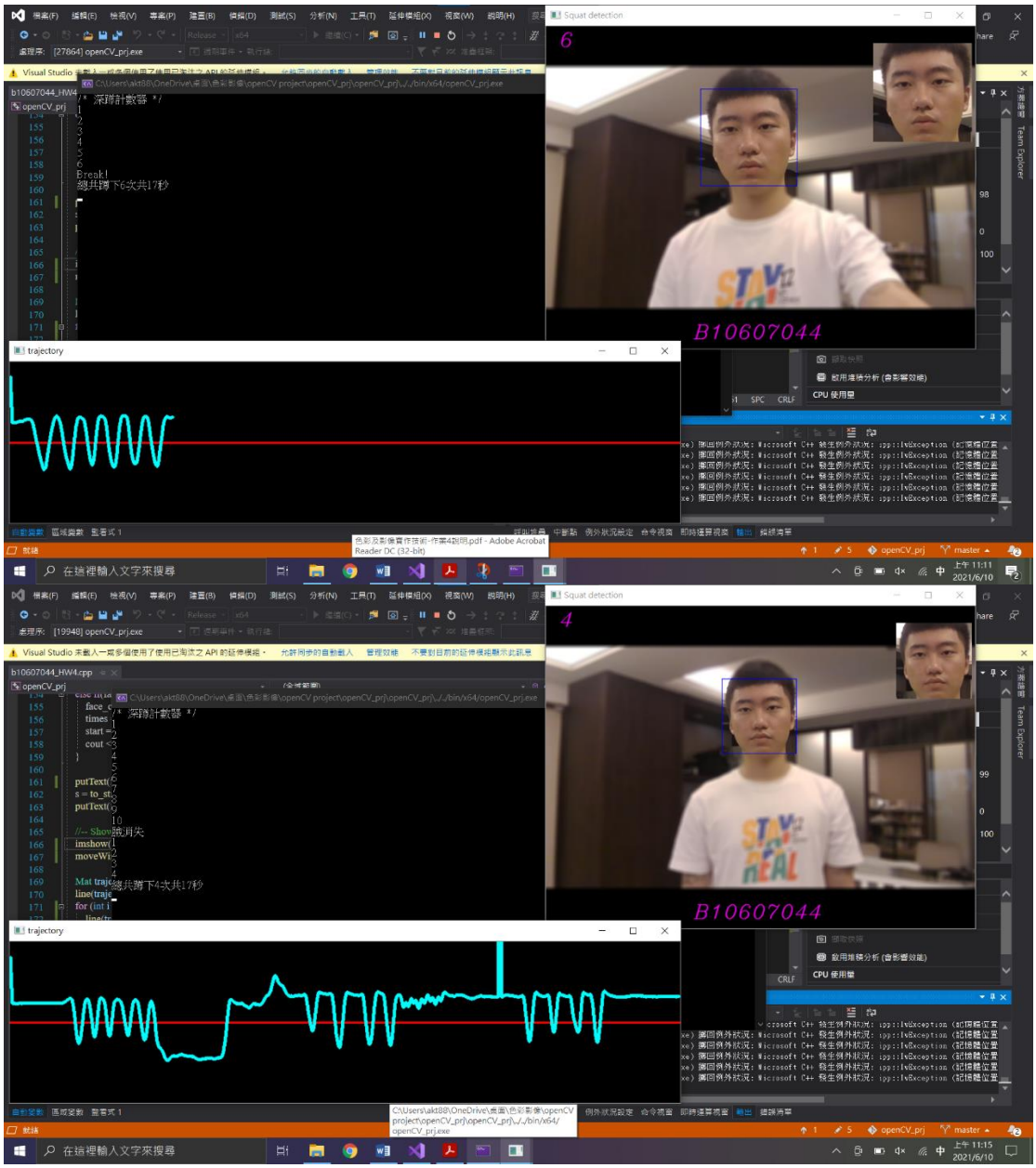
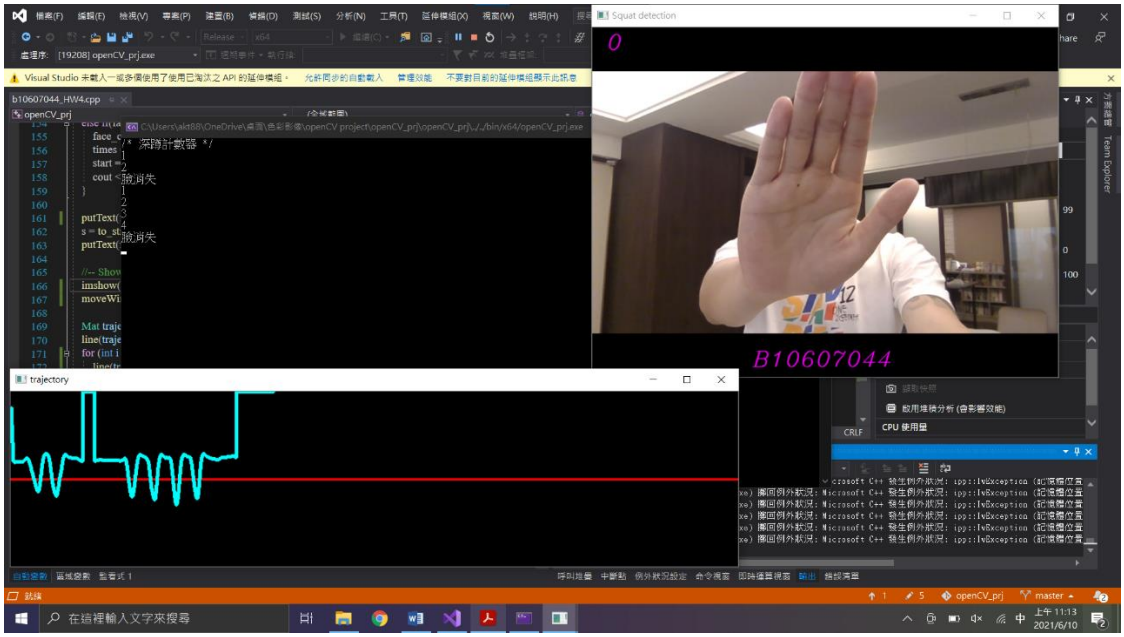
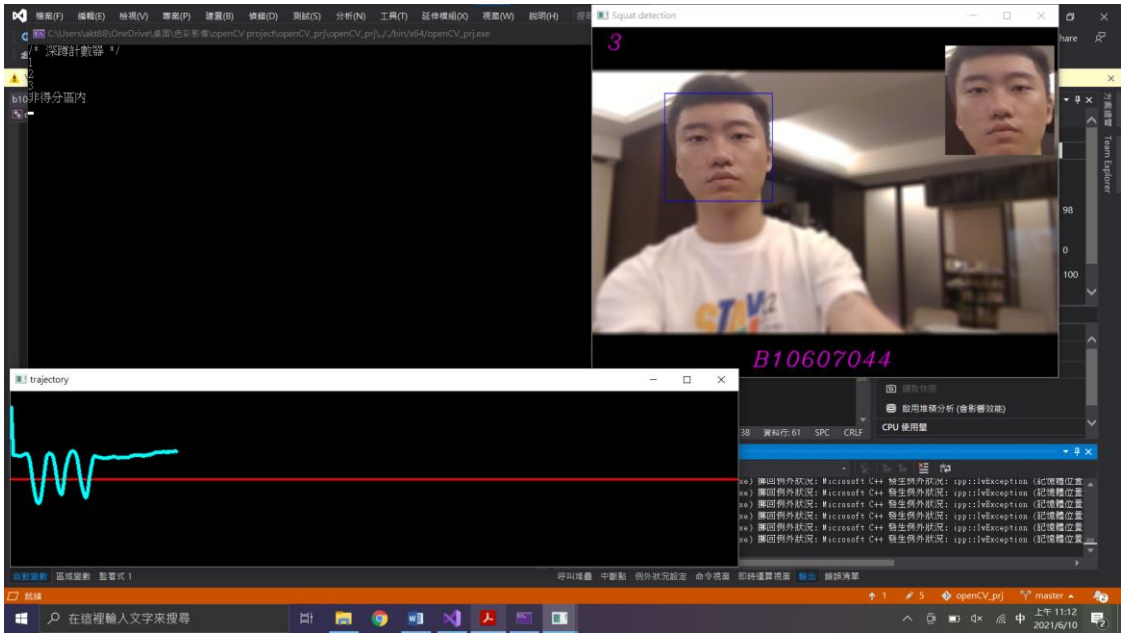
台科大 109 學年度 色彩及影像實作技術

作業四： OpenCV/C++深蹲計數器 B10607044 潘乃聿




Squat detection
0
B10607044
trajectory
/* 深蹲計數器 */
臉消失
臉消失


Squat detection
3
B10607044
trajectory
/* 深蹲計數器 */
非得分區內

程式碼註解

```
#include "opencv2/opencv.hpp"
#include <iostream>
#include <time.h>
#define FRAME_TOTAL 1000 //總共讀的Frame數

using namespace cv; //宣告 opencv 函式庫的命名空間
using namespace std; //宣告 C++函式庫的命名空間

/** Function Headers */
void detectAndDisplay(Mat frame); //偵測人臉與顯示的函式宣告

/** Global variables */
String face_cascade_name = "data\\haarcascade_frontalface_alt.xml";//正面人臉瀑布偵測訓練數據
//放專案下："lbpcascade_frontalface.xml"(跟sln檔放一起)
//相對路徑："data/lbpcascade_frontalface.xml"(放在下一層的data檔案夾)
//絕對路徑："D: / AAA / BBB / CCC / lbpcascade_frontalface.xml"(任意位置)

CascadeClassifier face_cascade; //建立瀑布分類器物件

Mat frame, face_im, face_ROI;
int cross=1; //是否跨越得分線
int frame_no = 0; //畫面編號
int counter = 0; //紀錄波型圖用
int times = 0; //得分
int outside=0; //是否在得分區內
int face_check = 0; //有無偵測到臉
string s; //得分字串
int y_trajectory[FRAME_TOTAL]; //垂直方向軌跡
Rect buffer[4]; //前四個畫面的人臉ROI位置(暫存，用於ROI軌跡平滑化)
vector<Rect> faces;//存臉
clock_t start, End; // 儲存時間用的變數


int main() {
    cout << "/* 深蹲計數器 */" << endl;
    VideoCapture capture;//讀取相機
```

```
	//-- 1. Load the cascades
	if (!face_cascade.load(face_cascade_name)) { printf("--(!)Error loading face cascade\n"); return -1; };

