

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**  
**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN LẬP TRÌNH TÍNH TOÁN**  
**NỘI SUY ĐƯỜNG CONG BÉZIER**

Người hướng dẫn: **PGS. TS. NGUYỄN TẤN KHÔI**

Sinh viên thực hiện:

**Võ Văn Đức**

**LỚP: 21T\_DT2 NHÓM: 11**

**Nguyễn Thị Thu Thảo**

**LỚP: 21T\_DT2 NHÓM: 11**

**Đà Nẵng, 07/2022**

## MỤC LỤC

MỤC LỤC .....	i
DANH MỤC HÌNH VẼ .....	ii
MỞ ĐẦU .....	i
1. TỔNG QUAN ĐỀ TÀI .....	1
2. CƠ SỞ LÝ THUYẾT .....	1
2.1. Ý tưởng.....	1
2.2. Cơ sở lý thuyết.....	1
3. TỔ CHỨC CẤU TRÚC DỮ LIỆU VÀ THUẬT TOÁN.....	6
3.1. Phát biểu bài toán .....	6
3.2. Cấu trúc dữ liệu .....	6
3.3. Thuật toán.....	7
4. CHƯƠNG TRÌNH VÀ KẾT QUẢ .....	15
4.1. Tổ chức chương trình .....	15
4.2. Ngôn ngữ cài đặt .....	16
4.3. Kết quả thực hiện.....	16
4.3.1. Giao diện chính của chương trình.....	16
4.3.2. Các kết quả thực thi của chương trình .....	19
4.3.3. Nhận xét đánh giá kết quả.....	21
5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN .....	22
5.1. Kết luận .....	22
5.2. Hướng phát triển.....	22
TÀI LIỆU THAM KHẢO .....	23
PHỤ LỤC .....	24

## DANH MỤC HÌNH VẼ

Hình 1. Đường cong tham số bậc nhất	1
Hình 2. Đường cong tham số bậc hai	2
Hình 3. Đường cong tham số bậc ba	3
Hình 4. Đường cong Bézier bậc $p = 3$ có 4 đỉnh điều khiển	5
Hình 5. Lưu đồ hoạt động của chương trình.	15
Hình 6.1. Giao diện chính của chương trình.	16
Hình 6.2. Giao diện khi lấy dữ liệu từ file.	16
Hình 6.3. File để lấy dữ liệu.	17
Hình 6.4. Giao diện khi nhập dữ liệu từ bàn phím	17
Hình 6.5. Giao diện có lưu kết quả vào file.	18
Hình 6.6. File dữ liệu mà kết quả lưu vào.	18
Hình 6.7. Giao diện khi lấy dữ liệu từ file.	18
Hình 7.1. Kết quả của dữ liệu 1.	19
Hình 7.2. Hình ảnh đường cong 1 sau khi tái tạo.	19
Hình 8.1. Kết quả của dữ liệu 2.	20
Hình 8.2. Hình ảnh đường cong 2 sau khi tái tạo.	20
Hình 9.1. Kết quả của dữ liệu 3.	21
Hình 9.2. Hình ảnh đường cong 3 sau khi tái tạo.	21

## MỞ ĐẦU

Đường cong Bezier là một trong những đường cong cơ bản nhất, thường được sử dụng trong đồ họa máy tính, hoạt hình, mô hình hóa, xử lý hình ảnh và nhiều lĩnh vực liên quan khác. Thông thường, đường cong được thiết kế để xấp xỉ một hình dạng trong thế giới thực mà mặt khác không có biểu diễn toán học hoặc biểu diễn của nó là không xác định hoặc quá phức tạp. Đường cong Bézier được đặt theo tên kỹ sư người Pháp Pierre Bézier, người đã sử dụng nó vào những năm 1960 để thiết kế đường cong cho thân xe ô tô Renault. Vì là đường cong được viết dưới dạng cơ sở inBernstein, chúng được biết đến từ nhiều năm trước. Tuy nhiên, những ứng dụng này đã sử dụng rộng rãi trong 30 năm gần đây. Do đó, trong bài báo cáo này, nhóm mình đã chọn đề tài “Nội suy đường cong Bézier” để thông qua đó tìm hiểu về đề tài cũng như cách ứng dụng vào thực tiễn. Phương pháp nghiên cứu đề tài này là đọc hiểu lý thuyết, thông qua đó tìm hiểu về: định điều khiển, bậc, phương trình, vector nút đồng nhất, tính chất, ứng dụng; triển khai các ví dụ rồi từ đó xây dựng thuật toán và lập trình tính toán tìm các điểm điều khiển khi biết các điểm mà đường cong sẽ đi qua. Cấu trúc của báo cáo này sẽ gồm năm phần: Tổng quan về dự án, Cơ sở lý thuyết, Tổ chức cấu trúc dữ liệu và thuật toán, Chương trình và kết quả, Kết luận và hướng phát triển.

## 1. TỔNG QUAN ĐỀ TÀI

Cơ sở toán học cho đường cong Bézier — đa thức Bernstein — được thiết lập vào năm 1912, nhưng đa thức không được áp dụng cho đồ họa cho đến khoảng 50 năm sau khi nhà toán học Paul de Casteljau vào năm 1959 phát triển thuật toán de Casteljau, một phương pháp ổn định về mặt số học để đánh giá các đường cong, và trở thành người đầu tiên áp dụng chúng vào thiết kế có sự hỗ trợ của máy tính tại hãng xe Pháp Citroën. Tuy nhiên, phương pháp của de Casteljau đã được cấp bằng sáng chế ở Pháp nhưng không được xuất bản cho đến những năm 80, trong khi các đa thức Bézier được kỹ sư người Pháp Pierre Bézier công bố rộng rãi vào những năm 1960, người đã phát hiện ra chúng một cách độc lập và sử dụng chúng để thiết kế thân ô tô tại Renault.

Đường cong Bézier đang đã sử dụng trong đồ họa máy tính để sản xuất đường cong có vẻ mịn hợp lý ở tất cả các tỷ lệ. Dạng tổng quát hóa của đường cong Bézier trong không gian nhiều chiều được gọi là mặt phẳng Bézier, trong đó tam giác Bézier là một trường hợp đặc biệt.

## 2. CƠ SỞ LÝ THUYẾT

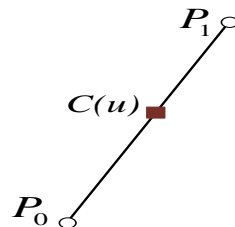
### 2.1. Ý tưởng

### 2.2. Cơ sở lý thuyết

#### 2.2.1 Một số trường hợp cụ thể

##### a) Đường cong Bézier tuyến tính

Đường cong Bézier tuyến tính là đường cong Bézier bậc 1 với  $n+1=2$  điểm điều khiển là  $P_0$  và  $P_1$ . Do đó có thể nói đây chính là đoạn thẳng đi qua 2 điểm  $P_0$  và  $P_1$ , hay phương trình đường cong Bézier tuyến tính là phương trình tham số kiểu đoạn thẳng.



Hình 1. Đường cong tham số bậc nhất

Xét đoạn thẳng có điểm đầu  $P_0$  và điểm cuối  $P_1$ ,  $C$  là điểm bất kỳ trên đoạn thẳng. Phương trình tham số của đoạn thẳng này được biểu diễn dưới dạng như sau:

$$C(u) = (1-u)P_0 + uP_1, \text{ với } u \in [0, 1]$$

Đặt:  $B_0 = 1 - u$

$$B_1 = u$$

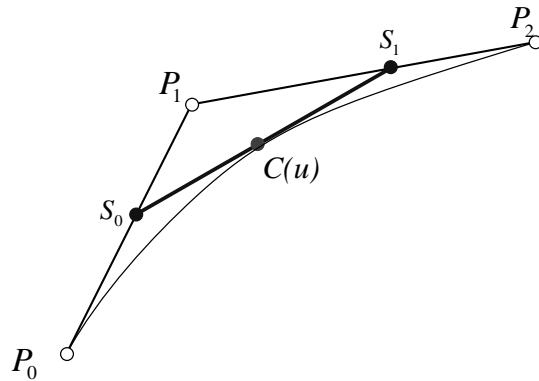
Khi đó,  $B_0$  và  $B_1$  là các hàm cơ sở

Thay lại vào phương trình trên:  $C(u) = B_0 P_0 + B_1 P_1 = \sum_{i=0}^1 B_i P_i$ . với  $u \in [0, 1]$

Như vậy, một điểm trên đường thẳng được xác định dựa vào các hàm cơ sở và các điểm tương ứng.

b) Đường cong Bézier bậc 2

Đường cong Bézier bậc 2 có  $n+1=3$  điểm điều khiển là  $P_0$ ,  $P_1$  và  $P_2$ . Xét đa giác gồm 3 đỉnh  $P_0$ ,  $P_1$  và  $P_2$ .



