TRƯỜNG ĐẠI HỌC NHA TRANG
KHOA CÔNG NGHỆ THÔNG TIN

CÀI ĐẶT THUẬT TOÁN XÉN TỈA ĐOẠN THẲNG BẰNG THUẬT TOÁN COHEN SUTHERLAND VÀ LIANG BARSKY

GVHD : **Ths. ĐOÀN VŨ THỊNH**

MSSV : **63132195**

Lớp : **63. CNTT - 4**

Sinh viên thực hiện: **Cao Nguyễn Quốc Lâm**

Khánh Hòa, tháng 1 năm 2024

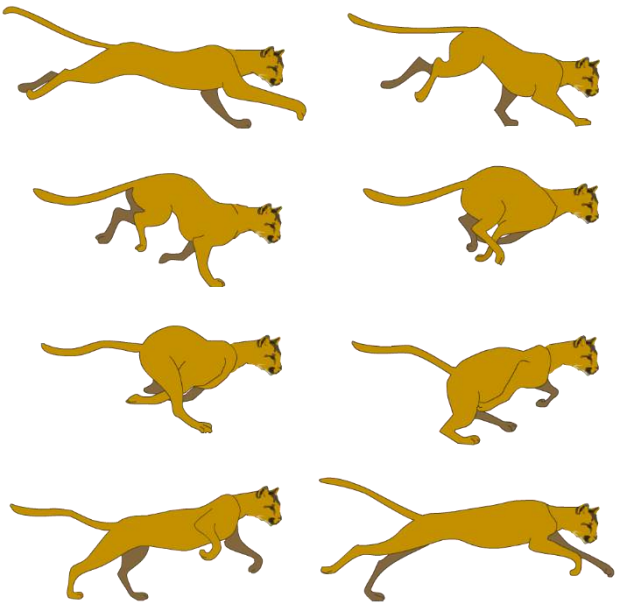
DANH MỤC HÌNH ẢNH

TÓM TẮT

Đồ họa máy tính, là một lĩnh vực độc đáo trong Công nghệ Thông tin, tập trung vào nghiên cứu, phát triển, và kết hợp đa dạng công cụ, từ mô hình lý thuyết đến phần mềm, nhằm tạo ra, lưu trữ, và xử lý các mô hình cũng như hình ảnh của các đối tượng, sự vật, và hiện tượng đa dạng trong cuộc sống, sản xuất, và nghiên cứu. Trong thế giới này, thuật toán xén tỉa đoạn thẳng - với các giải thuật như Cohen-Sutherland và Liang-Barsky - trở thành một phần quan trọng. Những thuật toán này không chỉ áp dụng rộng rãi trong các phần mềm thương mại mà còn trên các thiết bị xử lý đồ hoạ. Trong chuyến hành trình thực tập này, việc triển khai thuật toán xén tỉa đoạn thẳng đã giúp tôi hiểu rõ hơn về một trong những cơ sở của đồ họa máy tính. Quá trình này bao gồm từ thiết kế giao diện, triển khai thuật toán, đến việc hiển thị kết quả trực tiếp trên màn hình thông qua ngôn ngữ lập trình C++ và sử dụng ứng dụng DevC/C++ kết hợp với thư viện graphics.h. Kết quả cuối cùng không chỉ đáp ứng mọi yêu cầu đặt ra trong quá trình thực tập mà còn tạo ra một sản phẩm với giao diện đồ họa mê hoặc, thân thiện, và khả năng nhập dữ liệu linh hoạt từ chuột, đều là những ưu điểm nổi bật của sản phẩm. Điều này chứng minh khả năng ứng dụng thực tế và sự thành công của quá trình nghiên cứu và phát triển trong lĩnh vực đồ họa máy tính.

Chương 1. GIỚI THIỆU

Đồ họa máy tính là một lĩnh vực của Công nghệ thông tin, ở đó nghiên cứu, xây dựng và tập hợp các công cụ (mô hình lý thuyết và phần mềm) khác nhau để kiến tạo, xây dựng, lưu trữ và xử lý các mô hình và hình ảnh của các đối tượng, sự vật, hiện tượng trong cuộc sống, sản xuất, nghiên cứu. Đồ họa máy tính góp phần quan trọng làm cho giao tiếp giữa con người và máy tính trở nên thân thiện hơn. Từ đồ họa trên máy tính chúng ta có nhiều lĩnh vực có ứng dụng rất quan trọng của đồ họa máy tính trong thực tế như: tạo mô hình, hoạt cảnh, hỗ trợ thiết kế đồ họa, mô phỏng hình ảnh, chuẩn đoán hình ảnh (trong Y tế), huấn luyện đào tạo ảnh (quân sự, hàng không, …). Hình 1.1. là một ví dụ cụ thể của ứng dụng kỹ thuật đồ hoạ trong ngành điện ảnh. Trong khuôn khổ đợt thực tập cơ ppy, nhóm thực hiện đề tài "Cài đặt thuật toán xén đoạn thẳng bằng Cohen-Sutherland và Liang-Barsky".



Hình 1.1. Ứng dụng kỹ thuật đồ họa trong kỹ thuật hoạt hình

(Nguồn: https://gamedev.stackexchange.com/questions/79598/how-can-i-correctly-line-up-differently-sized-animation-frames-with-a-character)

Trước khi xén tỉa Sau khi xén tỉa

Hình 1.2. Ví dụ xén tỉa đoạn thẳng trong phần mềm PowerPoint 2016

Thuật toán xén tỉa trong đồ họa máy tính có tầm quan trọng rất lớn và được sử dụng rộng rãi trong các phần mềm phổ biến hiện này. Từ những phần mềm đơn giản 3 như Paint, Powerpoint trong bộ Office của Window đến những ứng dụng thiết kế đồ họa chuyên nghiệp như Photoshop, AutoCad. Hình 1.2 là ví dụ về kỹ thuật xén tỉa đa giác trong phần mềm MS Powerpoint 2013.

1.1 Thuật toán xén tỉa

Thuật toán xén tỉa, hoặc đơn giản là xén tỉa, là quá trình xác định và giữ lại các phần của hình ảnh bên trong một không gian xác định, trong khi loại bỏ phần bên ngoài. Trong ngữ cảnh này, "cửa sổ xén tỉa" là vùng được xác định để giữ lại, và cửa sổ này có thể mang nhiều hình dạng như hình chữ nhật, hình tròn, cửa sổ lồi hoặc lõm, tùy thuộc vào chiến lược xén tỉa cụ thể.

Chẳng hạn, nếu ta chọn cửa sổ xén tỉa là hình chữ nhật và đối tượng xén tỉa là đoạn thẳng, thuật toán xén tỉa sẽ giữ lại phần hình ảnh nằm bên trong hình chữ nhật này. Cùng một nguyên tắc, những đối tượng khác nhau như đường thẳng, đa giác, hình tròn, ký tự hoặc đường cong không đều cũng có thể được xén tỉa để giữ lại chỉ phần nằm trong vùng xén. Điều này không chỉ tạo ra sự tập trung vào những đối tượng quan trọng mà còn cho phép quản lý và hiển thị chỉ những phần cần thiết của hình ảnh, đồng thời giảm bớt tài nguyên tính toán và lưu trữ không cần thiết.

1.2 Thuật toán xén tỉa đoạn thẳng

Thuật toán xén tỉa đoạn thẳng đầu tiên ra đời vào năm 1967 được phát minh bởi Danny Cohen and Ivan Sutherland. Thuật toán xén tỉa đoạn thẳng là thuật toán xác định các điểm của đoạn thẳng nằm trong hay nằm ngoài của cửa sổ xén.

Trước khi xén tỉa Sau khi xén tỉa

Hình 1.3. Ví dụ về xén tỉa đoạn thẳng

Ý tưởng: Các đoạn thẳng có thể rơi vào các trường hợp sau:

Hiển thị (visible): cả hai đầu cuối của đoạn thẳng đều nằm bên trong cửa sổ Không hiển thị (invisible): đoạn thẳng xác định nằm ngoài cửa sổ. Điều này xảy ra khi đoạn thẳng từ (x1, y1) đến (x2, y2) thoả mãn bất kỳ một trong bốn bất đẳng thức sau:

x1, x2>xmax y1, y2> ymax

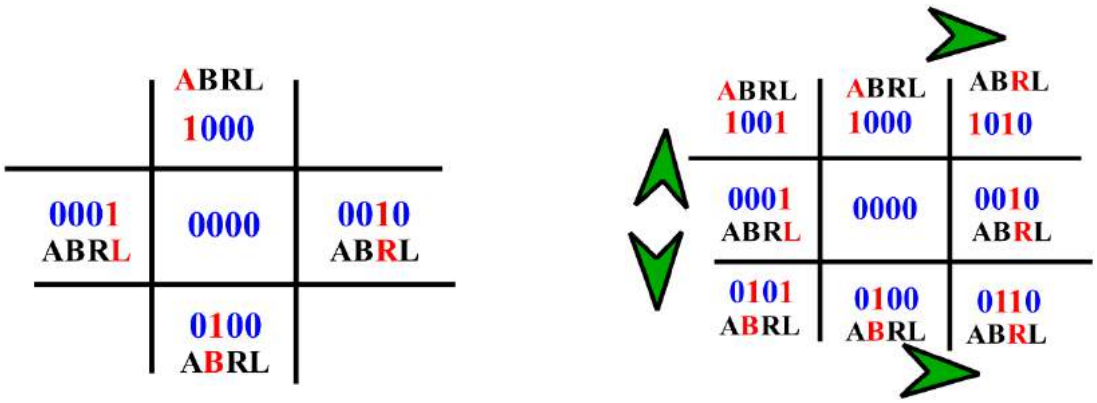
x1, x2< xmin y1, y2< ymin

Xén tỉa: đoạn thẳng cần xén tỉa

Thuật toán được chia thành các bước sau:

Bước 1: Gán mã vùng 4-bit cho mỗi điểm cuối của đoạn thẳng ABRL (Above – Below – Right – Left).

Bước 2: Mã vùng được xác định theo 9 vùng (hình 1.4) của mặt phẳng mà các điểm cuối nằm vào đó. Một bít được cài đặt true (1) hoặc false (0).



Hình 1.4. Mã vùng của 9 mặt phẳng vùng

Bước 3: Sử dụng các mã vùng để xác định các trường hợp của đoạn thẳng. Xét mã vùng của 2 điểm đầu cuối P1, P2 của đoạn thẳng cần xén. Ta có các trường hợp sau:

Nếu mã của P1 hoặc P2 đều = 0000 thì toàn bộ đoạn thẳng thuộc phần hiển thị.

Nếu mã của P1 và P2 có cùng một vị trí mà P1 AND P2 != 0000 => cùng phía.

Nếu không nằm trong 2 trường hợp sau đường thẳng cần được xén tỉa

Tìm giao điểm của đường thẳng với cửa sổ, (với phần mở rộng của đường biên).

Nếu: Bit 1 là 1: xén y = ymax

Bit 2 là 1: xén y = ymin

Bit 3 là 1: xén x = xmax

Bit 4 là 1: xén x = xmin

Tìm giao điểm (x,y) của (x1,y1) (x2,y2) với cửa sổ xén

Ta có: $m = \frac{y-y1}{x-x1}$

hay $y - y1 = m(x - x1) \rightarrow y = y1 + m(x - x1)$

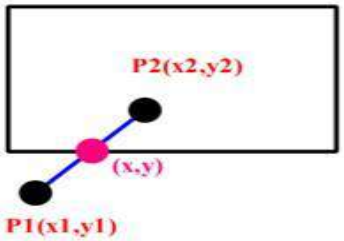
Nếu y nằm trên đường thẳng đứng: $\left[\begin{array}{l} x = xmin \\ x = xmax \end{array}\right.$

Nếu x nằm trên đường thẳng nằm ngang:

Từ phương trình: $m = \frac{y-y1}{x-x1} \rightarrow x - x1 = \frac{y-y1}{m}$

$hay\ x = x1 + \frac{(y - y1)}{m}$

và $\left[\begin{array}{l} y = xmin \\ y = ymax \end{array}\right.$



Hình 1.5 Ví dụ tìm giao điểm

Đối với thuật toán Cohen Shutherland, khi tìm giao điểm của đoạn thẳng cần xén tỉa với các cạnh của cửa sổ xén bằng cách dùng các tham số của phương trình đường thẳng (hệ số góc được tính bằng công thức xuất hiện phép chia). Dẫn tới không tối ưu về mặt thời gian.

