TRƯỜNG ĐẠI HỌC NHA TRANG

**KHOA CÔNG NGHỆ THÔNG TIN**



**CÀI ĐẶT THUẬT TOÁN XÉN TỈA ĐA GIÁC BẰNG**

**THUẬT TOÁN SUTHERLAND - HODGMAN**

**GVHD : ĐOÀN VŨ THỊNH**

**SVTH : Trần Đinh Trí**

**MSSV : 60131750**

**Lớp : 60.CNTT-1**

Khánh Hòa,ngày 17 tháng 01 năm 2021

Mục lục

[I. GIỚI THIỆU ĐỀ TÀI 3](#_Toc62451350)

[1.1. Thuật toán xén tỉa 3](#_Toc62451351)

[1.2. Thuật toán xén tỉa đoạn thẳng 4](#_Toc62451353)

[1.2.1 Thuật toán Cohen-Sutherland 5](#_Toc62451356)

[1.2.2 Thuật toán Cyrus-beck 7](#_Toc62451359)

[1.3 Thuật toán xén tỉa đa giác 8](#_Toc62451361)

[II) PHƯƠNG PHÁP NGHIÊN CỨU 12](#_Toc62451366)

[2.1. Nhập dữ liệu 12](#_Toc62451367)

[2.1.1. Đoạn thẳng nằm trên biên trái cửa sổ sắt : 13](#_Toc62451368)

[2.1.2. Đoạn thẳng nằm trên biên phải cửa sổ xén 15](#_Toc62451369)

[2.1.3 Đoạn thẳng nằm trên biên trên cửa sổ cắt 16](#_Toc62451370)

[2.1.4 Đoạn thẳng nằm trên biên dưới cửa sổ cắt 17](#_Toc62451371)

[2.1.5 Đoạn thẳng nằm ngoài cửa sổ xén 18](#_Toc62451372)

[III) KẾT QUẢ 18](#_Toc62451373)

[IV. THẢO LUẬN 19](#_Toc62451374)

# I. GIỚI THIỆU ĐỀ TÀI

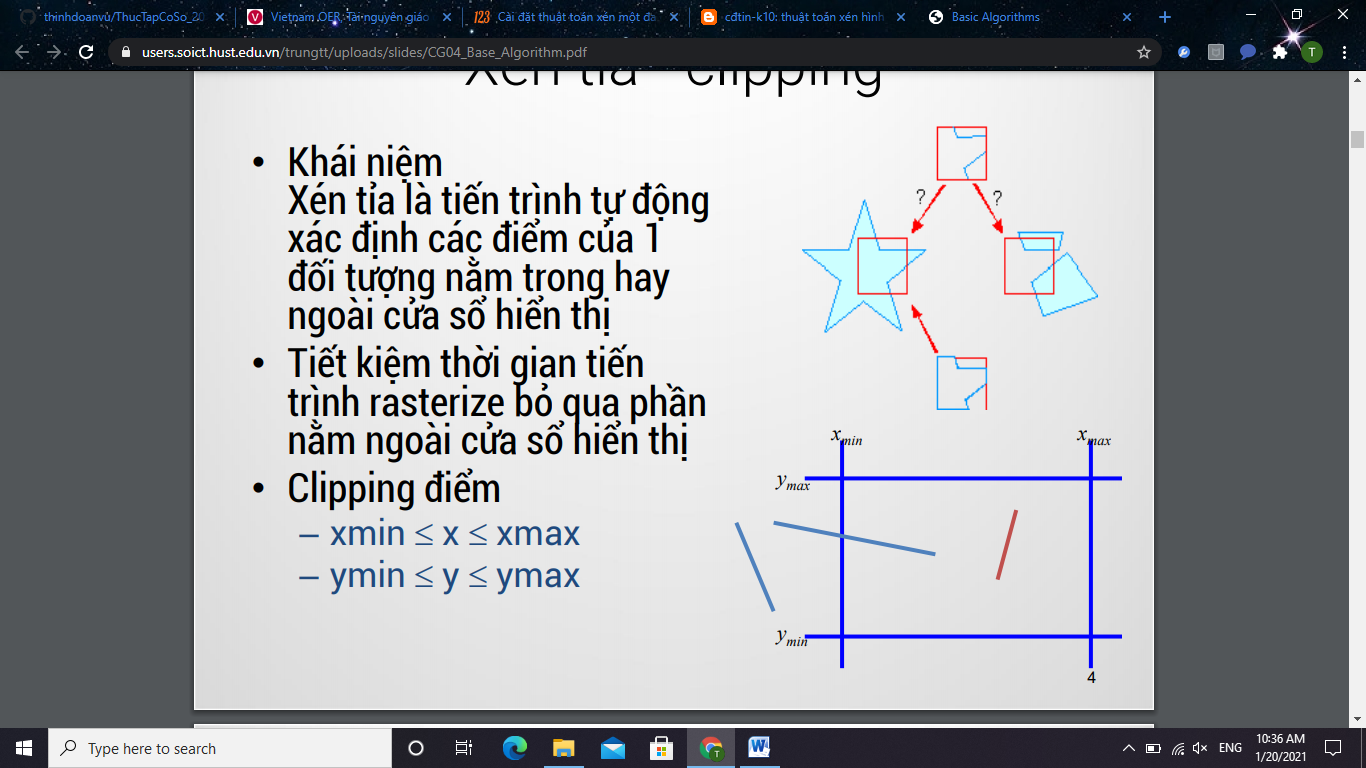
Cùng với sự phát triển kinh tế hiện nay là sự phất triển mạnh mẽ của ngành công nghệ thông tin. Các ứng dụng của công nghệ thông tin ngày càng nhiều và có mặt hầu hết trong các lĩnh vực cuộc sống.