Hình 2. Đường cong tham số bậc hai

Xét đa giác gồm 3 đỉnh theo thứ tự là  $P_0$ ,  $P_1$  và  $P_2$ .  $S_0$ ,  $S_1$  lần lượt là các điểm bất kỳ trên hai đoạn thẳng  $P_0P_1$  và  $P_1P_2$ .

Với  $u \in [0, 1]$ , ta có

- Phương trình tham số của  $P_0P_1$  là:

$$S_0(u) = (1-u) P_0 + u P_1$$

- Phương trình tham số của  $P_1P_2$  là:

$$S_1(u) = (1-u) P_1 + u P_2$$

Xét cạnh  $S_0S_1$ , điểm  $C$  trên cạnh này được xác định theo tham số  $u$  cũng là điểm nằm trên đường cong bậc hai có ba đỉnh điều khiển  $P_0$ ,  $P_1$ ,  $P_2$ . Tọa độ điểm  $C$  được biểu diễn như sau:

$$C(u) = (1-u) S_0 + u S_1$$

Triển khai phương trình trên, ta có:

$$C(u) = (1-u)((1-u)P_0 + uP_1) + u((1-u)P_1 + uP_2)$$

$$C(u) = (1-u)(P_0 - P_0u + P_1u) + u(P_1 - P_1u + P_2u)$$

$$C(u) = (1-u)^2 P_0 + (2u(1-u))P_1 + u^2 P_2$$

Đặt

$$B_0 = (1-u)^2$$

$$B_1 = 2u(1-u)$$

$$B_2 = u^2$$

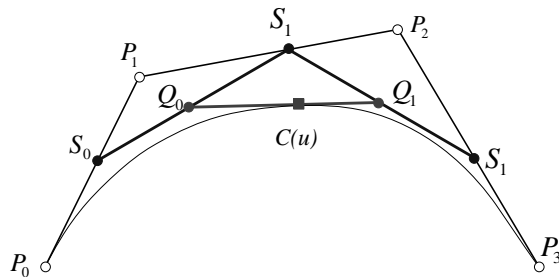
Khi đó, các hàm  $B_i$  là các hàm cơ sở

Thay lại vào phương trình trên, ta nhận được phương trình  $C(u)$  biểu diễn đường cong bậc hai (*quadratic*):

$$C(u) = \sum_0^2 B_i P_i \text{ với } u \in [0, 1]$$

### c) Đường cong Bézier khối

Đường cong Bézier tuyến tính là đường cong Bézier bậc 3 với  $n+1=4$  điểm điều khiển là  $P_0, P_1, P_2$  và  $P_3$ . Xét đa giác gồm 4 đỉnh  $P_0, P_1, P_2$  và  $P_3$



Hình 3. Đường cong tham số bậc ba

Xét đa giác gồm 4 đỉnh  $P_0, P_1, P_2$  và  $P_3$ .  $S_0, S_1, S_2$  lần lượt là các điểm bất kỳ trên hai đoạn thẳng  $P_0P_1, P_1P_2$  và  $P_2P_3$  cùng xác định theo tham số  $u$ .

Với  $u \in [0, 1]$ , ta có

-Phương trình tham số của  $P_0P_1$  là:

$$S_0(u) = (1-u)P_0 + uP_1$$

-Phương trình tham số của  $P_1P_2$  là:

$$S_1(u) = (1-u)P_1 + uP_2$$

-Phương trình tham số của  $P_2P_3$  là:

$$S_2(u) = (1-u)P_2 + uP_3$$

Xét cạnh  $S_0S_1$ , điểm  $C$  trên cạnh này được xác định theo tham số  $u$  cũng là điểm nằm trên đường cong bậc hai có ba đỉnh điều khiển  $P_0, P_1, P_2$ .

Gọi  $Q_0, Q_1$  lần lượt là các điểm trên cạnh  $S_0S_1$  và  $S_1S_2$  được biểu diễn theo tham số  $u$ . Các điểm  $Q_i$  này được biểu diễn như sau

$$Q_0(u) = (1-u)S_0 + uS_1$$

$$Q_1(u) = (1-u)S_1 + uS_2$$

Xét cạnh  $Q_0Q_1$ , điểm  $C$  trên cạnh này được xác định theo tham số  $u$  cũng là điểm nằm trên đường cong bậc ba có bốn đỉnh điều khiển  $P_0, P_1, P_2, P_3$

$$C(u) = (1-u)Q_0 + uQ_1$$

Triển khai phương trình trên, ta có:

$$C(u) = (1-u)[(1-u)^2P_0 + 2u(1-u)P_1 + u^2P_2] + u[(1-u)^2P_1 + 2u(1-u)P_2 + u^2P_3]$$

$$C(u) = (1-u)^3P_0 + 2u(1-u)^2P_1 + u^2(1-u)P_2 + u(1-u)^2P_1 + 2u^2(1-u)P_2 + u^3P_3$$

$$C(u) = (1-u)^3P_0 + 3u(1-u)^2P_1 + 3u^2(1-u)P_2 + u^3P_3$$

Đặt

$$B_0 = (1-u)^3$$

$$B_1 = 3u(1-u)^2$$

$$B_2 = 3u^2(1-u)$$

$$B_3 = u^3.$$

Khi đó, các hàm  $B_i$  là các hàm cơ sở

Thay lại vào phương trình trên, ta nhận được phương trình  $C(u)$  biểu diễn đường

cong bậc ba (quadratic):  $C(u) = \sum_{i=0}^3 B_i P_i$ . với  $u \in [0, 1]$

### 2.2.2 Phương trình đường cong Bézier tổng quát:

Một đường cong Bézier  $C(u)$  được xác định bằng tập hợp các điểm điều khiển  $P_0(x_0, y_0, z_0)$  đến  $P_n(x_n, y_n, z_n)$ , với  $n$  được gọi bậc của đường cong Bézier. Điểm kiểm soát



đầu và cuối là các điểm mút của đường cong, trong khi các điểm nằm giữa (nếu có) thường không nằm bên trên đường cong.

Phương trình biểu diễn đường cong Bézier bậc  $n$ :

$$C(u) = \sum_{i=0}^n P_i B_{i,n}(u) \quad \text{với } 0 \leq u \leq 1$$

Điểm  $P_i \in \mathbb{R}^3$  có tọa độ:  $P_i (x_i, y_i, z_i)$   $\forall x_i, y_i, z_i \in \mathbb{R}, \quad i=1,2,\dots,n$

Trong đó:

- $\{P_i\} (i = 0 \dots n)$  là các đỉnh điều khiển của đường cong;
- $B_{i,n}(u)$  là hàm trộn (*blending function*) hay còn gọi là đa thức Bernstein (*Bernstein polynomial*) có bậc  $n$ :

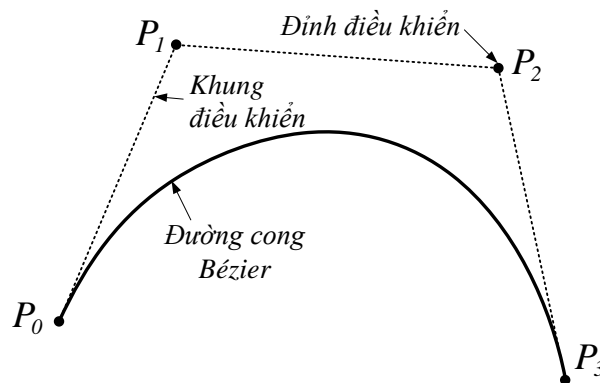
$$B_{i,n}(u) = \frac{n!}{i!(n-i)!} u^i (1-u)^{n-i} \quad i = 0 \dots n$$

Hàm trộn là một đa thức có bậc nhỏ hơn số lượng các điểm điều khiển

Đa giác điều khiển của đường cong Bézier là đa giác thu được khi nối các đỉnh điều khiển theo thứ tự lại với nhau.