	//-- 2. Read the video stream
	capture.open(0);
	if (!capture.isOpened()) { printf("--(!)Error opening video capture\n"); return -1; }

	start = clock();//紀錄開始時間

	while (frame_no< FRAME_TOTAL)//Frame數固定
	{
		capture.read(frame);//讀Frame

		if (frame.empty())//如果沒讀到
		{
			printf(" --(!) No captured frame -- Break!");
			break;
		}

		//-- 3. Apply the classifier to the frame
		detectAndDisplay(frame);

		frame_no++;//Frame數+1
		counter++;//波型數+1

		int c = waitKey(10);//讀鍵盤
		if ((char)c == 27) {//如果按ESC
			cout << "Break!" << endl;
			break; } // escape
	}
	End = clock();//結束時間

	cout << "總共蹲下" << times << "次共"<< round(double(End - start) / CLK_TCK) <<"秒" << endl;//
印出結果
	waitKey();//等待按鍵盤
	return 0;
}
```

```
//偵測人臉與顯示函式
void detectAndDisplay(Mat frame)
{
    /*人臉偵測部分*/
    Mat im_gray; //灰階影像物件
    cvtColor(frame, im_gray, COLOR_BGR2GRAY);//彩色影像轉灰階
    equalizeHist(im_gray, im_gray);//灰階值方圖等化(對比自動增強)

    //-- Detect faces
    face_cascade.detectMultiScale(im_gray, faces, 1.1, 3, 0, Size(80, 80));

    //有人臉的處理
    if (faces.size() > 0) {
        face_check = 1;//有偵測到臉
        face_ROI = frame(Rect(faces[0].x, faces[0].y, faces[0].width, faces[0].height));//將人臉部分存
ROI
        face_ROI.convertTo(face_im, face_im.type());//將人臉影像存到變數

        //將人臉roi複製到buffer[0]
        buffer[0].x = faces[0].x;
        buffer[0].y = faces[0].y;
        buffer[0].width = faces[0].width;
        buffer[0].height = faces[0].height;

