

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



Môn học: Đồ họa máy tính

---

## BÁO CÁO BÀI TẬP THỰC HÀNH 3

Thực hiện các phép biến đổi đối tượng 2D

---

Giảng viên phụ trách: Thầy Trần Thái Sơn - Thầy Võ Hoài Việt



## Thành viên

Thứ tự	Họ và tên	MSSV
1	Phạm Long Khánh	21120479



## Mục lục

<b>1</b>	<b>Cài đặt thư viện và môi trường</b>	<b>4</b>
<b>2</b>	<b>Xử lý chương trình ( Người điều khiển)</b>	<b>5</b>
2.1	Thừa kế từ chương trình trước . . . . .	5
2.2	Một số hàm chú ý . . . . .	5
<b>3</b>	<b>Lớp Matrix và các phép biến đổi</b>	<b>7</b>
3.1	Lớp Matrix . . . . .	7
3.2	Các phép biến đổi . . . . .	7
3.3	Phép tịnh tiến . . . . .	8
3.4	Phép quay . . . . .	9
3.5	Phép co giãn . . . . .	10
<b>4</b>	<b>Demo Hướng dẫn sử dụng</b>	<b>11</b>
4.1	Demo . . . . .	11
4.2	Hướng dẫn sử dụng . . . . .	11
<b>5</b>	<b>Các nguồn tham khảo</b>	<b>14</b>



# 1 Cài đặt thư viện và môi trường

Phần lớn chương trình như vẽ hình , tô màu và chọn hình được lấy lại từ đồ án 2. Đồ án này khác ở file Matrix.h nhằm thực hiện một số biến đổi đối với các hình đã được vẽ.

## 2 Xử lý chương trình ( Người điều khiển)

### 2.1 Thừa kế từ chương trình trước

Đa số phần xử lý chương trình này sẽ được lấy lại từ đề án 2, tuy nhiên có một vài thay đổi ở một số hàm để có thể điều khiển các phép biến đổi.

### 2.2 Một số hàm chú ý

- Keyboard

```
1 static void keyboard(unsigned char key, int x, int y) {
2     double angle = 0.0;
3     double ds = 1.0;
4     if (isSelectedMode && selectedShape)
5     {
6         switch (key) {
7             case 'l': // Clockwise rotation
8                 angle = 1.0;
9                 break;
10            case 'r':
11                angle = -1.0; // Counterclockwise rotation
12                break;
13            case '+':
14                ds = 1.1;
15                break;
16            case '-':
17                ds = 0.9;
18                break;
19        }
20        if (angle != 0.0)
21            selectedShape->rotate(angle);
22        else
23            selectedShape->scale(ds, ds);
24        glutPostRedisplay();
25    }
26 }
```

Hàm này thêm một số nút để xoay trái (l), xoay phải(r), phóng to(+) và thu nhỏ(-) hình được chọn. Nếu không có hình được chọn, sẽ không có gì xảy ra.

- Keyboard

```
Title

1 static void specialKeyboard(int key, int x, int y) {
2     double dx = 0.0, dy = 0.0;
3     if (isSelectedMode && selectedShape)
4     {
5         switch (key) {
6             case GLUT_KEY_RIGHT:
7                 dx = 1.0;
8                 break;
9             case GLUT_KEY_LEFT: // Right arrow
10                dx = -1.0;
11                break;
12             case GLUT_KEY_DOWN:
13                 dy = 1.0;
14                 break;
15             case GLUT_KEY_UP: // Right arrow
16                 dy = -1.0;
17                 break;
18         }
19         selectedShape->translate(dx, dy);
20         glutPostRedisplay();
21     }
22 }
```

Hàm này thêm một số nút để di chuyển trái phải lên xuống hình được chọn bằng các nút mũi tên. Nếu không có hình được chọn, sẽ không có gì xảy ra.

