Київський національний університет імені Т. Шевченка
Факультет комп'ютерних наук та кібернетики

Комп'ютерна графіка

Лабораторна робота №3

Звіт

Виконав: студент групи ІПС-31

Мисечко Артемій

Київ-2020

Умова лабораторної роботи:

Геометричний пошук.

Локалізація точки на планарному розбитті методом деталізації тріангуляції.

Алгоритм розв'язання:

1. Вказані координати точок на площині та їх з'єднання між собою.
2. Сортуємо усі точки по осі Oy, при рівності значень по Ox.
3. Будуємо планарний граф (використовуємо списки суміжності), де вершини є задані точки, а ребра - поєднуючі їх відрізки (з самого початку вважаємо, що граф є регуляризований).
4. Будуємо граф деталізації тріангуляції:
   a. Охоплюємо граф навколо трикутником і поєднуємо його вершини з вершинами графу;
   b. Розбиваємо усі багатокутники у графі на *y*-монотонні:
      i. Кожній вершині задаємо тип: *start, end, split, merge, common*.
      ii. *Start* - це вершина з найбільшим значенням осі *Oy* (належить трикутнику який охоплює граф);
      iii. *End* - це вершина з найбільшим значенням осі *Oy* (належить трикутнику який охоплює граф);
      iv. *Split* - це вершина в яку входять ребра з вершин координати яких по осі *Oy* менші за дану;
      v. *Merge* - це вершина в яку входять ребра з вершин координати яких по осі *Oy* більші за дану;
      vi. *Common* - це вершина в яку входять ребра з вершин координати яких по осі *Oy* менші та більші за дану;
      vii. Кожному ребру надаємо помічника - це вершина, яка лежить правіше від даного ребра, що була оброблена замітаючою прямою;
      viii. Замітаючою прямою йдемо по осі *Oy* від найбільшої до найменшої координати. Зберігаємо усі поточні ребра у дереві (відсортовані зліва на право), для швидкого їх знаходження;
      ix. Обробляємо вершину в залежності від її типу;
      x. На виході маємо, що усі *split* та *merge* вершини перетворюються на *common*.
   c. Кожний *y*-монотонний прямокутник розбиваємо на трикутники:
      i. Зберігаємо вершини в стеці (з самого початку вносимо 2 найбільші за *Oy* вершини). Розділяємо багатокутник на 2 ланцюги;
      ii. Визначаємо якому ланцюгу належить поточна вершина і перша вершина у стеці;

iii. Якщо вони з різних ланцюгів, то поєднуємо поточну точку з усіма іншими до кінця стеку. Заносимо в стек останню в стеці, що була і поточну;
iv. Якщо вони з однакових ланцюгів, то циклічно перевіряємо кут повороту поточної вершини з двома верхніми у стеці. В залежності від кута, та якому ланцюгу належить вершини, додаємо нове ребро і виносимо верхню вершину зі стеку, чи заносимо поточну вершину у стек;
v. Останню вершину обробляємо окремо: поєднуємо з усіма, що є стеці(і не є суміжними).

d. Маємо триангуляцію рівня h;
e. Якщо є минулий рівень тріангуляції, то поєднуємо трикутники, які перетинаються між собою на різних рівнях;
f. Вилучаємо усі несуміжні вершини з найбільшою кількістю суміжних вершин(крайні, що належать трикутнику не чіпаємо);
g. Повторюємо алгоритм тріангуляції для наступного рівня (Повертаємося на пункт 4b) до тих пір поки не лишиться один трикутник;

5. Пошук на тріангуляції починаємо з останнього рівня. При перевірці, що точка є в трикутнику, переходимо у трикутники попереднього рівня, що перетинають поточний, і так до тих пір поки не дійдемо до останнього рівня.

Передобробка: O(nlogn) - побудова графу трінгуляції.

Алгоритм пошуку: O(log(n)) - час пошуку точки.

Витрати по пам'яті: O(n) - зберігання вершин, ланцюгів та графу.