

Міністерство освіти і науки, молоді та спорту України

Національний Технічний Університет України

«Київський Політехнічний Інститут»

Навчально-науковий комплекс

«Інститут прикладного системного аналізу»

Кафедра системного проектування

«**ОБЧИСЛЮВАЛЬНА ГЕОМЕТРІЯ»**

Методичні вказівки до виконання лабораторних робіт для студентів напряму підготовки 6.050101 «Комп'ютерні науки», спеціальностей 8.05010102 «Інформаційні технології проектування» та 8.05010103 «Системне проектування» денної та заочної форм навчання

Склав: доц. Харченко Костянтин Васильович

Київ – 2012

Обчислювальна геометрія: Методичні вказівки до виконання лабораторних робіт для студентів напряму підготовки 6.050101 «Комп'ютерні науки», спеціальностей 8.05010102 «Інформаційні технології проектування» та 8.05010103 «Системне проектування» денної та заочної форм навчання / Укл. Харченко К.В. – К. : НТУУ «КПІ», 2012 р. – \_\_ c.

Рекомендовано Методичною радою НТУУ «КПІ»

(Протокол № \_\_ від \_\_.\_\_.201\_ р.)

|  |  |  |
| --- | --- | --- |
| *Укладачі:* |  | *Харченко Костянтин Васильович, кандидат технічних наук* |

ЗМІСТ

ЛАБОРАТОРНА РОБОТА №1

**Визначення точки перетину двох відрізків. Визначення кута між двома відрізками.**

**Мета роботи:** ознайомитися з основними базовими об’єктами та алгоритмами обчислювальної геометрії. Набути практичний досвід програмування базових алгоритмів обчислювальної геометрії на об’єктно-орієнтованій мові програмування.

**Задача:** Дослідити алгоритм та написати програму для визначення точки перетину двох відрізків. Функція повинна повертати значення True / False і координати точки перетину (як аргументи), якщо відрізки перетинаються. Відрізки задаються у вигляді координат їх вершин. Дослідити алгоритм та написати програму для визначення кута між двома відрізками. Функція повинна повертати значення кута. Відрізки вважати векторами, та кут визначити як кут між векторами. Кут має приймати значення від 0 до 360 градусів. Відлік починається від першого вектору проти годинникової стрілки до другого вектора). Відрізки задаються координатами їх вершин.

**Короткі теоретичні відомості**

Рівняння прямої, що проходить через дві точки (відрізки задані двома своїми кінцевими точками):

\left(x - x_1 \right)/ \left(x_2 - x_1 \right) = \left(y - y_1 \right)/ \left(y_2 - y_1 \right),

де x_1, y_1 и x_2, y_2 — координати, відповідно, першої та другої точок відрізка.

Скалярний добуток векторів:

http://www.pm298.ru/Math/f319.JPG

де http://www.pm298.ru/Math/f320.JPG - кут між векторами http://www.pm298.ru/Math/f304.JPG та http://www.pm298.ru/Math/f306.JPG

**Завдання**

1. На основі рівняння прямої скласти систему рівнянь для визначення точки перетину двох відрізків. Скласти додаткові умови для аналізу розташування точки перетину прямих у межах першого та другого відрізку.
2. На основі скалярного добутку векторів скласти рівняння для визначення кута між двома векторами. Значення кута підліковується від першого вектора до другого вектора проти годинникової стрілки та має знаходитися в межах від 0 до 360 градусів (0..2Pi).
3. Скласти блок-схему алгоритму.
4. Скласти тестову програму на об’єктно орієнтованій мові програмування.
5. Провести тестування тестової програми на різноманітних розташуваннях відрізків на площині. Визначити потрібні частинні випадки розташування відрізків на площині. Для тестування знаходження кута між векторами знайти значення кутів для векторів <a,b та <b,a.

**Зміст звіту**

1. Алгоритм і постановка задачі
2. Блок-схема алгоритму
3. Основні моменти вихідного тексту програми
4. Тестовий набір вхідних даних
5. Геометрична ілюстрація набору вхідних даних
6. Висновки

**Контрольні питання**

1. Чому вертикальне розташування векторів може приводити тестову програму до арифметичних винятків роботи та як їх уникнути за допомогою додаткових умов?
2. Чому тестові відрізки з невеличким (1e-6 градуса) нахилом можуть спричиняти до винятків роботи тестової програми?
3. Який діапазон значень кута між двома векторами? Чому в лабораторній роботі вимагається знайти кут між відрізками зі значенням кута від 0 до 360 градусів?

ЛАБОРАТОРНА РОБОТА №2

**Метод локалізації точки в простому багатокутнику.**

**Мета роботи:** ознайомитися з алгоритмом локалізації точки в простому багатокутнику при одноразовому запиті. Набути практичний досвід програмування базових алгоритмів обчислювальної геометрії на об’єктно-орієнтованій мові програмування.