Để khắc phục điều đó thuật toán Liang-Barsky ra đời. thuật toán Liang-Barsky (được đặt theo tên của You-Dong Liang và Brian A. Barsky) sử dụng phương trình tham số của đường thẳng và bất đẳng thức mô tả phạm vi của cửa sổ xén để xác định các giao điểm giữa đường thẳng và cửa sổ xén. Thuật toán này hiệu quả hơn đáng kể so với Cohen-Sutherland.

Ý tưởng: 3 bước để thực hiện:

Bước 1: Xét đường thẳng đi qua 2 điểm (x_1, y_1) và (x_2, y_2) như sau:

- Gọi t=0 tại điểm (x_1, y_1) và t=1 tại điểm (x_2, y_2)
- Xét 1 điểm thuộc đường thẳng có tọa độ (x, y). Khi đó:
- $x = t.x2 + (1-t)x1 = t.x2 + x1 - t.x1 = x1 + t(x2 - x1) = x1 + t.\Delta x$
- $y = y1 + t.\Delta y; với\ 0 < t < 1$

Bước 2: Xét cửa sổ clip được giới hạn bởi (x_{wmin}, y_{wmin}), (x_{wmax}, y_{wmax})

- Do vậy:

$$xwmin \le x \le xwmax$$
$$ywmin \le y \le ywmax$$

- Ta có 4 bất đẳng thức sau:
  - $x1 + t.\Delta x \ge xwmin$
  - $x1 + t.\Delta x \le xwmax$
  - $y1 + t.\Delta y \ge ywmin$
  - $y1 + t.\Delta y \le ywmax$
- Tổng quát ta có: $t.pk \le qk\ với\ k = 1,2,3,4$

Từ đó, ta có:

$p_1 = -\Delta x$	$q1 = x_1 - x_{wmin}$
$p_2 = \Delta x$	$q2 = x_{wmax} - x_1$
$p_3 = -\Delta y$	$q3 = y_1 - y_{wmin}$
$p_4 = \Delta y$	$q4 = y_{wmax} - y_1$

Bước 3: Kiểm tra vị trí của đoạn thẳng so với cửa sổ. Ta có các trường hợp sau:

- Nếu $p_k = 0$: điều đó tương đương với việc đoạn thẳng đang xét song song với cạnh thứ k của hình chữ nhật clipping.

❶ Nếu qk < 0 đoạn thẳng nằm ngoài cửa sổ (hệ bất phương trình trên vô nghiệm)

❷ Nếu qk > 0 thì đoạn thẳng nằm trong 1 phần của cửa sổ clipping

❸ Nếu qk = 0 thì đoạn thẳng nằm trên đường biên của cạnh k

- Nếu p_k <0, từ bất đẳng thức $t.pk \leq qx \rightarrow xác\ định\ t1$.

  ❹ Ta có: $t1 = \max(0, \frac{qk}{pk})$

  $x = x1 + t1.\Delta x; y = y1 + t1.\Delta y$

- Nếu p_k >0, từ bất đẳng thức $t.pk \leq qx \rightarrow xác\ định\ t2$.

❺ Ta có: $t2 = \min(1, \frac{qk}{pk})$

$x = x1 + t2.\Delta x; y = y1 + t2.\Delta y$

Nếu $t1 < 0$ và $t2 > 1$ thì cả 2 điểm đều nằm bên ngoài cửa sổ cliping

Nếu $t1 = 0$ và $t2 = 1$ thì cả 2 điểm đều nằm bên trong cửa sổ cliping

Nếu $0 < t1$ và $t2 < 1$ thì cả 2 điểm đều cần được xem xét

Thuật toán này giúp tối ưu hóa việc vẽ đoạn thẳng trên màn hình bằng cách loại bỏ những phần không cần thiết nằm ngoài cửa sổ.

- Nhận xét: $t1 = \max(0, \frac{qk}{pk})$; $t2 = \min\left(1, \frac{qk}{pk}\right)$
  - Nếu t1 thay đổi giá trị tăng dần từ 0 đến max $(\frac{qk}{pk})$, có nghĩa là đường thẳng có chiều từ noài vào trong cửa sổ clipping.
  - Nếu t2 thay đổi giá trị giảm dần từ 1 đến $min(\frac{qk}{pk})$, có nghĩa là đường thẳng có chiều trong cửa sổ clipping ra ngoài.

Nếu t1 thay đổi giá trị tăng dần từ 0 đến max $(\frac{qk}{pk})$ còn t2 thay đổi giá trị giảm dần từ 1 đến $min(\frac{qk}{pk})$, có nghĩa là đường thẳng có chiều từ ngoài cửa sổ clipping ra bên ngoài cửa sổ clipping.

1.3 Dev C++

Bloodshed Dev-C ++ (https://www.bloodshed.net/devcpp.html) là môi trường phát triển tích hợp (IDE) đầy đủ tính năng cho ngôn ngữ lập trình C/C ++ sử dụng Mingw của GCC (Bộ sưu tập trình biên dịch GNU) làm trình biên dịch. Dev-C ++ cũng có thể kết hợp với Cygwin hoặc bất kỳ trình biên dịch dựa trên GCC nào khác.

Các tính năng của Dev-C++:

Hỗ trợ trình biên dịch dựa trên GCC

Gỡ lỗi tích hợp (sử dụng GDB- General DeBug)

Quản lý dự án

Trình chỉnh sửa cú pháp

Trình duyệt lớp

Hoàn thành mã

Danh sách chức năng

Hồ sơ hỗ trợ

Nhanh chóng tạo Windows, console, thư viện tĩnh và DLL

Hỗ trợ các mẫu để tạo các loại dự án của riêng bạn

Tạo Makefile

Chỉnh sửa và biên dịch các tệp Tài nguyên

Quản lý công cụ

Hỗ trợ in

Tìm và thay thế mã lệnh

Hỗ trợ CVS

1.4 Thư viện Graphics.h

Với việc sử dụng DevC++ làm trình biên dịch cho các thuật toán và cài đặt, có một thách thức là không thể thực hiện trực tiếp trên môi trường Windows. Để giải quyết vấn đề này, Michael đã phát triển một môi trường giả lập đồ họa cho Borland C và tạo ra thư viện có tên là Graphics.h. Để tối ưu hóa sự tương thích trên Windows, Michael đã thay đổi thư viện BGI, chuyển đổi nó thành thư viện mới mang tên WinBGIm. Điều này cho phép người dùng DevC++ sử dụng các hàm đặc biệt của Borland một cách

thuận lợi, mà không gặp phải các vấn đề không tương thích trên nền tảng Windows. (https://github.com/thinhdoanvu/ComputerGraphics)

Chương 2. PHƯƠNG PHÁP NGHIÊN CỨU

2.1 Cài đặt DevC và thư viện graphics.h

Tải file cài đặt phần mềm DevC++ theo đường dẫn trong mục 1.4. Sau đó mở file vừa tải, và tiến hành cài đặt. Thư viện graphics.h được tiến hành cài đặt theo các bước:

Bước 1: Copy 6-ConsoleAppGraphics và ConsoleApp_cpp_graph

Paste C:\Program Files\Dev-Cpp\Templates

Bước 2: Copy graphics và winbgim

PasteC:\ProgramFiles\Dev-Cpp\MinGW64\x86_64-w64 mingw32\include

Bước 3: Copy libbgi.a

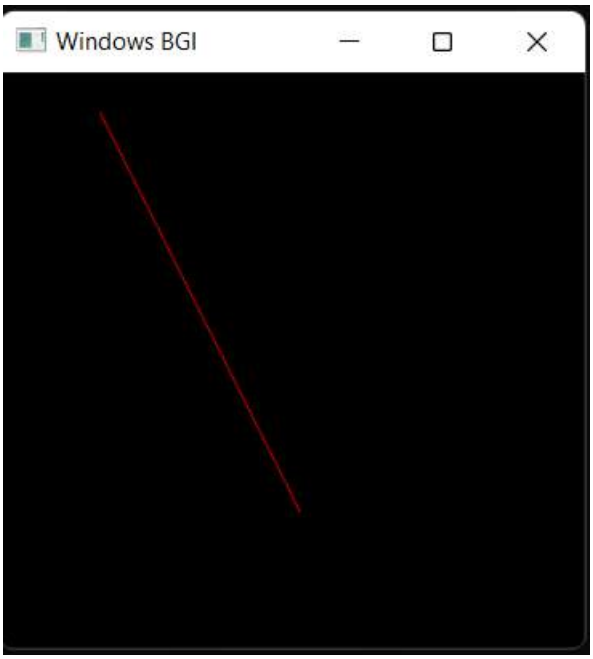
Paste C:\Program Files\Dev-Cpp\MinGW64\x86_64-w64-mingw32\lib

Bước 4: Ở DevC++ => New Project => Console Graphics Application

Bước 5: Thay đổi Tools – Complier Option: TDM – GCC 4.9.2 32 bit Release

Bước 6: Sử dụng đoạn code mẫu bên dưới để test thư viện Graphics.h

```
#include<graphics.h> // Thư viện graphics.h
int main(){
    initwindow(300,300);// Tạo cửa sổ BGI
    setcolor(RED); // Hàm thay đổi màu sắc
    line(50,20,150,220); // Hàm vẽ đoạn thẳng
    getch();
}
```



Hình 2.1. Ví dụ minh họa thư viện graphics.h

2.2 Cài đặt thuật toán

2.2.1. Khai báo các biến toàn cục

```
FILE *fp;// file
struct Button
{
    int x1, y1, x2, y2;
};
struct point {
    int x,y;
};
int color = 0;
int n=2;
int n1 = 0;
int xmin, xmax,ymin,ymax;
point p1, p2;
point pt[2];
point poly[20];
point poly1[20];
point clip[4];
int code1[10], code2[10];
int tong1, tong2;
int temp=0,temp1=0,temp2=0, temp3 = 0;
```

2.2.2. Nhập dữ liệu đầu vào

Nhập dữ liệu đầu vào bằng bàn phím

```
void KhoiTao()
{
    printf("\nNHAP PHAI THOA MAN DIEU KIEN \n|| 80<X<550 && 10<Y<370
||\n");
    do
    {
        printf("Nhap hinh chu nhat Clipping\n");
        printf("xmin = "); scanf("%d",&pt[0].x);
        printf("ymin = "); scanf("%d",&pt[0].y);
        printf("xmax = "); scanf("%d",&pt[1].x);
        printf("ymax = "); scanf("%d",&pt[1].y);
    }
    while(80 >= pt[0].x || 550<=pt[0].x  ||  10>=pt[0].y || 370 <=
pt[0].y || 80 >= pt[1].x || 550<=pt[1].x  ||  10>=pt[1].y || 370 <=
pt[1].y);
    //Ve cua so cut
    setcolor(GREEN);
    rectangle(pt[0].x,pt[0].y,pt[1].x,pt[1].y);
    printf("nhap so canh cua da giac :");
    scanf("%d",&n1);
    int i = 0;
    while(i<n1)
    {   printf("Nhap toa do dinh thu %d\n",i+1);
        printf("x = "); scanf("%d",&poly[i].x);
        printf("y = "); scanf("%d",&poly[i].y);
        if(80<poly[i].x && 550>poly[i].x
           &&10<poly[i].y && 370 > poly[i].y)
        {
            i++;
        }
        else {
            printf(" diem ban vua nhap khong nam trong cua so , vui
long nhap lai \n");
        }

    }
```

```
    for (int i=0; i<n1; i++)
    {   int k=(i+1)%n1;
        line(poly[i].x,poly[i].y,poly[k].x,poly[k].y);
    }
}
```

