计算机图形学: 实验练习一

Edited by 李民选

Edited on 2021 年 3 月 9 日

王长波 *2021 Spring*

华东师范大学
计算机科学与技术学院

李民选

10185102208

目录

1 实验目的

熟悉 Windows 操作系统下图形 API 的基本功能。

2 实验内容与实验步骤

1. 创建 Visual Studio 项目并配置;
2. 创建第一个窗体;
3. 定义窗体过程 (Windows Procedure);
4. 在窗体中使用 SetPixel;
5. 实现画笔工具。

3 实验环境

说明: 由于没有配置 Visual Studio 2019，这里采用的 IDE 为 CodeBlocks 16.01

1. OS: Windows 10 Home Edition
2. IDE: CodeBlocks 16.01
3. Compiler: Standard MinGW 32-bit Edition GCC 4.9 Series

4 实验过程与分析

1. 窗口初始化 & 创建窗口

   1) 首先要创建窗口类并注册窗口

```
BOOL InitApplication(HINSTANCE hInstance) // 应用初始化
{
       WNDCLASS wc; // Data structure of the window class
       wc.style         = CS_HREDRAW|CS_VREDRAW;
       wc.lpfnWndProc   = (WNDPROC)MainWndProc; // Name of the Window Function
       wc.cbClsExtra    = 0;
       wc.cbWndExtra    = 0;
       wc.hInstance     = hInstance;
       wc.hIcon         = LoadIcon (NULL, IDI_APPLICATION);
       wc.hCursor       = LoadCursor(NULL, IDC_ARROW);
       wc.hbrBackground = (HBRUSH)GetStockObject(WHITE_BRUSH);
       wc.lpszMenuName  = NULL;
       wc.lpszClassName = TEXT("My1stWClass"); // Name of the window class
       return RegisterClass(&wc); }
```

2) 然后将窗口实例化

```
BOOL InitInstance(HINSTANCE hInstance, int nCmdShow) // 实例初始化
{
       HWND hWnd = CreateWindow(TEXT("My1stWClass"), // Name of the window class
       TEXT("Lab0.HelloWindow ***李民选***10185102208***"), // Title of the window
       WS_OVERLAPPEDWINDOW,
       CW_USEDEFAULT,
       CW_USEDEFAULT,
       CW_USEDEFAULT,
       CW_USEDEFAULT,
       NULL,
       NULL,
       hInstance,
       NULL                                    );
       if (!hWnd) return FALSE;
       ShowWindow(hWnd, nCmdShow);
       UpdateWindow(hWnd);
       return TRUE;
}
```

3) 创建窗口

```
int WINAPI WinMain(HINSTANCE hInstance, // 入口函数
       HINSTANCE,
       LPSTR    lpCmdLine,
       int      nCmdShow ){
       if (!InitApplication(hInstance)) // 应用初始化
       return FALSE;

       if (!InitInstance(hInstance,nCmdShow)) // 实例初始化
       return FALSE;

       MSG msg;
       while (GetMessage(&msg, NULL, 0, 0)) // 消息循环
       {
       TranslateMessage(&msg);
       DispatchMessage(&msg);
       }

       return (int)msg.wParam;
}
```

2. 完善窗口回调函数

由于本次实验要用 GDI 库中的 SetPixel 方法来画点，并跟随鼠标的移动轨迹画点。因此我们只要在 WM_LBUTTONDOWN 消息与 WM_MOUSEMOVE 消息下添加相应的功能即可。

这里的颜色是 RGB 模式，三个数值是随机生成的 0 255 以内的数，因此像素点的颜色也会随机变化。

```
// 窗口过程函数
LRESULT CALLBACK MainWndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam
    ){
        HDC hdc;
        int mouse_x;
        int mouse_y;
        int cur_x;
        int cur_y;

        int color1 = rand()%255;
        int color2 = rand()%255;
        int color3 = rand()%255;

        switch (message) {
                case WM_LBUTTONDOWN:
                        hdc = GetDC(hWnd);
                        mouse_x = LOWORD(lParam);
                        mouse_y = HIWORD(lParam);
                        /*for(int i = mouse_x-5; i<=mouse_x+5; i++)*/ SetPixel(hdc,
                            mouse_x, mouse_y, RGB(color1, color2, color3));
                        return 0;

                case WM_MOUSEMOVE:
                        cur_x = LOWORD(lParam);
                        cur_y = HIWORD(lParam);
                        SetPixel(hdc, cur_x, cur_y, RGB(color2, color1, color3));
                        return 0;

                case WM_DESTROY: // 窗口关闭
                        PostQuitMessage(0);
                        return 0;

                default: // 缺省消息的处理
                        return DefWindowProc(hWnd, message, wParam, lParam);
        }
}
```

5 实验结果总结

可以看到，当运行的窗口出现后，然后在空白区域内，按下鼠标，然后移动鼠标，就可以看到一个一个的点绘制在空白区上。松开鼠标左键后，就停止绘制点。

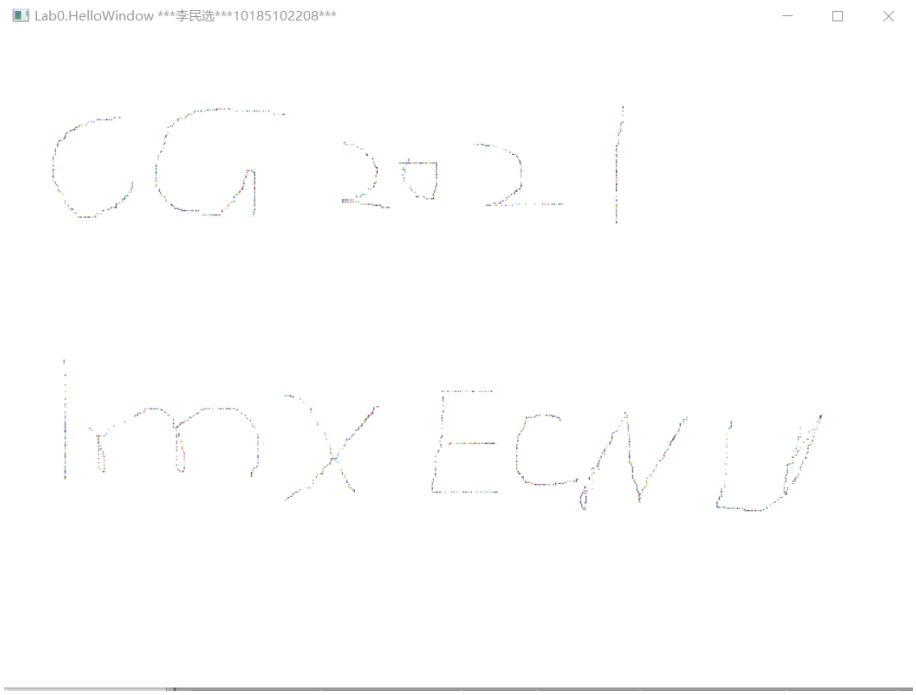

图 1: 实验结果示例

6 附录

1. https://docs.microsoft.com/en-us/windows/win32/learnwin32/creating-a-window

2. https://docs.microsoft.com/en-us/cpp/windows/walkthrough-creating-windows-desktop-appli
view=msvc-160&viewFallbackFrom=vs-2019

3. https://docs.microsoft.com/en-us/windows/win32/direct2d/comparing-direct2d-and-gdi

4. https://docs.microsoft.com/en-us/windows/win32/api/wingdi/nf-wingdi-setpixel