Với đề tài : **“Cài đặt thuật toán xén tỉa đa giác bằng thuật toán Sutherland-Hogman”** , ý nghĩa của thuật toán xén hình trong công nghệ thông tin là không hề nhỏ, đặc biệt đối với màn hình máy tính thì nó có ý nghĩa vô cùng quan trọng.

Qua đề tài này em mong sẽ giúp ích được cho mọi người hiểu thêm về thuật toán xén hình.

## 1.1. Thuật toán xén tỉa

Xén tỉa là tiến trình xác định các điểm của một đối tượng nằm trong hay ngoài cửa sổ hiển thị. Nằm trong được hiển thị, nằm ngoài loại bỏ. Việc loại từng điểm ảnh của đối tượng thường chậm nhất là khi đối tượng mà phần lớn nằm ngoài cửa sổ hiển thị.



Hình 1.1. Ví dụ minh họa cho thuật toán xén tỉa đa giác

Clipping điểm

xmin ≤ x ≤ xmax

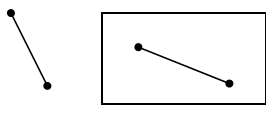
ymin ≤ y ≤ ymax



*Hình 1.2. Ví dụ minh họa*

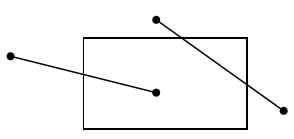
Nhưng trước hết muốn xén tỉa một đa giác thì chúng ta nên tìm hiểu cách xén tỉa một đoạn thẳng bởi vì xén tỉa đoạn thẳng là nền tảng để chúng ta đi xén tỉa một đa giác.

## 1.2. Thuật toán xén tỉa đoạn thẳng

****Các đoạn thẳng không cắt cửa sổ thị: nằm trong hoàn toàn, hoặc ngoài hoàn toàn.

Hình 1.3. Các đoạn thẳng nằm trong hoặc nằm ngoài

Nếu đoạn thẳng cắt cửa sổ thì phân chia qua điểm cắt phần nằm trong và phần nằm ngoài



Hình 1.4. Các đoạn thẳng cắt cửa sổ

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

**TH1:** Hiển thị (visible): cả hai đầu cuối của đoạn thẳng đều nằm bên trong cửa sổ.

**TH2:** 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

Thao tác xén hình là một trong nhưng thao tác cơ bản của quá trình hiển thị đối tượng. Ý tưởng chung của thao tác xén tỉa đoạn thẳng đó là phép toán tìm giao điểm giữa đoạn thẳng với biên. Với các đoạn thẳng đặt biệt như nằm trong hoàn toàn hoặc nằm ngoài hoàn toàn cửa sổ (ví dụ như là hình 1.3, 1.4). Đối với các đoạn thẳng cắt cửa sổ thì phải tìm giao điểm.

Người ta thường sử dụng phương trình tham số của đoạn thẳng trong việc tìm giao điểm của đoạn thẳng với cửa sổ.