Một điểm trên đường cong Bézier  $C(u)$  tại tham số  $u$  có tọa độ:

$$C(u) = \begin{pmatrix} C_x(u) = \sum_{i=0}^n x_i B_{i,n}(u) \\ C_y(u) = \sum_{i=0}^n y_i B_{i,n}(u) \\ C_z(u) = \sum_{i=0}^n z_i B_{i,n}(u) \end{pmatrix}$$



Hình 4. Đường cong Bézier bậc  $p = 3$  có 4 đỉnh điều khiển

Để biểu diễn tham số  $u$  trong đoạn  $[a, b]$  thay vì trong đoạn  $[0, 1]$ , ta đặt:

$$u = \frac{u - a}{b - a}.$$

Hàm trộn  $B_{i,n}(u)$  xác định đường cong Bézier trong miền tham số  $[a, b]$ :

$$B_{i,n}(u) = \frac{n!}{i!(n-i)!} \left( \frac{u-a}{b-a} \right)^i \left( 1 - \frac{u-a}{b-a} \right)^{n-i}$$

### 3. TỔ CHỨC CẤU TRÚC DỮ LIỆU VÀ THUẬT TOÁN

#### 3.1. Phát biểu bài toán

- Input: Tập  $n$  điểm  $A_i(x_i, y_i, z_i)$  mà ta cần xây dựng một đường cong Bézier nội suy đi qua chúng.
- Output: Tập 4 đỉnh điều khiển  $P_0(x_0, y_0, z_0)$  đến  $P_3(x_3, y_3, z_3)$  của một đường cong Bézier bậc 3 nội suy.

#### 3.2. Cấu trúc dữ liệu

Sử dụng phương pháp nội suy bình phương bé nhất.

Đối với các điểm nằm trên đường cong, ta dùng một biến kiểu `int` để lưu số điểm và một mảng hai chiều có kiểu dữ liệu là `double` để lưu tọa độ, trong đó có kích thước tối đa là 100 dòng (lưu được tối đa 100 điểm) và 3 cột (tương ứng với  $x, y, z$ ). Số dòng có thể mở rộng ra nếu cần thiết.

`int n;`

`double a[100][3];`

Đối với các đỉnh điều khiển, ta dùng mảng hai chiều có kiểu dữ liệu `double` với kích thước 4 dòng (tương ứng với 4 đỉnh) và 3 cột (tương ứng với  $x, y, z$ ) để lưu tọa độ. Số dòng có thể mở rộng ra nếu cần thiết.

`double P[4][3];`

Dùng thêm một số mảng hai chiều có kiểu dữ liệu `double` để tạo các ma trận dùng trong việc giải hệ phương trình hai ẩn số.

`double B[2][2], D[2][3];`

### 3.3. Thuật toán

#### 3.3.1 Ví dụ minh họa

##### Đề bài:

Cho 4 điểm  $A_0(10,0,0)$ ,  $A_1(20,30,0)$ ,  $A_2(50,40,0)$ ,  $A_3(80,0,0)$  là các điểm thuộc đường cong Bézier bậc 3. Tìm các điểm điều khiển  $P_0, P_1, P_2, P_3$  của đường cong này.

##### Cách giải:

- Phương trình đường cong bezier bậc 3 là:

$$C(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t) P_2 + t^3 P_3$$

$$P_0 = A_0 ; P_3 = A_3;$$

- Do:  $P_0P_1$  là tiếp tuyến của đường cong tại  $P_0 \rightarrow P_1 = \alpha_1 t_1 + P_0$

Với  $t_1$  là vector đơn vị của  $P_0P_1$

$P_2P_3$  là tiếp tuyến của đường cong tại  $P_3 \rightarrow P_2 = \alpha_2 t_2 + P_3$

Với  $t_2$  là vector đơn vị của  $P_3P_2$ ;

- Sử dụng phương pháp bình phương bé nhất ta có:

$S = (A_i - Q(t_i))^2$  nhỏ nhất. Với  $A_i$  là các điểm trên đường cong,  $Q(t_i)$  là các hình chiếu của  $A_i$  lên đường cong.

$S$  đạt giá trị nhỏ nhất khi đạo hàm của nó có giá trị là 0;

$$\frac{S'}{\alpha_1} = 0; \frac{S'}{\alpha_2} = 0$$

$$\rightarrow B_0 = \sum_{t=0}^1 (1-t)^3$$

$$\rightarrow B_1 = \sum_{t=0}^1 3(1-t)^2 t$$

$$\rightarrow B_2 = \sum_{t=0}^1 3(t)^2 (1-t)$$

$$\rightarrow B_3 = \sum_{t=0}^1 (t)^3$$

$$\begin{cases} \sum_{i=0}^{n-1} B_1 * B_1 * \alpha_1 t_1 + \sum_{i=0}^{n-1} B_1 * B_2 * \alpha_2 t_2 = \sum_{i=0}^{n-1} (A_i - (P_0 * B_0 + P_0 * B_1 + P_3 * B_2 + P_3 * B_3)) * B_1 \\ \sum_{i=0}^{n-1} B_1 * B_2 * \alpha_1 t_1 + \sum_{i=0}^{n-1} B_2 * B_2 * \alpha_2 t_2 = \sum_{i=0}^{n-1} (A_i - (P_0 * B_0 + P_0 * B_1 + P_3 * B_2 + P_3 * B_3)) * B_2 \end{cases}$$

Thay các giá trị của hệ rồi giải hệ, ta được các điểm điều khiển:

$$P_0 = (10,00 ; 0,00 ; 0,00 );$$

$$P_1 = (3,33 ; 30,00 ; 0,00 );$$

$$P_2 = (56,67 ; 75,00 ; 0,00 );$$

$P_3 = (80,00 ; 0,00 ; 0,00 )$ ;

### 3.3.2 Thuật toán

Input: Tập hợp các điểm trên đường cong Bezier lấy dạng file hoặc nhập từ bàn phím.

Output: 4 đỉnh điều khiển của đường cong Bézier

**B1**: Lấy dữ liệu là tập hợp  $n$  điểm  $A[0 \dots (n-1)]$  với  $4 \leq n$

- Gián tiếp (lấy dữ liệu từ file dữ liệu):

Ta có hàm để lấy dữ liệu từ file như sau:

```
void readfile(double a[100][3],int *n)
{
    FILE *f;
    char file[20];
    printf("Nhap ten file doc du lieu: ");
    fflush(stdin);
    gets(file);
    f=fopen(file,"r");

    fscanf (f,"%d ",n);
    for (int i=0;i<*n;i++)
        for (int j=0;j<3;j++)
            fscanf(f,"%lf",&a[i][j]);
    fclose(f);
}
```

- Trực tiếp (lấy dữ liệu từ bàn phím):

```
void input(double a[100][3], int *n)
{
    printf("Nhap so diem tren duong cong: ");
    scanf ("%d",n);

    printf("nhap toa do cac diem tren duong cong:\n");
    for (int i=0;i<*n;i++)
    {
        printf("Nhap toa do diem thu %d:\n",i);
        for (int j=0;j<3;j++)
            scanf("%lf",&a[i][j]);
    }
}
```

**B2**: Gán tọa độ các đỉnh điều khiển đầu và cuối  $P_0 \equiv A_0$ ,  $P_3 \equiv A_{n-1}$ :

```
for (int i=0;i<3;i++)
    P[0][i]=a[0][i];
for (int i=0;i<3;i++)
    P[3][i]=a[n-1][i];
```

**B3**: Chia đều tham số  $t$  trong khoảng  $0 \leq t \leq 1$

Chia khoảng  $[0, 1]$  thành  $(n-1)$  khoảng bằng nhau. Theo đó, với mỗi điểm  $A_i$ , ta sẽ xác định được một  $t_i$  tương ứng.

