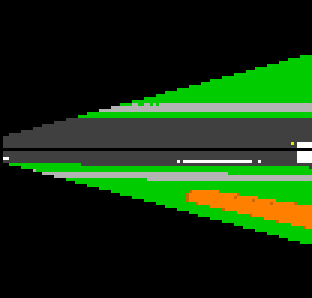
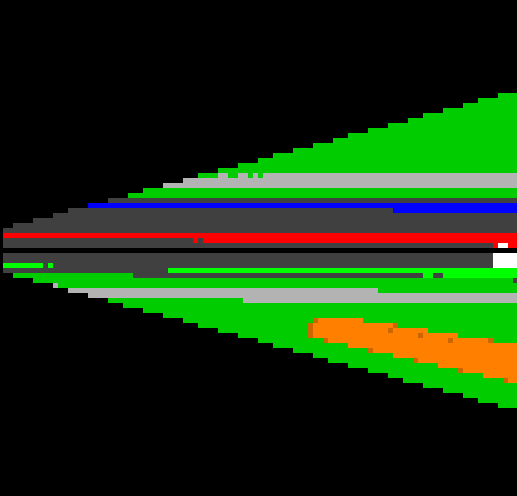
電機所控制組 110318051 張皓博

HW3\_handmade講解

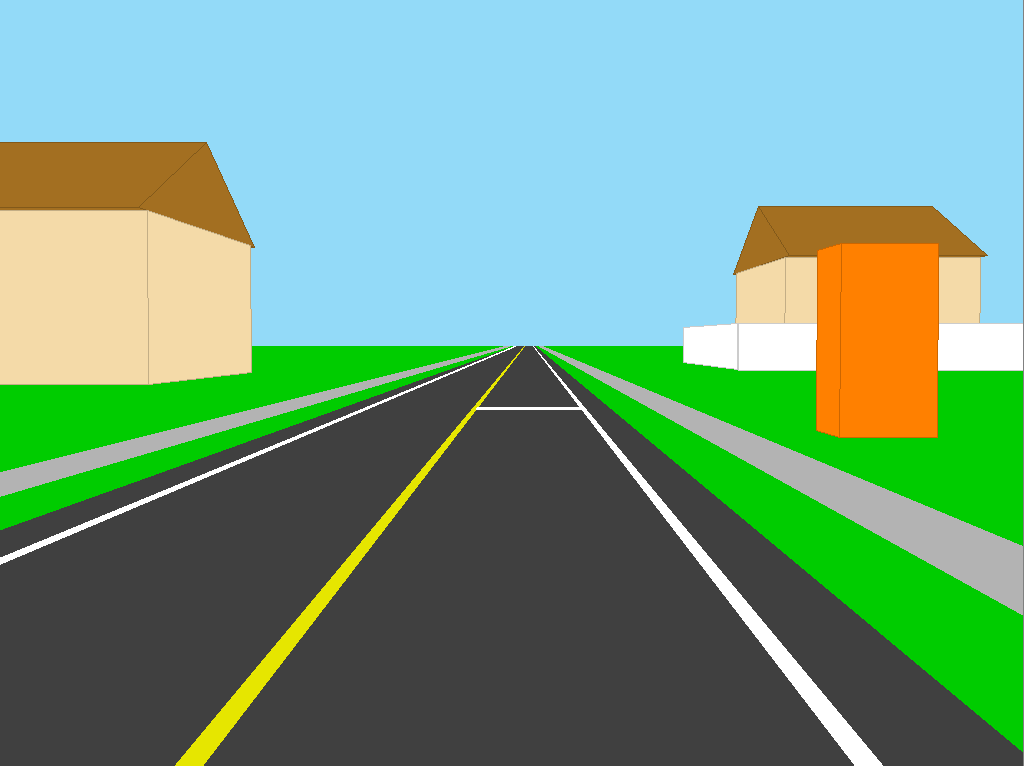
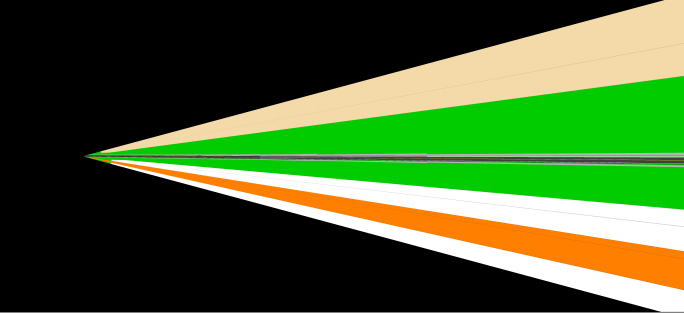
我使用Visual Studio來寫這個C++作業，執行方式為把.exe檔跟road.raw放在同個資料夾底下後，執行.exe即可產生輸出圖檔。