X = x1+t( x2-x1) = x1 + tDx1, Dx = x2-x1

Y = y1+t( y2-y1) = y1 + tDy1, Dy = y2-y1 0<=t<=1

1.2.1 Thuật toán Cohen-Sutherland

Đây là một trong những thuật toán ra đời sớm nhất. Bằng cách kéo dài các biên của cửa sổ, người ta chia mặt phẳng thành chín vùng gồm cửa sổ và 8 vùng xung quanh.

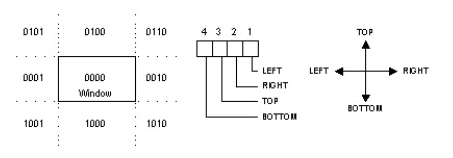
**Khái niệm mã vùng:** mã vùng là một con số 4 bit nhị phân sẽ được gán cho mỗi vùng để mô tả vị trí tương đối của vùng đó so với cửa sổ. Bằng cách đánh số từ 1 đến 4 theo thứ tự từ phải sang trái, các bit theo mã vùng được dùng theo quy ước sau để chỉ một trong bốn vị trí tương đối của vùng so với cửa sổ bao gồm: trái, phải, trên, dưới.

**Bit 1:** trái (LEFT)

**Bit 2:** phải (RIGHT)

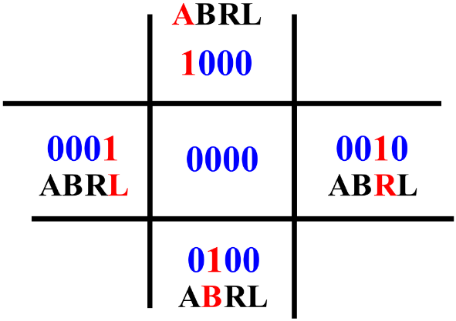
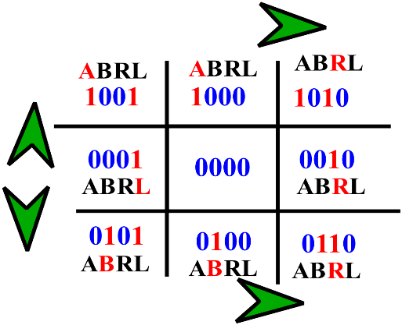
**Bit 3:** trên (TOP)

**Bít 4:** dưới (BOTTOM)



Hình 1.5. Mã vùng quy định vị trí tương đối của vùng so với cửa sổ

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



Hình 1.6. Mã vùng được xác định theo 9 vùng

**Các trường hợp xén:**

|  |  |
| --- | --- |
| ***Trường hợp 1:*** Đoạn thẳng có 2 điểm hoàn toàn nằm trong cửa sổ nên **không cần xén.** |  |
| ***Trường hợp 2:*** Đoạn thẳng có 2 điểm cùng nằm ngoài về một phía ngoài của cửa sổ và sẽ bị xén mất. |  |
| ***Trường hợp 3:*** Nếu đoạn thẳng cắt biên cửa sổ bên trái thì tọa độ tại điểm đó được xác định như sau: |  |
| ***Trường hợp 4:*** Nếu đoạn thẳng cắt biên cửa sổ bên phải thì tọa độ tại điểm đó được xác định như sau: |  |
| ***Trường hợp 5:*** Nếu đoạn thẳng cắt biên cửa sổ bên trên thì tọa độ tại điểm đó được xác định như sau: |  |
| ***Trường hợp 6:*** Nếu đoạn thẳng cắt biên cửa sổ bên dưới thì tọa độ tai điểm đó được xác định như sau: |  |

**Xác định chiều đoạn thẳng:**

**TH1: đoạn thẳng cắt biên bên trái:** Nếu xP1 <= xwmin và xP2 >= xwmin  thì đoạn thẳng có chiều từ ngoài vào trong. Ký hiệu OUT2IN.

**TH2: đoạn thẳng cắt biên bên phải:** Nếu xP1 > xwmax và xP2 <= xwmax  thì đoạn thẳng có chiều từ ngoài vào trong. Ký hiệu OUT2IN.

**TH3: đoạn thẳng cắt biên bên trên:** Nếu yP1 > ywmax và yP2 <= ywmax  thì đoạn thẳng có chiều từ ngoài vào trong. Ký hiệu OUT2IN.

