

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC



BÁO CÁO MÔN HỌC

Môn học: Mô hình mô phỏng

**ĐỀ TÀI: MÔ PHỎNG DÒNG CHẢY MỘT ĐOẠN
SÔNG CỬU LONG**

<i>Giảng viên hướng dẫn:</i>	Tạ Thị Thanh Mai
<i>Họ và tên sinh viên:</i>	Nguyễn Lâm Tùng
<i>MSSV:</i>	20154256

Hà Nội – 2019

MỤC LỤC

I. Giới thiệu	2
II. Cơ sở lý thuyết.....	3
2.1. Vận tốc dòng chảy	3
2.2. Đặc điểm chất lưu	7
III. Mô hình và mô phỏng	8
3.1. Mô hình.....	8
3.2. Mô phỏng.....	8
3.2.1. Xây dựng lưới	8
3.2.2. Quản lý biên của lưới.....	15

I. Giới thiệu

- Sông Mekong là con sông dài thứ 12 (~4350 km) và có lưu lượng nước đứng thứ 10 trên Thế Giới, con sông chảy qua 5 nước, bắt đầu từ tỉnh Thanh Hải, Trung Quốc, đi qua Lào, Myanmar, Thái Lan, Campuchia và đổ ra Việt Nam.
- Đoạn sông Mekong đi qua Việt Nam được gọi là Sông Lớn hoặc sông Cái, có 9 nhánh sông đổ ra biển, từ đó người ta còn gọi đó là sông Cửu Long.
- Sông Cửu Long có lưu lượng nước rất lớn, 6000m³/s về mùa khô và lên đến 120,000 m³/s vào mùa mưa, chuyên chở phù sa về đồng bằng Đông Nam Bộ, do đó dòng chảy của sông cũng thay đổi quanh năm, tùy vào lượng mưa cũng như lượng phù sa có trong nước.
- Bài báo cáo sẽ mô phỏng một cách rất đơn giản dòng chảy của một phần nhỏ sông Cửu Long, đó là những nhánh sông ở phía cao nhất phân chia 2 tỉnh Tiền Giang và Bến Tre.



nguồn: Google Maps

II. Cơ sở lý thuyết

Trên thực tế, dòng chảy của chất lưu phụ thuộc vào rất nhiều yếu tố, có thể kể đến một số yếu tố như:

- Địa hình:
 - Độ dốc: theo nguyên lý vật lý tự nhiên, thì nước chảy chỗ trũng, độ dốc có sự tác động lớn đến tốc độ dòng chảy, chỗ nào có độ dốc càng cao, vận tốc sẽ càng lớn.
 - Độ sâu: chỗ nào càng sâu, nước càng chảy chậm, có thể tạo xoáy.
 - Điều kiện biên, bờ: ví dụ dòng chảy trên một mương máng được xây bằng xi măng sẽ khác so với dòng chảy trên một con sông có bờ là đất, đất thấm thấu nước cũng như là nước có thể tràn lên bờ.
- Đặc điểm chất lưu: Tùy từng loại chất lưu mà dòng chảy của chúng sẽ khác nhau. Ví dụ như sữa, dầu thô,... đặc nên có dòng chảy chậm hơn so với nước cất, dầu, benzen,...
- Thời tiết:
 - Lượng mưa ảnh hưởng trực tiếp tới dòng chảy, mưa càng nhiều thì nước chảy càng mạnh, là nguyên nhân gây ra lũ lụt.
 - Các chấn động địa chất dưới lòng sông, hồ.
 - Gió thổi mạnh cũng tác động đến hướng của dòng chảy.
- ...

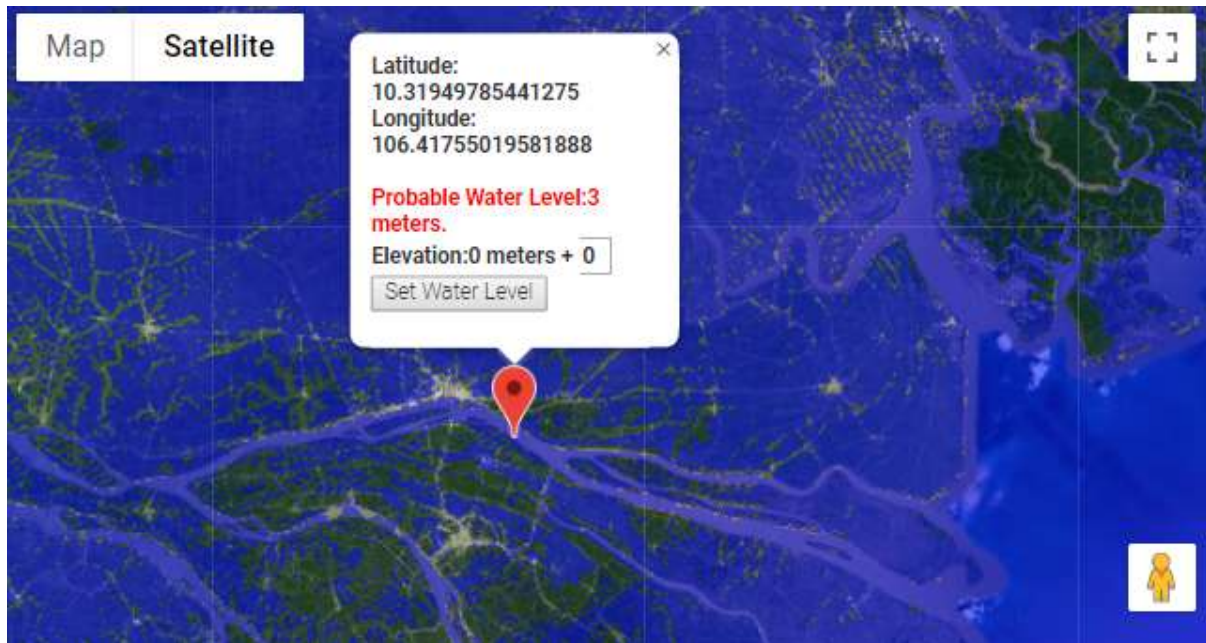
2.1. Vận tốc dòng chảy

Vận tốc dòng chảy trên một đoạn sông có giá trị không cố định, cách xác định cũng rất phức tạp. Nhưng để đơn giản hơn cho việc mô phỏng, ta sẽ xác định vận tốc trung bình của dòng chảy để đại diện cho vận tốc dòng chảy trên cả đoạn sông. Để xác định được vận tốc trung bình của dòng chảy, chúng ta sẽ phải tính thời gian để một vật thể trôi tự do từ điểm đầu đến điểm cuối của đoạn sông, và tính độ dài tương đối của đoạn sông đó.

Thời gian để một vật thể trôi tự do từ đầu này sang đầu kia được tính thông qua các tham số về địa hình, vận tốc ban đầu. Các tham số về địa hình cần cho tính toán đó là độ cao của điểm đầu so với điểm cuối, và độ dài quãng đường phải đi.

- Độ cao:

Chúng ta sẽ xét độ cao của điểm đầu so với điểm cuối của đoạn sông. Điểm cuối của đoạn sông là biển, do đó độ cao của điểm đầu so với điểm cuối sẽ chính là độ cao so với mực nước biển của điểm đầu.



nguồn: floodmap.net

Theo số liệu của floodmap (trang web tính toán dự báo về lũ lụt) thì điểm đầu chúng ta xét là vị trí gần Cù Lao Tam Hiệp có độ cao so với mực nước biển rơi vào khoảng 3m.

- Quãng đường:

Theo như tính toán một cách tương đối dựa trên Google Maps, thì đoạn sông chúng ta xét có độ dài khoảng 43km. Từ đó ta sẽ sử dụng tham số này để tính thời gian trung bình của một vật chảy tự do từ đầu sông này sang đầu sông kia.

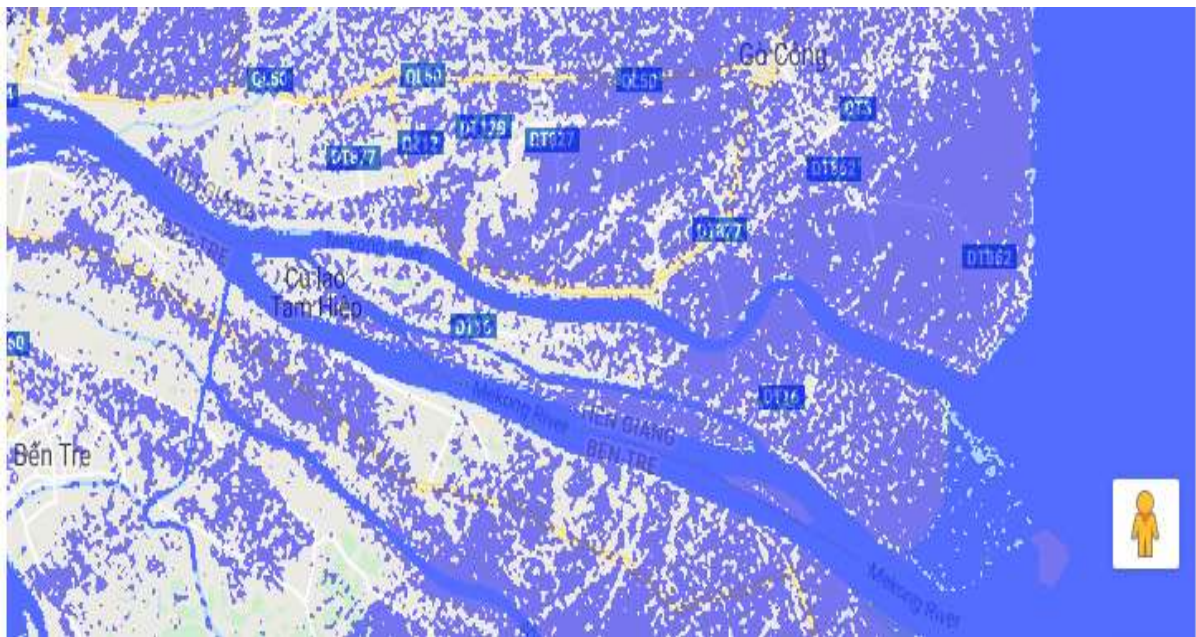


nguồn: Google Maps

- Vận tốc ban đầu:

Nếu coi như vận tốc của nước chỉ phụ thuộc vào độ cao của bờ bên này so với bờ bên kia thì nếu độ cao 2 bên bờ là như nhau, vận tốc nước sẽ bằng 0. Ví dụ như mưa rơi theo phương thẳng đứng, khi tiếp xúc với mặt nghiêng, chỗ nước đó sẽ tự khắc chảy xuống phía nào thấp hơn, còn nếu mưa rơi trên mặt phẳng, thì chỗ nước đó gần như không thay đổi vị trí theo thời gian sau khi rơi xuống.