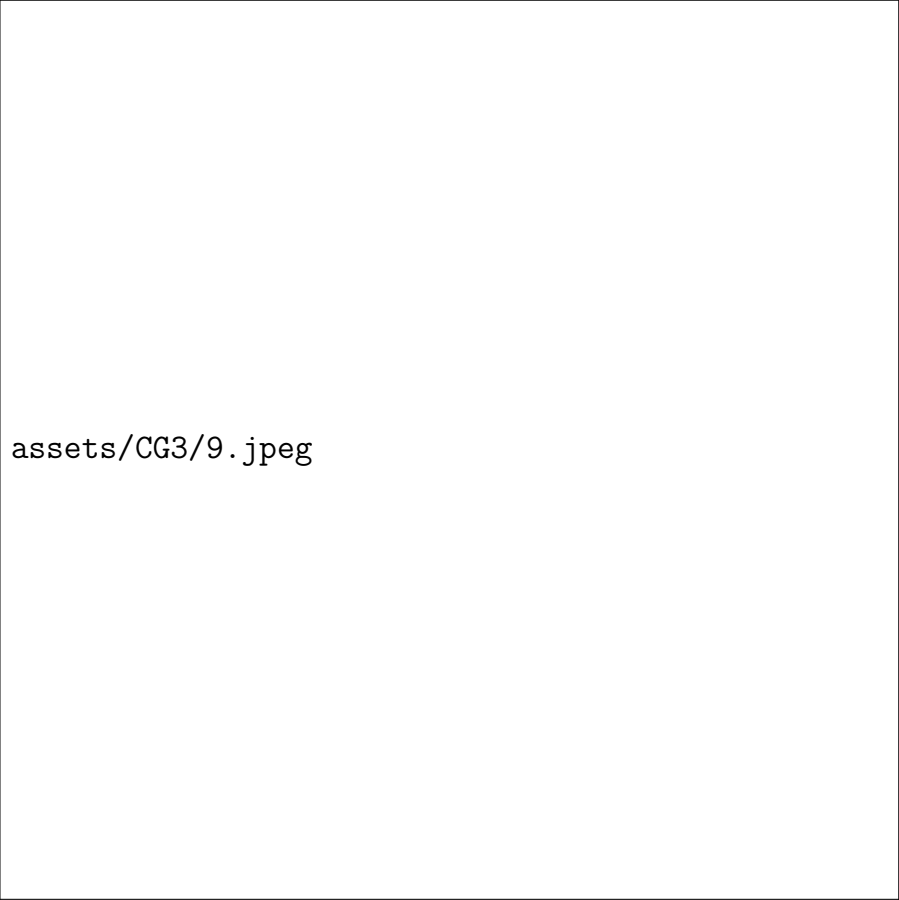
## 3 Lớp Matrix và các phép biến đổi

### 3.1 Lớp Matrix

- Đây là lớp lưu ma trận chứa các biến đổi đối với các điểm của một hình.
- Với mỗi 1 phép biến đổi sinh ra, giá trị ma trận biến đổi mới là tích của ma trận biến đổi đang lưu giữ với ma trận biến đổi.
- Mô phỏng theo lớp System.Drawing.Drawing2D.Matrix trong C#.
- Đối tượng sẽ được khởi tạo với Ma trận đơn vị có kích thước 3x3.

### 3.2 Các phép biến đổi

- Trước khi đi vào các phép biến đổi, ta sẽ làm rõ cách một hình được vẽ, từ đó sẽ dễ hiểu hơn cho các phép biến đổi.



assets/CG3/9.jpeg

- Một hình sẽ được cấu tạo bởi 2 phần, đường viền và màu được vẽ ở trong. Do đó khi các phép biến đổi được áp dụng, nó sẽ áp dụng lên cả 2 thành phần này.
- Các phép biến đổi sẽ được thực hiện (sử dụng hàm TransformPoints và TransformPixels) bằng cách nhân từng điểm trong 2 phần này với ma trận được tạo sẵn từ trước.

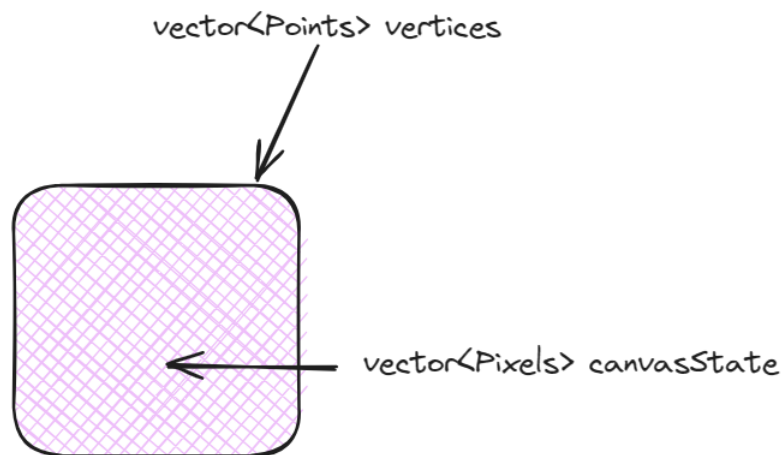
### 3.3 Phép tịnh tiến

$$M = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{pmatrix}$$

Đây là ma trận được sử dụng cho phép quay, với  $t_x$  và  $t_y$  lần lượt là khoảng cách dịch chuyển theo trục tung và trục hoành của hệ tọa độ.