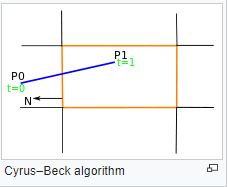
**TH4: đoạn thẳng cắt biên bên dưới:** Nếu yP1 < ywmin và yP2 >= ywmin  thì đoạn thẳng có chiều từ ngoài vào trong. Ký hiệu OUT2IN.

1.2.2 Thuật toán Cyrus-beck

Thuật toán Cyrus-Beck là thuật toán cắt dòng tổng quát. Nó được thiết kế để hiệu quả hơn thuật toán Cohen-Sutherland, sử dụng cắt lặp đi lặp lại. Cyrus-Beck là một thuật toán tổng quát và có thể được sử dụng với cửa sổ cắt đa giác lồi, không giống như Cohen-Sutherland, chỉ có thể được sử dụng trên một khu vực cắt hình chữ nhật.

Ở đây phương trình tham số của một đường trong mặt phẳng:





Hình 1.7. Cửa sổ cắt tỉa đoạn thẳng

Ngoài ra còn có còn có nhiều thuật toán để xén tỉa đa giác khác nhau như: Nicholl-Lee-Nicholl, Fast Clipping , O(lg N), Skala, See also, references,… Các thuật toán xén tỉa này cũng được cải thiện đáng kể về độ chính xác và thời gian xử lí. Xén tỉa đa giác là một trong những phần mở rộng của thuật toán xén tỉa đoạn thẳng. Có thể xem tại đường link: ( <https://en.wikipedia.org/>).

## 1.3 Thuật toán xén tỉa đa giác

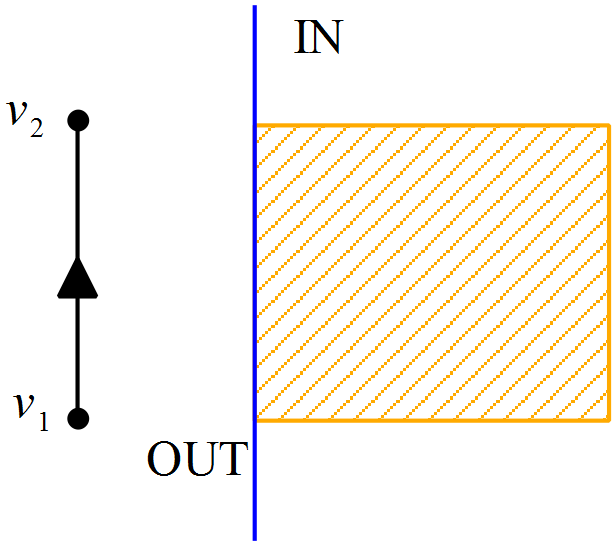
[Ivan Sutherland](https://en.wikipedia.org/wiki/Ivan_Sutherland) và Gary W. Hodgman (1974) đề xuất thuật toán xén tỉa đa giác bằng cách lần lượt mở rộng từng đoạn thẳng của *đa giác lồi* và chỉ chọn các đỉnh từ đa giác cần xén nằm ở phía có thể nhìn thấy. Hạn chế của thuật toán này là nếu đa giác cần xén là đa giác lõm, đa giác mới có thể có các cạnh trùng nhau.

Qui ước xác định chiều các điểm của đa giác theo chiều kim đồng hồ như sau:

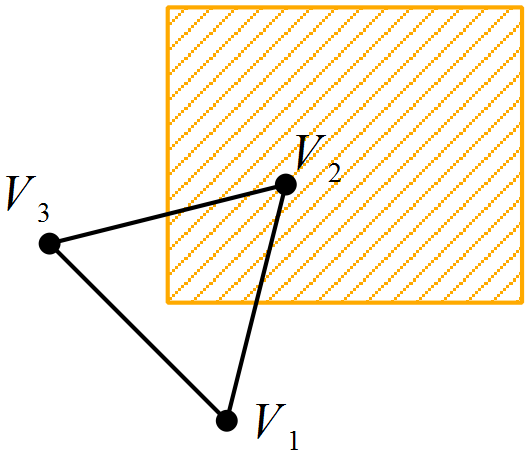
Theo qui ước: một đa giác với các đỉnh , ....., (các cạnh là và ) được gọi là theo hướng dương nếu các hình theo thứ tự đã cho tạo thành mạch ngược chiều kim đồng hồ.