Do đó ta sẽ xét sự chênh lệch về độ cao trên một đoạn rất nhỏ tại thời điểm ta coi là nơi dòng nước xuất phát. Đó là đoạn nước sông trước khi nó chảy đến Cù Lao Tam Hiệp.



nguồn: *floodmap.net*

Giải thích một chút, trên bản đồ có 2 phần, phần nước và phần đất. Phần đất lại có 2 màu, đó là màu trắng và màu đen, màu trắng là những điểm có độ cao khoảng 3m, còn màu đen là các điểm thấp hơn 3m. Nhìn vào phía trái, đó là chỗ ta coi như là điểm dòng nước xuất phát, mật độ điểm trắng dày đặc, khác hẳn so với phần giữa (đen trắng đan xen) và phần phía bên phải (gần như là đen, nghĩa là vùng trũng). Do đó ta có thể coi như đây là vùng nước rất bằng phẳng, vì vậy ta có thể coi như vận tốc dòng nước ở vùng này là khá nhỏ. Để đơn giản hơn cho việc tính toán, ta sẽ coi vận tốc tại vùng này xấp xỉ 0.

- Thời gian trung bình:

Để tính thời gian trung bình của một vật trôi tự do từ điểm đầu sang điểm cuối, ta có thể tương đương với thời gian trung bình của một vật rơi tự do từ độ cao h nào đó, với gia tốc trọng trường được tính nhờ độ dốc (độ cao địa hình và độ dài đoạn sông).

- Gia tốc: $g_0 = 9.81 \times \sin(\arctan(3/43000)) \approx 0.0006844$
- Quãng đường: vì độ cao 3m rất nhỏ so với chiều dài 43km, do đó ta có thể xấp xỉ quãng đường đi của vật thể là 43km.

$$\Rightarrow \text{thời gian trung bình là } t = \sqrt{\frac{2h}{g_0}} \approx 11209 \text{ (s)}$$

Ta đã tính được thời gian trung bình của một vật thể trôi tự do từ điểm đầu đến điểm cuối, do đó ta sẽ tính được vận tốc trung bình của dòng chảy:

$$v = s/t = 3.836 \text{ m/s}$$

2.2. Đặc điểm chất lưu

Nước sông chủ yếu là nước mưa, mà nước mưa nguyên chất ta coi đó là nước cất. Trong nước sông có một lượng phù sa, nhưng nếu xét về mặt tỉ lệ khối lượng, chúng không thực sự đáng kể (khoảng 0.00002%). Do đó để cho đơn giản, ta coi nước sông là nước cất.

Vùng đồng bằng Đông Nam Bộ có nhiệt độ cao quanh năm, ta có thể lấy nhiệt độ trung bình nơi đây rơi vào khoảng 30°C.

Do đó tham số chúng ta cần sử dụng để mô phỏng sẽ là độ nhớt của nước cất ở nhiệt độ 30°C, nên ta có $\text{viscosity} = 8 \times 10^{-7} \text{ m}^2/\text{s}$.

III. Mô hình và mô phỏng

3.1. Mô hình

Để mô phỏng dòng chảy, ta sử dụng phương trình Navier - Stokes

$$\begin{cases} u_t - \mu \Delta u + u \nabla u + \nabla p = f \\ \operatorname{div}(u) = 0 \end{cases}$$

Trong đó: $u = u(x,t)$ là vận tốc nước $(\in R^d)$

$p = p(x,t)$ là áp suất $(\in R)$

μ là độ nhớt (viscosity) $(\in R)$

3.2. Mô phỏng

Bài báo cáo này sẽ mô phỏng hiện tượng dòng chảy trên sông bằng ngôn ngữ lập trình Freefem++.

3.2.1. Xây dựng lưới

Có nhiều cách để xây dựng một lưới. Đối với việc tạo lưới từ một ảnh chụp vệ tinh thì ta rất khó có thể xây dựng được lưới bằng các hàm thông thường. Do đó ta cần sử dụng những công cụ chuyên biệt hơn được tích hợp sẵn trong Freefem++. Ta sẽ sử dụng công cụ isoline và ppm2rnm để xây dựng lưới (trong đó isoline là công cụ để nhận biết các đường đồng mức trên ảnh, còn ppm2rnm là công cụ để đọc ảnh dạng Portable PixMap). Việc xây dựng một lưới mô phỏng phải trải qua 2 bước: xử lý ảnh và viết chương trình.

a, Xử lý ảnh

- Bước 1: Chọn một bức ảnh bất kỳ chúng ta muốn tách vật thể trong ảnh ra làm lưới mô phỏng.
- Bước 2: Convert ảnh đó sang dạng *.ppm (Portable PixMap) hoặc *.pgm (Portable GrayMap), hoặc *.pbm (Portable Bitmap) bằng một công cụ nào đó (có thể là tool trên internet, có thể là phần mềm Photoshop, GIMP, ...). Thông thường đối với lưới 2D tạo từ một ảnh, người ta hay sử dụng ảnh *.pgm nhất.
- Bước 3: (bước này có thể bỏ qua) Sử dụng một phương pháp nào đó để làm cho vật thể trong ảnh trở nên nổi bật và rõ ràng hơn so với nền. Mình gợi ý cho các

bạn một phương pháp rất đơn giản, đó là Thresholding. Về cơ bản là ảnh cấp xám là một tập hợp các pixel, mỗi pixel mang một giá trị đặc trưng cho màu, dải màu trải từ 0 đến 255 (đối với ảnh 8 bit) với số 0 là màu đen, số 255 là màu trắng (ảnh cấp xám là ảnh có tối đa 256 màu), các bạn sẽ chọn một Threshold (ngưỡng) ở giữa 0 và 255 (ví dụ là 120 chẳng hạn), khi đó phương pháp Thresholding này sẽ convert ảnh xám của bạn sang ảnh binary (nghĩa là chỉ có 2 màu đen và trắng chứ không còn là 256 màu như ở ảnh ban đầu) bằng cách chọn các pixel có độ xám nhỏ hơn hoặc bằng ngưỡng Threshold, cho chúng bằng 0 hết, ngược lại pixel nào có độ xám lớn hơn Threshold, cho chúng bằng 255 hết.

- Bước 4: (bước này có thể bỏ qua) loại bỏ đi những chi tiết thừa trong ảnh. Nếu ảnh của chúng ta là ảnh có vật thể đen, nền trắng thì chúng ta nên tô trắng hết những chi tiết thừa đi bằng một công cụ chỉnh sửa ảnh nào đó. Ngược lại, nếu ảnh của chúng ta có vật thể trắng trên nền đen thì chúng ta sẽ tô đen những vật thể thừa đi.

Dưới đây là một ví dụ mình làm về việc tạo lưới 2D từ ảnh

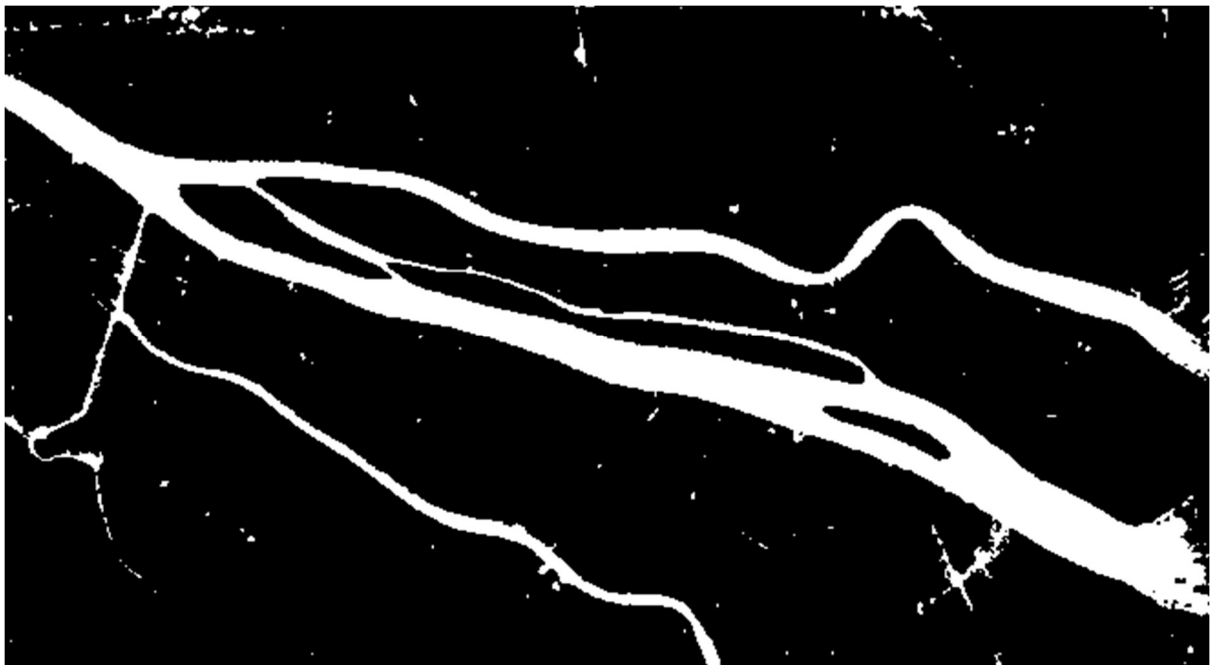
- Bước 1: Mình tải một ảnh chụp vệ tinh từ trên Internet về