        //將buffer[0]到[3]的 x, y, width, height 四項目分別平均，取代faces[0]，目的是減少人臉矩
形框的抖動
        for (int i = 1; i < 4; i++) {
            faces[0].x += buffer[i].x;
            faces[0].y += buffer[i].y;
            faces[0].width += buffer[i].width;
            faces[0].height += buffer[i].height;
        }
        faces[0].x /= 4;
        faces[0].y /= 4;
        faces[0].width /= 4;
        faces[0].height /= 4;

        //搬移buffer的資料: 使buffer[1]到[3]分別儲存[0]到[2]的資料，也就是最近三個畫面的人臉
```

roi

```
for (int i = 3; i >0; i--) {
    buffer[i].x = buffer[i-1].x;
    buffer[i].y = buffer[i-1].y;
    buffer[i].width = buffer[i-1].width;
    buffer[i].height = buffer[i-1].height;
}
/*計分部分*/
//計算人臉中心點
//根據 ROI 計算人臉中心
Point leftup(faces[0].x, faces[0].y);
Point rightdown(faces[0].x + faces[0].width, faces[0].y + faces[0].height);
Point center(faces[0].x + faces[0].width / 2, faces[0].y + faces[0].height / 2);

//在y_trajectory垂直方向軌跡陣列，記錄這個畫面的人臉垂直方向中心點
y_trajectory[frame_no] = center.y/2;

if (center.x > 200 && center.x < frame.size().width - 200 && center.y > 50)//得分區域判定
{
    outside = 1;//無越界
    if (cross == 0 && center.y < frame.size[0] / 2) {//如已到下界又回上界
        cross = 1;
        times++;//得分
        cout << times << endl;
    }
    else if (cross == 1 && center.y > frame.size[0]/2) //如已到上界又回下界
        cross = 0;
}
else if(outside==1)//如原本在得分區後超出得分區
{
    outside = 0;
    cout   << "非得分區內"<<endl;
}

/*繪圖部分*/
//模糊背景圖
blur(frame, frame, Size(7, 7));
//將清晰的人臉複製到模糊背景圖的人臉位置