Nếu bàn tay dọc theo bất kỳ cạnh hoặc cũng chỉ về bên trong đa giác.

Quy tắc khác: Tính tổng các cạnh của đa giác: (x2 − x1)(y2 + y1). Nếu kết quả cho số dương thì chiều của đa giác thuận chiều kim đồng hồ và ngược lại.

***Các trường hợp trong giải thuật hogman:***

|  |  |
| --- | --- |
| **TH1:** Nếu đỉnh v1 nằm bên ngoài và đỉnh v2 nằm bên trong của cạnh và có chiều từ trái sang phải thì giao điểm v’1 và v2 được lưu lại. |  |
| **TH2:** Nếu đỉnh v1 nằm bên trong và đỉnh v2 nằm bên ngoài của cạnh và có chiều từ phải sang trái thì giao điểm v’1 và v1 được lưu lại. |  |
| **TH3:** Nếu cả 2 đỉnh v1 và v2 cùng nằm về phía bên trong của cạnh thì cả 2 đỉnh v1 và v2 **đều được** lưu lại. |  |
| **TH4:** Nếu cả 2 đỉnh v1 và v2 cùng nằm về phía bên ngoài của cạnh thì cả 2 đỉnh v1 và v2 **không được** lưu lại. |  |



**Ví dụ minh họa:**Cho đa giác dương sau và cửa sổ cắt, hãy xác định các đỉnhcắt:

|  |  |
| --- | --- |
| ***Bước 0:*** Áp dụng quy tắc bàn tay trái cho đa giác dương để xác định chiều của các cạnh. |  |
| ***Bước 1: Cắt bên trái***  Xét cạnh V1-V2 (In-In): V2.  Xét cạnh V2-V3 (In-Out): V’2.  Xét cạnh V3-V1(Out-In): V3-V3’. |  |
| ***Bước 2. Cắt bên phải***  Xét cạnh V1-V2 (In-In): V2.  Xét cạnh V2-V’2 (In-In): V’2.  Xét cạnh V’2-V’3 (In-In): V’3.  Xét cạnh V’3-V1(In-In): V1. |  |
| ***Bước 3. Cắt bên trên***  Xét cạnh V1-V2 (In-In): V2.  Xét cạnh V2-V’2 (In-In): V’2.  Xét cạnh V’2-V’3 (In-In): V’’3.  Xét cạnh V’3-V1(Out-Out): NULL. |  |
| ***Bước 4. Cắt bên dưới***  Xét cạnh V1-V2 (Out-In): V’1V2.  Xét cạnh V2-V’2 (In-In): V’2.  Xét cạnh V’2-V’3 (In-Out): V’’2.  Xét cạnh V’3-V1(Out-Out): NULL. |  |

**Ví dụ 2:**

|  |  |
| --- | --- |
| Cho đa giác ABCD với tọa độ lần lượt là (20,200), (120,350), (300,220), (120,80) và cửa sổ cắt xyzt với tọa độ lần lượt là (50,120), (180,120), (180,280), (50,280). |  |

**Bước 1: Ta xét cạnh bên trái**

**Cạnh AB**: nằm ngoài cửa sổ theo chiều Out-In

Tìm giao điểm A’ thuộc đường thẳng AB

Vậy A’(50,245)

**Cạnh DA**: nằm ngoài cửa sổ theo chiều In-Out

Tìm giao điểm A” thuộc đường thẳng DA

Vậy A”(50,164)

**Bước 2: Ta xét cạnh bên phải**

**Cạnh BC:** nằm ngoài hoàn toàn cửa sổ nên ta không tạo ra điểm mới cho đa giác sau khi xén tỉa

**Cạnh CD:** nằm ngoài cửa sổ và chiều Out-In

Tìm giao điểm C’ thuộc đường thẳng CD

Vậy C’(180,)

**Bước 3: Ta xét cạnh bên trên**

**Cạnh AB**: nằm ngoài cửa sổ chiều In-Out

Tìm giao điểm B’ thuộc đường thẳng AB

Vậy B’()