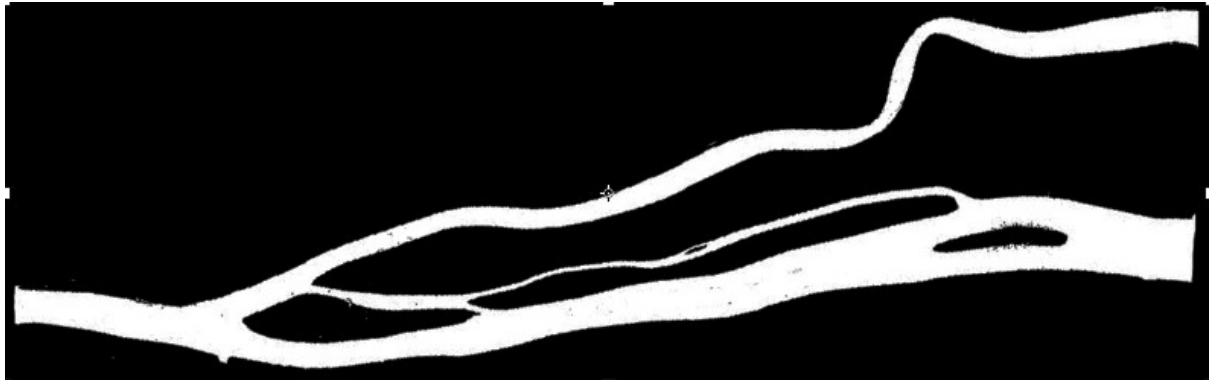
- Bước 2: Mình convert ảnh này sang dạng *.pgm



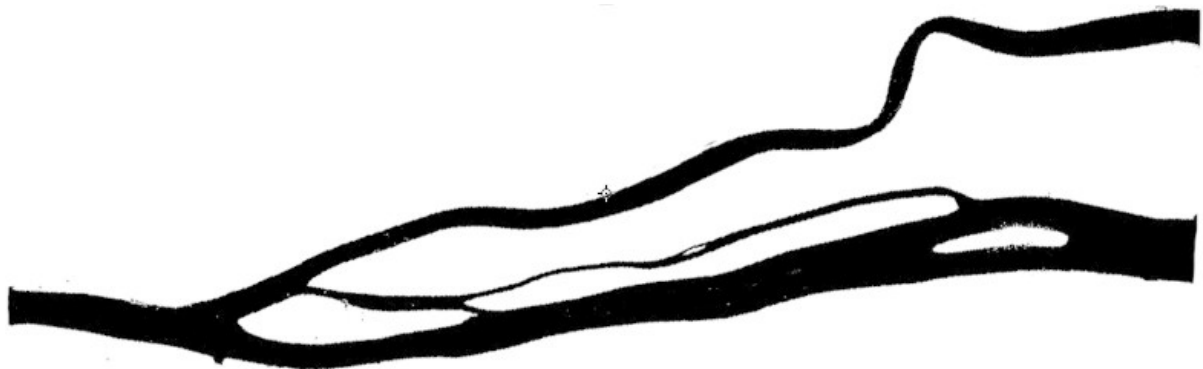
- Bước 3: Mình sử dụng công cụ Threshold trên Photoshop để tạo ảnh đen trắng, ở đây mình chọn Threshold = 120. Các bạn có thể làm theo bằng cách mở ảnh pgm bằng Photoshop, chọn Image -> Adjustment -> Threshold.... Hoặc sử dụng một công cụ Thresholding online trên Internet, việc này khá đơn giản.



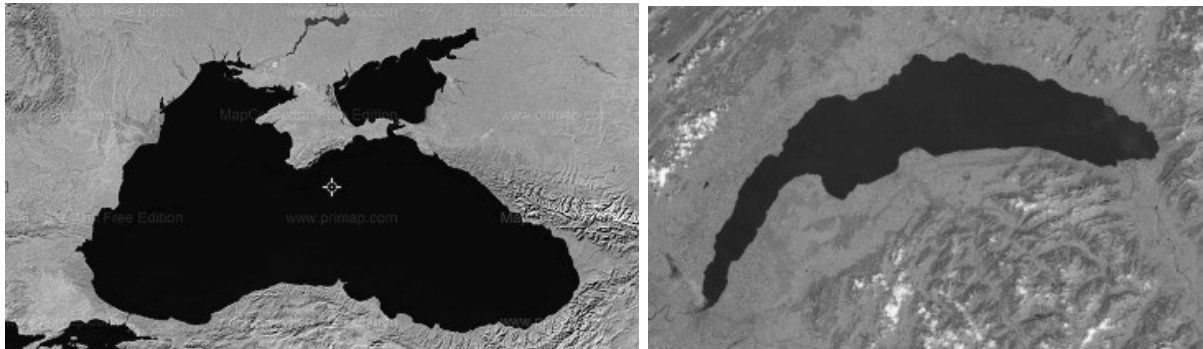
- Bước 4: Mình dùng một chút công cụ trong Photoshop để tô đen những phần trắng không cần thiết. (Mình có xoay ảnh đi một chút để sau đó dễ xử lý điều kiện biên hơn).



Hoặc các bạn có thể tạo ảnh có vật thể đen nền trắng, không nhất thiết là phải vật thể trắng nền đen đâu.



Lưu ý: Nếu ảnh pgm của bạn có độ rõ ràng đủ cao thì bạn có thể bỏ qua bước 3 và bước 4. Ví dụ như những ảnh như thế này.



b, Viết chương trình:

Dưới đây là một ví dụ mình đã tạo một lưới từ ảnh chụp một đoạn sông Cửu Long.

1.	load "ppm2rnm"
2.	load "isoline"
3.	
4.	real[int, int] Curves(3, 1);
5.	int[int] be(1);
6.	real[int, int] ff1("SongCuuLong.pgm");
7.	int nx = ff1.n, ny = ff1.m;
8.	mesh Th = square(nx-1, ny-1, [(nx-1) * (x), (ny-1) * (1-y)]);
9.	fespace Vh(Th, P1);
10.	Vh f1;
11.	f1[] = ff1;
12.	int nc = isoline(Th, f1, iso=0.4, Curves, beginend=be);
13.	cout << "number of isoline curves: " << nc << endl;
14.	
15.	border c0(t=0,1){int c = 0; P=Curve(Curves,be[2 * c],be[2 * c+1]-1,t);}
16.	border c1(t=0,1){int c = 1; P=Curve(Curves,be[2 * c],be[2 * c+1]-1,t);}
17.	border c2(t=0,1){int c = 2; P=Curve(Curves,be[2 * c],be[2 * c+1]-1,t);}
18.	border c3(t=0,1){int c = 3; P=Curve(Curves,be[2 * c],be[2 * c+1]-1,t);}
19.	border c4(t=0,1){int c = 4; P=Curve(Curves,be[2 * c],be[2 * c+1]-1,t);}
20.	border c5(t=0,1){int c = 5; P=Curve(Curves,be[2 * c],be[2 * c+1]-1,t);}
21.	plot(c0(1000));
22.	plot(c1(1000));
23.	plot(c2(1000));
24.	plot(c3(1000));
25.	plot(c4(1000));
26.	plot(c5(1000));
27.	Th = buildmesh(c0(-2000) + c2(-300) + c3(-300) + c4(-300) + c5(-300));
28.	plot(Th);

Sau đây mình sẽ giải thích ngắn gọn về đoạn chương trình trên, để sau này các bạn có thể sử dụng lại đoạn code này cùng với những chỉnh sửa hợp lý.

- Dòng 1,2: Khai báo plugins ppm2rnm và isoline.
- Dòng 4: Tạo mảng 2 chiều Curves để lưu các điểm trên các isoline.
- Dòng 5: Tạo mảng để lưu cặp điểm xuất phát và kết thúc của các isoline (be: begin-end)

Giải thích cụ thể hơn về dòng 4 và dòng 5 như sau:

+ Dòng 4 tạo ra mảng để lưu các điểm thuộc các isoline độc lập với nhau (một ảnh có thể có nhiều isoline, ví dụ như một ảnh có 2 miền không giao nhau thì ít nhất có 3 isoline, isoline đầu tiên đó là khung của bức ảnh là một hình chữ nhật và 2 isoline khác là biên bao quanh 2 miền không giao nhau kia). Curves(3,1) có nghĩa là một điểm P_i thuộc một Curve có 3 giá trị, tọa độ x, tọa độ y và độ dài đường đi từ điểm khởi đầu P_0 tới điểm P_i .

+ Dòng 5 tạo ra mảng để lưu cặp điểm khởi đầu kết thúc của các isoline (từ đó xác định hướng của đường cong là cùng chiều hay ngược chiều kim đồng hồ).

Các isoline có dạng $line = p_b^c, p_1^c, p_2^c, \dots, p_n^c, p_e^c$ trong đó:

- c là số thứ tự của isoline (0,1,2,...)
- p_b^c là điểm khởi đầu (begin) của isoline thứ c , được tính bằng $be[2*c]$
- p_e^c là điểm kết thúc (end) của isoline thứ c , được tính bằng $be[2*c + 1]$
- p_i^c là các điểm ở giữa, mang 3 giá trị:

$Curves(0,i)$ là tọa độ x của điểm p_i

$Curves(1,i)$ là tọa độ y của điểm p_i

$Curves(2,i)$ là độ dài quãng đường từ điểm p_b đến điểm p_i (giá trị này của $p_b = 0$, của p_e là lớn nhất vì quãng đường di chuyển của p_b đến p_e là xa nhất.