**Задача:** дослідити алгоритм локалізації точки в простому багатокутнику при одноразовому запиті. Реалізувати тестову програму на об’єктно-орієнтованій мові програмування.

**Короткі теоретичні відомості**

Простий багатокутник не має самоперетинів.

Належність точки простому багатокутнику. Дано простий багатокутник P і точка z. Визначити, чи знаходиться z всередині P. Розглянемо випадок унікального запиту. Проведемо через точку z горизонталь l. Якщо l не перетинає P, то z – зовнішня точка. Нехай тепер l перетинає P. Розглянемо спочатку випадок, коли l не проходить через вершини P. Нехай L - число точок перетину l з кордоном P лівіше z. Оскільки P обмежений, лівий кінець l лежить поза P. Рухаючись з - ∞ направо, при першому перетині кордону потрапляємо всередину P, при другому - виходимо назовні з P, при третьому - знову всередину і т.д. Тому z лежить всередині P тоді і тільки тоді, коли L непарне. Тепер розглянемо вироджений випадок, коли l проходить через вершини P. Нескінченно малий поворот l навколо z проти годинникової стрілки не змінить локалізації точки, але усуне виродженість. Тепер видно наступне. Якщо обидві вершини ребра, належать l, то його не слід враховувати. Якщо ж тільки одна вершина ребра лежить на l, то перетин слід врахувати, якщо це вершина з більшою ординатою, і ігнорувати в іншому випадку.

**Завдання**

1. На основі теоретичного матеріалу скласти алгоритм роботи програми для локалізації точки у простому багатокутнику.
2. Скласти блок-схему алгоритму.
3. Скласти тестову програму на об’єктно орієнтованій мові програмування.
4. Провести тестування тестової програми на різноманітних варіантах багатокутника та розташування точки для локалізації. Скласти перелік частинних випадків та продемонструвати роботу тестової програми для них.

**Зміст звіту**

1. Алгоритм і постановка задачі
2. Блок-схема алгоритму
3. Основні моменти вихідного тексту програми
4. Тестовий набір вхідних даних
5. Геометрична ілюстрація набору вхідних даних
6. Висновки

**Контрольні питання**

1. Як визначити складність роботи алгоритму локалізації точки в багатокутнику O(f(n)) ?
2. Чому алгоритм має таку складність?
3. Які частинні випадки має алгоритм локалізації точки в багатокутнику?
4. Які функції з попередніх лабораторних робіт були використані в алгоритмі?

ЛАБОРАТОРНА РОБОТА №3

**Знаходження опуклої оболонки для множини точок на плоскості.**

**Мета роботи:** ознайомитися з алгоритмом знаходження опуклої оболонки для множини точок на плоскості. Набути практичний досвід програмування базових алгоритмів обчислювальної геометрії на об’єктно-орієнтованій мові програмування.

**Задача:** дослідити алгоритм знаходження опуклої оболонки для множини точок на плоскості. Реалізувати тестову програму на об’єктно-орієнтованій мові програмування.

**Короткі теоретичні відомості**

Нехай на площині задані k різних точок p1, p2, ..., pk. Множина точок

p = α1p1 + α2p2 + ... + αkpk,

(αi ∈ R, αi ≥ 0, α1 + α2 + ... + αk = 1)

називається опуклою множиною, породженим точками p1, p2, ..., pk, а точка p називається опуклою комбінацією точок p1, p2, ..., pk. Опуклою оболонкою множини точок L на площині (позначається conv (L)) називається найменша опукла множина, що містить L. Надалі ми розглядаємо лише кінцеві множини L. Опукла оболонка кінцевої множини точок на площині є опуклим багатокутником і навпаки, кожен опуклий багатокутник є опуклою оболонкою деякої множини точок. Задача побудови опуклої оболонки ставиться наступним чином. Для

множини L з n точок потрібно побудувати опуклу оболонку conv (L) в

вигляді повного опису кордону. Цей опис кордону є впорядкована підмножина точок з L, званих граничними точками. Решта точки з L, що не є граничними, називаються внутрішніми.

Метод Джарвіса (загортання подарунка). Припустимо, що знайдена найменша в лексикографічному порядку точка p1 заданої множини L. Це означає, що p1 має мінімальну ординату (тобто координату y) серед точок з L. Ця точка завідомо гранична, тобто є вершиною опуклої оболонки. Знайдемо тепер наступну за нею вершину p2 опуклої оболонки. Точка p2 - це точка, яка має найменший позитивний полярний кут відносно точки p1 як початку координат. Далі наступна гранична точка p3 вибирається таким чином, щоб вектор p2 p3 мав найменший позитивний кут щодо вектора p1p2. Аналогічно шукаються інші граничні точки: точка pk +1 вибирається так, щоб вектор pk pk +1 мав найменший позитивний кут щодо вектора pk-1pk для k ≥ 2. Алгоритм Джарвіса обходить опуклу оболонку, породжуючи в потрібному порядку послідовність крайніх точок по одній на кожному кроці. Так як всі точки з L можуть виявитися граничними, а алгоритм Джарвіса витрачає на знаходження однієї граничної точки лінійний час, то загальний час виконання алгоритму в гіршому випадку становить O (n2). Однак, якщо в дійсності число вершин опуклої оболонки одно h, то час виконання алгоритму Джарвіса буде O (nh), тобто він дуже ефективний, якщо h мале.

