ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

PHÂN TÍCH VÀ THIẾT KẾ THUẬT TOÁN
CS112.P11.KHTN

BTVN NHÓM 2

Nhóm 7 :
Hoàng Đức Dũng - 23520328
Nguyễn Văn Hồng Thái - 23521418

Giảng viên :
Nguyễn Thanh Sơn

Ngày 2 tháng 12 năm 2024

Mục lục

1 Bài toán khu vườn

Ông Nhân có một khu vườn với n cái cây, mỗi cây được biểu diễn bởi tọa độ $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ trên mặt phẳng tọa độ. Ông Nhân muốn cột một sợi dây để bao quanh toàn bộ các cây, sao cho độ dài dây cần dùng là ngắn nhất.

Yêu cầu

1. Tìm độ dài ngắn nhất của sợi dây để bao quanh tất cả các cây.
2. Tìm thứ tự các đỉnh mà dây đi qua (theo ngược chiều kim đồng hồ).
3. Viết chương trình Python tính toán và hiển thị kết quả.

Phân tích thuật toán

Ý tưởng chính

Bài toán yêu cầu tìm **bao lồi (Convex Hull)** của tập hợp các điểm trên mặt phẳng tọa độ:

- **Bao lồi** là đa giác nhỏ nhất bao quanh tất cả các điểm.
- Chu vi của bao lồi chính là độ dài ngắn nhất của sợi dây.

Phương pháp giải

Áp dụng thuật toán **Andrew's Monotone Chain** với các bước như sau:

1. Sắp xếp toàn bộ các điểm theo hoành độ x. Nếu x bằng nhau, sắp theo tung độ y.
2. Xây dựng bao lồi theo hai phần:
   - **Hull dưới**: Duyệt từ trái sang phải để tìm phần dưới của bao lồi.
   - **Hull trên**: Duyệt từ phải sang trái để tìm phần trên của bao lồi.
3. Kết hợp hai phần để tạo thành bao lồi hoàn chỉnh.
4. Tính chu vi của bao lồi bằng cách cộng tổng khoảng cách Euclid giữa các đỉnh.

Tích chéo (Cross Product)

Tích chéo của hai vector $\overrightarrow{OA}$ và $\overrightarrow{OB}$:

$$\text{Cross\_Product}(O, A, B) = (A_x - O_x) \cdot (B_y - O_y) - (A_y - O_y) \cdot (B_x - O_x)$$

- Nếu Cross_Product > 0: Điểm B nằm bên trái đoạn OA.
- Nếu Cross_Product < 0: Điểm B nằm bên phải đoạn OA.
- Nếu Cross_Product $= 0$: Điểm B thẳng hàng với OA.

Các bước thực hiện

1. **Sắp xếp các điểm:** Sắp xếp n điểm theo x tăng dần. Nếu x bằng nhau, sắp theo y.
2. **Xây dựng hull dưới:**
   - Duyệt từ trái sang phải.
   - Loại bỏ các điểm không cần thiết bằng cách kiểm tra tích chéo.
3. **Xây dựng hull trên:**
   - Duyệt từ phải sang trái.
   - Loại bỏ các điểm không cần thiết tương tự hull dưới.
4. Tính chu vi: Cộng tổng khoảng cách giữa các cặp điểm liên tiếp trên bao lồi.

Mã Python đầy đủ

```python
import math

def cross_product(o, a, b):
    """Tinh tich cheo giua vector OA va OB."""
    return (a[0] - o[0]) * (b[1] - o[1]) - (a[1] - o[1]) * (b[0] -
        o[0])

def convex_hull(points):
    """Tim bao loi tu tap diem."""
    # Buoc 1: Sap xep diem theo x, neu x bang thi theo y
    points = sorted(points)

    # Buoc 2: Tim hull duoi
    lower = []
    for p in points:
        while len(lower) >= 2 and cross_product(lower[-2], lower
            [-1], p) <= 0:
            lower.pop()
        lower.append(p)

    # Buoc 3: Tim hull tren
    upper = []
    for p in reversed(points):
        while len(upper) >= 2 and cross_product(upper[-2], upper
            [-1], p) <= 0:
            upper.pop()
        upper.append(p)

    # Buoc 4: Ket hop hai phan hull
    return lower[:-1] + upper[:-1]
```

```
28
29 def perimeter(hull):
30     """Tinh chu vi cua da giac bao loi."""
31     return sum(math.dist(hull[i], hull[(i+1) % len(hull)]) for i
       in range(len(hull)))
32
33 # Nhap du lieu
34 n = int(input("Nhap so dinh: "))
35 trees = []
36 print("Nhap cac dinh (x y):")
37 for i in range(n):
38     x, y = map(float, input().split())
39     trees.append((x, y))
40
41 # Tim bao loi va tinh chu vi
42 hull = convex_hull(trees)
43 length = perimeter(hull)
44
45 # In ket qua
46 print("Do dai day ngan nhat:", length)
47 print("Danh sach dinh theo nguoc chieu kim dong ho:", hull)
```

Listing 1: Thuật toán Convex Hull để tìm độ dài dây ngắn nhất

Kết quả

- **Độ dài dây ngắn nhất:** Kết quả in ra từ hàm `perimeter(hull)`.
- **Danh sách các đỉnh:** Kết quả in ra từ hàm `convex_hull(points)`, sắp xếp theo chiều ngược kim đồng hồ.

