Министерство образования Республики Беларусь

Учреждение образования

«Брестский Государственный технический университет»

Кафедра ИИТ

**Лабораторная работа №7**

По дисциплине «Графический интерфейс интеллектуальных систем»

Тема: «Алгоритмы построения триангуляции Делоне»

**Выполнил:**

Студент 3 курса

Группы ИИ-21

Пучинский А.А.

**Проверил:**

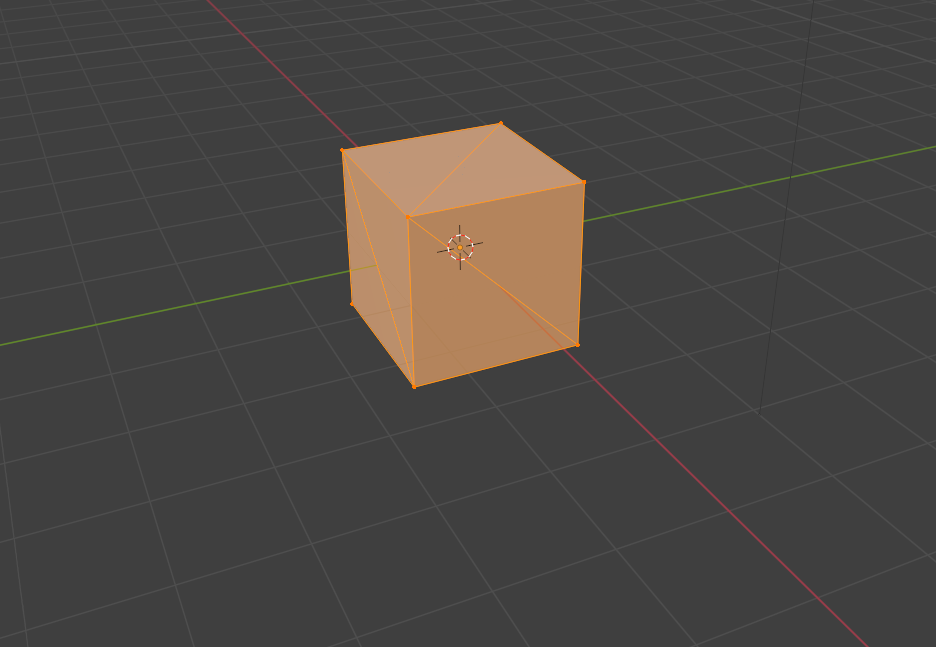
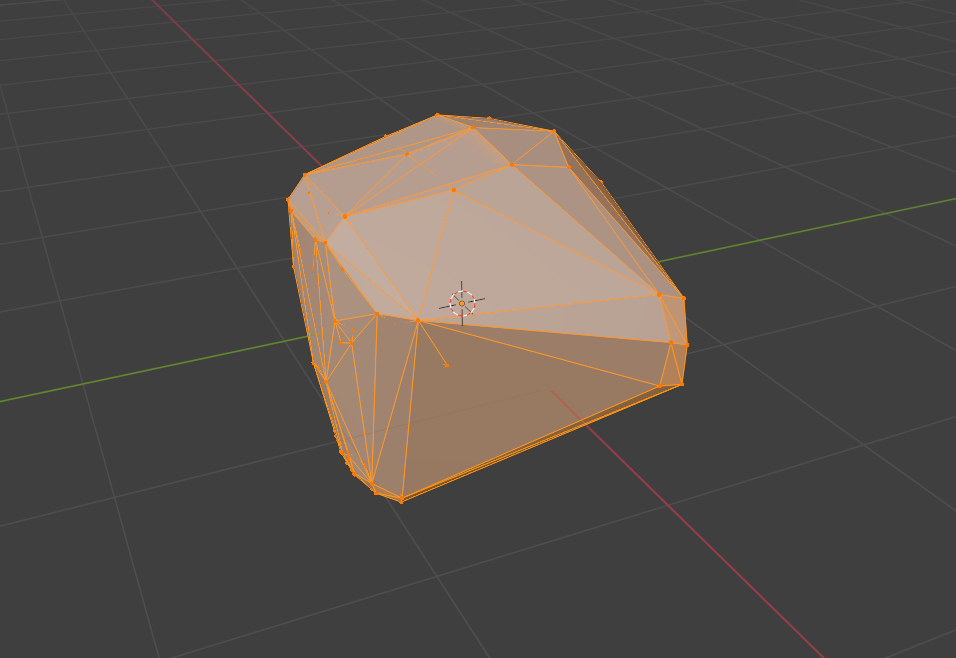
Михно Е. В.

Брест 2024

**Цель работы:** научиться использовать алгоритм построения триангуляции Делоне.

Ход работы:

После применения Алгоритма видим результат на разных фигурах:



Код алгоритма:  
import bpy

import bmesh

def convex\_hull\_merge(verts):

"""

Применяет выпуклый полосовый алгоритм слияния к набору вершин.

Args:

verts: список вершин.

Returns:

list: список вершин оболочки.

"""

# Базовый случай: если вершин меньше 4, то это уже оболочка.

if len(verts) < 4:

return verts

# Отсортировать вершины по координатам X.

verts.sort(key=lambda v: v.co.x)

# Разделить список вершин на две половины.

mid = len(verts) // 2

left\_verts = verts[:mid]

right\_verts = verts[mid:]

# Рекурсивно применить алгоритм к обеим половинам.

left\_hull = convex\_hull\_merge(left\_verts)

right\_hull = convex\_hull\_merge(right\_verts)

# Объединить левую и правую оболочки с помощью алгоритма слияния.

return merge\_hulls(left\_hull, right\_hull)

def merge\_hulls(left\_hull, right\_hull):

"""

Объединяет две оболочки в одну выпуклую оболочку.

Args:

left\_hull: список вершин левой оболочки.

right\_hull: список вершин правой оболочки.

Returns:

list: список вершин общей оболочки.

"""

# Найти самую правую и самую левую вершины оболочек.

rightmost\_left = max(left\_hull, key=lambda v: v.co.x)

leftmost\_right = min(right\_hull, key=lambda v: v.co.x)

# Найти вершины ниже линии, соединяющей две вершины.

lower\_left = max([v for v in left\_hull if v.co.x <= leftmost\_right.co.x], key=lambda v: v.co.y)

lower\_right = min([v for v in right\_hull if v.co.x >= rightmost\_left.co.x], key=lambda v: v.co.y)

# Объединить оболочки.

merged\_hull = [v for v in left\_hull if v != lower\_left] + [lower\_left, lower\_right] + [v for v in right\_hull if v != lower\_right]

return merged\_hull

# Получить активный объект.

obj = bpy.context.active\_object

if obj and obj.type == 'MESH' and obj.data.vertices:

# Получить данные сетки объекта.

mesh = obj.data

# Создать бмеш-представление сетки.

bm = bmesh.new()

bm.from\_mesh(mesh)

# Получить все вершины сетки объекта.

verts = bm.verts

# Применить алгоритм слияния оболочек к вершинам.

hull\_verts = convex\_hull\_merge(verts)

# Создать оболочку из вершин.

bmesh.ops.convex\_hull(bm, input=hull\_verts)

# Обновить сетку Blender.

bm.to\_mesh(mesh)

mesh.update()

bm.free()

else:

print("Active object is not a mesh or has no vertices.")

**Вывод:** научился использовать алгоритм построения триангуляции Делоне.