**Cạnh BC:** nằm ngoài hoàn toàn cửa sổ cắt nên sẽ không tạo điểm mới

**Bước 4: Xét cạnh bên dưới**

**Cạnh DA**: nằm ngoài cửa sổ và chiều Out-In

Tìm giao điểm D’ thuộc đường thẳng DA

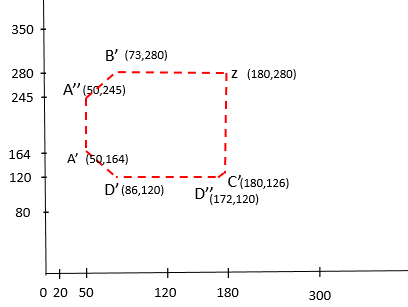
Vậy D’(, 120)

**Cạnh CD:** nằm ngoài cửa sổ và chiều In-Out

Tìm giao điểm D” thuộc đường thẳng CD

Vậy D”(, 120)

Sau khi thực hiện xong ta được đa giác mới A’’B’zC’D”D’A’ có tọa độ lần lượt là (50,245), (73,280), (180,280), (180,126), (172,120), (86,120), (50,164)

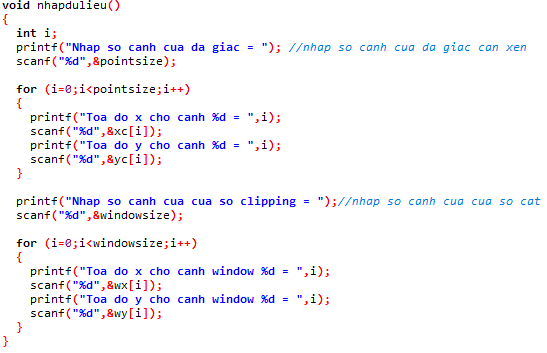


[Vatti](https://en.wikipedia.org/wiki/Vatti_clipping_algorithm) được phát triển vào năm 1992 bởi Bala R. vatti, thuật toán giải quyết hạn chế của các loại đa giác có thể được sử dụng làm đối tượng hoặc clip. Ngay cả các đa giác phức tạp (tự giao nhau) và đa giác lõm có thể được xử lý (<https://en.wikipedia.org/wiki/Vatti_clipping_algorithm>).

**II) PHƯƠNG PHÁP NGHIÊN CỨU**

## 2.1. Nhập dữ liệu

Để thực hiện các thao tác xén tỉa đa giác thì trước hết cần nhập dữ liệu các tọa độ điểm của cạnh cần vẽ và cửa sổ cắt.



Hình 2.1. Nhập tọa độ đa giác và cửa sổ cẳt

Sau khi nhập dữ liệu tạo độ các điểm có thể rơi vào những trường hợp như sau:

### 

### 2.1.1. Đoạn thẳng nằm trên biên trái cửa sổ sắt :

|  |  |
| --- | --- |
| Trường hợp 1: đoạn thẳng P1P2 nằm trên biên trái cửa sổ xén có chiều từ ngoài vào trong. | Hình 2.2. Ảnh minh họa TH1 |
| Hình 2.3. Code minh họa cho trường hợp 1 | |
| Trường hợp 2 : đoạn thẳng đoạn thẳng P1P2 nằm trên biên trái cửa sổ xén có chiều từ trong ra ngoài. | Hình 2.4. Ảnh minh họa TH2 |
| Hình 2.5. Code minh họa cho trường hợp 1 | |

Hình 2.6. Code minh họa cho trường hợp 2

### 2.1.2. Đoạn thẳng nằm trên biên phải cửa sổ xén

|  |  |
| --- | --- |
| **Trường hợp 1**: đoạn thẳng P1P2 nằm trên biên phải cửa sổ xén có chiều từ trong ra ngoài | Hình 2.6. Ảnh minh họa TH1 |
| Hình 2.7. Code minh họa cho TH1 | |
| **Trường hợp 2:** đoạn thẳng P1P2 nằm trên biên phải cửa sổ xén có chiều từ trong ra ngoài. | Hình 2.8. Ảnh minh họa TH2 |
| Hình 2.9. Code minh họa cho TH2 | |

### 2.1.3 Đoạn thẳng nằm trên biên trên cửa sổ cắt

|  |  |
| --- | --- |
| **Trường hợp 1**: đoạn thẳng P1P2 nằm trên biên trên cửa sổ xén có chiều từ trong ra ngoài | Hình 2.10. Ảnh minh họa TH1 |
| Hình 2.11. Code minh họa cho TH1 | |
| **Trường hợp 2**: đoạn thẳng P1P2 nằm trên biên trên cửa sổ xén có chiều từ ngoài vào trong | Hình 2.12. Ảnh minh họa TH2 |
| Hình 2.13. Code minh họa cho TH2 | |