### 3.4 Phép quay



Đây là ma trận được sử dụng cho phép quay, trong đó  $\alpha$  là góc quay. Tuy nhiên để quay quanh chính hình nó. Ta cần tịnh tiến hình đó về gốc tọa độ, thực hiện phép quay và cuối cùng là tịnh tiến nó lại vị trí ban đầu.

```
Title
1 matrix.translate(-xC, -yC);
2 matrix.rotate(rotationAngle);
3 matrix.translate(xC, yC);
4 vertices = matrix.TransformPoints(vertices);
```

Đây là đoạn mã minh họa, các phép biến đổi được lần lượt được nhân theo ma trận đúng thứ tự như thứ tự miêu tả ở trên. Với  $x_C$  và  $y_C$  lần lượt là vị trí của hình vẽ.

### 3.5 Phép co giãn

## Matrix

$$M = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Đây là ma trận được sử dụng cho phép quay, trong đó  $s_x$  và  $s_y$  lần lượt là độ co giãn theo trục tung và trục hoành của hệ tọa độ. Tuy nhiên, để hình được biến đổi co giãn giữ nguyên vị trí ban đầu của nó thì ta cần phải dùng phép tịnh tiến .

```
Title
1 double newXC = scaleX * xC;
2 double newYC = scaleY * yC;
3
4 Matrix matrix;
5 matrix.scale(scaleX, scaleY);
6 matrix.translate(xC - newXC, yC - newYC);
7 filledPixels = matrix.TransformPixels(filledPixels);
```

Đây là đoạn mã minh họa, các phép biến đổi được lần lượt được nhân theo ma trận đúng thứ tự như thứ tự miêu tả ở trên. Với  $x_C$  và  $y_C$  lần lượt là vị trí của hình.

## 4 Demo Hướng dẫn sử dụng

### 4.1 Demo

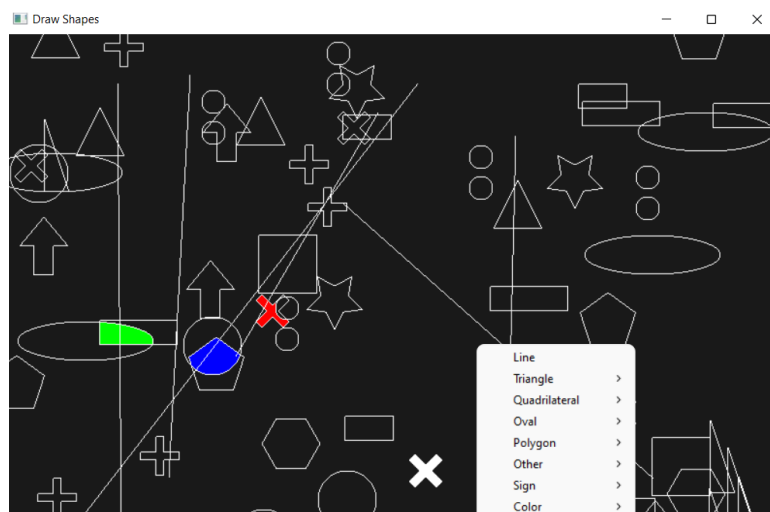
Đây là demo của chương trình: [demo](#).

### 4.2 Hướng dẫn sử dụng

1. Mở folder chứa file build ở dạng Release và chạy file exe.

```
.\CG-Lab2-2.exe
Time taken by boundaryFill: 845213 microseconds
Time taken by boundaryFill: 1426756 microseconds
Time taken by boundaryFill: 1666589 microseconds
```

2. Sau đó ấn chuột phải để mở menu.



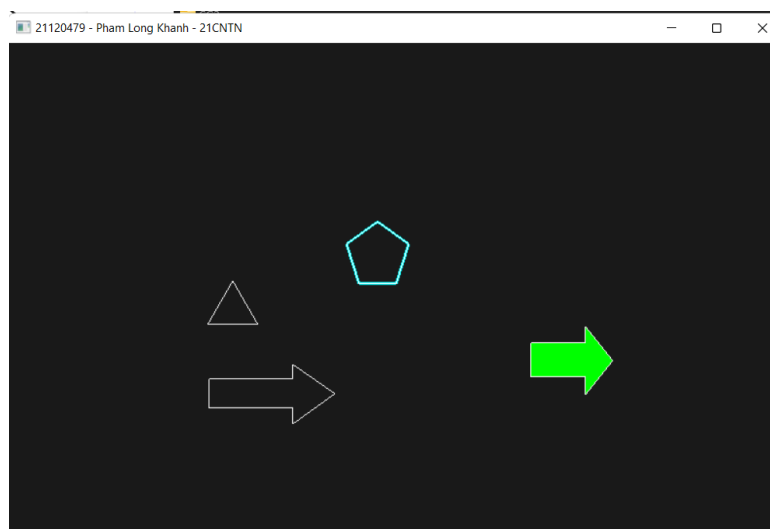
3. Tiếp theo chọn yêu cầu muốn thực hiện.