Ta có hàm chia đều tham số  $t$ :

```
double t(int n, int i)
{
    return (double)i/(n-1);
}
```

**B4:** Tính đỉnh  $P_1$  và  $P_2$ :

Đỉnh  $P_1$  và  $P_2$  được định vị một khoảng cách  $\alpha$  trên các vector tiếp tuyến, trái và phải tương ứng

$$P_1 = \alpha_1 t_1 + P_0, \quad P_2 = \alpha_2 t_2 + P_3$$

Từ phương pháp bình phương bé nhất, ta có hệ phương trình sau:

$$\begin{cases} \sum_{i=0}^{n-1} B_1 * B_1 * \alpha_1 t_1 + \sum_{i=0}^{n-1} B_1 * B_2 * \alpha_2 t_2 = \sum_{i=0}^{n-1} (A_i - (P_0 * B_0 + P_0 * B_1 + P_3 * B_2 + P_3 * B_3)) * B_1 \\ \sum_{i=0}^{n-1} B_1 * B_2 * \alpha_1 t_1 + \sum_{i=0}^{n-1} B_2 * B_2 * \alpha_2 t_2 = \sum_{i=0}^{n-1} (A_i - (P_0 * B_0 + P_0 * B_1 + P_3 * B_2 + P_3 * B_3)) * B_2 \end{cases}$$

→ Đưa về dạng ma trận như sau

$$\begin{pmatrix} \sum_{i=0}^{n-1} B_1 * B_1 & \sum_{i=0}^{n-1} B_1 * B_2 \\ \sum_{i=0}^{n-1} B_1 * B_2 & \sum_{i=0}^{n-1} B_2 * B_2 \end{pmatrix} * \begin{pmatrix} \alpha_1 t_1 \\ \alpha_2 t_2 \end{pmatrix} = \begin{pmatrix} \sum_{i=0}^{n-1} (A_i - (P_0 * B_0 + P_0 * B_1 + P_3 * B_2 + P_3 * B_3)) * B_1 \\ \sum_{i=0}^{n-1} (A_i - (P_0 * B_0 + P_0 * B_1 + P_3 * B_2 + P_3 * B_3)) * B_2 \end{pmatrix}$$

→ Tương đương với phương trình sau:

$$\begin{aligned} \begin{pmatrix} \alpha_1 t_1 \\ \alpha_2 t_2 \end{pmatrix} &= \begin{pmatrix} \sum_{i=0}^{n-1} B_1 * B_1 & \sum_{i=0}^{n-1} B_1 * B_2 \\ \sum_{i=0}^{n-1} B_1 * B_2 & \sum_{i=0}^{n-1} B_2 * B_2 \end{pmatrix}^{-1} * \begin{pmatrix} \sum_{i=0}^{n-1} (A_i - (P_0 * B_0 + P_0 * B_1 + P_3 * B_2 + P_3 * B_3)) * B_1 \\ \sum_{i=0}^{n-1} (A_i - (P_0 * B_0 + P_0 * B_1 + P_3 * B_2 + P_3 * B_3)) * B_2 \end{pmatrix} \\ &= \frac{1}{\sum_{i=0}^{n-1} (B_1 * B_1) * \sum_{i=0}^{n-1} (B_2 * B_2) - (\sum_{i=0}^{n-1} B_1 * B_2)^2} * \begin{pmatrix} \sum_{i=0}^{n-1} B_2 * B_2 & -\sum_{i=0}^{n-1} B_1 * B_2 \\ -\sum_{i=0}^{n-1} B_1 * B_2 & \sum_{i=0}^{n-1} B_1 * B_1 \end{pmatrix} \\ &\quad * \begin{pmatrix} \sum_{i=0}^{n-1} (A_i - (P_0 * B_0 + P_0 * B_1 + P_3 * B_2 + P_3 * B_3)) * B_1 \\ \sum_{i=0}^{n-1} (A_i - (P_0 * B_0 + P_0 * B_1 + P_3 * B_2 + P_3 * B_3)) * B_2 \end{pmatrix} \end{aligned}$$

Đặt:

$$A = \begin{pmatrix} \alpha_1 t_1 \\ \alpha_2 t_2 \end{pmatrix}$$

$$B = \frac{1}{\sum_{i=0}^{n-1} (B1 * B1) * \sum_{i=0}^{n-1} (B2 * B2) - (\sum_{i=0}^{n-1} B1 * B2)^2} * \begin{pmatrix} \sum_{i=0}^{n-1} B2 * B2 & -\sum_{i=0}^{n-1} B1 * B2 \\ -\sum_{i=0}^{n-1} B1 * B2 & \sum_{i=0}^{n-1} B1 * B1 \end{pmatrix}$$

$$D = \begin{pmatrix} \sum_{i=0}^{n-1} (Ai - (P0 * B0 + P0 * B1 + P3 * B2 + P3 * B3)) * B1 \\ \sum_{i=0}^{n-1} (Ai - (P0 * B0 + P0 * B1 + P3 * B2 + P3 * B3)) * B2 \end{pmatrix}$$

Vậy phương trình của chúng ta có dạng:

$$A=B*D$$

Ta có hàm tính ma trận B:

```
void MTB(int n)
{
    double sum1=tongB1B1(n);
    double sum2=tongB2B2(n);
    double sum3=tongB1B2(n);
    double sum=sum1*sum2-sum3*sum3;
    B[0][0]=sum2/sum;
    B[1][0]=-sum3/sum;
    B[0][1]=-sum3/sum;
    B[1][1]=sum1/sum;
}
```

Ta có hàm tính ma trận D:

```
void MTD(double a[100][3], int n)
{
    for (int k=0; k<3; k++)
    {
        double sum1=0;
        for (int i=0; i<n; i++)
            sum1+=(a[i][k]-(a[0][k]*B0(t(n, i))+a[0][k]*B1(t(n, i))+a[n-1][k]*B2(t(n, i))+a[n-1][k]*B3(t(n, i))))*B1(t(n, i));
        D[0][k]=sum1;

        double sum2=0;
        for (int i=0; i<n; i++)
            sum2+=(a[i][k]-(a[0][k]*B0(t(n, i))+a[0][k]*B1(t(n, i))+a[n-1][k]*B2(t(n, i))+a[n-1][k]*B3(t(n, i))))*B2(t(n, i));
        D[1][k]=sum2;
    }
}
```

Từ đó, ta tính được ma trận A, suy ra  $\alpha_1 t_1$  và  $\alpha_2 t_2$ .

Ta có đoạn lệnh tính ma trận A:

```
double alpha_t[2][3];
for (int i=0; i<2; i++)
    for (int j=0; j<3; j++)
        alpha_t[i][j]=B[i][0]*D[0][j]+B[i][1]*D[1][j];
```

Lại có:  $P_1 = \alpha_1 t_1 + P_0$ ,  $P_2 = \alpha_2 t_2 + P_3$  nên ta tính được  $P_1$  và  $P_2$ .

Ta có đoạn lệnh tính ma trận P là tọa độ các điểm điều khiển:

```
double P[4][3];
for (int i=0;i<3;i++)
    P[0][i]=a[0][i];
for (int i=0;i<3;i++)
    P[3][i]=a[n-1][i];
for (int i=0;i<3;i++)
    P[1][i]=alpha_t[0][i]+P[0][i];
for (int i=0;i<3;i++)
    P[2][i]=alpha_t[1][i]+P[3][i];
```

**B5:** Xuất kết quả.

**B6:** Lưu kết quả vào file dữ liệu.

Ta có hàm lưu dữ liệu vào file

```
void writefile(double P[4][3])
{
    FILE *f;
    char file[20];
    printf("Nhap ten file ghi du lieu: ");
    fflush(stdin);
    gets(file);
    f=fopen(file,"w");

    fprintf(f,"Cac diem dieu khien la:\n");
    for (int i=0;i<4;i++)
        fprintf(f,"P[%d]=(%10.21f,%10.21f,%10.21f)\n", i, P[i][0], P[i][1], P[i][2]);
    printf("File cua ban da duoc luu.\n");
    fclose(f);
}
```

### 3.3.3 Độ phức tạp của thuật toán

#### a) Các hàm chính

- void MTB ( double a[100][3] , int n )
- void MTD ( double a[100][3] , int n )
- void output( double a[100][3] , int n )

#### ❖ Hàm MTB

```
void MTB(int n)
{
    double sum1=tongB1B1(n);
    double sum2=tongB2B2(n);
    double sum3=tongB1B2(n);
    double sum=sum1*sum2-sum3*sum3;
    B[0][0]=sum2/sum;
    B[1][0]=-sum3/sum;
    B[0][1]=-sum3/sum;
    B[1][1]=sum1/sum;
}
```

Hàm này dùng để tính toán các phần tử của ma trận B. Độ phức tạp của các hàm  $tongB1B1$ ,  $tongB2B2$ ,  $tongB1B2$  đều là  $n$ .

Độ phức tạp  $O(n)$ .

❖ Hàm MTD

```
void MTD(double a[100][3], int n)
{
    for (int k=0; k<3; k++)
    {
        double sum1=0;
        for (int i=0; i<n; i++)
            sum1+=(a[i][k]-(a[0][k]*B0(t(n, i))+a[0][k]*B1(t(n, i))+a[n-1][k]*B2(t(n, i))+a[n-1][k]*B3(t(n, i))))*B1(t(n, i));
        D[0][k]=sum1;

        double sum2=0;
        for (int i=0; i<n; i++)
            sum2+=(a[i][k]-(a[0][k]*B0(t(n, i))+a[0][k]*B1(t(n, i))+a[n-1][k]*B2(t(n, i))+a[n-1][k]*B3(t(n, i))))*B2(t(n, i));
        D[1][k]=sum2;
    }
}
```

Hàm này được sử dụng để tìm ma trận D, trong đó có sử dụng 2 vòng lặp for từ 0 đến  $(n-1)$  nhưng không lồng nhau.

