ĐỒ HỌA MÁY TÍNH

Lab 2: Tô màu đối tượng 2D

1. Nội dung

a. Thuật toán tô màu theo đường bao (Boundary Fill)

Đường biên của vùng tô được xác định bởi tập các đỉnh của một đa giác, đường biên trong thuật toán được mô tả bằng một giá trị duy nhất đó là màu của tất cả các điểm thuộc về đường biên.

Bắt đầu từ điểm nằm bên trong vùng tô, ta sẽ kiểm tra các điểm lân cận của nó đã được tô màu hay có phải là điểm biên hay không, nếu không phải là điểm đã tô và không phải là điểm biên ta sẽ tô màu nó. Quá trình này được lặp lại cho tới khi nào không còn tô được điểm nào nữa thì dừng. Bằng cách này, toàn bộ các điểm thuộc vùng tô được kiểm tra và sẽ được tô hết.

Mã giả (C++):

```
void BoundaryFill (int x, int y, RGBColor F_Color, RGBColor B_Color)
{
    RGBColor currentColor;

    currentColor = GetPixel(x, y);

    if(!IsSameColor(currentColor, B_Color) && !IsSameColor(currentColor, F_Color))
    {
        PutPixel(x, y, F_Color);
        FillLeft(x - 1, y, F_Color, B_Color);
        FillTop(x, y + 1, F_Color, B_Color);
        FillRight(x + 1, y, F_Color, B_Color);
        FillBottom(x, y - 1, F_Color, B_Color);
    }
}
```

Trong đó

- RGBColor is (R,G,B) struct defined as:

```
typedef struct _RGBColor
{
    unsigned char r;
    unsigned char g;
    unsigned char b;
}RGBColor;
```

- IsSameColor: hàm tự cài đặt so sánh 2 giá trị màu
- FillLeft, FillRight, FillTop, FillBottom: được cài đặt tương tự hàm BoundaryFill.

b. GetPixel(x, y)

Hàm GetPixel lấy giá trị màu RGB tại tọa độ (x, y)

```
RGBColor GetPixel(int x, int y)
{
    unsigned char * ptr = new unsigned char [3];

    glReadPixels(x, h -  y, 1, 1, GL_RGB, GL_UNSIGNED_BYTE, ptr);

    RGBColor color;
    color.r = ptr[0];
    color.g = ptr[1];
    color.b = ptr[2];

    return color;
}
```

Lưu ý: Do hàm glReadPixels đi từ dưới lên theo trục Oy, nên chúng ta phải lật ngược giá trị y (y' = h - y), trong đó h là chiều cao của cửa sổ vẽ.

c. PutPixel(x, y)

```
void PutPixel(int x, int y, RGBColor color)
{
    unsigned char * ptr = new unsigned char [3];
    ptr[0] = color.r;
    ptr[1] = color.g;
    ptr[2] = color.b;

    glRasterPos2i(x, y);
    glDrawPixels(1, 1, GL_RGB, GL_UNSIGNED_BYTE, ptr);

    glFlush();
}
```

d. Mouse click event

```
void XuLyMouse(int button, int state, int x, int y)
{
    if(button == GLUT_LEFT_BUTTON && state == GLUT_DOWN)
    {
        BoundaryFill(x, y, F_Color, B_Color);
    }
}
```

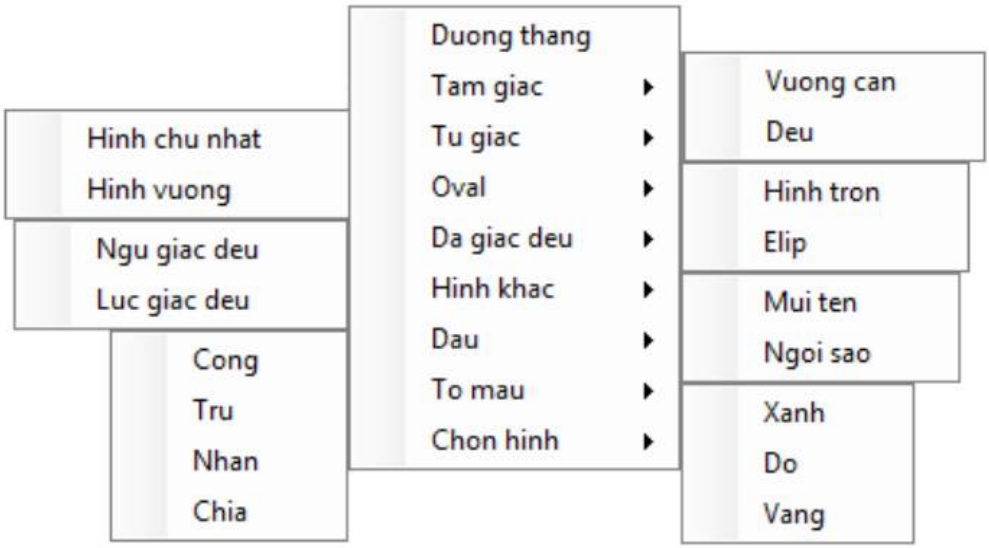
Note:

- Preset: gluOrtho2D(0.0, w, h, 0.0);
- w, h is window's width & height
- F_Color, B_Color are fill color & boundary color

2. Bài tập

Xây dựng chương trình OpenGL có chức năng

- Tự động vẽ các đối tượng hình học cơ bản theo bảng kê bên dưới từ Menu chính của chương trình
- Khi người dùng click chuột vào bên trong của đối tượng thực hiện chọn đối tượng đó
- Dùng Menu tô màu để chọn màu tô cơ bản cho đối tượng đã chọn



Lưu ý: Thuật boundary fill là thuật toán đệ quy, để tránh lỗi tràn stack (stack overflow) nên vẽ các đối tượng tự động với kích thước pixel nhỏ.

Các thuật toán tự tham khảo:

- Flood Fill Coloring Algorithm
- Scan Line Coloring Algorithm

Gợi ý tra cứu:

- http://www.codeproject.com/KB/GDI/QuickFill.aspx
- https://docs.microsoft.com/en-us/dotnet/framework/winforms/controls/walkthrough-providing-standard-menu-items-to-a-form

Output:

- Chương trình được build dưới dạng **Release**, có file dll đi kèm
- Màn hình **OpenGL**: đối tượng được vẽ & tô màu, thời gian tô màu tính bằng mili-second (ms)

- File báo cáo (Word):
    - Nhận xét về độ chính xác của các hàm tô màu đã cài đặt.
    - Nêu các ưu nhược điểm của thuật toán này so với các thuật toán lý thuyết khác được học.

Quy định nộp bài:

- Gồm 3 thư mục, được nén vào file **MSSV_Lab2.zip**
    - Doc: chứa báo cáo
    - Release: chứa chương trình chạy và thư viện đi kèm
    - Source: chứa source code chương trình **(đã xóa các file biên dịch trung gian)**
- Thời gian nộp bài
    - Theo thông báo trên link nộp bài trong moodle