2 Khu vườn giao nhau

Ông Nhân có hai mảnh đất, mỗi mảnh đất được mô tả bởi một đa giác lồi trong mặt phẳng tọa độ.

- Mảnh đất thứ nhất có các đỉnh $A_1(x_1, y_1), A_2(x_2, y_2), \ldots, A_m(x_m, y_m)$.
- Mảnh đất thứ hai có các đỉnh $B_1(x'_1, y'_1), B_2(x'_2, y'_2), \ldots, B_n(x'_n, y'_n)$.
- Các đa giác được biểu diễn theo thứ tự ngược chiều kim đồng hồ.

Nhiệm vụ: Tính diện tích phần giao nhau giữa hai đa giác lồi.

Phân tích thuật toán

Ý tưởng chính

- Sử dụng thuật toán **Sutherland-Hodgman** để cắt đa giác.

- Cắt đa giác thứ nhất (A) bằng các cạnh của đa giác thứ hai (B) để tìm ra đa giác giao nhau.
- Tính diện tích đa giác giao nhau bằng công thức diện tích đa giác.

Phương pháp giải

1. Dùng thuật toán Sutherland-Hodgman để xác định đa giác giao nhau:
   - Khởi tạo đa giác giao là toàn bộ đa giác thứ nhất (A).
   - Lần lượt sử dụng từng cạnh của đa giác thứ hai (B) để cắt đa giác giao.
2. Sau khi tìm được đa giác giao, tính diện tích của nó:
   - Áp dụng công thức diện tích đa giác:

$$\text{Area} = \frac{1}{2}\left|\sum_{i=1}^{n}(x_i y_{i+1} - y_i x_{i+1})\right|$$

   với $(x_{n+1}, y_{n+1}) = (x_1, y_1)$.

Thuật toán Sutherland-Hodgman

Thuật toán hoạt động như sau:

1. Khởi tạo đa giác giao là đa giác thứ nhất (A).
2. Với mỗi cạnh của đa giác thứ hai (B):
   - Kiểm tra từng đỉnh của đa giác giao hiện tại:
     - Nếu đỉnh nằm trong nửa mặt phẳng do cạnh đang xét chia, giữ lại.
     - Nếu đỉnh nằm ngoài, tính giao điểm của cạnh với đường thẳng chia và thêm giao điểm vào.
3. Kết quả sau khi cắt lần lượt tất cả các cạnh của B sẽ là đa giác giao.

Mã Python đầy đủ

```
def cross(o, a, b):
    """Tich cheo cua vector OA va OB."""
    return (a[0] - o[0]) * (b[1] - o[1]) - (a[1] - o[1]) * (b[0] -
        o[0])

def intersect(p1, p2, q1, q2):
    """Tinh giao diem giua 2 doan thang."""
    A1, B1, C1 = p2[1] - p1[1], p1[0] - p2[0], p2[0] * p1[1] - p1
       [0] * p2[1]
```

```
    A2, B2, C2 = q2[1] - q1[1], q1[0] - q2[0], q2[0] * q1[1] - q1[0] * q2[1]
    det = A1 * B2 - A2 * B1
    if det == 0:
        return None
    x = (B2 * C1 - B1 * C2) / det
    y = (A1 * C2 - A2 * C1) / det
    return (x, y)

def is_inside(p, edge_start, edge_end):
    """Kiem tra xem diem p co nam trong nua mat phang khong."""
    return cross(edge_start, edge_end, p) >= 0

def sutherland_hodgman(subject, clip):
    """Thuat toan Sutherland-Hodgman."""
    output = subject
    for i in range(len(clip)):
        input_list = output
        output = []
        edge_start, edge_end = clip[i], clip[(i + 1) % len(clip)]
        for j in range(len(input_list)):
            current_point = input_list[j]
            prev_point = input_list[j - 1]
            if is_inside(current_point, edge_start, edge_end):
                if not is_inside(prev_point, edge_start, edge_end):
                    output.append(intersect(prev_point, current_point, edge_start, edge_end))
                output.append(current_point)
            elif is_inside(prev_point, edge_start, edge_end):
                output.append(intersect(prev_point, current_point, edge_start, edge_end))
    return output

def polygon_area(polygon):
    """Tinh dien tich da giac."""
    n = len(polygon)
    return abs(sum(polygon[i][0] * polygon[(i + 1) % n][1] - polygon[i][1] * polygon[(i + 1) % n][0] for i in range(n)) / 2)

# Nhap so dinh va toa do cac diem
m = int(input("Nhap so dinh cua da giac A: "))
polygon_a = []
print("Nhap toa do cac dinh cua da giac A (x y):")
for _ in range(m):
    x, y = map(float, input().split())
    polygon_a.append((x, y))

n = int(input("Nhap so dinh cua da giac B: "))
```

```
polygon_b = []
print("Nhap toa do cac dinh cua da giac B (x y):")
for _ in range(n):
    x, y = map(float, input().split())
    polygon_b.append((x, y))

# Tim da giac giao
intersection = sutherland_hodgman(polygon_a, polygon_b)

# Tinh dien tich phan giao
area = polygon_area(intersection)

# In ket qua
print("Dien tich phan giao:", area)
```

Listing 2: Thuật toán Sutherland-Hodgman và tính diện tích

Kết quả

- **Đa giác giao nhau:** Danh sách các đỉnh của đa giác giao in ra từ hàm `sutherland_hodgman`.
- **Diện tích giao nhau:** Kết quả in ra từ hàm `polygon_area`.