Độ Phức Tạp  $O(n)$ .

❖ Hàm output



```

void output(double a[100][3],int n)
{
    MTB(n);
    MTD(a,n);

    // ma tran cot cua an (alpha*t)
    double alpha_t[2][3];
    for (int i=0;i<2;i++)
        for (int j=0;j<3;j++)
            alpha_t[i][j]=B[i][0]*D[0][j]+B[i][1]*D[1][j];

    // ma tran cua cac diem dieu khien
    double P[4][3];
    for (int i=0;i<3;i++)
        P[0][i]=a[0][i];
    for (int i=0;i<3;i++)
        P[3][i]=a[n-1][i];
    for (int i=0;i<3;i++)
        P[1][i]=alpha_t[0][i]+P[0][i];
    for (int i=0;i<3;i++)
        P[2][i]=alpha_t[1][i]+P[3][i];

    // xuat cac diem dieu khien
    printf ("Cac diem dieu khien la : \n");
    for (int i=0;i<4;i++)
        printf("P[%d]=(%10.2lf,%10.2lf,%10.2lf)\n",i,P[i][0],P[i][1],P[i][2]);
    printf("\n");

    //xuat lua chon luu ket qua
    int tmp;
    printf("Ban co muon luu ket qua khong?\n");
    printf("1, Co\n");
    printf("2, Khong\n");
    printf("Nhap lua chon: ");
    scanf("%d",&tmp);
    printf("\n");
    if (tmp==1)
        writefile(P);
    printf("Nhiem vu hoan thanh.\n");
}

```

Đây là hàm quan trọng nhất của chương trình. Hàm này chứa hàm MTB và MTD dùng để xử lý và in ra kết quả của chương trình. Các vòng lặp for trong hàm có độ phức tạp tối đa là  $O(6)$ , nhưng độ phức tạp của hàm MTB và hàm MTD là  $O(n)$  nên độ phức tạp của hàm này phụ thuộc vào độ lớn của  $n$ .

Độ phức tạp:  $O(n) + O(6)$ .

b) Các hàm hỗ trợ để tính toán

- double t (int i)
- double B0 (double u); double B1 (double u); double B2 (double u);  
double B3 (double u);
- double tongB0(double u); double tongB1(double u); double tongB2(double u);  
double tongB3(double u);
- double tongB1B1(double a[][3],int n); double tongB1B2(double a[][3],int n);  
double tongB2B2(double a[][3] , int n);

❖ Hàm t

```
double t(int n, int i)
{
    return (double)i/(n-1);
}
```

Hàm này dùng để xác định  $t_i$  tương ứng với mỗi điểm  $A_i$  và chỉ thực hiện 1 phép tính.

Độ phức tạp  $O(1)$ .

❖ Các hàm B0; B1; B2; B3

Vì các hàm đều tương tự nhau nên ta chỉ cần xét hàm B0:

```
double B0(double u)
{
    return (1-u)*(1-u)*(1-u);
}
```

Hàm này dùng để tính  $B_{0,n}(t)$  và chỉ thực hiện 1 phép tính.

Độ phức tạp  $O(1)$ .

❖ Các hàm tongB0; tongB1; tongB2; tongB3;

Vì các hàm đều tương tự nhau nên ta chỉ cần xét hàm tongB0:

```
double tongB0(int n)
{
    double sum=0;
    for (int i=0;i<n;i++)
    {
        double u=t(n,i);
        sum+=B0(u);
    }
    return sum;
}
```

Hàm này dùng để tính  $\sum_{t=0}^1 (B_{0,n}(t))=B_0$ , có sử dụng một vòng lặp từ 0 đến (n-1).

Độ phức tạp  $O(n)$ .

❖ Các hàm tongB1B1; tongB1B2; tongB2B2;

Vì các hàm đều tương tự nhau nên ta chỉ cần xét hàm tongB1B1:

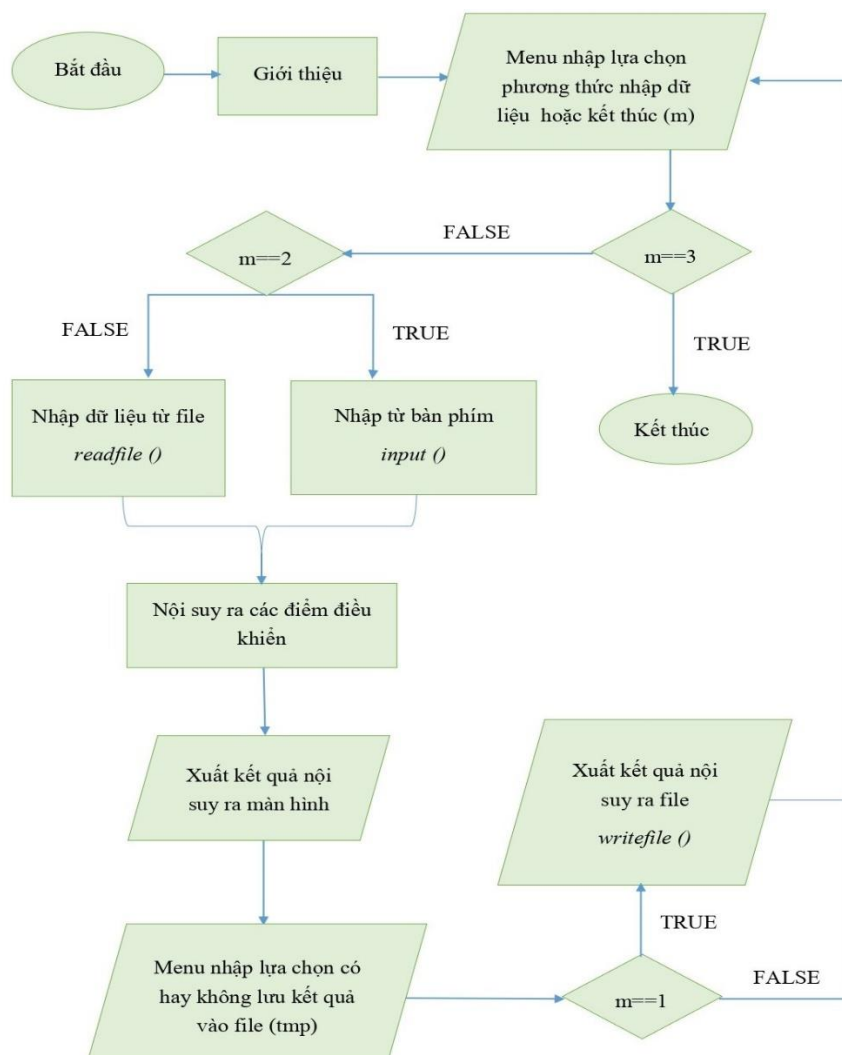
```
double tongB1B1(int n)
{
    double sum=0;
    for (int i=0;i<n;i++)
    {
        double u=t(n,i);
        sum+=B1(u)*B1(u);
    }
    return sum;
}
```

Hàm này dùng để tính  $\sum_{i=0}^{n-1} (B_1 * B_1)$ , có sử dụng một vòng lặp từ 0 đến (n-1).

Độ phức tạp  $O(n)$ .

## 4. CHƯƠNG TRÌNH VÀ KẾT QUẢ

### 4.1. Tổ chức chương trình



Hình 5. Lưu đồ hoạt động của chương trình.

## **4.2. Ngôn ngữ cài đặt**

Sử dụng ngôn ngữ C.

## **4.3. Kết quả thực hiện**

### **4.3.1. Giao diện chính của chương trình**

```
NOI SUY DUONG CONG BEZIER
Nguoi huong dan: PGS. TS. NGUYEN TAN KHOI
Sinh vien thuc hanh:
VO VAN DUC - 21T_DT2
NGUYEN THI THU THAO - 21T_DT2

-----MENU-----
1: Lay du lieu tu file
2: Nhap du lieu tu ban phim
3: Thoat

Nhap lua chon cua ban: _
```

*Hình 6.1. Giao diện chính của chương trình.*

Mở đầu, chương trình sẽ đưa ra 3 lựa chọn và yêu cầu chúng ta nhập vào. Để nhập dữ liệu vào, ta có 2 lựa chọn là 1 và 2.