**Завдання**

1. На основі теоретичного матеріалу скласти алгоритм роботи програми для знаходження опуклої оболонки.
2. Скласти блок-схему алгоритму.
3. Скласти тестову програму на об’єктно орієнтованій мові програмування.
4. Провести тестування тестової програми на різноманітних варіантах множин точок з різними параметрами n та h. Скласти перелік частинних випадків та продемонструвати роботу тестової програми для них.

**Зміст звіту**

1. Алгоритм і постановка задачі
2. Блок-схема алгоритму
3. Основні моменти вихідного тексту програми
4. Тестовий набір вхідних даних
5. Геометрична ілюстрація набору вхідних даних
6. Висновки

**Контрольні питання**

1. Яку складність має алгоритм Джарвіса?
2. Яку складність має алгоритм Грехема? Чому алгоритм Грехема працює швидше?
3. Які частинні випадки має алгоритм Джарвіса?
4. Чому алгоритм є стабільним до вхідного набору даних та має порівняно невелику кількість частинних випадків?

ЛАБОРАТОРНА РОБОТА №4

**Побудова сітки кінцевих елементів для методів кінцевих елементів.**

**Мета роботи:** ознайомитися з методом побудови сітки кінцевих елементів для методів кінцевих елементів за допомогою алгоритму «вирівнювання-виїмка». Набути практичний досвід програмування базових алгоритмів обчислювальної геометрії на об’єктно-орієнтованій мові програмування.

**Задача:** дослідити алгоритм знаходження опуклої оболонки для множини точок на плоскості. Реалізувати тестову програму на об’єктно-орієнтованій мові програмування.

**Короткі теоретичні відомості**

Метод вирівнювання-виїмка дозволяє заповнювати трикутниками багатокутник, який позначає границю об’єкту моделювання для МКЕ. До форми та розміру кінцевих елементів накладаються певні вимоги: трикутники мають наближатися до рівностороннього трикутника, а розмір сторони трикутника має наближуватися до еталонного значення параметру довжини трикутника. Багатокутник заповнюється трикутними кінцевими елементами за таким алгоритмом:

1. Знаходимо найменший внутрішній кут багатокутника.
2. Якщо кут є меншим 75 градусів, то будуємо один трикутник всередину області моделювання – «вирівнювання»
3. Якщо кут є більшим за 75 градусів, то будуємо два трикутника всередину області моделювання. Для цього проведемо бісектрису з кута та відкладемо на ній відрізок 1.2-1.4 d, де d – еталонна довжина кінцевого елементу – «виїмка».
4. Переходимо до п.1, якщо область моделювання не є трикутником з розмірами, що наближаються до еталонного.

**Завдання**

1. На основі теоретичного матеріалу скласти алгоритм побудови сітки трикутних кінцевих елементів.
2. Скласти блок-схему алгоритму.
3. Скласти тестову програму на об’єктно орієнтованій мові програмування.
4. Провести тестування тестової програми на різноманітних варіантах багатокутника області моделювання. Скласти перелік частинних випадків та продемонструвати роботу тестової програми для них.

**Зміст звіту**

1. Алгоритм і постановка задачі
2. Блок-схема алгоритму
3. Основні моменти вихідного тексту програми
4. Тестовий набір вхідних даних
5. Геометрична ілюстрація набору вхідних даних
6. Висновки

**Контрольні питання**

1. Як впливає коефіцієнт видовження середнього ребра 1.2-1.4 на розмір та форму кінцевих елементів у випадку «виїмки»?
2. Як впливає змінення значення мінімального кута 75 градусів на форму та розміри кінцевих елементів?
3. Які є частинні випадки побудови «вирівнювання» та «виїмки»?
4. Які існують подальші методи покращення сітки кінцевих елементів?

**Рекомендована література**

1. Ф. Препарата, М. Шеймос. Вычислительная геометрия: введение. “Мир”, Москва, 1989 г.
2. А.Ахо. Дж. Хопкрофт, Дж. Ульман. Построение и анализ вычислительных алгоритмов. “Мир”, Москва, 1979 г.
3. М. Ласло. Вычислительная геометрия и компьютерная графика на С++. М., “Бином,” 1997 г.

4. Фокс А., Пратт М., Вычислительная геометрия. Применение в проектировании и на производстве. Пер. с англ. М., “Мир”, 1982.