```

```
            face_im.copyTo(frame(Rect(buffer[1].x, buffer[1].y, buffer[1].width, buffer[1].height)));
            //繪製人臉矩形框
            cv::rectangle(frame, leftup, rightdown, Scalar(255, 0, 0));
            //將人臉複製到模糊背景圖的右上角
            face_im.copyTo(frame(Rect(frame.size[1] - 5 - buffer[1].width, 25, buffer[1].width,
buffer[1].height)));
        }
        else if(face_check==1){//如果偵測到臉後又消失
            face_check = 0;
            times = 0;
            start = clock();
            cout << "臉消失" << endl;
        }

        putText(frame, "B10607044", Point(frame.size[1]/2 - 100, frame.size[0]-15),
FONT_HERSHEY_COMPLEX | FONT_ITALIC, 1, CV_RGB(255, 0, 255), 1, LINE_AA);//寫學號
        s = to_string(times);
        putText(frame, s, Point(20, 30), FONT_HERSHEY_COMPLEX | FONT_ITALIC, 1, CV_RGB(255, 0,
255), 1, LINE_AA);//寫分數

        //-- Show what you got
        imshow("Squat detection", frame);
        moveWindow("Squat detection", 800, 0);

        Mat trajectory(Size(FRAME_TOTAL,frame.size[0]/2), frame.type(), Scalar(0));//創黑影像
        line(trajectory, Point(0, trajectory.size[0]/2), Point(trajectory.size[1], trajectory.size[0]/2),
CV_RGB(255, 0, 0),2);//劃界線
        for (int i = 1; i < counter; i++) {//畫波型
            line(trajectory, Point(i, y_trajectory[i-1]), Point(i, y_trajectory[i]), CV_RGB(0, 255, 255), 4);
        }
        imshow("trajectory", trajectory);
        moveWindow("trajectory", 0, 500);

}
```