- Nếu nhập lựa chọn 1 (Lấy dữ liệu từ file):

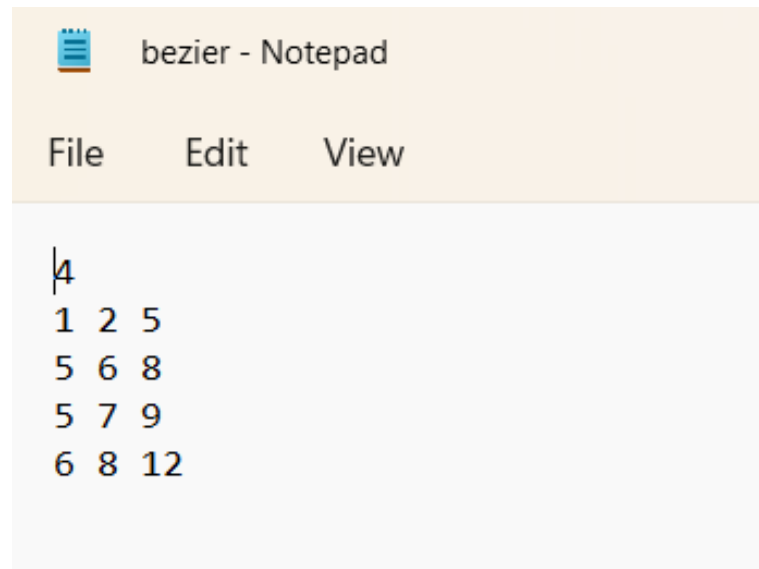
```
NOI SUY DUONG CONG BEZIER
Nguoi huong dan: PGS. TS. NGUYEN TAN KHOI
Sinh vien thuc hanh:
VO VAN DUC - 21T_DT2
NGUYEN THI THU THAO - 21T_DT2

-----MENU-----
1: Lay du lieu tu file
2: Nhap du lieu tu ban phim
3: Thoat

Nhap lua chon cua ban: 1

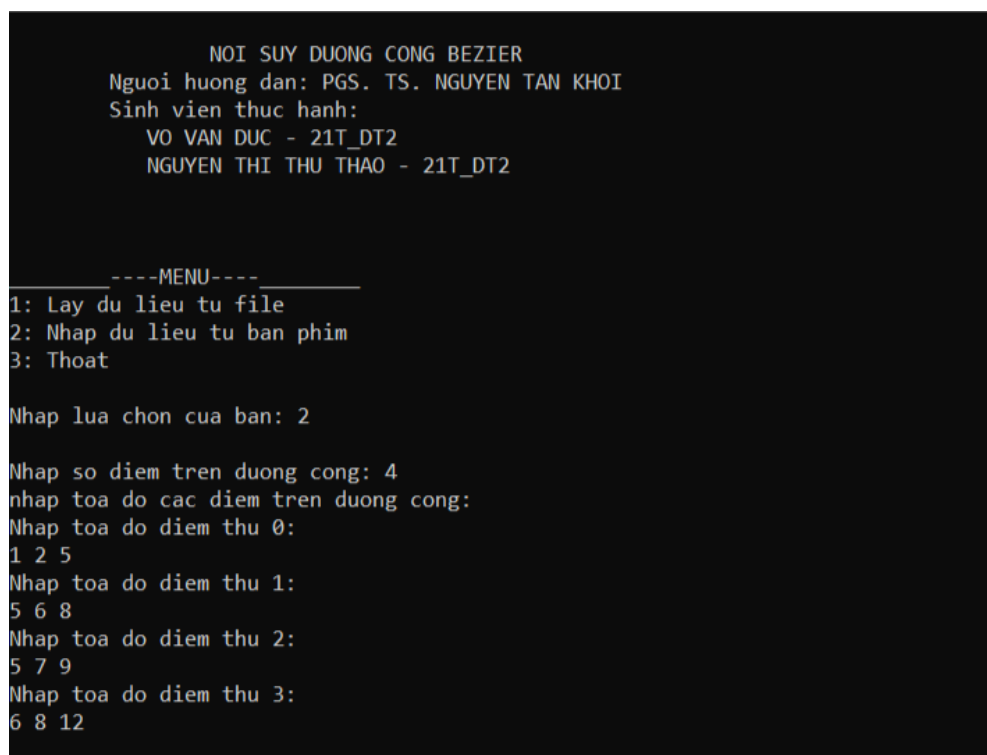
Nhap ten file doc du lieu: bezier.txt_
```

*Hình 6.2. Giao diện khi lấy dữ liệu từ file.*



Hình 6.3. File để lấy dữ liệu.

- Nếu nhập lựa chọn 2 (Nhập dữ liệu từ bàn phím)



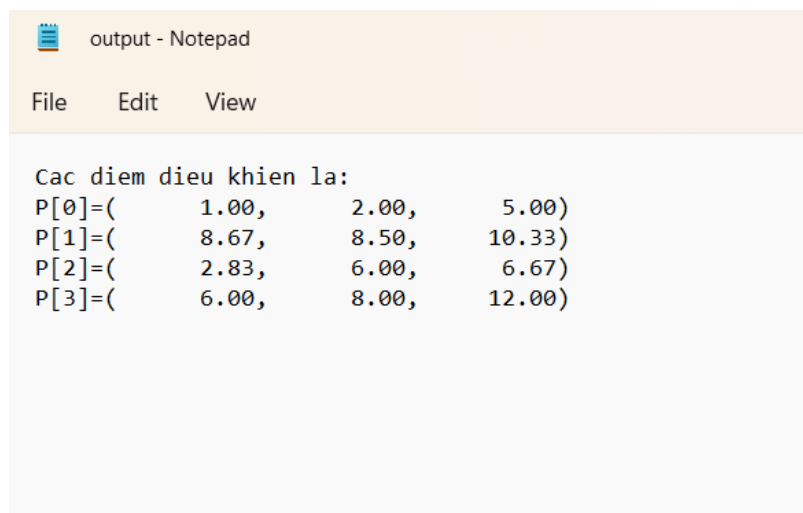
Hình 6.4. Giao diện khi nhập dữ liệu từ bàn phím

Sau khi xuất kết quả, chương trình đưa ra 2 lựa chọn về việc có hay không lưu kết quả và yêu cầu chúng ta nhập vào.

- Nếu nhập lựa chọn 1 (Có lưu kết quả)

```
Ban co muon luu ket qua khong?  
1, Co  
2, Khong  
Nhap lua chon: 1  
  
Nhap ten file ghi du lieu: output.txt  
File cua ban da duoc luu.  
Nhiem vu hoan thanh.
```

Hình 6.5. Giao diện có lưu kết quả vào file.



output - Notepad

File Edit View

Cac diem dieu khien la:

P[0]=(	1.00,	2.00,	5.00)
P[1]=(	8.67,	8.50,	10.33)
P[2]=(	2.83,	6.00,	6.67)
P[3]=(	6.00,	8.00,	12.00)

Hình 6.6. File dữ liệu mà kết quả lưu vào.

- Nếu nhập lựa chọn 2 (Không lưu kết quả)

```
Ban co muon luu ket qua khong?  
1, Co  
2, Khong  
Nhap lua chon: 2  
  
Nhiem vu hoan thanh.
```

Hình 6.7. Giao diện khi lấy dữ liệu từ file.

#### 4.3.2. Các kết quả thực thi của chương trình

##### a. Dữ liệu 1:

Cho 4 điểm  $A_0(10,0,0)$ ,  $A_1(20,30,0)$ ,  $A_2(50,40,0)$ ,  $A_3(80,0,0)$  là các điểm thuộc đường cong Bézier bậc 3. Tìm các điểm điều khiển  $P_0, P_1, P_2, P_3$  của đường cong này.