Nhập dữ liệu đầu vào bằng đọc file

```
void readfile() {
    FILE *fp;
    fp = fopen("xentia.inp", "r");
    if (fp == NULL) {
        printf("File not found or cannot be opened.\n");
        return;
    }
    // Read clipping window coordinates
    if (fscanf(fp, "%d %d %d %d", &pt[0].x, &pt[0].y, &pt[1].x,
&pt[1].y) != 4) {
        printf("Error reading coordinates of clipping window.\n");
        fclose(fp);
        return;
    }
    // Draw the clipping window
    setcolor(RED);
    rectangle(pt[0].x, pt[0].y, pt[1].x, pt[1].y);
    poly[0]={poly[0].x,poly[0].y};
    poly[1]={poly[1].x,poly[1].y};
    hoandoi();
    for (int i=2; i<=n-2 ; i+=2)
    {
        tohog(pt[i],pt[i+1]);
        for (int j=1; j<=4; j++)
        {
            code1[j] = 0;
            code2[j] = 0;
        }
    }
```

```
    // Read the number of vertices
    if (fscanf(fp, "%d", &n1) != 1) {
        printf("Error reading the number of vertices.\n");
        fclose(fp);
        return;
    }
    // Read and draw
    for (int i = 0; i < n1; i++) {
        if (fscanf(fp, "%d %d", &poly[i].x, &poly[i].y) != 2) {
            printf("Error reading coordinates of vertex %d.\n", i +
1);
            fclose(fp);
            return;
        }
        printf("\nToa do thu %d: (%d, %d)\n", i + 1, poly[i].x,
poly[i].y);
    }
    // Draw the line
    setcolor(GREEN);
    for (int i = 0; i < n1; i++) {
        int k = (i + 1) % n1;
        line(poly[i].x, poly[i].y, poly[k].x, poly[k].y);
    }
    // Perform polygon clipping using Cohen-Sutherland
    addpointwd();
    Cohen_Sutherland();
    fclose(fp);
}
```

Nhập dữ liệu đầu vào bằng chuột qua giao diện

```
struct Button
{
    int x1, y1, x2, y2;
};// Tọa độ điểm đầu – điểm cuối của 1 nút


bool check_button(Button bt,point p)
```

```
{//Kiểm tra điểm p có nằm trong nút không? Nút p là tọa độ của chuột
       if(bt.x1<=p.x && bt.y1<=p.y&& bt.x2>=p.x&& bt.y2>=p.y)
          return true;
       return false;
}

if (check_button(btLine,p))
            {
                while(1)
                {
                    delay(0.1);
                    point p1,p2;
                    if (ismouseclick(WM_MOUSEMOVE))
                    {
                        getmouseclick(WM_MOUSEMOVE, x, y);
                        clearmouseclick(WM_MOUSEMOVE);
                        setcolor(15);
                        printf(s,"\n Toa Do (%d,%d) \n",x,y);
                        outtextxy(300,40,s);
                    }

                    if (ismouseclick(WM_LBUTTONDOWN))
                    {
                        getmouseclick(WM_LBUTTONDOWN, x, y);
                        clearmouseclick(WM_LBUTTONDOWN);
                        p1 = {x,y};
                        if (!check_button(btboder,p1)) break;
                    }

                    if(ismouseclick(WM_LBUTTONUP))
                    {
                        getmouseclick(WM_LBUTTONUP, x, y);
                        clearmouseclick(WM_LBUTTONUP);
                        p2 = {x,y};
                        if (!check_button(btboder,p2)) break;
                        setcolor(color);
                        line(p1.x,p1.y,p2.x,p2.y);
                        pt[n]={p1.x,p1.y};
```

```
                    pt[n+1]={p2.x,p2.y};
                    n+=2;
                }
            }
        }
```

2.2.3. Cài đặt thuật toán xén tỉa

Thuật toán xén tỉa đoạn thẳng bằng Cohen – Sutherland

```
int x_intersect(point p1,point p2, point p3, point p4)
{  // hàm trả về toạ độ x - giao điểm của 2 đoạn thẳng p1p2 với p3p4
   // p1, p2 là toạ độ của 1 cạnh cửa sổ xén.
  // p3, p4 là toạ đồ của 1 cạnh đa giác cần xén
    int num = (p1.x*p2.y - p1.y*p2.x) * (p3.x-p4.x) -
              (p1.x-p2.x) * (p3.x*p4.y - p3.y*p4.x);
    int den = (p1.x-p2.x) * (p3.y-p4.y) - (p1.y-p2.y) * (p3.x-p4.x);
    return round(num/den);
}
int y_intersect(point p1,point p2, point p3, point p4)
{
    int num = (p1.x*p2.y - p1.y*p2.x) * (p3.y-p4.y) -
              (p1.y-p2.y) * (p3.x*p4.y - p3.y*p4.x);
    int den = (p1.x-p2.x) * (p3.y-p4.y) - (p1.y-p2.y) * (p3.x-p4.x);
    return round(num/den);
}
void cllip(point p1,point p2)
{
    int n2 = 0;
    for (int i = 0; i < n1; i++)
    {
        int k = (i+1) % n1;
        int ix = poly[i].x, iy = poly[i].y;
        int kx = poly[k].x, ky = poly[k].y;
        int i_pos = (p2.x-p1.x) * (iy-p1.y) - (p2.y-p1.y) * (ix-
p1.x);
        int k_pos = (p2.x-p1.x) * (ky-p1.y) - (p2.y-p1.y) * (kx-
p1.x);
```

```
        // Case 1 : Khi cả hai điểm đều ở bên trong
        if (i_pos < 0  && k_pos < 0)
        {
            poly1[n2].x = kx;
            poly1[n2].y = ky;
            n2++;
        }
        // Case 2: Khi chỉ có điểm đầu tiên ở bên ngoài, điểm thứ 2
nằm trong so với cạnh của cửa sổ xén đang xét
        else if (i_pos >= 0  && k_pos < 0)
        {
            poly1[n2].x = x_intersect(p1,p2,poly[i],poly[k]);
            poly1[n2].y = y_intersect(p1,p2,poly[i],poly[k]);
            n2++;

            poly1[n2].x = kx;
            poly1[n2].y = ky;
            n2++;
        }
        // Case 3: Khi chỉ có điểm thứ hai ở bên ngoài
        else if (i_pos < 0  &&  k_pos >= 0)
        {   poly1[n2].x = x_intersect(p1,p2,poly[i],poly[k]);
            poly1[n2].y = y_intersect(p1,p2,poly[i],poly[k]);
            n2++;
        }

        else
        {
        //nothing;
        }
    }
    n1 = n2;
    printf("%dN1\n = ",n1);
    for (int i = 0; i < n1; i++)
    {
        poly[i] = poly1[i];
        printf("\n%d--%d\n",poly[i].x,poly[i].y);
    }
```

```
}

void Cohen_Sutherland()
{
    for (int i=0; i<4; i++)
    {
        int k = (i+1) % 4;
        cllip(clip[i],clip[k]);
    }
    setcolor(RED);
    setlinestyle(0,0,3);
    for (int i = 0; i < n1; i++)
    {   int k= (i+1)%n1;
        line(poly[i].x,poly[i].y,poly[k].x,poly[k].y);
    }
}
```

Thuật toán xén tỉa đoạn thẳng bằng Liang – Barsky

```
void Liang_Barsky(point p1, point p2) {
    int dx = p2.x - p1.x;
    int dy = p2.y - p1.y;
    int p[4], q[4], t1 = 0, t2 = 1;

    p[0] = -dx;
    p[1] = dx;
    p[2] = -dy;
    p[3] = dy;

    q[0] = p1.x - xmin;
    q[1] = xmax - p1.x;
    q[2] = p1.y - ymin;
    q[3] = ymax - p1.y;

    for (int i = 0; i < 4; i++) {
        if (p[i] == 0 && q[i] < 0) {
            printf("Line is parallel to the clipping window and
outside\n");
```

```
            return;
        }

        float r = (float)q[i] / p[i];

        if (p[i] < 0 && r > t1)
            t1 = r;
        else if (p[i] > 0 && r < t2)
            t2 = r;
    }

    // Check if the line is outside the clipping window
    if (t1 > t2) {
        printf("Line is outside the clipping window\n");
        return hoandoi();
    }


    // Clipping coordinates
    int x1 = p1.x + t1 * dx;
    int y1 = p1.y + t1 * dy;
    int x2 = p1.x + t2 * dx;
    int y2 = p1.y + t2 * dy;

    printf("Line is inside the clipping window\n");
    printf("Clipped coordinates: (%d, %d) to (%d, %d)\n", pt[0].x,
pt[0].y, pt[1].x, pt[1].y);

    for (int i=0; i<4; i++)
    {
        int k = (i+1) % 4;

        cllip(clip[i],clip[k]);

    }
    setcolor(RED);
    setlinestyle(0,0,3);
    for (int i = 0; i < n1; i++)
```

```
    {   int k= (i+1)%n1;
        line(poly[i].x,poly[i].y,poly[k].x,poly[k].y);
    }
}
```

2.3 Cài đặt giao diện

Vẽ giao diện chính

```
void GUI(){
    setcolor(BLUE);
    //boder
    setlinestyle(0,1,3);
    rectangle(80,10,550,370);
    //button THUC
    setcolor(WHITE);
    settextstyle(2,0,7);
    outtextxy(18,32,"THUC");
    //button TAP y+40
    setcolor(WHITE);
    settextstyle(2,0,7);
    outtextxy(22,72,"TAP");
    //button CO y+40
    setcolor(WHITE);
    settextstyle(2,0,7);
    outtextxy(26,112,"CO");
    //button CO y+40
    setcolor(WHITE);
    settextstyle(2,0,7);
    outtextxy(26,152,"SO");
    //button CO y+40
    setcolor(WHITE);
    settextstyle(2,0,7);
    outtextxy(26,192,"2");
    //button CO y+40
    setcolor(WHITE);
    settextstyle(2,0,7);
    outtextxy(52,192,"0");
    //button CO y+40
```

```
setcolor(WHITE);
settextstyle(2,0,7);
outtextxy(26,232,"2");
//button CO y+40
setcolor(WHITE);
settextstyle(2,0,7);
outtextxy(52,232,"3");
//button back
setcolor(GREEN);
settextstyle(2,0,7);
rectangle(580,10,680,50);
outtextxy(610,20,"New");
//button load file : +60 for top and bottom
setcolor(WHITE);
settextstyle(2,0,7);
rectangle(580,70,680,110);
outtextxy(590,80,"Load File");
//button : +60 for top and bottom
setcolor(WHITE);
settextstyle(2,0,7);
rectangle(580,130,680,170);
outtextxy(590,140,"Keyboard");
//button fraphic
setcolor(WHITE);
settextstyle(2,0,7);
rectangle(580,190,680,230);
outtextxy(600,200,"Graphic");
// button ho ten GVHD:
setcolor(WHITE);
settextstyle(2,0,7);
outtextxy(200,380,"GVHD: Doan Vu Thinh");
// button ho ten:
setcolor(WHITE);
settextstyle(2,0,7);
outtextxy(160,400,"FullName: Cao Nguyen Quoc Lam");
// button MSSV:
setcolor(WHITE);
settextstyle(2,0,7);
```

```
    outtextxy(210,420,"MSSV: 63132195");
    // button Lop:-
    setcolor(WHITE);
    settextstyle(2,0,7);
    outtextxy(200,440,"Class: 63.CNTT - 4");
}
```

Vẽ các nút trong giao diện sử dụng chuột

```
void button(Button bt, int color_fill)
{

    setlinestyle(0,0,2);
    setfillstyle(1,color_fill);
    setcolor(15);
    rectangle(bt.x1, bt.y1, bt.x2, bt.y2);
}
```

Vẽ hình ảnh bên trong nút ở giao diện sử dụng chuột

```
//------------Button ve doan thang-------------
void button_line(Button bt, int color_fill, int color_line)
{
    button(bt, color_fill);
    setcolor(color_line);
    line((bt.x2-bt.x1)/5*2, (bt.y2-bt.y1)/5+bt.y1, (bt.x2-bt.x1),
(bt.y2-bt.y1)/5*4+bt.y1);

}
```

Cài đặt hiệu ứng hover

```
while(1)
   {
        delay(0.1);
        if (ismouseclick(WM_MOUSEMOVE))
        {
            int x,y;
            char s[100];
            getmouseclick(WM_MOUSEMOVE, x, y);
            point p = {x,y};
            //Hover button Liang
            if (check_button(btLiang,p))
            {
                setcolor(15);
                button_liang(btLiang, 5 , 3);
                temp3=1;
            }
            else if (!check_button(btLiang, p) && temp3==1)
            {
                setcolor(15);
                button_liang(btLiang, 2 , 15);
                temp3=0;
            }
```

Chương 3. KẾT QUẢ

3.1 Giao diện chính

Hình 3.1. Giao diện chính

L_Click vào mục "Load File" sẽ chạy giao diện load file để xén tỉa đoạn thẳng được nhập dữ liệu sẵn trong file xentia.inp

L_Click vào mục "Keybroad" sẽ chạy giao diện nhập bằng tay và không cho thực hiện bất kỳ hành động nào trên giao diện chính nữa.