Do đó đây là giá trị để biết được quãng đường di chuyển từ p_b đến p_e sẽ đi qua các điểm trung gian theo thứ tự nào).

- Dòng 6: đọc ảnh và ghi vào mảng 2 chiều.
- Dòng 7: lấy kích cỡ của ảnh (n là chiều rộng, m là chiều cao) theo pixel.
- Dòng 8: tạo lưới hình chữ nhật có kích cỡ giống ảnh.
- Dòng 9, 10, 11, 12: xác định số lượng các isoline và ghi lại các giá trị đặc trưng của isoline vào 2 mảng $Curves[*,*]$ và $be[*]$. Tại dòng 12 các bạn có thể thay đổi giá trị iso, giá trị này nhỏ hơn thì lưới tạo được sẽ mảnh hơn.
- Dòng 13: đây là dòng thử, không ảnh hưởng đến việc tạo lưới. Để chúng ta biết được chương trình đã xác định ra bao nhiêu isoline.
- Dòng 15,16,17,18,19,20: đây là các dòng để tạo biên, ví dụ của mình với bức ảnh dòng sông có khoảng 30 isoline, nghĩa là có thể tạo được 30 biên. Tuy nhiên những biên có số thứ tự càng nhỏ thì những vùng có biên đó càng có diện tích lớn và nổi bật, cho nên chúng ta sẽ thử một vài biên có thứ tự nhỏ xem biên nào sẽ có ích cho việc tạo lưới.

- Dòng 21,22,23,24,25,26: đây là các dòng thử, không ảnh hưởng đến việc tạo lưới. Để chúng ta biết chính xác biên đó bao quanh vùng nào, để sau này chúng ta lựa chọn có sử dụng biên đó để dựng lưới hay không.
- Dòng 27: sau khi biết được biên nào bao quanh vùng nào rồi, ta sẽ lựa chọn những biên phù hợp để dựng lưới. Lưu ý tham số đi kèm với các biên, có thể đó là số âm, hoặc là số dương tùy vào cặp điểm p_b và p_e của từng isoline, cho nên để biết chính xác isoline đó có biên chạy cùng hay ngược chiều kim đồng hồ, các bạn nên thử bằng các tạo lưới bằng mỗi biên đó thôi (thay số dương mà nó đúng thì biên đó chạy ngược chiều kim đồng hồ, và ngược lại). Lưu ý thứ 2 đó là tham số đó không được quá nhỏ (đó là số điểm trên biên để tạo thành lưới hữu hạn các tam giác), vì khi mỗi isoline được tạo, nó cũng được tạo bởi hữu hạn các điểm, cho nên tham số này bắt buộc phải lớn hơn hoặc bằng số điểm tạo nên isoline thì mới có thể tạo được một lưới hữu hạn các tam giác (ví dụ như một biên hình vuông thì ít nhất trên biên của nó cũng phải có 4 điểm thì mới tạo thành lưới các tam giác được).
- Dòng 28: hiện lưới vừa tạo.
- Kết quả:



3.2.2. Quản lý biên của lưới

Khi viết chương trình sử dụng một lưới nào đó để mô phỏng, chúng ta luôn quan tâm đến các điều kiện biên. Tuy nhiên khi tạo lưới từ một ảnh như trên thì label của các điểm nằm trên biên được đánh tự động (ví dụ như tạo lưới vuông bằng lệnh square thì các điểm trên biên được đánh các số 1,2,3,4 một cách tự động). Nếu chúng ta đặt một điều kiện duy cho toàn bộ biên của lưới thì chúng ta không cần quan tâm đến label của các điểm trên biên, mà chúng ta gọi thẳng tên của biên và sau đó đặt điều kiện, (ví dụ như $on(c, u = 0)$). Tuy nhiên nếu chúng ta chỉ muốn đặt điều kiện lên một phần của biên thì đó là một vấn đề không dễ, lúc này một cách khả thi là chúng ta sửa đổi label của các điểm trên đoạn biên cần đặt điều kiện, rồi sau đó đặt điều kiện lên các điểm trên biên có label đó. Hoặc một cách nữa là chúng ta sẽ xấp xỉ đoạn biên đó bằng một số hữu hạn các cạnh, các cạnh mà chúng ta đã tạo bằng các lệnh border, khi đó ta đã biết được border nào có label bao nhiêu rồi, sau đó thay vì đặt điều kiện lên đoạn biên ban đầu, chúng ta sẽ đặt điều kiện lên tập hợp các cạnh kia.

a, Quản lý label của các điểm trên biên

Với một lưới được dựng từ ảnh thì label của các điểm trên biên được đánh hoàn toàn tự động, khác với việc chúng ta xây dựng lưới từ một tập hợp các border, chúng ta có thể tự đặt label. Vì vậy muốn đặt điều kiện lên một đoạn biên, ta phải chỉnh sửa label của các điểm trên đoạn biên đó.

Vì lưới là một đoạn sông ở phía trên khá phức tạp, và khi hiển thị tại báo cáo sẽ thiếu rõ ràng và tổng quát, do đó mình sẽ demo phương pháp này trên một lưới đơn giản hơn. Các bạn hoàn toàn có thể áp dụng phương pháp này trên một lưới phức tạp hơn nhiều.

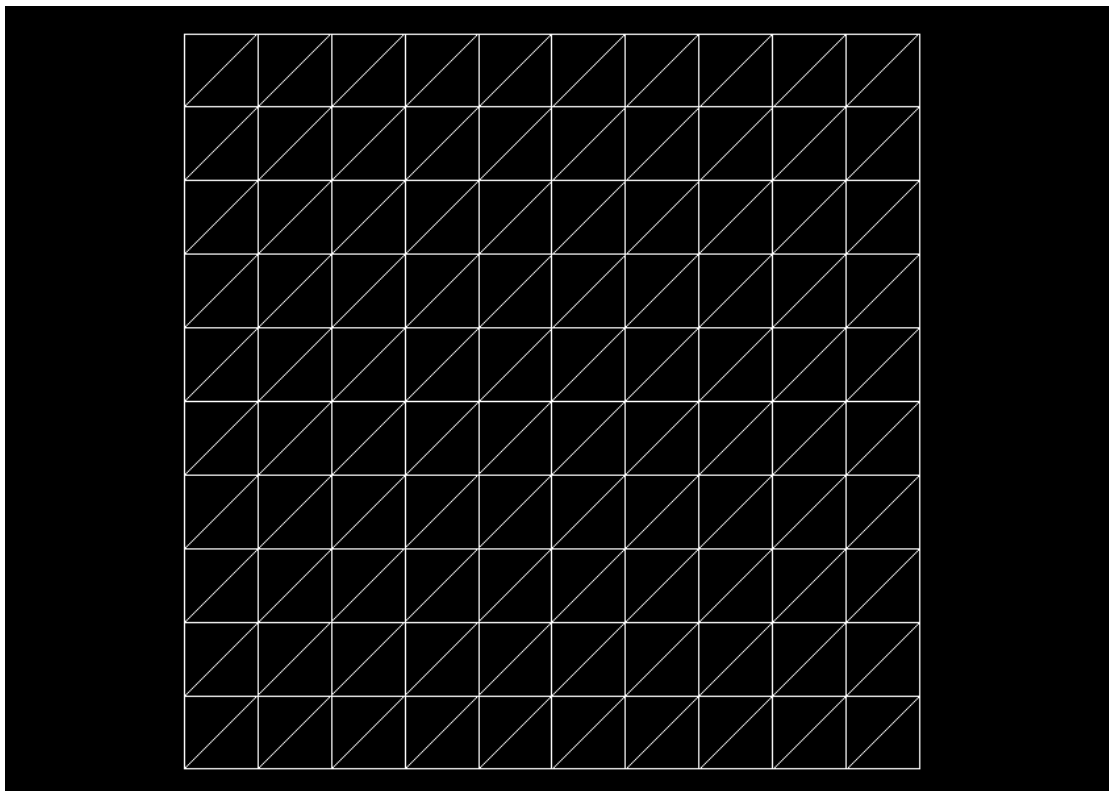
Ví dụ ta có một lưới vuông:

```
mesh Th = square(10,10);  
savemesh(Th, "Th.mesh");
```