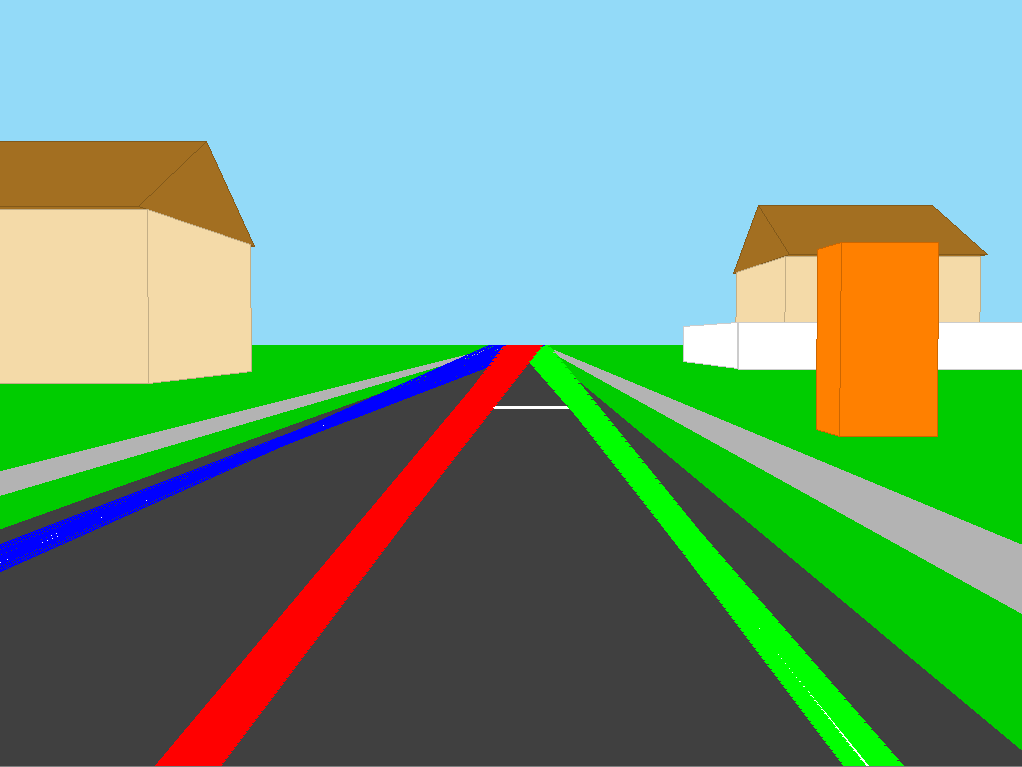
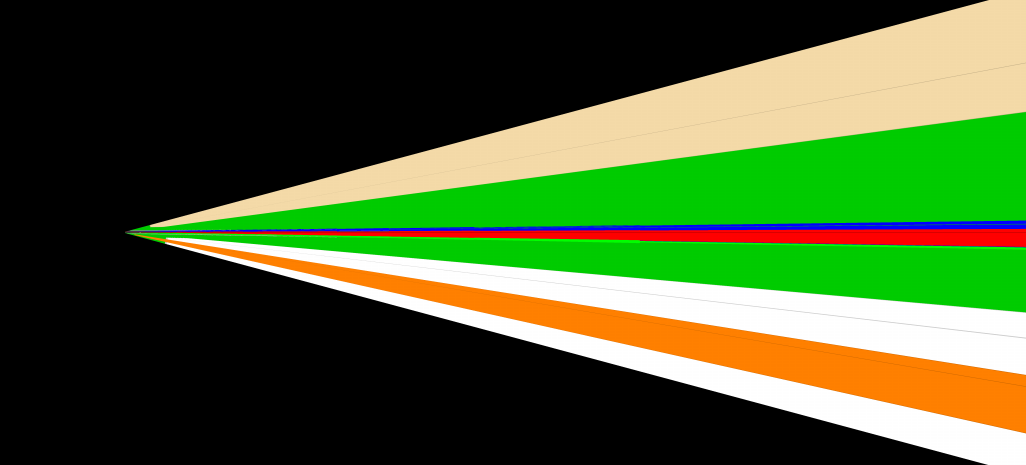
由於這項作業的圖片大小會一直變換，不想調整header故採用.raw。