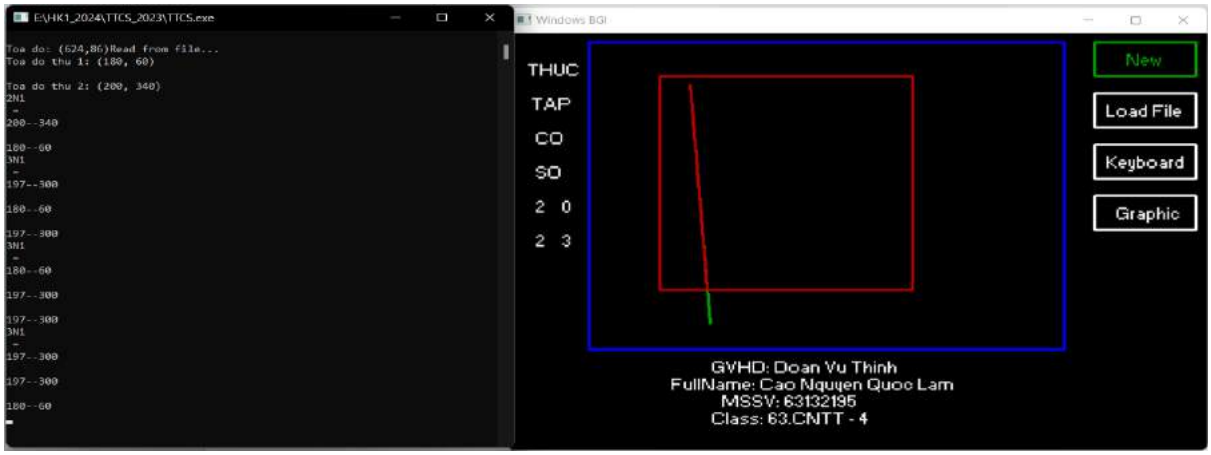
L_Click vào mục "Graphic" sẽ dẫn tới giao diện đồ họa sử dụng chuột để thực hiện thao tác

3.2 Giao diện đọc từ file

```
150 50 400 300
2
180 60
200 340
```

Hình 3.2.1. Dữ liệu trong file xentia.inp

Khi L_Click vào button Load File thì cửa sổ console hiển thị kết quả đọc từ file xentia.inp và cửa sổ BGI sẽ hiển thị được kết quả đồ họa thể hiện thuật toán xén tỉa đoạn thẳng nằm trực tiếp trên giao diện chính.

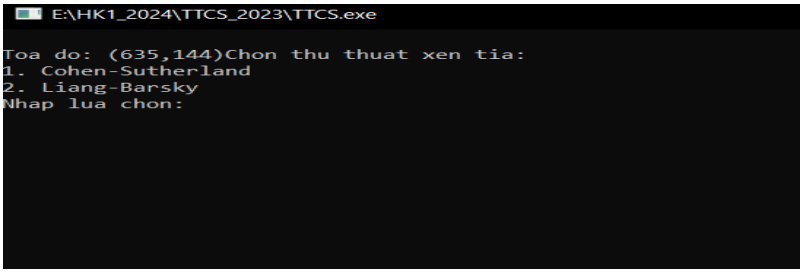


Hình 3.2.2. Giao diện khi load file

3.3 Giao diện nhập dữ liệu từ bàn phím

Khi L_Click vào button keybroad thì cửa console cho phép nhập dữ liệu. Lúc này chỉ có thể thao tác nhập dữ liệu bên trên cửa sổ console và giao diện chính tạm thời không thể sử dụng được.

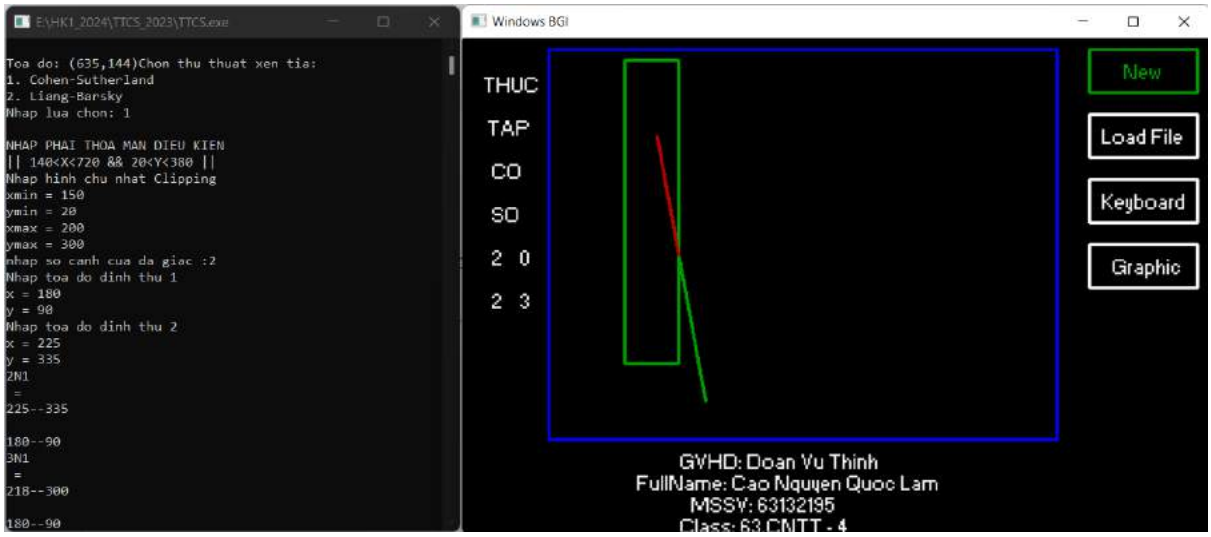
Trước khi nhập dữ liệu sẽ cho người nhập lựa chọn 1 trong 2 thuật toán



Hình 3.3.1. Người dùng lựa chọn thuật toán xén tỉa

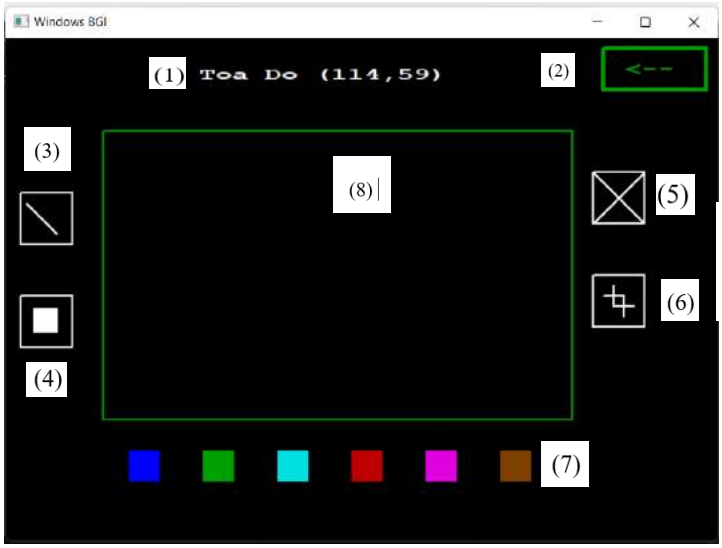
Lựa chọn và sau khi nhập xong, nhấn phím enter để kết thúc việc nhập và lưu dữ liệu

Sau đó tiến hành xén đoạn thẳng vừa nhập và hiển thị lên màn hình BGI



Hình 3.3.2. Giao diện nhập từ bàn phím

3.4 Giao diện đồ họa sử dụng chuột



(1) Trạng thái con trỏ chuột

(2) Button quay trở lại giao diện chính

(3) Button vẽ đoạn thẳng

(4) Button để xóa màn hình

(5) Button vẽ cửa sổ xén và thực hiện thuật toán Liang – Barsky

(6) Button vẽ cửa sổ xén và thực hiện thuật toán Cohen - Sutherland

(7) Bảng màu để vẽ

(8) Border hiển thị

Hình 3.4.1. Giao diện đồ họa sử dụng chuột

Trạng thái con trỏ chuột

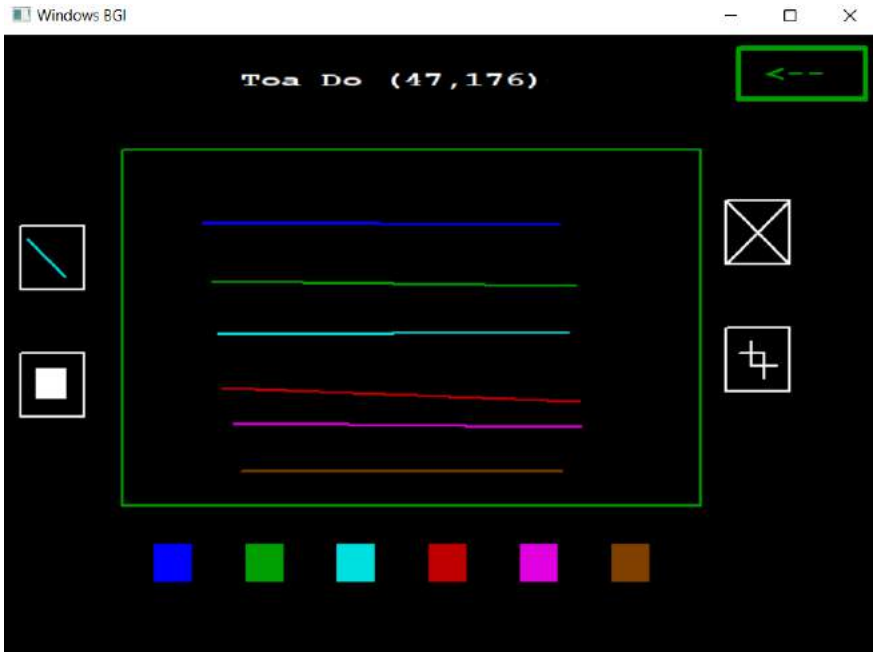
Khi di chuyển con trỏ chuột thì sẽ hiển thị đúng vị trí con trỏ chuột

Button quay lại

Khi L_Click vào button "< --" sẽ quay trở lại menu chính lúc ban đầu

Màu vẽ

Khi L_Click vào các button màu vẽ phía dưới sẽ thay đổi màu vẽ vừa click vào



Hình 3.4.2. Minh họa sử dụng button màu

Button vẽ đoạn thẳng

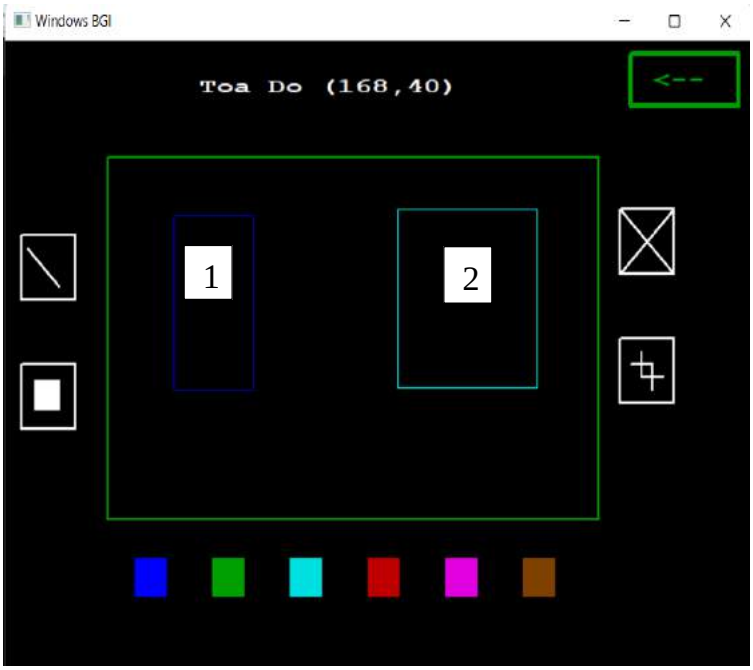
Trước khi L_Click vào button vẽ đoạn thẳng thì trước hết cần chọn màu cho đoạn thẳng đó trước nếu như không chọn màu thì không thể vẽ đoạn thẳng được và L_Click lần sau ra ngoài phạm vi khung vẽ thì chế độ đó tắt

Sau khi L_Click vào button màu và L_Click vào button vẽ đoạn thẳng thì sau đó L_Click giữ kéo đến vị trí của điểm thứ hai thì thả ra để vẽ, có thể vẽ được nhiều đoạn thẳng trên khung vẽ.

