmull–Clark Subdivision** and verify quads are generated consistently.

6. Compare mesh statistics (vertex count, face count, edge count) before and after each operation using the `statistics()` function.

Correctness should be verified both visually (mesh smoothness, quad/triangle structure) and numerically (surface area, volume consistency).

4 Submission Format

Submit your source code as a single ZIP file on SCeLE and be prepared for a **live demo** where your team explains and demonstrates your implementation. This assignment is done in **pairs**.

ZIP Archive

- **Filename:** `<NPM1>-<NPM2>-A2-mesh-editor.zip`

Live Demo (pair)

- **Schedule:** TBA by the TAs on SCeLE.

Happy Rendering!