Kết quả:

```
-----MENU-----
1: Lay du lieu tu file
2: Nhap du lieu tu ban phim
3: Thoat

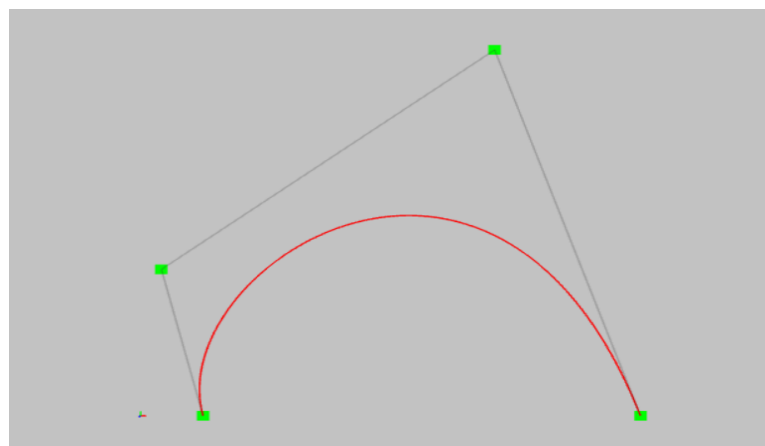
Nhap lua chon cua ban: 1

Nhap ten file doc du lieu: bezier1.txt
Cac diem tren duong cong la:
10.00  0.00  0.00
20.00  30.00  0.00
50.00  40.00  0.00
80.00  0.00  0.00
Cac diem dieu khien la :
P[0]=( 10.00, 0.00, 0.00)
P[1]=( 3.33, 30.00, 0.00)
P[2]=( 56.67, 75.00, 0.00)
P[3]=( 80.00, 0.00, 0.00)

Ban co muon luu ket qua khong?
1, Co
2, Khong
Nhap lua chon: 2

Nhiem vu hoan thanh.
```

Hình 7.1. Kết quả của dữ liệu 1.



Hình 7.2. Hình ảnh đường cong 1 sau khi tái tạo.

##### b. Dữ liệu 2:

Cho 4 điểm  $A_0(2,5,7)$ ,  $A_1(6,8,10)$ ,  $A_2(11,10,16)$ ,  $A_3(15,17,20)$  là các điểm thuộc đường cong Bézier bậc 3. Tìm các điểm điều khiển  $P_0$ ,  $P_1$ ,  $P_2$ ,  $P_3$  của đường cong này.

Kết quả:

```
-----MENU-----
1: Lay du lieu tu file
2: Nhap du lieu tu ban phim
3: Thoat

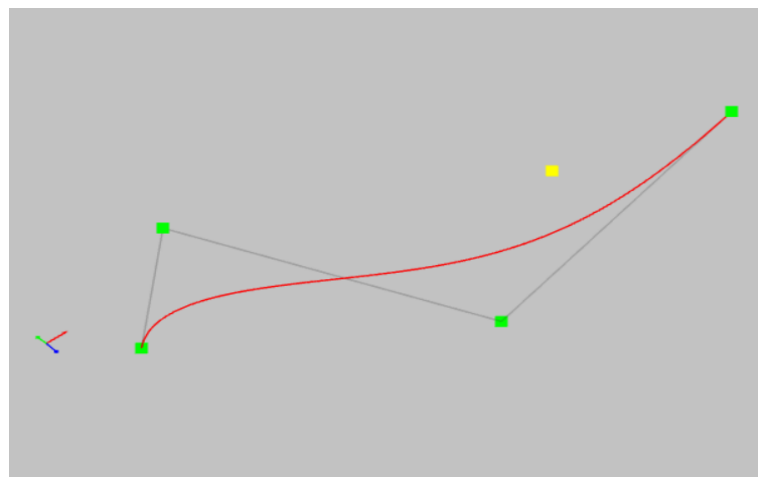
Nhap lua chon cua ban: 1

Nhap ten file doc du lieu: bezier2.txt
Cac diem tren duong cong la:
2.00    5.00    7.00
6.00    8.00    10.00
11.00   10.00   16.00
15.00   17.00   20.00
Cac diem dieu khien la :
P[0]=(    2.00,    5.00,    7.00)
P[1]=(    4.83,   10.50,    6.83)
P[2]=(   12.17,    5.50,   18.67)
P[3]=(   15.00,   17.00,   20.00)

Ban co muon lưu ket qua khong?
1, Co
2, Khong
Nhap lua chon: 2

Nhiem vu hoan thanh.
```

Hình 8.1. Kết quả của dữ liệu 2.



Hình 8.2. Hình ảnh đường cong 2 sau khi tái tạo.

c. Dữ liệu 3:

Cho 6 điểm  $A_0(11,15,13)$ ,  $A_1(15,19,22)$ ,  $A_2(16,20,25)$ ,  $A_3(21,17,25)$ ,  $A_4(25,15,19)$ ,  $A_5(30,12,12)$  là các điểm thuộc đường cong Bézier bậc 3. Tìm các điểm điều khiển  $P_0$ ,  $P_1$ ,  $P_2$ ,  $P_3$  của đường cong này.



Kết quả:

```
-----MENU-----
1: Lay du lieu tu file
2: Nhap du lieu tu ban phim
3: Thoat

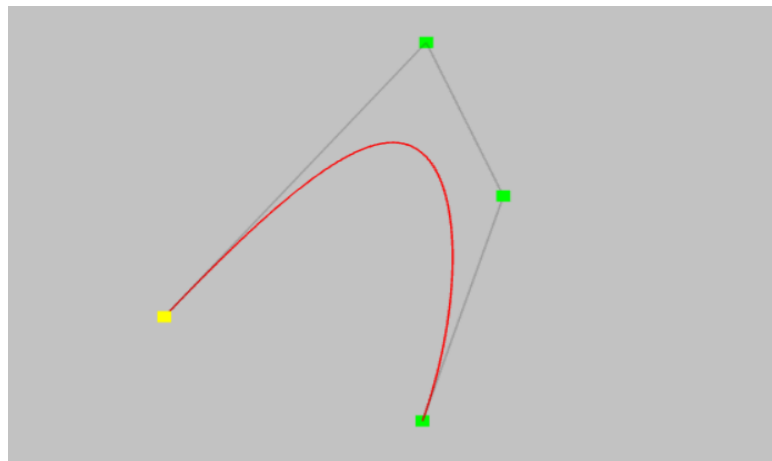
Nhap lua chon cua ban: 1

Nhap ten file doc du lieu: bezier3.txt
Cac diem tren duong cong la:
11.00  15.00  13.00
15.00  19.00  22.00
16.00  20.00  25.00
21.00  17.00  25.00
25.00  15.00  19.00
30.00  12.00  12.00
Cac diem dieu kien la :
P[0]=( 11.00, 15.00, 13.00)
P[1]=( 16.20, 25.77, 33.01)
P[2]=( 20.31, 15.33, 26.29)
P[3]=( 30.00, 12.00, 12.00)

Ban co muon luu ket qua khong?
1, Co
2, Khong
Nhap lua chon: 2

Nhiem vu hoan thanh.
```

Hình 9.1. Kết quả của dữ liệu 3.



Hình 9.2. Hình ảnh đường cong 3 sau khi tái tạo.

#### 4.3.3. Nhận xét đánh giá kết quả

- Chương trình cho ra kết quả xấp xỉ chính xác.
- Số điểm thuộc đường cong càng nhiều thì sai số của chương trình càng lớn.

## **5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

### **5.1. Kết luận**

- Sau một thời gian thực hiện đồ án cơ sở PBL1, nhóm chúng em đã rèn luyện được nhiều kỹ năng cho bản thân như nghiên cứu tài liệu, tư duy logic, làm việc nhóm,...
- Nhóm em đã thành công trong việc hoàn thành một chương trình nội suy đường cong Bézier bằng phương pháp bình phương bé nhất.
- Thông qua đề tài, nhóm em đã nắm vững hơn kiến thức về môn “Phương pháp tính” và “Cấu trúc dữ liệu”, đồng thời biết thêm nhiều kỹ thuật trong C.