Button vẽ khung xén

Khi L_Click button vẽ cửa sổ xén trên màn hình và L_Click lần sau ra ngoài phạm vi khung vẽ thì chế độ đó tắt

Sau khi L_Click vào button sau đó đưa con trỏ chuột vào khung vẽ và L_Click giữ kéo đến vị trí của điểm thứ hai thì thả ra để vẽ cửa sổ xén



1. Cửa sổ cắt của thuật toán Cohen – Sutherland
2. Cửa sổ cắt của thuật toán Liang – Barsky

Hình 3.4.3. Minh họa sử dụng button vẽ cửa sổ xén của 2 thuật toán

Button xóa màn hình

Khi L_Click vào màn hình toàn bộ dữ liệu hình vẽ trước đó được hiển thị trên khung vẽ sẽ bị xóa, và reset bộ nhớ về trạng thái ban đầu.

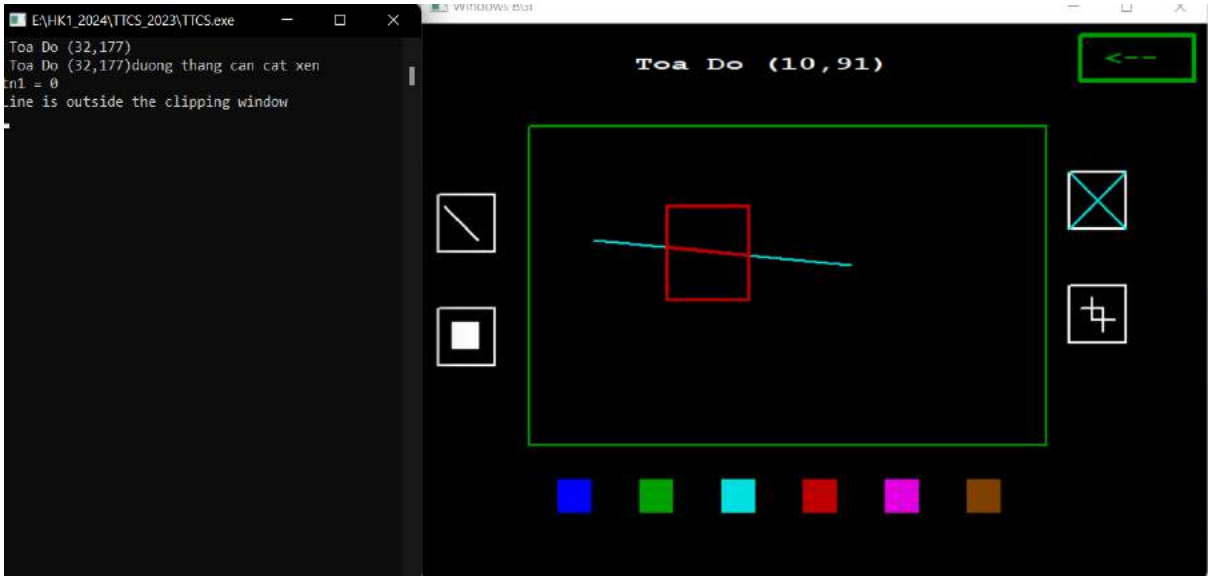
3.5 Xén tỉa đoạn thẳng bằng đồ họa

Để xén tỉa một hay nhiều đoạn thẳng thì cần thực hiện 2 bước

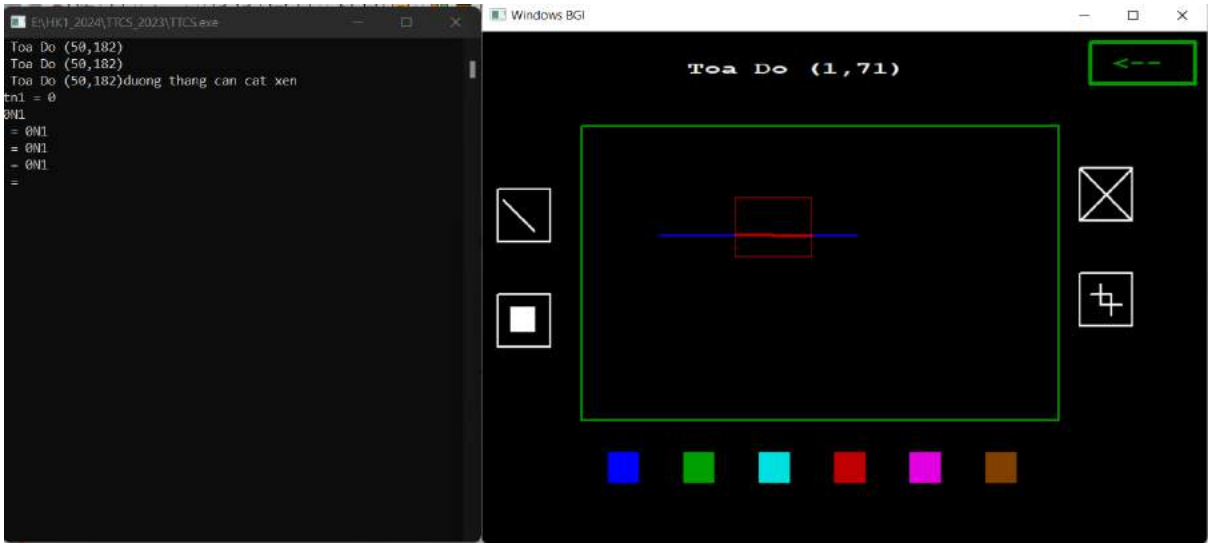
Bước 1: Vẽ các đoạn thẳng xem mục 3.4.2

Bước 2: Vẽ cửa sổ xén xem mục 3.4.3

Sau khi vẽ cửa sổ xén thì phần đoạn thẳng nằm trong cửa sổ xén sẽ chuyển màu đỏ.



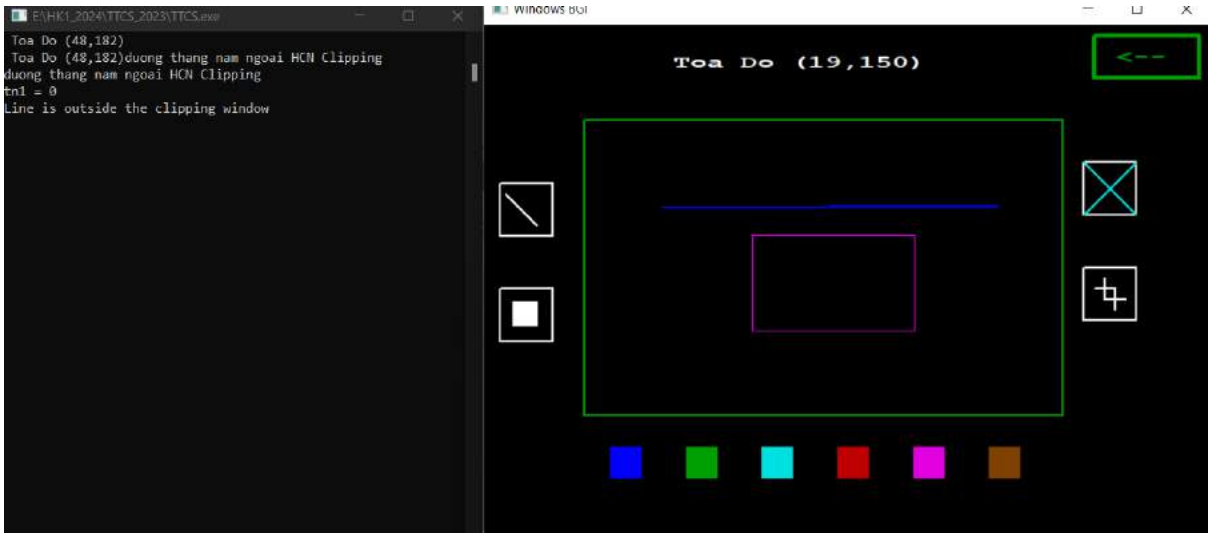
Hình 3.5.1. Minh họa xén tỉa đoạn thẳng bằng thuật toán Liang – Barsky



Hình 3.5.2. Minh họa xén tỉa đoạn thẳng bằng thuật toán Cohen – Sutherland

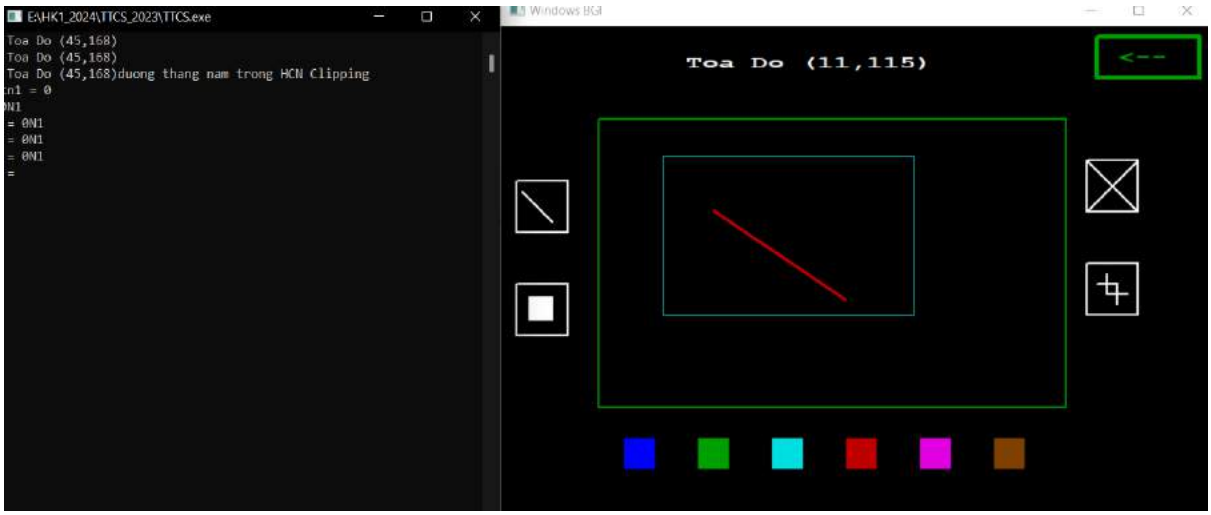
Các trường hợp của xén tỉa đoạn thẳng

TH1: Đoạn thẳng nằm ngoài cửa sổ xén



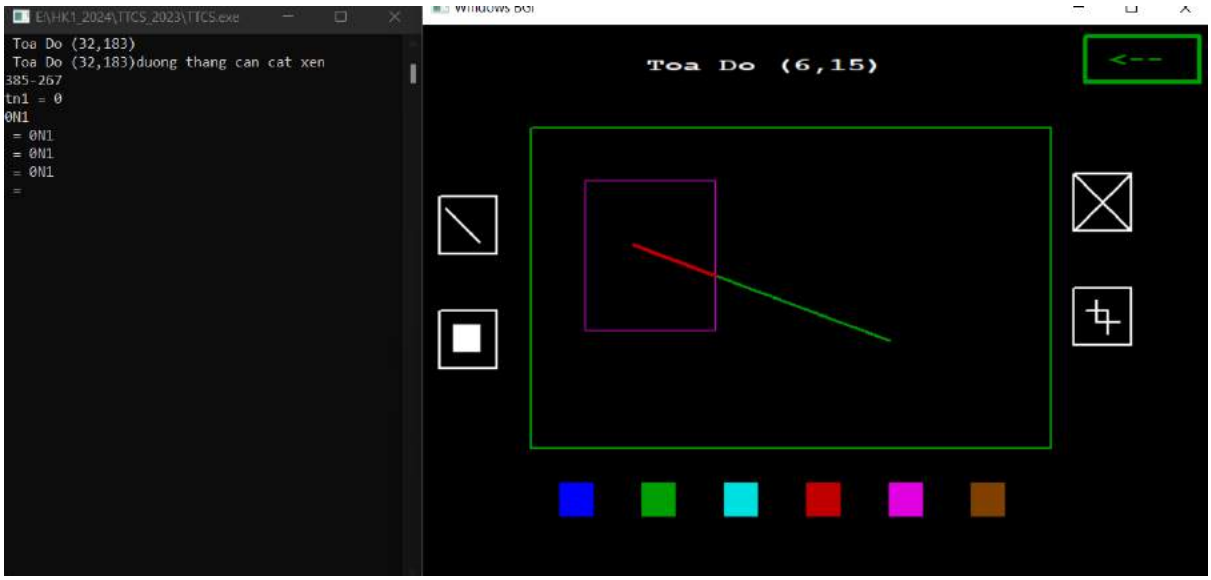
Hình 3.5.3. Minh họa trường hợp đoạn thẳng nằm ngoài cửa sổ cắt