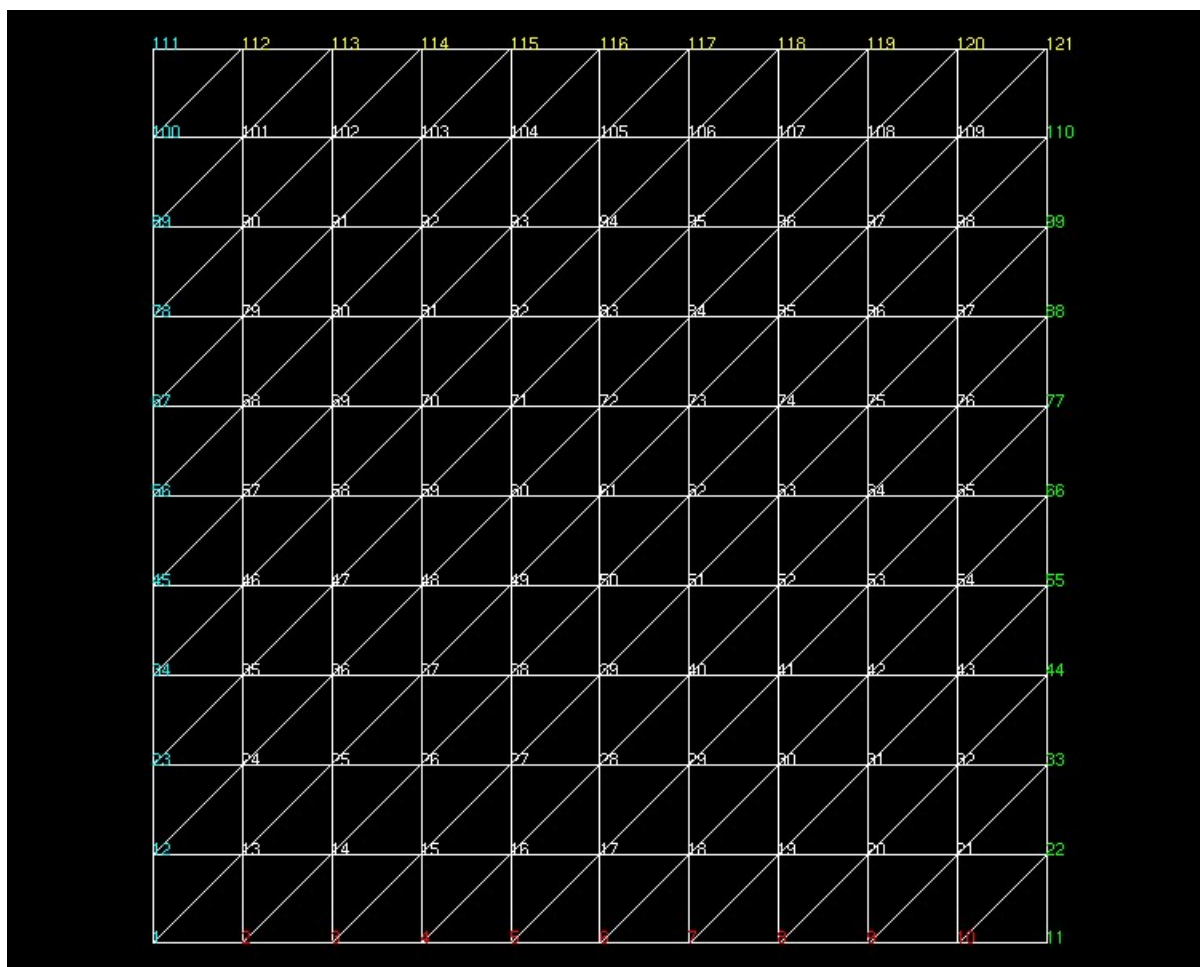
Lệnh savemesh là để lưu lại lưới ở dạng *.mesh, sau đó chúng ta có thể sử dụng lại ở một nơi khác mà không cần phải mất công tạo lại lưới nữa, thay vì tạo lưới chúng ta chỉ cần khai báo một câu lệnh đọc lưới (readmesh) là xong, việc này rất hữu ích trong việc chúng ta có một lưới rất phức tạp mà lại muốn sử dụng lưới này ở nhiều bài toán khác nhau.

Mở lưới vừa tạo lên bằng câu lệnh trên cmd (Windows)

```
ffmedit Th.mesh
```

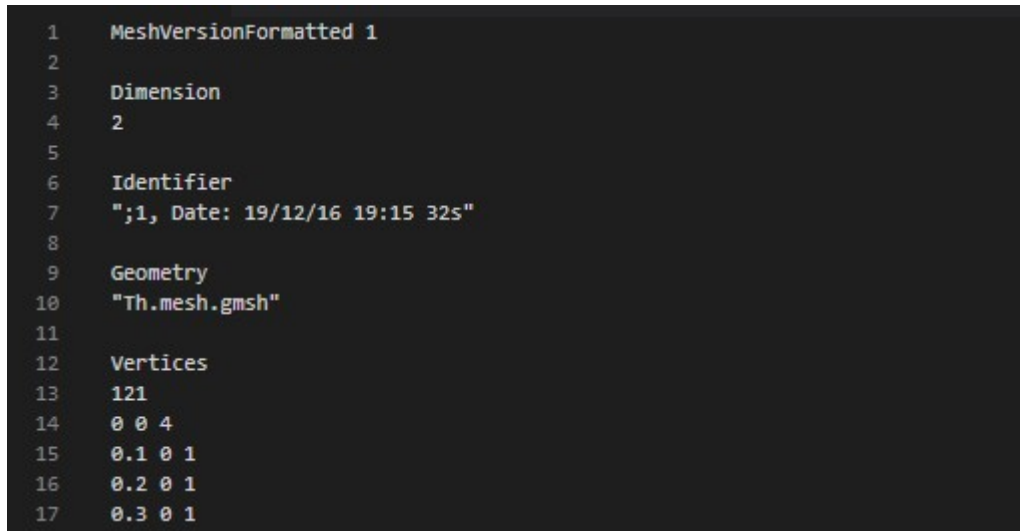


Dùng tổ hợp phím Shift+P để hiện lên số thứ tự của tất cả các điểm trong lưới



Chúng ta có một lưới gồm hữu hạn các điểm, mỗi điểm có số thứ tự riêng biệt, những điểm có cùng màu là những điểm có cùng label (điểm màu trắng có label = 0, màu đỏ có label = 1, màu xanh lá có label = 2, màu vàng có label = 3, màu xanh dương có label = 4). Ví dụ ta muốn đặt điều kiện Dirichlet thuần nhất cho cạnh bên phải, ta dùng lệnh $on(2, u = 0)$. Nhưng nếu ta muốn đặt điều kiện lên một đoạn bất kỳ, ví dụ như đoạn biên từ điểm 4 đến điểm 8 chẳng hạn. Chúng ta có thể có một cách đó là tách biên dưới (từ điểm 1 đến điểm 11) thành 3 phần liên nhau bằng 3 lệnh border (1 đến 4, 4 đến 8, và 8 đến 11) và đặt điều kiện lên phần giữa, tuy nhiên với một lưới tạo bởi ảnh thì điều đó là không thể, cách duy nhất là sửa label các điểm đó về cùng một label, rồi sau đó đặt điều kiện biên dựa vào label vừa đặt.

Ta mở file Th.mesh ở cùng folder với file chương trình *.edp bằng một công cụ soạn thảo (Word, Sublime Text, Notepad, VSCode,...), nhưng mình khuyên các bạn nên sử dụng phần mềm soạn văn bản nào có hiển thị số dòng (line number). Ở đây mình dùng VSCode.



```
1 MeshVersionFormatted 1
2
3 Dimension
4 2
5
6 Identifier
7 ";1, Date: 19/12/16 19:15 32s"
8
9 Geometry
10 "Th.mesh.gmsh"
11
12 Vertices
13 121
14 0 0 4
15 0.1 0 1
16 0.2 0 1
17 0.3 0 1
```

Mở file Th.mesh lên chúng ta sẽ thấy dữ liệu được chia thành 12 phần

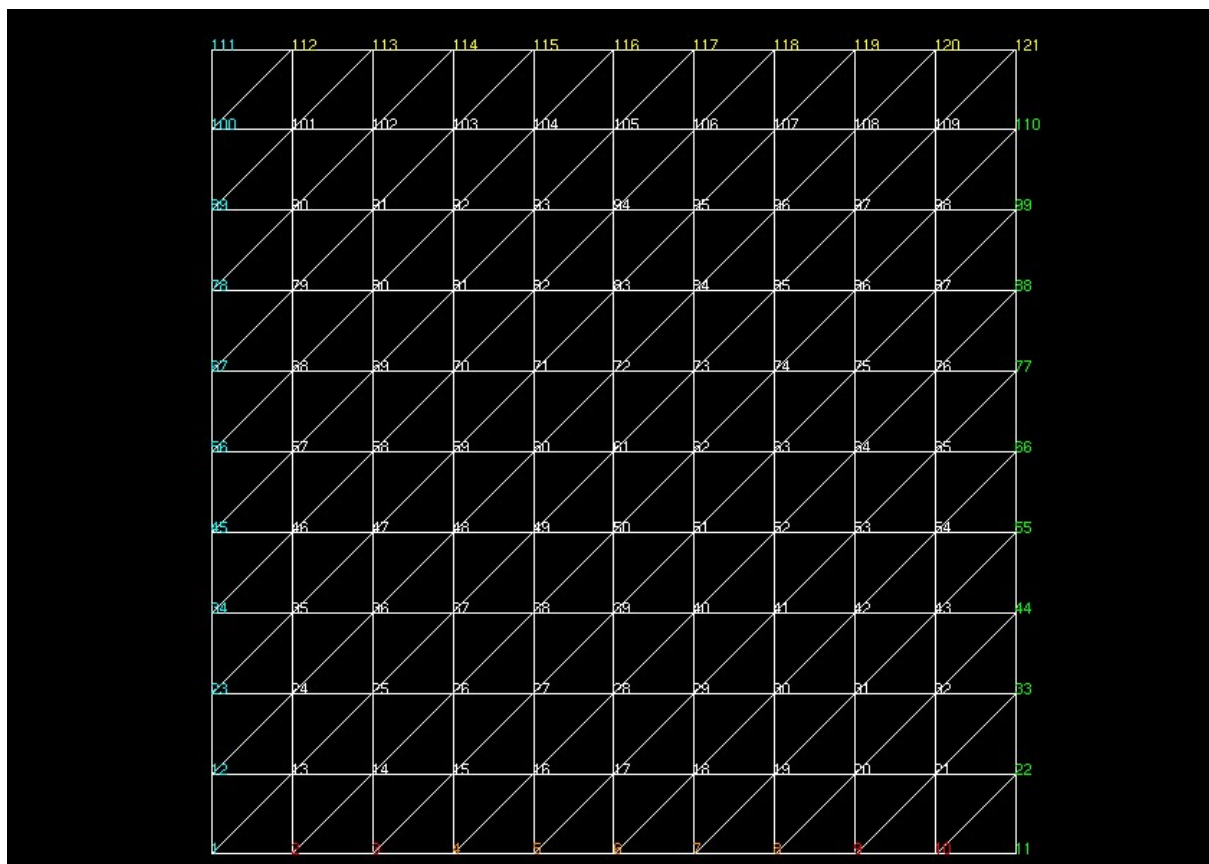
- MeshVersionFormatted
- Dimension
- Identifier
- Geometry
- Vertices
- Edges
- Triangles
- SubDomainFromMesh
- SubDomainFromGeom
- VertexOnGeometricVertex
- VertexOnGeometricEdge
- EdgeOnGeometricEdge

Chúng ta sẽ chỉ chú ý vào phần Vertices (các đỉnh tam giác) mà thôi, với mỗi điểm sẽ tương ứng với mỗi dòng, mỗi dòng có 3 con số, số đầu tiên là tọa độ x, số thứ 2 là tọa độ y, số thứ 3 là label. Ví dụ điểm số 1 có tọa độ (0,0) và label 4.