- Nếu là vẽ hình, chọn hình muốn vẽ sau đó ấn chuột trái để bắt đầu vẽ hình. Giữ chuột trái sau khi nhấn và rê để xác định kích thước của hình đang muốn vẽ
- Nếu là tô màu, chọn điểm bắt đầu tô

- **Chú ý:** Khi tô màu nên chọn những nơi bị giới hạn diện tích và diện tích không được quá lớn. Nếu không sẽ bị tràn stack, vì thuật toán tô màu có sử dụng đệ quy.
- Ngoài ra, thời gian chạy thuật toán tô màu sẽ được in ở màn hình Console.

```
Time taken by boundaryFill: 631779 microseconds
Time taken by boundaryFill: 1888444 microseconds
Time taken by boundaryFill: 930788 microseconds
Time taken by boundaryFill: 375415 microseconds
Time taken by boundaryFill: 416123 microseconds
Time taken by boundaryFill: 849223 microseconds
Time taken by boundaryFill: 632673 microseconds
Time taken by boundaryFill: 1043448 microseconds
Time taken by boundaryFill: 377066 microseconds
Time taken by boundaryFill: 2215867 microseconds
Time taken by boundaryFill: 1321883 microseconds
Time taken by boundaryFill: 633 microseconds
```

4. Click vào bên trong hình để chọn hình. Hình được chọn sẽ có đường viền khác với các hình còn lại. Mỗi lần chỉ được chọn 1 hình duy nhất.



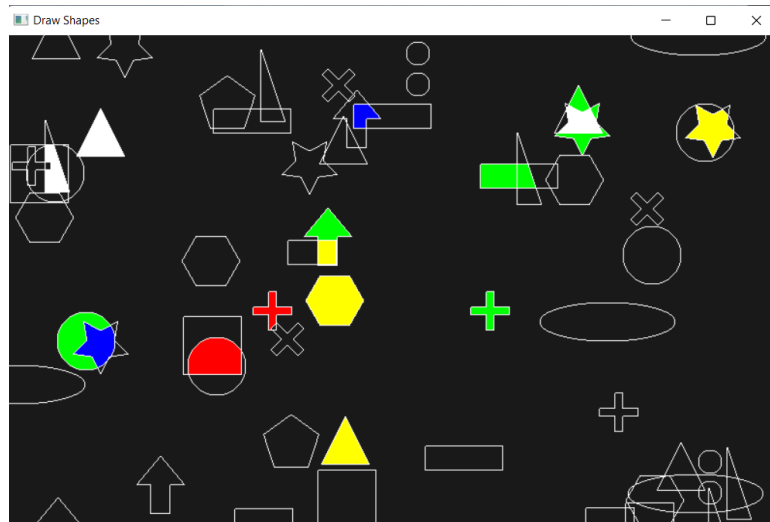
5. Để thực hiện được các phép biến đổi, bắt buộc phải chọn hình, nếu không khi thực hiện sẽ không có gì xảy ra.

Đây là danh sách các nút:

- Quay trái phải (mỗi lần 1 độ):  $\langle l \rangle$ ,  $\langle r \rangle$
- Tịnh tiến theo 4 chiều mũi tên (mỗi lần 1 đơn vị tọa độ): 4 phím mũi tên

- Phóng to thu nhỏ cả 2 chiều (mỗi lần 0.1): <+>, <->

Đây là một demo sau khi vẽ các hình và tô màu một số vùng.



## 5 Các nguồn tham khảo

1. Slide bài giảng của thầy Trần Thái Sơn - Đại học Khoa học Tự nhiên - Đại học Quốc gia Thành phố Hồ Chí Minh (chỉ trên moodle môn học)
2. Slide bài giảng của thầy Trần Võ Hoài Việt - Đại học Khoa học Tự nhiên - Đại học Quốc gia Thành phố Hồ Chí Minh (chỉ trên moodle môn học)
3. Đồ họa máy tính - Dương Anh Đức - - Đại học Khoa học Tự nhiên - Đại học Quốc gia Thành phố Hồ Chí Minh
4. [GeeksForGeeks](#)
5. [Followtutorials](#)
6. [BoundaryFillAlgorithm](#)
7. [OpenGL Programming Guide](#)
8. [Matrix Class C](#)