TH2: Đoạn thẳng nằm hoàn toàn bên trong cửa sổ cắt



Hình 3.5.4. Minh họa trường hợp đoạn thẳng nằm bên trong cửa sổ cắt

TH3: Đoạn thẳng nằm cả trong lẫn ngoài



Hình 3.5.5. Minh họa trường hợp đoạn thẳng nằm cả trong lẫn ngoài cửa sổ cắt

Chương 4. KẾT LUẬN

Thuật toán xén tỉa đoạn thẳng Cohen – Sutherland và Liang – Barsky được cài đặt theo các yêu cầu của đề tài thực tập cơ sở. Việc nhập dữ liệu được thông qua bàn phím và sử dụng chuột trái, phải. Giao diện thiết kế sử dụng hoàn toàn bằng thư viện graphics.h. Các trường hợp xén tỉa đa giác đều được mô hình hoá và cài đặt, trong khuôn khổ báo cáo này, hạn chế của demo là chưa thực sự đẹp và hoàn hảo, chưa khắc phục được việc vẽ đoạn thẳng hay cửa sổ cắt khi không chọn màu, dù đã được chỉ ra nhưng chưa thể giải quyết trong đợt thực tập lần này. Trong quá trình hoàn thành phần mềm với kinh nghiệm thực tế chưa nhiều nên báo cáo không tránh khỏi những sai sót, rất mong sự góp ý của các thầy cô bộ môn.

Em xin chân thành cảm ơn.

TÀI LIỆU THAM KHẢO

[1] Bài giảng Kỹ thuật đồ họa, Đoàn Vũ Thịnh, 2019

[2] Hướng dẫn giải chi tiết và lập trình Kỹ thuật đồ hoạ, Đoàn Vũ Thịnh, 2021

[3] Computer Graphics, C Version (2nd Edition), Donald Hearn, 1996

[4] Giải thuật và lập trình, Lê Minh Hoàng, 2006

PHỤ LỤC

A1. Code các button được sử dụng

```
//----------------------------Khai báo các kieu button-----------//
void bt_color(Button bt, int color_fill)
{
    setlinestyle(1,2,2);
    setfillstyle(1,color_fill);
    bar(bt.x1, bt.y1, bt.x2, bt.y2);
}

//------Kiếm tra xem chuột có nằm trong button không ?-----------//
bool check_button(Button bt,point p)
{
       if(bt.x1<=p.x && bt.y1<=p.y&& bt.x2>=p.x&& bt.y2>=p.y)
          return true;
       return false;
}

// Button back
void button2(Button bt, int color_fill)
{
    setcolor(color_fill);
    setlinestyle(2,3,4);
    rectangle(bt.x1, bt.y1, bt.x2, bt.y2);
}

//-----Button mặc định---------------//
void button(Button bt, int color_fill)
{
    setlinestyle(0,0,2);
    setfillstyle(1,color_fill);
    setcolor(15);
    rectangle(bt.x1, bt.y1, bt.x2, bt.y2);
}
```

```
//------------Button vẽ đoạn thẳng-------------//
void button_line(Button bt, int color_fill, int color_line)
{
    button(bt, color_fill);
    setcolor(color_line);
    line((bt.x2-bt.x1)/5*2, (bt.y2-bt.y1)/5+bt.y1, (bt.x2-bt.x1),
(bt.y2-bt.y1)/5*4+bt.y1);

}
//----------Button dùng code Liang------------
void button_liang(Button bt, int color_fill, int color_line)
{
    button(bt, color_fill);
    setcolor(color_line);
    line(bt.x1, bt.y1, bt.x2, bt.y2);
    line(bt.x1, bt.y2, bt.x2, bt.y1);
}

 //----------Button dùng code Cohen---------------
void button_cohen(Button bt, int color_fill, int color_line)
{
    button(bt, color_fill);
    settextjustify(1, 1);
    setfillstyle(1,color_line);
 -----------//Thực hiện vẽ hình bên trong button-----------
    int space = 10;
    line(bt.x1 + space, bt.y1 + 2*space, bt.x2 - 2*space, bt.y1 +
2*space);
    line(bt.x1 + 2*space, bt.y2 - 2*space, bt.x2 - space, bt.y2 -
2*space);
    line(bt.x1 + 2*space, bt.y1 + space, bt.x1 + 2*space, bt.y2 -
2*space);
    line(bt.x2 - 2*space, bt.y1 + 2*space, bt.x2 - 2*space, bt.y2 -
space);

}
```

```
//-----------------Button Delete------------------
void button_del(Button bt, int color_fill, int color_line)
{
    button(bt, color_fill);
    settextjustify(1, 1);
    setfillstyle(1,color_line);
    bar(bt.x1 + (bt.x2-bt.x1)/4, bt.y1 + (bt.y2-bt.y1)/4, bt.x1 +
(bt.x2-bt.x1)/4*3, bt.y1 + (bt.y2-bt.y1)/4*3);
}

//-------------Khung vẽ trong giao diện graphic-------------------
void boder(Button bt, int color_line)
{
    setcolor(color_line);
    rectangle(bt.x1, bt.y1, bt.x2, bt.y2);
}

//-------------------Button quay lại-----------------------------
void button_back(Button bt, int color_fill, int color_line)
{
    button2(bt, color_fill);
    settextjustify(1, 1);
    setfillstyle(1,color_line);
    settextstyle(0,0,2);
    outtextxy((bt.x1)+45, (bt.y1)+25,"<--");
}
```

A2. Code giao diện chính

```
void GUI(){
    setcolor(BLUE);
    //boder
    setlinestyle(0,1,3);
    rectangle(80,10,550,370);
```

```
//button THUC
setcolor(WHITE);
settextstyle(2,0,7);
outtextxy(18,32,"THUC");
//button TAP y+40
setcolor(WHITE);
settextstyle(2,0,7);
outtextxy(22,72,"TAP");
//button CO y+40
setcolor(WHITE);
settextstyle(2,0,7);
outtextxy(26,112,"CO");
//button CO y+40
setcolor(WHITE);
settextstyle(2,0,7);
outtextxy(26,152,"SO");
//button CO y+40
setcolor(WHITE);
settextstyle(2,0,7);
outtextxy(26,192,"2");
//button CO y+40
setcolor(WHITE);
settextstyle(2,0,7);
outtextxy(52,192,"0");
//button CO y+40
setcolor(WHITE);
settextstyle(2,0,7);
outtextxy(26,232,"2");
//button CO y+40
setcolor(WHITE);
settextstyle(2,0,7);
outtextxy(52,232,"3");
//button back
setcolor(GREEN);
settextstyle(2,0,7);
rectangle(580,10,680,50);
outtextxy(610,20,"New");
//button load file : +60 for top and bottom
```

```
    setcolor(WHITE);
    settextstyle(2,0,7);
    rectangle(580,70,680,110);
    outtextxy(590,80,"Load File");
    //button : +60 for top and bottom
    setcolor(WHITE);
    settextstyle(2,0,7);
    rectangle(580,130,680,170);
    outtextxy(590,140,"Keyboard");
    //button fraphic
    setcolor(WHITE);
    settextstyle(2,0,7);
    rectangle(580,190,680,230);
    outtextxy(600,200,"Graphic");
    // button ho ten GVHD:
    setcolor(WHITE);
    settextstyle(2,0,7);
    outtextxy(200,380,"GVHD: Doan Vu Thinh");
    // button ho ten:
    setcolor(WHITE);
    settextstyle(2,0,7);
    outtextxy(160,400,"FullName: Cao Nguyen Quoc Lam");
    // button MSSV:
    setcolor(WHITE);
    settextstyle(2,0,7);
    outtextxy(210,420,"MSSV: 63132195");
    // button Lop:-
    setcolor(WHITE);
    settextstyle(2,0,7);
    outtextxy(200,440,"Class: 63.CNTT - 4");
}
```

A3. Code sử dụng chuột trên giao diện chính

```
void clickmouse(){
    int x_mouse;
```

```
    int y_mouse;
    while(1){
        //get even mouse click
        if(ismouseclick(WM_LBUTTONDOWN)){
            getmouseclick(WM_LBUTTONDOWN,x_mouse,y_mouse);
            printf("\nToa do: (%d,%d)",x_mouse,y_mouse);
            //check button back
            if(x_mouse > 580 && x_mouse <680 && y_mouse > 10 &&
y_mouse < 50){ //Reset
                printf("Resetting....");
                resetAll();
            }
            //check button
            if(x_mouse > 580 && x_mouse <680 && y_mouse > 70 &&
y_mouse < 110){ //load file
                printf("Read from file...");
                readfile();
            }
            // check button keyboard
            if(x_mouse > 580 && x_mouse <680 && y_mouse > 130 &&
y_mouse < 170){ //keyboard
                setcolor(RED);
                keyboard();
            }
            // check button graphic
            if(x_mouse > 580 && x_mouse <680 && y_mouse > 190 &&
y_mouse < 230){ //Graphic
                printf("Running demo...");
                cleardevice();
                Graphic();
            }
        }
        //Check right mouse click
        if(ismouseclick(WM_RBUTTONDOWN)){
            getmouseclick(WM_RBUTTONDOWN,x_mouse,y_mouse);
            printf("\nExiting...");
            break;
        }delay(100);}}// next click mouse
```