產出兩張圖：crop\_normal.raw(104x100)、crop\_label.raw(104x100)。

過程圖片 image(1024x768) IPM(6859x3120)：

這次有建立兩個structure，分別為uv\_points和xy\_points，用來存放image coordinate跟world coordinate的二維空間座標點。

總共有11個function：

1. **double cot(double x):**

因為c++的cmath本身沒有cot，利用tan做一個轉換寫成cot的function。

1. **vector<unsigned char> rawRead(const char\* pictureName, int width, int height):**

先利用fopen開啟傳入的Image，然後啟用讀取二進位的模式，利用fread函式將raw data存成一維vector。

1. **void rawWrite(char const\* pictureName, vector <unsigned char> pixels\_Output, unsigned int width, unsigned int height):**

可以傳入圖片名稱、圖片的raw data資訊以及圖片的長寬，方式跟rawRead是一樣的，不同的是從fread變成fwrite，將改造後的raw data依序寫回圖片。

1. **vector<vector<unsigned char>> oneDimension2twoDimension(vector<unsigned char> pixels\_input, int width, int height):**

用以把一維vector轉為二維vector的function。

1. **vector<unsigned char> twoDimension2oneDimension(vector<vector<unsigned char>> pixels\_input2D, int width, int height):**

用以把二維vector轉為一維vector的function。

1. **vector<unsigned char> rgb2binary(vector<unsigned char> pixels\_input, int width, int height):**

**這裡我把rgb2gray跟gray2binary寫在一起成為rgb2binary**

gray = 0.2989 \* r + 0.5870 \* g + 0.1140 \* b

用以把RGB圖轉為二值圖，利用權值計算後轉成灰階，再利用閥值轉換成二值化圖，在這裡我比較暴力的把閥值直接設為三條標線的值，以快速找標線。

1. **vector<unsigned char> EdgeDetection(vector<unsigned char> pixels\_input, int width, int height):**

運用 [2 -1 -1; -1 2 -1; -1 -1 2]的filter找出斜線邊緣。

1. **void DrawLines(vector<vector<unsigned char>>& pixels\_label2D, int row, int col, int color):**

用於HoughTransform時的對線上色，因為比較繁雜，拉出來一個function寫。

1. **vector<unsigned char> HoughTransform(vector<unsigned char> pixels\_input, vector<unsigned char>pixels\_edge, int width, int height):**

這裡利用hough transform的演算法，找到經過EdgeDetection後的直線，再利用斜率去區分三條線，分別上色。

1. **xy\_points uv2xy(double u, double v, int width, int height):**

用於image coordinate轉換world coordinate的點對點function。

1. **uv\_points xy2uv(double x, double y, int width, int height):**

用於world coordinate轉換image coordinate的點對點function。

1. **vector<unsigned char> forwardWarping(vector<unsigned char> image\_coordinate, int width, int height):**

先用uv2xy轉到world coordinate，找出照片的大小，創建一張新的影像，再利用 xy2uv(inverse warping)，找到原圖的點的RGB值，填回IPM image

1. **vector<unsigned char> crop(vector<unsigned char> pixels\_input, int width, int** height):

單純簡單的把原IPM image需要的範圍框框四點記下，再寫入一張新的圖片做回傳。

這次的作業花了超級無敵多時間在研究，雖然論文有把公式都寫出來，但是在座標選擇以及縮放調整時一直出錯，思路需要非常清楚才能將需要轉換的座標範圍選清楚，另外就是畫標線的部分，花了四天的時間在研究霍夫轉換的演算法，雖然能簡單理解了轉換的原理，但在xy的方向、rho的大小、theta的範圍研究許久後，才成功找出我要的線段。