Công việc bây giờ của chúng ta là sửa label cho các điểm thuộc cạnh từ điểm 4 đến điểm 8 (nghĩa là 5 điểm 4,5,6,7,8), chúng ta đặt 5 điểm này cùng một label, ví dụ là label = 5. Đỉnh 1 ở dòng 14 thì đỉnh 4,5,6,7,8 ở dòng 17,18,19,20,21 (đỉnh i nằm ở dòng $i + 13$) đó là lý do tại sao mình khuyên các bạn nên đọc file mesh bằng một Text Editor có hiển thị được line number. Mình thì thường xóa đi một số dòng trắng để tiện cho việc tìm thứ tự điểm, ví dụ mình xóa đi dòng 2,5,8 là các dòng trắng, để điểm 1 nằm ở dòng 11, từ đó chúng ta có thể tự tìm các điểm thứ i sẽ nằm ở dòng $i + 10$.

```
9 Vertices
10 121
11 0 0 4
12 0.1 0 1
13 0.2 0 1
14 0.3 0 5
15 0.4 0 5
16 0.5 0 5
17 0.6 0 5
18 0.7 0 5
19 0.8 0 1
20 0.9 0 1
```

Sau khi sửa label của 5 điểm. Ta lưu lại và mở lại mesh bằng ffmedit Th



Các bạn có thể thấy đoạn gồm 5 điểm 4,5,6,7,8 đã chuyển sang màu cam, vì lúc này đoạn đó có các điểm mang label = 5. Nếu muốn sử dụng lưới này vào một bài toán cụ thể, ta sẽ khai báo lưới như sau:

```
mesh Th = readmesh("Th.mesh");
```

Vì lúc này file Th.mesh của chúng ta đã được thay đổi.

Sau đó ví dụ muốn đặt điều kiện biên Dirichlet lên đoạn biên từ điểm 4 đến điểm 8 này, ta chỉ cần sử dụng câu lệnh:

```
on(5, u = 0);
```

Vậy là ta đã đặt điều kiện biên lên một đoạn bất kỳ được rồi.

Nhưng với một lưới dày, mịn, việc sửa label của một đoạn biên đồng nghĩa với việc sửa label một cách thủ công một số lượng lớn các điểm. Nếu may mắn số thứ tự các điểm trên đoạn biên đó là liên tiếp, thì mình gợi ý cho các bạn một phương pháp nhanh và hiệu quả, đó là sử dụng bảng tính Sheet (Excel, Google Sheet, Libre Office Calc,...). Ví dụ như ở bài trên, ta phải sửa label một tập hợp các điểm mang số thứ tự từ 4 đến 8.

Các bước làm như sau:

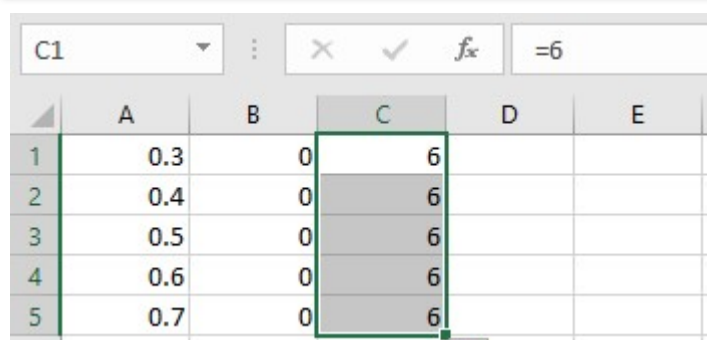
Cut đoạn chứa các điểm cần sửa vào Text Editor nào đó để tạo bảng, mình dùng MSWord

0.3	0	5
0.4	0	5
0.5	0	5
0.6	0	5
0.7	0	5

Copy sang Excel

	A	B	C	D
1	0.3	0	5	
2	0.4	0	5	
3	0.5	0	5	
4	0.6	0	5	
5	0.7	0	5	

Ví dụ ta muốn sửa tất cả label của đoạn này đều bằng 6, ta gõ hàm =6 vào ô ở hàng đầu tiên, cột cuối cùng. Giữ chuột ở góc phải phía dưới của ô đó, kéo xuống đến dòng cuối cùng.



	A	B	C	D	E
1	0.3	0	6		
2	0.4	0	6		
3	0.5	0	6		
4	0.6	0	6		
5	0.7	0	6		

Copy cả bảng này vào đúng vị trí đã cut ở file mesh

```
9 Vertices
10 121
11 0 0 4
12 0.1 0 1
13 0.2 0 1
14 0.3 0 6
15 0.4 0 6
16 0.5 0 6
17 0.6 0 6
18 0.7 0 6
19 0.8 0 1
20 0.9 0 1
```

Vậy là ta đã sửa được label của một dãy các điểm mà không cần phải sửa label từng điểm một cách thủ công. Với ví dụ này thì việc sửa label 5 điểm là một công việc khá dễ, có thể sửa thủ công, nhưng với đoạn biên mà có số lượng điểm lớn, thì phương pháp sử dụng bảng tính Sheet là rất nhanh và chính xác.

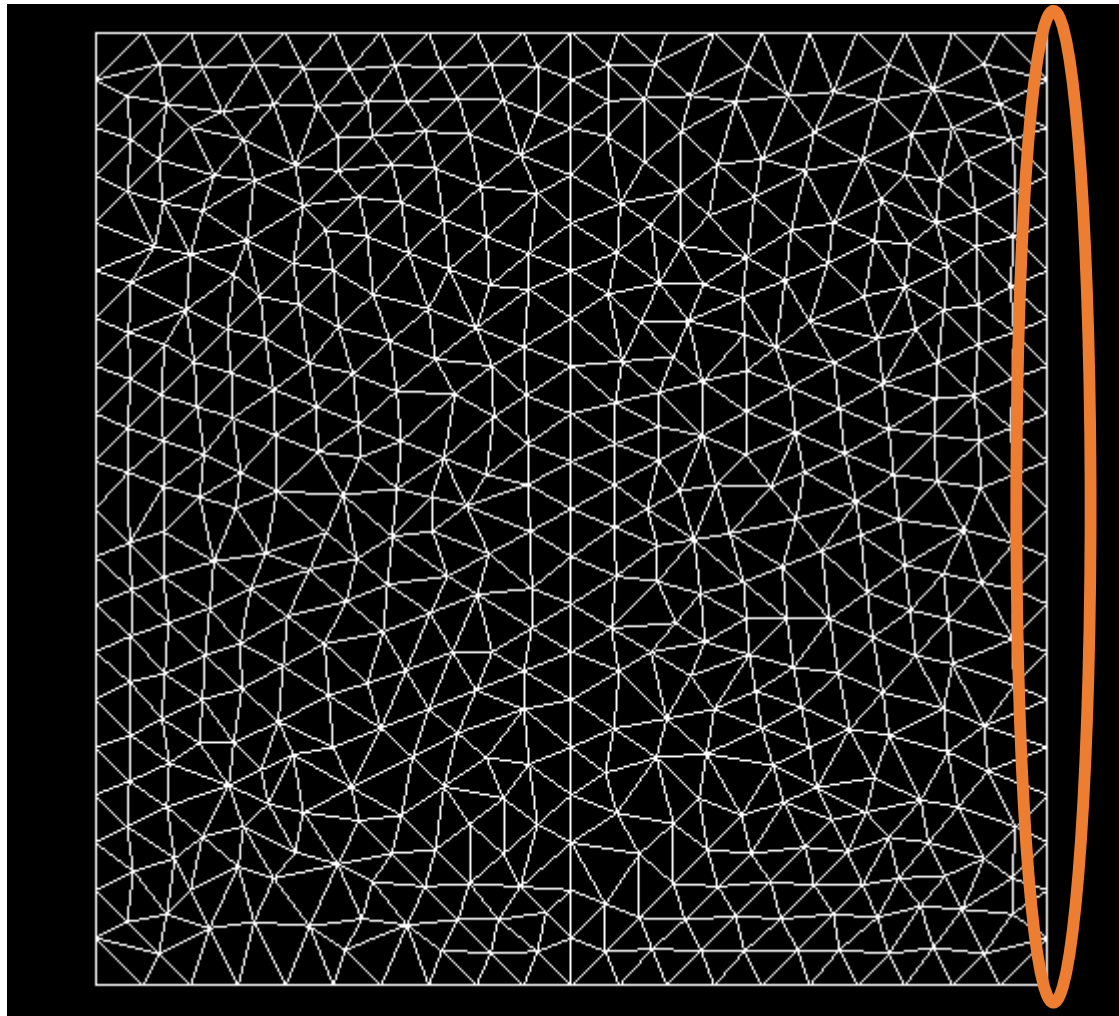
Với một lưới được xây dựng từ một ảnh, cách thức quản lý label của các điểm là hoàn toàn tương tự, tuy nhiên nếu như số lượng mắt lưới quá nhiều sẽ gây khó khăn cho chúng ta trong việc chỉnh sửa label của những điểm trên biên. Do đó chúng ta sẽ phải làm lưới thưa hơn để việc chỉnh sửa trở nên đơn giản hơn, hoặc chúng ta có thể sử dụng phương pháp tiếp theo sẽ được nêu sau đây.