A4. Code thuật toán Cohen - Sutherland

```
point kiemtra(point p1, point p2, point p3)
{
    int t1 = (p1.x - p2.x)*(p1.x - p2.x) + (p1.y - p2.y)*(p1.y -
p2.y);
    int t2 = (p1.x - p3.x)*(p1.x - p3.x) + (p1.y - p3.y)*(p1.y -
p3.y);
    if (t1 > t2 ) return p2;
    return p3;

}

void hoandoi()
{
    if (pt[0].x > pt[1].x) swap(pt[0].x, pt[1].x);
    if (pt[0].y > pt[1].y) swap(pt[0].y, pt[1].y);
    rectangle(pt[0].x, pt[0].y, pt[1].x, pt[1].y);
}

void MaCode(point p1, point p2)
{
    tong1 = 0;
    tong2 = 0;
    if (p1.y> pt[1].y) code1[1] = 1;
    if (pt[0].y > p1.y) code1[2] = 1;
    if (p1.x > pt[1].x) code1[3] = 1;
    if (pt[0].x > p1.x) code1[4] = 1;

    if (p2.y > pt[1].y) code2[1] = 1;
    if (pt[0].y > p2.y) code2[2] = 1;
    if (p2.x > pt[1].x) code2[3] = 1;
    if (pt[0].x > p2.x) code2[4] = 1;

    int mu2 = 1;
    for (int i = 4; i > 0; i--)
    {
        tong1 += mu2*code1[i];
```

```
        tong2 += mu2*code2[i];
        mu2 *= 2;
    }
}

 void CatXen(point p1, point p2)
{
    if ((tong1+tong2) == 0)
    {
        printf("duong thang nam trong HCN Clipping\n");
        setcolor(RED);
        setlinestyle(0,0,3);
        line(p1.x,p1.y,p2.x,p2.y);

    }
    else if ((tong1&tong2) != 0)
    {
        printf("duong thang nam ngoai HCN Clipping\n");
    }

    else
    {
        printf("duong thang can cat xen\n");
        float m = 1.0*(p2.y-p1.y)/(p2.x-p1.x);
        float x,y;
        if(tong1 != 0)
        {
            if(code1[1]==1 && code1[4]==1)
            {
                point b1, b2;
                x = p1.x+1.0*(pt[1].y-p1.y)/m;
                b1.x = round(x);
                b1.y= pt[1].y;
                printf("%d-%d\n",b1.x,b1.y);
                y = p1.y+(pt[0].x-p1.x)*m;
                b2.y= round(y);
                b2.x=pt[0].x;
                p1=kiemtra(p1,b1,b2);
```

```
}
else if(code1[1]==1 && code1[3]==1)
{
    point b1, b2;
    x = p1.x+1.0*(pt[1].y-p1.y)/m;
    b1.x = round(x);
    b1.y= pt[1].y;

    printf("%d-%d\n",b1.x,b1.y);

    y = p1.y+(pt[1].x-p1.x)*m;
    b2.y= round(y);
    b2.x=pt[1].x;
    p1=kiemtra(p1,b1,b2);

}
else if(code1[2]==1 && code1[3]==1)
{
    point b1, b2;
    x = p1.x+1.0*(pt[0].y-p1.y)/m;
    b1.x = round(x);
    b1.y= pt[0].y;

    printf("%d-%d\n",b1.x,b1.y);

    y = p1.y+(pt[1].x-p1.x)*m;
    b2.y= round(y);
    b2.x=pt[1].x;
    p1=kiemtra(p1,b1,b2);

}
else if(code1[2]==1 && code1[4]==1)
{
    point b1, b2;
    x = p1.x+1.0*(pt[0].y-p1.y)/m;
    b1.x = round(x);
```

```
            b1.y= pt[0].y;

            printf("%d-%d\n",b1.x,b1.y);

            y = p1.y+(pt[0].x-p1.x)*m;
            b2.y= round(y);
            b2.x=pt[0].x;
            p1=kiemtra(p1,b1,b2);

        }

        else if(code1[1] == 1)
        {
            x = p1.x+1.0*(pt[1].y-p1.y)/m;
            p1.x = round(x);
            p1.y= pt[1].y;
        }
        else if (code1[4] == 1)
        {
            y = p1.y+(pt[0].x-p1.x)*m;
            p1.y= round(y);
            p1.x=pt[0].x;
        }
        else if (code1[2] == 1)
        {
            x = p1.x+1.0*(pt[0].y-p1.y)/m;
            p1.x = round(x);
            p1.y= pt[0].y;
        }
        else if (code1[3] == 1)
        {
            y = p1.y+(pt[1].x-p1.x)*m;
            p1.y= round(y);
            p1.x=pt[1].x;
        }
    }

    if(tong2 != 0)
```

```
{
    if(code2[1]==1 && code2[4]==1)
    {
        point b1, b2;
        x = p2.x+1.0*(pt[1].y-p2.y)/m;
        b1.x = round(x);
        b1.y= pt[1].y;
        printf("%d-%d\n",b1.x,b1.y);
        y = p2.y+(pt[0].x-p2.x)*m;
        b2.y= round(y);
        b2.x=pt[0].x;
        p2=kiemtra(p2,b1,b2);

    }
    else if(code2[1]==1 && code2[3]==1)
    {
        point b1, b2;
        x = p2.x+1.0*(pt[1].y-p2.y)/m;
        b1.x = round(x);
        b1.y= pt[1].y;

        printf("%d-%d\n",b1.x,b1.y);

        y = p2.y+(pt[1].x-p2.x)*m;
        b2.y= round(y);
        b2.x=pt[1].x;
        p2=kiemtra(p2,b1,b2);

    }
    else if(code2[2]==1 && code2[3]==1)
    {
        point b1, b2;
        x = p2.x+1.0*(pt[0].y-p2.y)/m;
        b1.x = round(x);
        b1.y= pt[0].y;

        printf("%d-%d\n",b1.x,b1.y);
```

```
        y = p2.y+(pt[1].x-p2.x)*m;
        b2.y= round(y);
        b2.x=pt[1].x;
        p2=kiemtra(p2,b1,b2);

    }
    else if(code2[2]==1 && code2[4]==1)
    {
        point b1, b2;
        x = p2.x+1.0*(pt[0].y-p2.y)/m;
        b1.x = round(x);
        b1.y= pt[0].y;

        printf("%d-%d\n",b1.x,b1.y);

        y = p2.y+(pt[0].x-p2.x)*m;
        b2.y= round(y);
        b2.x=pt[0].x;
        p2=kiemtra(p2,b1,b2);

    }
    else if(code2[1] == 1)
    {
        x=p2.x+1.0*(pt[1].y-p2.y)/m;
        p2.x = round(x);
        p2.y = pt[1].y;
    }
    else if (code2[2] == 1)
    {
        x = p2.x+1.0*(pt[0].y-p2.y)/m;
        p2.x = round(x);
        p2.y = pt[0].y;
    }
    else if (code2[3] == 1)
    {
        y = p2.y + (pt[1].x-p2.x)*m;
        p2.y = round(y);
        p2.x = pt[1].x;
```

```
            }
            else if (code2[4] == 1)
            {
                y = p2.y + (pt[0].x-p2.x)*m;
                p2.y = round(y);
                p2.x = pt[0].x;
            }
        }
        int mid_x = round((p1.x+p2.x)/2);
        int mid_y = round((p1.y+p2.y)/2);
        if (pt[0].x < mid_x && mid_x < pt[1].x && pt[0].y < mid_y &&
mid_y < pt[1].y)
        {
            setcolor(RED);
            setlinestyle(0,0,3);
            line(p1.x,p1.y,p2.x,p2.y);
        }
    }
}

void tohog(point p1, point p2)
{
    MaCode(p1,p2);
    CatXen(p1,p2);
}

void addpointwd()
 {
    clip[0] = pt[0];
    clip[1].x = pt[0].x;
    clip[1].y = pt[1].y;
    clip[2] = pt[1];
    clip[3].x = pt[1].x;
    clip[3].y = pt[0].y;
 }

int x_intersect(point p1,point p2, point p3, point p4)
{
```

```
    int num = (p1.x*p2.y - p1.y*p2.x) * (p3.x-p4.x) -
              (p1.x-p2.x) * (p3.x*p4.y - p3.y*p4.x);
    int den = (p1.x-p2.x) * (p3.y-p4.y) - (p1.y-p2.y) * (p3.x-p4.x);
    return round(num/den);
}

int y_intersect(point p1,point p2, point p3, point p4)
{
    int num = (p1.x*p2.y - p1.y*p2.x) * (p3.y-p4.y) -
              (p1.y-p2.y) * (p3.x*p4.y - p3.y*p4.x);
    int den = (p1.x-p2.x) * (p3.y-p4.y) - (p1.y-p2.y) * (p3.x-p4.x);
    return round(num/den);
}

void cllip(point p1,point p2)
{
    int n2 = 0;
    for (int i = 0; i < n1; i++)
    {
        int k = (i+1) % n1;
        int ix = poly[i].x, iy = poly[i].y;
        int kx = poly[k].x, ky = poly[k].y;
        int i_pos = (p2.x-p1.x) * (iy-p1.y) - (p2.y-p1.y) * (ix-
p1.x);
        int k_pos = (p2.x-p1.x) * (ky-p1.y) - (p2.y-p1.y) * (kx-
p1.x);
        // Case 1 : When both points are inside
        if (i_pos < 0  && k_pos < 0)
        {
            poly1[n2].x = kx;
            poly1[n2].y = ky;
            n2++;
        }
        // Case 2: When only first point is outside
        else if (i_pos >= 0  && k_pos < 0)
        {
            poly1[n2].x = x_intersect(p1,p2,poly[i],poly[k]);
            poly1[n2].y = y_intersect(p1,p2,poly[i],poly[k]);
```

```
            n2++;

            poly1[n2].x = kx;
            poly1[n2].y = ky;
            n2++;
        }
        // Case 3: When only second point is outside
        else if (i_pos < 0  &&  k_pos >= 0)
        {   poly1[n2].x = x_intersect(p1,p2,poly[i],poly[k]);
            poly1[n2].y = y_intersect(p1,p2,poly[i],poly[k]);
            n2++;
        }

        else
        {
        //nothing;
        }
    }
    n1 = n2;
    printf("%dN1\n = ",n1);
    for (int i = 0; i < n1; i++)
    {
        poly[i] = poly1[i];
        printf("\n%d--%d\n",poly[i].x,poly[i].y);
    }
}

void Cohen_Sutherland()
{
    for (int i=0; i<4; i++)
    {
        int k = (i+1) % 4;
        cllip(clip[i],clip[k]);
    }
    setcolor(RED);
    setlinestyle(0,0,3);
    for (int i = 0; i < n1; i++)
    {   int k= (i+1)%n1;
```

```
        line(poly[i].x,poly[i].y,poly[k].x,poly[k].y); }}
```

A5. Code thuật toán Liang – Barsky

```
void Liang_Barsky(point p1, point p2) {
    int dx = p2.x - p1.x;
    int dy = p2.y - p1.y;
    int p[4], q[4], t1 = 0, t2 = 1;

    p[0] = -dx;
    p[1] = dx;
    p[2] = -dy;
    p[3] = dy;

    q[0] = p1.x - xmin;
    q[1] = xmax - p1.x;
    q[2] = p1.y - ymin;
    q[3] = ymax - p1.y;

    for (int i = 0; i < 4; i++) {
        if (p[i] == 0 && q[i] < 0) {
            printf("Line is parallel to the clipping window and
outside\n");
            return;
        }

        float r = (float)q[i] / p[i];

        if (p[i] < 0 && r > t1)
            t1 = r;
        else if (p[i] > 0 && r < t2)
            t2 = r;
    }

    // Check if the line is outside the clipping window
    if (t1 > t2) {
        printf("Line is outside the clipping window\n");
```

```
        return hoandoi();
    }


    // Clipping coordinates
    int x1 = p1.x + t1 * dx;
    int y1 = p1.y + t1 * dy;
    int x2 = p1.x + t2 * dx;
    int y2 = p1.y + t2 * dy;

    printf("Line is inside the clipping window\n");
    printf("Clipped coordinates: (%d, %d) to (%d, %d)\n", pt[0].x,
pt[0].y, pt[1].x, pt[1].y);

    for (int i=0; i<4; i++)
    {
        int k = (i+1) % 4;

        cllip(clip[i],clip[k]);

    }
    setcolor(RED);
    setlinestyle(0,0,3);
    for (int i = 0; i < n1; i++)
    {   int k= (i+1)%n1;
        line(poly[i].x,poly[i].y,poly[k].x,poly[k].y);
    }
}
```