### **5.2. Hướng phát triển**

- Tối ưu thuật toán, từ đó giảm thiểu thời gian xử lý.
- Mở rộng chương trình để ứng dụng được cho việc tìm nhiều đỉnh điều khiển hơn.
- Có thể phát triển thêm nhiều tính năng cũng như mở rộng ra nhiều nền tảng hơn

## **TÀI LIỆU THAM KHẢO**

- [1] XI.8 An algorithm for automatically fitting digitized curves
- [2] Đỗ Thị Tuyết Hoa, Bài giảng môn Phương pháp tính, 2007
- [3][https://en.wikipedia.org/wiki/B%C3%A9zier\\_curve#Cubic\\_B%C3%A9zier\\_curves](https://en.wikipedia.org/wiki/B%C3%A9zier_curve#Cubic_B%C3%A9zier_curves)
- [4] <http://nurbscalculator.in/>, 21/11/2019

## PHỤ LỤC

```
# include<stdio.h>

// B la nghich dao cua ma tran he so cua he phuong trinh hai an (alpha1*t1) va
(alpha2*t2)

// D la ma tran cot cua he so tu do cua he phuong trinh hai an (alpha1*t1) va
(alpha2*t2)

double B[2][2],D[2][3];

// cac ham ho tro tinh toan

// t tuong ung voi diem a[i] nam tren duong cong
double t(int n, int i);

// tinh B[i,n](u)_ da thuc Bernstein (u dai dien cho t ung voi moi diem)
double B0(double u);
double B1(double u);
double B2(double u);
double B3(double u);

// tinh tong cac B[i,n] cua tat ca cac u
double tongB0(int n);
double tongB1(int n);
double tongB2(int n);
double tongB3(int n);

double tongB1B1(int n);
double tongB1B2(int n);
double tongB2B2(int n);

// tinh ma tran B va ma tran D
void MTB(int n);
void MTD(double a[100][3],int n);

//ham tinh toan va xuat ket qua
void output(double a[100][3],int n);

//cac ham ho tro nhap xuat

//ham nhap du lieu tu ban phim
void input(double a[100][3], int *n);

//ham doc du lieu tu file
```

```
void readfile(double a[100][3],int *n);

//ham luu du lieu vao file
void writefile(double P[4][3]);

int main()
{
    // m la lua chon
    // n la so cac diem nam tren duong cong
    // a la toa do cac diem nam tren duong cong
    int m,n;
    double a[100][3];

    printf("\n\t\tNOI SUY DUONG CONG BEZIER\n");
    printf("\tNguoi huong dan: PGS. TS. NGUYEN TAN KHOI\n");
    printf("\tSinh vien thuc hanh:\n");
    printf("\t    VO VAN DUC - 21T_DT2\n");
    printf("\t    NGUYEN THI THU THAO - 21T_DT2\n");
    printf("\n\n");
    do
    {
        printf ("\n_____----MENU----_____\n");
        printf("1: Lay du lieu tu file\n");
        printf("2: Nhap du lieu tu ban phim\n");
        printf("3: Thoat\n\n");
        printf("Nhap lua chon cua ban: ");
        scanf ("%d", &m);
        printf("\n");

        //nhap du lieu
        switch (m)
        {
            // lay du lieu tu file txt
            case 1:
                readfile(a, &n);

                printf("Cac diem tren duong cong la:\n");
                for (int i=0;i<n;i++)
                {
                    for (int j=0;j<3;j++)
                        printf (".2lf\t",a[i][j]);
                    printf ("\n");
                }
            }
        }
    }
```

```
        break;

        // nhap du lieu truc tiep.....

        case 2:
            input(a,&n);

            printf("Cac diem tren duong cong la:\n");
            for (int i=0;i<n;i++)
            {
                for (int j=0;j<3;j++)
                    printf ( "%.2lf\t",a[i][j]);
                printf ("\n");
            }
            break;

            case 3: return 0;
        }

        //xuat ket qua
        output(a,n);

    } while (1);

    return 0;
}

//.....
double t(int n, int i)
{
    return (double)i/(n-1);
}

//.....
double B0(double u)
{
    return (1-u)*(1-u)*(1-u);
}

double B1(double u)
{
    return 3*(1-u)*(1-u)*u;
}
```

```
double B2(double u)
{
    return 3*(1-u)*u*u;
}

double B3(double u)
{
    return u*u*u;
}

//.....
double tongB0(int n)
{
    double sum=0;
    for (int i=0;i<n;i++)
    {
        double u=t(n,i);
        sum+=B0(u);
    }
    return sum;
}

double tongB1(int n)
{
    double sum=0;
    for (int i=0;i<n;i++)
    {
        double u=t(n,i);
        sum+=B1(u);
    }
    return sum;
}

double tongB2(int n)
{
    double sum=0;
    for (int i=0;i<n;i++)
    {
        double u=t(n,i);
        sum+=B2(u);
    }
    return sum;
}
```

```
double tongB3(int n)
{
    double sum=0;
    for (int i=0;i<n;i++)
    {
        double u=t(n,i);
        sum+=B3(u);
    }
    return sum;
}

//.....
double tongB1B1(int n)
{
    double sum=0;
    for (int i=0;i<n;i++)
    {
        double u=t(n,i);
        sum+=B1(u)*B1(u);
    }
    return sum;
}

double tongB2B2(int n)
{
    double sum=0;
    for (int i=0;i<n;i++)
    {
        double u=t(n,i);
        sum+=B2(u)*B2(u);
    }
    return sum;
}

double tongB1B2(int n)
{
    double sum=0;
    for (int i=0;i<n;i++)
    {
        double u=t(n,i);
        sum+=B1(u)*B2(u);
    }
}
```



```
        return sum;
    }

    //.....

void MTB(int n)
{
    double sum1=tongB1B1(n);
    double sum2=tongB2B2(n);
    double sum3=tongB1B2(n);
    double sum=sum1*sum2-sum3*sum3;
    B[0][0]=sum2/sum;
    B[1][0]=-sum3/sum;
    B[0][1]=-sum3/sum;
    B[1][1]=sum1/sum;
}

//.....

void MTD(double a[100][3], int n)
{
    for (int k=0; k<3; k++)
    {
        double sum1=0;
        for (int i=0; i<n; i++)
            sum1+=(a[i][k]-(a[0][k]*B0(t(n, i))+a[0][k]*B1(t(n, i))+a[n-1][k]*B2(t(n, i))+a[n-1][k]*B3(t(n, i))))*B1(t(n, i));
        D[0][k]=sum1;

        double sum2=0;
        for (int i=0; i<n; i++)
            sum2+=(a[i][k]-(a[0][k]*B0(t(n, i))+a[0][k]*B1(t(n, i))+a[n-1][k]*B2(t(n, i))+a[n-1][k]*B3(t(n, i))))*B2(t(n, i));
        D[1][k]=sum2;
    }
}

//.....

void readfile(double a[100][3],int *n)
{
    FILE *f;
    char file[20];
    printf("Nhap ten file doc du lieu: ");
    fflush(stdin);
    gets(file);
```

```
f=fopen(file,"r");

fscanf (f,"%d ",n);
for (int i=0;i<*n;i++)
    for (int j=0;j<3;j++)
        fscanf(f,"%lf ",&a[i][j]);
fclose(f);
}

void writefile(double P[4][3])
{
    FILE *f;
    char file[20];
    printf("Nhap ten file ghi du lieu: ");
    fflush(stdin);
    gets(file);
    f=fopen(file,"w");

    fprintf(f,"Cac diem dieu khien la:\n");
    for (int i=0;i<4;i++)
        fprintf(f,"P[%d]=(%10.2lf,%10.2lf,%10.2lf)\n", i, P[i][0], P[i][1],
P[i][2]);
    printf("File cua ban da duoc luu.\n");
    fclose(f);
}

//.....
void input(double a[100][3], int *n)
{
    printf("Nhap so diem tren duong cong: ");
    scanf ("%d",n);

    printf("nhap toa do cac diem tren duong cong:\n");
    for (int i=0;i<*n;i++)
    {
        printf("Nhap toa do diem thu %d:\n",i);
        for (int j=0;j<3;j++)
            scanf("%lf",&a[i][j]);
    }
}

//.....
void output(double a[100][3],int n)
```

```
{
    MTB(n);
    MTD(a,n);

    // ma tran cot cua an (alpha*t)
    double alpha_t[2][3];
    for (int i=0;i<2;i++)
        for (int j=0;j<3;j++)
            alpha_t[i][j]=B[i][0]*D[0][j]+B[i][1]*D[1][j];

    // ma tran cua cac diem dieu khien
    double P[4][3];
    for (int i=0;i<3;i++)
        P[0][i]=a[0][i];
    for (int i=0;i<3;i++)
        P[3][i]=a[n-1][i];
    for (int i=0;i<3;i++)
        P[1][i]=alpha_t[0][i]+P[0][i];
    for (int i=0;i<3;i++)
        P[2][i]=alpha_t[1][i]+P[3][i];

    // xuat cac diem dieu khien
    printf ("Cac diem dieu khien la : \n");
    for (int i=0;i<4;i++)
        printf("P[%d]=(%10.2lf,%10.2lf,%10.2lf)\n",i,P[i][0],P[i][1],P[i][2]);
    printf("\n");

    //xuat lua chon luu ket qua
    int tmp;
    printf("Ban co muon luu ket qua khong?\n");
    printf("1, Co\n");
    printf("2, Khong\n");
    printf("Nhap lua chon: ");
    scanf("%d",&tmp);
    printf("\n");
    if (tmp==1)
        writefile(P);
    printf("Nhiem vu hoan thanh.\n");
}
```