b, Xấp xỉ đoạn biên bằng hữu hạn các cạnh

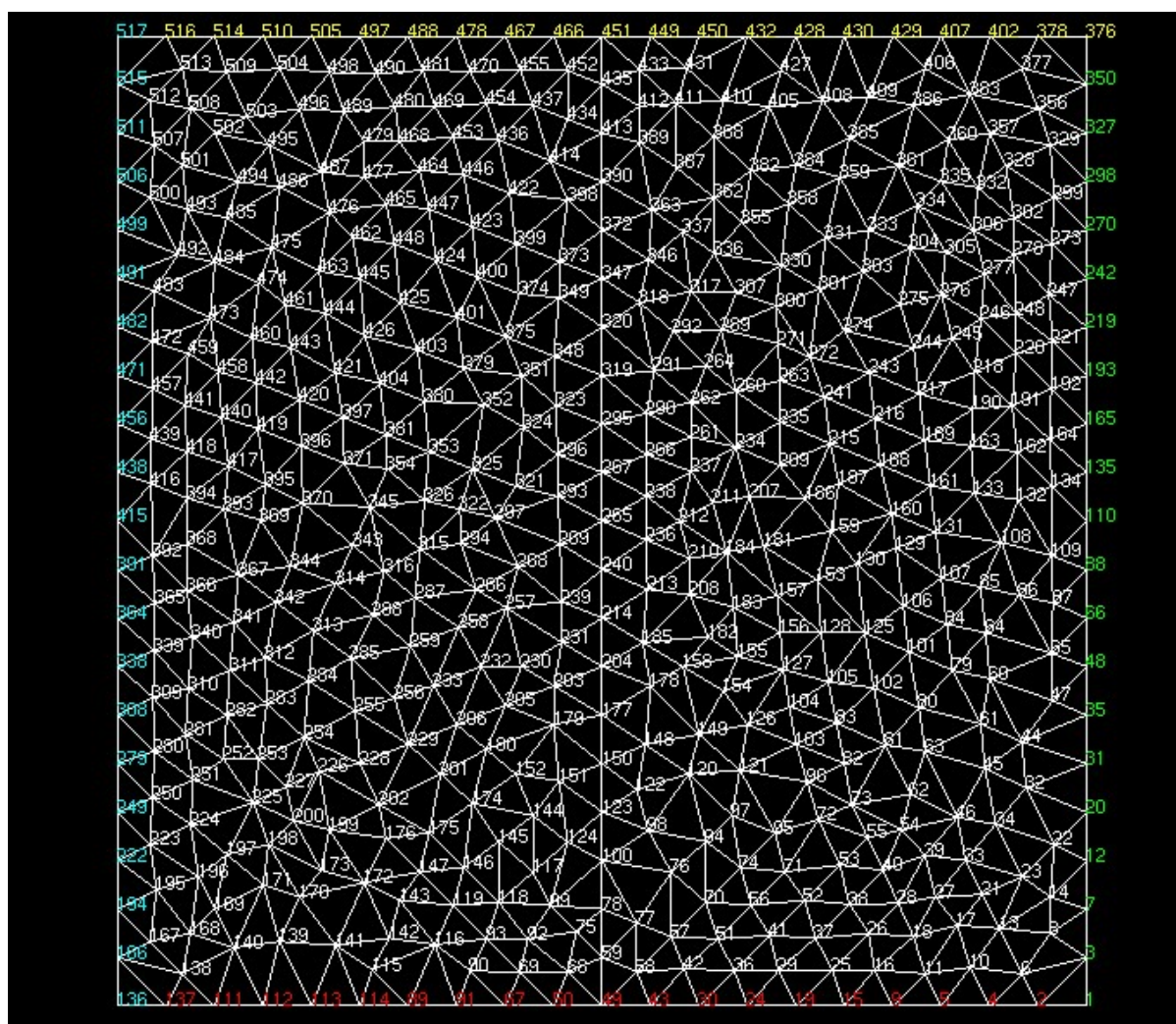
Ví dụ chúng ta muốn đặt điều kiện lên một đoạn biên có rất nhiều điểm nằm trên đó, và label của chúng hoặc là không giống nhau (nhưng thường thì giống nhau) hoặc là trùng với label của các điểm khác mà chúng ta không cần đặt điều kiện biên. Nếu áp dụng phương pháp sửa đổi label của tất cả các điểm đó thì rất mất thời gian. Vì vậy chúng ta phải xử lý theo cách khác.

Sau đây là một ví dụ minh họa

Ta muốn đặt điều kiện biên lên cạnh phải của lưới vuông dưới. Giả sử như ta không thể đặt điều kiện biên theo tên border được (ví dụ $on(c, u = 0)$), vì một lưới tạo bởi ảnh không được cấu thành từ tập các border, mà chỉ là tập hợp các điểm. Cụ thể hơn là ta muốn mô phỏng một dòng chảy xuất phát từ biên trái chảy từ trái sang phải, ta muốn đặt điều kiện Newman thuần nhất cho biên phải.

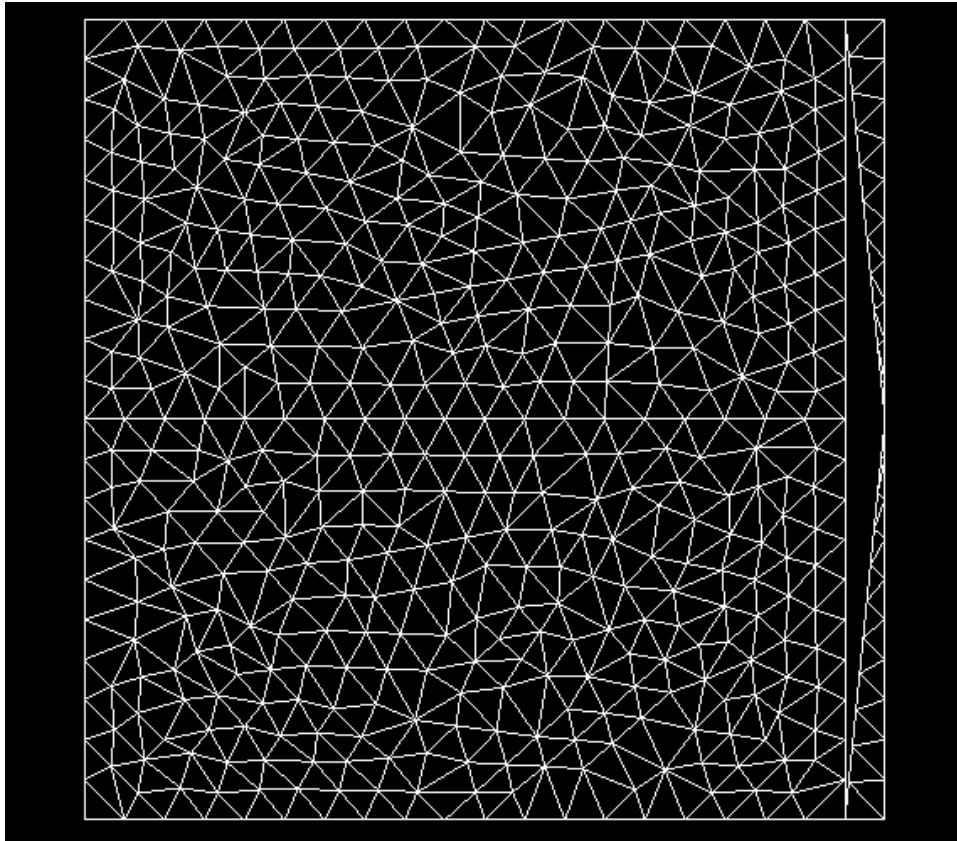


Dưới đây là số thứ tự cùng với màu label các điểm

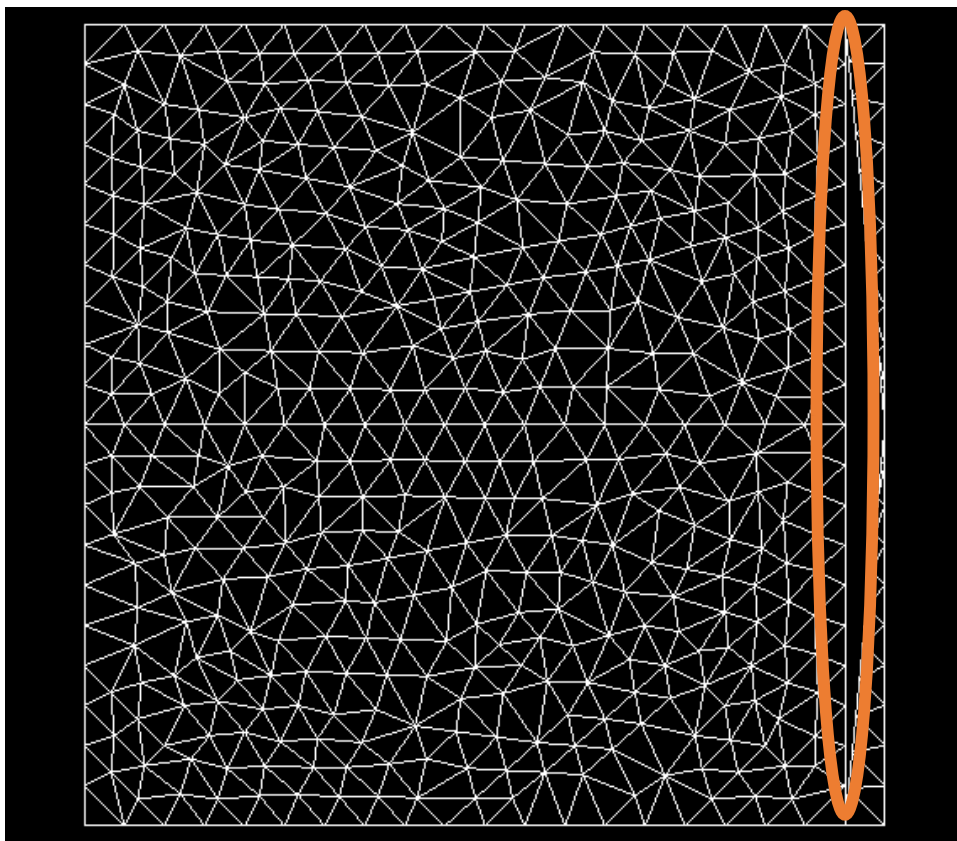


Số lượng điểm trên biên này khá nhiều, đồng thời label của chúng không riêng biệt. Thêm vào đó số thứ tự cũng không liên tục (1, 3, 7, 12, 20, ..., 350, 376) nên ta không thể sử dụng bảng tính Sheet để sửa hàng loạt được, cho nên phải sửa label của tất cả các điểm này một cách thủ công, vậy thì rất mất thời gian.