### 2.1.4 Đoạn thẳng nằm trên biên dưới cửa sổ cắt

|  |  |
| --- | --- |
| **Trường hợp 1**: đoạn thẳng P1P2 nằm trên biên dưới cửa sổ xén có chiều từ ngoài vào trong | Hình 2.14. Ảnh minh họa TH1 |
| Hình 2.15. Code minh họa cho TH1 | |
| **Trường hợp 2**: đoạn thẳng P1P2 nằm trên biên dưới cửa sổ xén có chiều từ trong ra ngoài | Hình 2.16. Ảnh minh họa TH1 |
| Hình 2.17. Code minh họa cho Th2 | |

### 2.1.5 Đoạn thẳng nằm ngoài cửa sổ xén

|  |  |
| --- | --- |
| Hình 2.18. Code minh họa | Hình 2.19. Ảnh minh họa |

2.1.6 Đoạn thẳng nằm trong cửa sổ xén

|  |  |
| --- | --- |
| Hình2.20. Code minh họa | Hình 2.21. Ảnh minh họa |

# III) KẾT QUẢ

|  |  |  |
| --- | --- | --- |
| Hình A | Hình B | Hình C |

# Hình A : Trường hợp đa giác giao biên cửa sổ cắt với hình màu xanh là cửa sổ cắt , hình màu đỏ là đa giác và hình màu vàng là kết quả sau khi cắt.

# Hình B : Trường hợp đa giác nằm ngoài cửa sổ cắt với hình màu xanh là cửa sổ cắt và hình màu đỏ là đa giác.

# Hình C : Trường hợp đa giác nằm trong cửa sổ cắt với hình màu xanh là cửa sổ cắt và hình màu vàng vừa là đa giác vừa là kết quả sau khi cắt.

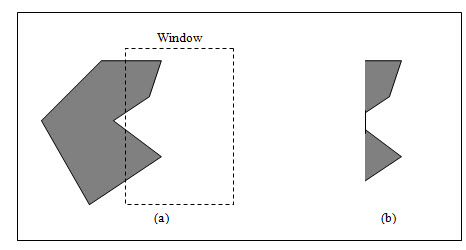
# IV. THẢO LUẬN

Thuật toán xén tỉa đa giác **Sutherland-Hodgman** ra đời từ rất sớm và kế thừa những ưu điểm và cải tiến những hạn chế từ những thuật toán, giải thuật xén tỉa trước đó nên nó rất phổ biến và hữu ích để cắt đa giác

Bên cạnh những thuận lợi, ưu điểm mà thuật toán Sutherland-Hodgman đem lại trong việc xén tỉa thì nó cũng có những hạn chế, nhược điểm chưa được khắc phục như đòi hỏi một lượng bộ nhớ đáng kể. Kết quả sau khi thuật toán này thực hiện được lưu trữ trong bộ nhớ. Vì vậy, lãng phí bộ nhớ để lưu trữ đa giác trung gian

Và điểm hạn chế thứ hai chính là vấn đề của bài toán trên, đối với đa giác lõm thì thuật toán Sutherland-Hodgman thì khi cắt nó có thể có các cạnh trùng nhau, chồng chéo nhau.

Trong ví dụ dưới đây ta thấy sau khi thực hiện việc cắt tỉa đa giác ở hình a) thì ta được hai đa giác tách biệt nối với nhau bằng một đoạn thẳng ở hình b) nhưng đoạn thẳng được nối ở giữa là dư thừa



*Hình 4.1. Hạn chế của thuật toán Sutherland-Hodgman*

*Nguồn:* [*https://voer.edu.vn/*](https://voer.edu.vn/)

# Và thuật toán [Weiler–Atherton](https://en.wikipedia.org/wiki/Weiler%E2%80%93Atherton) ra đời và khắc phục được điểm hạn chế này bằng cách trả về một tập hợp các đa giác được chia, nhưng phức tạp hơn và đắt hơn về mặt tính toán và thuật toán này chưa được thực hiện trên máy tính. Trong tương lai thuật toán sẽ được cải thiện và thực hiện tối ưu hơn.