A6. Code giao diện load file

```
void readfile() {
    FILE *fp;
    fp = fopen("xentia.inp", "r");
```

```
    if (fp == NULL) {
        printf("File not found or cannot be opened.\n");
        return;
    }

    // Read clipping window coordinates
    if (fscanf(fp, "%d %d %d %d", &pt[0].x, &pt[0].y, &pt[1].x,
&pt[1].y) != 4) {
        printf("Error reading coordinates of clipping window.\n");
        fclose(fp);
        return;
    }

    // Draw the clipping window
    setcolor(RED);
    rectangle(pt[0].x, pt[0].y, pt[1].x, pt[1].y);
    poly[0]={poly[0].x,poly[0].y};
    poly[1]={poly[1].x,poly[1].y};
    hoandoi();
    for (int i=2; i<=n-2 ; i+=2)
    {
        tohog(pt[i],pt[i+1]);
        for (int j=1; j<=4; j++)
        {
            code1[j] = 0;
            code2[j] = 0;
        }
    }

    // Read the number of vertices
    if (fscanf(fp, "%d", &n1) != 1) {
        printf("Error reading the number of vertices.\n");
        fclose(fp);
        return;
    }

    // Read and draw
    for (int i = 0; i < n1; i++) {
```

```
        if (fscanf(fp, "%d %d", &poly[i].x, &poly[i].y) != 2) {
            printf("Error reading coordinates of vertex %d.\n", i +
1);
            fclose(fp);
            return;
        }
        printf("\nToa do thu %d: (%d, %d)\n", i + 1, poly[i].x,
poly[i].y);
    }
    setcolor(GREEN);
    for (int i = 0; i < n1; i++) {
        int k = (i + 1) % n1;
        line(poly[i].x, poly[i].y, poly[k].x, poly[k].y);
    }
    addpointwd();
    Cohen_Sutherland();
    fclose(fp);
}
```

A7. Code giao diện nhập tay

```
void MenuXenTia() {
    printf("Chon thu thuat xen tia:\n");
    printf("1. Cohen-Sutherland\n");
    printf("2. Liang-Barsky\n");
}
```

```
void KhoiTao()

{
    printf("\nNHAP PHAI THOA MAN DIEU KIEN \n|| 80<X<550 && 10<Y<370
||\n");
    do
    {
        printf("Nhap hinh chu nhat Clipping\n");
        printf("xmin = "); scanf("%d",&pt[0].x);
        printf("ymin = "); scanf("%d",&pt[0].y);
        printf("xmax = "); scanf("%d",&pt[1].x);
        printf("ymax = "); scanf("%d",&pt[1].y);
    }
    while(80 >= pt[0].x || 550<=pt[0].x  ||  10>=pt[0].y || 370 <=
pt[0].y || 80 >= pt[1].x || 550<=pt[1].x  ||  10>=pt[1].y || 370 <=
pt[1].y);
    //Ve cua so cut
    setcolor(GREEN);
    rectangle(pt[0].x,pt[0].y,pt[1].x,pt[1].y);
    printf("nhap so canh cua da giac :");
    scanf("%d",&n1);
    int i = 0;
    while(i<n1)
    {   printf("Nhap toa do dinh thu %d\n",i+1);
        printf("x = "); scanf("%d",&poly[i].x);
        printf("y = "); scanf("%d",&poly[i].y);
        if(80<poly[i].x && 550>poly[i].x  &&  10<poly[i].y && 370 >
poly[i].y)
        {
            i++;
        }
        else {
            printf(" diem ban vua nhap khong nam trong cua so , vui
long nhap lai \n");
        }
    }
    for (int i=0; i<n1; i++)
    {   int k=(i+1)%n1;
```

```
            line(poly[i].x,poly[i].y,poly[k].x,poly[k].y);
        }
    }

    void keyboard(){
        MenuXenTia();
        int choice;
        printf("Nhap lua chon: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                KhoiTao();
                addpointwd();
                Cohen_Sutherland();
                break;
            case 2:
                KhoiTao();
                addpointwd();
                Liang_Barsky(p1,p2);
                break;
            default:
                printf("Lua chon khong hop le.\n");
            }
    }
```

A8. Code giao diện đồ họa sử dụng chuột

```
void Graphic(){

    int size_x = 50, size_y = 50, space = 30;
    int x1_line = 15, y1_line = 150, x2_line = 65, y2_line = 200;
    int x1_liang =570, y1_liang =130, x2_liang =620, y2_liang = 180;
    int x1_cohen =570, y1_cohen =230, x2_cohen =620, y2_cohen = 280;
    int x1_del = 15, y1_del = 250, x2_del = 65, y2_del = 300;
    int x1_bk = 580, y1_bk = 10, x2_bk = 680, y2_bk = 50;
    int x1_boder =95, y1_boder =90, x2_boder =550, y2_boder =370;
```

```
Button btLiang = {x1_liang, y1_liang, x2_liang, y2_liang};
Button btLine = {x1_line, y1_line, x2_line, y2_line};
Button btCohen = {x1_cohen, y1_cohen, x2_cohen, y2_cohen};
Button btboder = {x1_boder, y1_boder, x2_boder, y2_boder};
Button btDel = {x1_del, y1_del, x2_del, y2_del};
Button btback = {x1_bk, y1_bk, x2_bk, y2_bk};
Button btcolor[16];

button_liang(btLiang, 2 , 15);
button_line(btLine, 2 , 15);
button_cohen(btCohen, 2 , 15);
boder(btboder, 2);
button_back(btback,2,15);
button_del(btDel, 2 , 15);

//----------------------Ve cac button color
int x1 = 120, y1 = 400, x2 = x1 + 30, y2=y1+30;
int spcolor = 12;
for (int i = 1 ;i<7; i++)
{
    btcolor[i] = {x1, y1, x2, y2};
    bt_color(btcolor[i],i);
    x1+= 60+spcolor; x2+=60+spcolor;
}

//------------------theo tac tren giao dien chinh
while(1)
{
    delay(0.1);
    if (ismouseclick(WM_MOUSEMOVE))
    {
        int x,y;
        char s[100];
        getmouseclick(WM_MOUSEMOVE, x, y);
        point p = {x,y};
        //Hover button Liang
        if (check_button(btLiang,p))
        {
```

```
        setcolor(15);
        button_liang(btLiang, 5 , 3);
        temp3=1;
    }
    else if (!check_button(btLiang, p) && temp3==1)
    {
        setcolor(15);
        button_liang(btLiang, 2 , 15);
        temp3=0;
    }
    //Hover button Line
    if (check_button(btLine,p))
    {
        setcolor(15);
        button_line(btLine, 5 , 3);
        temp=1;
    }
    else if (!check_button(btLine, p)&& temp==1 )
    {
        setcolor(15);
        button_line(btLine, 2 , 15);
        temp=0;
    }
    //Hover button Cohen
    if (check_button(btCohen,p))
    {
        setcolor(15);
        button_cohen(btCohen, 5 , 3);
        temp1=1;
    }
    else if (!check_button(btCohen, p) && temp1==1 )
    {
        setcolor(15);
        button_cohen(btCohen, 2 , 15);
        temp1=0;
    }
    //Hover button Del
    if (check_button(btDel,p))
```

```
        {
            setcolor(15);
            button_del(btDel, 5 , 3);
            temp2=1;
        }
        else if (!check_button(btDel, p) && temp2==1)
        {
            setcolor(15);
            button_del(btDel, 2 , 15);
            temp2=0;
        }
        //Trang thai con tro?
        setcolor(15);
        sprintf(s,"\n Toa Do (%d,%d)",x,y);
        outtextxy(300,40,s);
    }

    if (ismouseclick(WM_LBUTTONUP))
    {
        int x,y;
        char s[100];
        getmouseclick(WM_LBUTTONUP, x, y);
        clearmouseclick(WM_LBUTTONUP);
        clearmouseclick(WM_LBUTTONDOWN);
        point p = {x,y};
        if (check_button(btLine,p))
        {
            while(1)
            {
                delay(0.1);
                point p1,p2;
                if (ismouseclick(WM_MOUSEMOVE))
                {
                    getmouseclick(WM_MOUSEMOVE, x, y);
                    clearmouseclick(WM_MOUSEMOVE);
                    setcolor(15);
                    printf(s,"\n Toa Do (%d,%d) \n",x,y);
                    outtextxy(300,40,s);
```

```
            }

            if (ismouseclick(WM_LBUTTONDOWN))
            {
                getmouseclick(WM_LBUTTONDOWN, x, y);
                clearmouseclick(WM_LBUTTONDOWN);
                p1 = {x,y};
                if (!check_button(btboder,p1)) break;
            }

            if(ismouseclick(WM_LBUTTONUP))
            {
                getmouseclick(WM_LBUTTONUP, x, y);
                clearmouseclick(WM_LBUTTONUP);
                p2 = {x,y};
                if (!check_button(btboder,p2)) break;
                setcolor(color);
                line(p1.x,p1.y,p2.x,p2.y);
                pt[n]={p1.x,p1.y};
                pt[n+1]={p2.x,p2.y};
                n+=2;
            }
        }
    }

    if (check_button(btDel,p))
    {
        cleardevice();
        Graphic();
    }

    if (check_button(btCohen,p))
    {
        while(1)
        {
            delay(0.01);
            point p1,p2;
            if (ismouseclick(WM_MOUSEMOVE))
```

```
{
    getmouseclick(WM_MOUSEMOVE, x, y);
    clearmouseclick(WM_MOUSEMOVE);
    setcolor(15);
    sprintf(s,"\t Toa Do (%d,%d) \t",x,y);
    outtextxy(300,40,s);
}
if (ismouseclick(WM_LBUTTONDOWN))
{
    getmouseclick(WM_LBUTTONDOWN, x, y);
    clearmouseclick(WM_LBUTTONDOWN);
    p1 = {x,y};
    if (!check_button(btboder,p1)) break;
}

if(ismouseclick(WM_LBUTTONUP))
{
    getmouseclick(WM_LBUTTONUP, x, y);
    clearmouseclick(WM_LBUTTONUP);
    p2 = {x,y};
    if (!check_button(btboder,p2))
    break;
    setcolor(color);
    setlinestyle(0,0,0);
    rectangle(p1.x,p1.y,p2.x,p2.y);
    pt[0]={p1.x,p1.y};
    pt[1]={p2.x,p2.y};
    hoandoi();
    for (int i=2; i<=n-2 ; i+=2)
    {
        tohog(pt[i],pt[i+1]);
        for (int j=1; j<=4; j++)
        {
            code1[j] = 0;
            code2[j] = 0;
        }
    }
    printf("tn1 = %d\n",n1);
```

```
                addpointwd();
                Cohen_Sutherland() ;
            }
        }
    }

    if (check_button(btLiang,p))
    {
        while(1)
        {
            delay(0.01);
            point p1,p2;
            if (ismouseclick(WM_MOUSEMOVE))
            {
                getmouseclick(WM_MOUSEMOVE, x, y);
                clearmouseclick(WM_MOUSEMOVE);
                setcolor(15);
                sprintf(s,"\t Toa Do (%d,%d) \t",x,y);
                outtextxy(300,40,s);
            }
            if (ismouseclick(WM_LBUTTONDOWN))
            {
                getmouseclick(WM_LBUTTONDOWN, x, y);
                clearmouseclick(WM_LBUTTONDOWN);
                p1 = {x,y};
                if (!check_button(btboder,p1)) break;
            }
            if(ismouseclick(WM_LBUTTONUP))
            {
                getmouseclick(WM_LBUTTONUP, x, y);
                clearmouseclick(WM_LBUTTONUP);
                p2 = {x,y};
                if (!check_button(btboder,p2))
                break;
                setcolor(color);
                setlinestyle(0,0,0);
                rectangle(p1.x,p1.y,p2.x,p2.y);
                pt[0]={p1.x,p1.y};
```

```
                        pt[1]={p2.x,p2.y};
                        hoandoi();
                        for (int i=2; i<=n-2 ; i+=2)
                        {
                            tohog(pt[i],pt[i+1]);
                            for (int j=1; j<=4; j++)
                            {
                                code1[j] = 0;
                                code2[j] = 0;
                            }
                        }
                        printf("tn1 = %d\n",n1);
                        addpointwd();
                        Liang_Barsky(p1,p2) ;
                    }
                }
            }

            for (int i=1; i<7; i++)
            {
                if(check_button(btcolor[i],p))
                {
                    color = i;
                    break;
                }
            }
            if(check_button(btback,p))
            {
                cleardevice();
                goto out;
            }
        }
    }
    out:
    cleardevice();
}
```

A9. Chương trình chính

```
int main(){
    //draw interface
    initwindow(700,500);
    GUI();
    // Using mouse
    clickmouse();
    getch();
}
```