Sau đây là một phương pháp tạo lưới mới để dễ dàng hơn trong việc xử lý điều kiện biên. Dưới đây là lưới mới, mô phỏng trên lưới mới này sẽ cho kết quả gần giống với mô phỏng trên lưới cũ ban đầu.



Thay vì đặt điều kiện lên biên bên phải ở ảnh ban đầu, ta sẽ đặt điều kiện lên cạnh được khoanh đỏ dưới đây

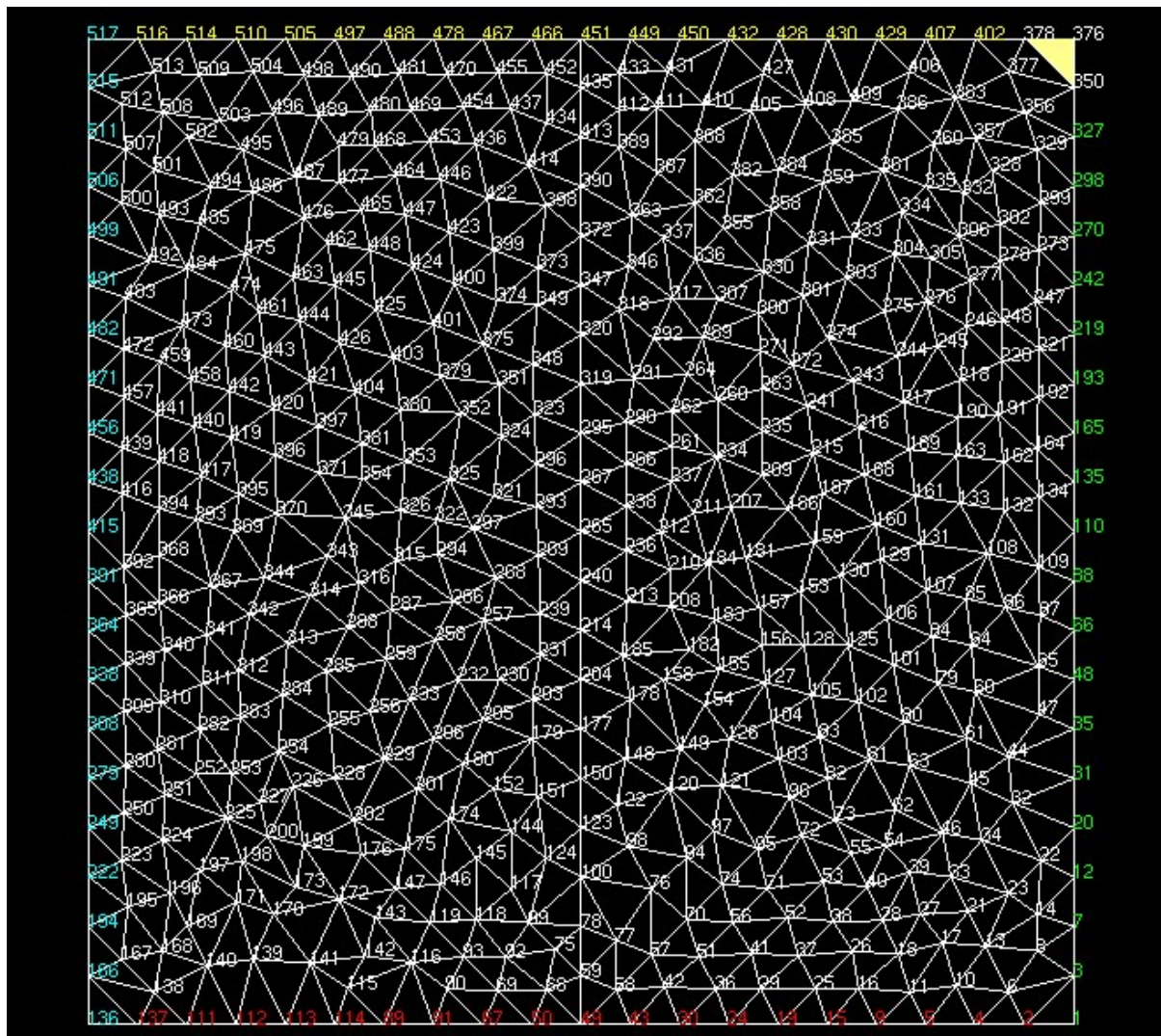


Việc đặt điều kiện sang cạnh này sẽ làm cho mô phỏng của chúng ta gần giống với mô phỏng với lưới ban đầu cùng điều kiện biên đặt ở biên phải. Việc đặt điều kiện lên cạnh trên là công việc đơn giản, vì cạnh đó ta đã tạo bằng border. Vậy làm thế nào để ta tạo ra được một lưới mới như trên.

Việc đầu tiên ta phải xấp xỉ đoạn biên cần đặt điều kiện bằng một hữu hạn các cạnh được nối bởi các điểm trong lưới. Ở ví dụ trên, ta xấp xỉ biên phải bằng một cạnh nối từ điểm số 2 tới điểm số 378, ví điểm số 2 và điểm số 378 là 2 điểm trên biên mà gần với 2 đầu của biên cần đặt điều kiện nhất (điểm 1 và điểm 376). Để mô phỏng giống với lưới ban đầu, ta gần như phải loại bỏ phần lưới giữa biên cần đặt điều kiện và tập các cạnh xấp xỉ, do đó ta mới cần tạo một lỗ như vậy ở lưới ban đầu (vì ví dụ ta cần làm đó là tạo dòng chảy từ trái sang phải và điều kiện biên Newman thuần nhất với biên phải).

Để tạo được một lỗ như vậy chúng ta chỉ cần 3 điểm, 2 điểm trên biên sao cho 2 điểm đó gần với 2 đầu của biên cần đặt điều kiện nhất, và 1 điểm bất kỳ thuộc vào biên cần đặt điều kiện (ở đây ta chọn điểm số 110).

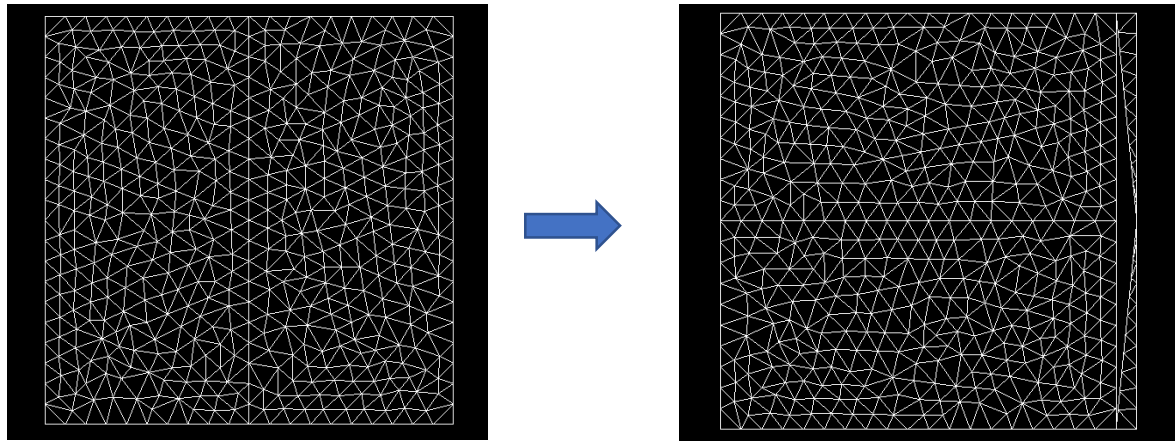
Việc tạo các border này khá đơn giản, đó chỉ là một đường thẳng đi qua 2 điểm đã biết trước tọa độ. Nếu muốn biết tọa độ của một điểm, ta có thể mở file Th.mesh để xem, hoặc có một cách nhanh hơn đó là mở mesh bằng medit, giữ Shift, chuột nhấn vào một tam giác có chứa đỉnh đó, tọa độ của đỉnh đó sẽ hiện lên màn hình console của cmd.



```
Picking result :
Triangle 672 : 350, 376, 378    ref : 0 [DEFAULT_MAT]
vertex 350 : 1.000000 0.950000 0.000000    ref 2
vertex 376 : 1.000000 1.000000 0.000000    ref 3
vertex 378 : 0.950000 1.000000 0.000000    ref 3
```

Như vậy ta có thể biết được mình vừa nhấn vào tam giác thứ 672, với tọa độ 3 đỉnh là giá trị của 2 cột đầu. Khi có tọa độ các điểm, ta xây dựng border bằng công thức $(x - x_A)(y_B - y_A) = (y - y_A)(x_B - x_A)$ với điểm thứ nhất có tọa độ (x_A, y_A) và điểm thứ 2 có tọa độ (x_B, y_B) .

Cuối cùng ta có được một lưới mới với điều kiện biên mới nhưng mô phỏng thì gần giống bài toán ban đầu với lưới cũ và điều kiện biên cũ.



Với một lưới phức tạp, ta hoàn toàn có thể áp dụng phương pháp sửa đổi lưới ban đầu sang một lưới mới với điều kiện biên mới như trên.

Bài mô phỏng dòng chảy đoạn sông của mình cũng áp dụng phương pháp này để đặt điều kiện biên Newman thuần nhất lên một số đoạn biên, và kết quả mô phỏng khá hợp lý.

