方塊圖形識別
(Using Visual studio C++ with Opencv)

何清模

December 24, 2023

Abstract

本作品是利用 Visual studio C++ 作為開發環境，並借助跨平台的電腦視覺函式庫 Opencv 的工具，將點陣圖型式的彩色照片讀取進來後，配合適當的演算法去辨識矩形塊及計算其個數，並將結果顯示出視窗上面。

Contents

Chapter 1

標頭檔及參考函式庫設定

```
#include <iostream>
#include "opencv2/opencv.hpp"
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/core/utility.hpp>

using namespace std;
using namespace cv;
```

Chapter 2

int main()

2.1 Setting before image processing

```
Mat image = imread("c:/opencv/squares3.jpg", IMREAD_COLOR); // read image form destination

Mat RGB[3]; //declare an array for color image

split(image, RGB); // divided the color image into  RGB image

Mat imageBlue = RGB[0]; // declare an array for original grey  image (Blue)

Mat imageBlueBin = imageBlue.clone(); // declare an array and copy a grey image
                                      //from imageBlue for processing

int row = imageBlue.size().height; // declare and read the height  of image

int col = imageBlue.size().width; // declare and read the width of image

int threshold = 50; // declare and set the proper threhold for binary processing

int SquareMask[9] = { 255, 255, 255, 255, 0, 0, 255, 0, 0 }; // declare and set the mask of square detection

int LocalMask[9] = { -col - 1, -col, -col + 1 , -1, 0, 1, col - 1, col, col + 1 }; //declare and set the mask of
                                                                                   //local searching based on each processing pixel
```

2.2 mplement Mean filter
=> sharp the original grey image

```
int sum = 0;
for (int i = 1; i < row - 1; i++)
{
    uchar* data = imageBlue.ptr(i);
    uchar* dataBin = imageBlueBin.ptr(i);
    for (int j = 1; j < col - 1; j++)
    {
        for (int k = 0; k < 9; k++)
        {
            sum = sum + data[j + LocalMask[k]];
        }
        dataBin[j] = sum / 9;
        sum = 0;
    }
}
```

2.3 implement binary processing

```
for (int i = 0; i < row; i++)
{
    uchar* dataBin = imageBlueBin.ptr(i);
    for (int j = 0; j < col; j++)
    {
        if (dataBin[j] <= threshold)
        {
            dataBin[j] = 0;
        }
        else
        {
            dataBin[j] = 255;
        }
    }
}
```

2.4 implement Square detection algorithm => calculate the number of squares

```
float count = 0;
int Squarecount = 0;
for (int i = 1; i < row - 1; i++)
{
    uchar* dataBin = imageBlueBin.ptr(i);
    for (int j = 1; j < col - 1; j++)
    {
        if (dataBin[j] == 0)
        {
            for (int k = 0; k < 9; k++)
            {
                if (SquareMask[k] == dataBin[j + LocalMask[k]])
                {
                    count++;
                }
            }
            if ((count / 9) == 1.0)
            {
                Squarecount++;
            }
            count = 0;
        }
    }
}
```

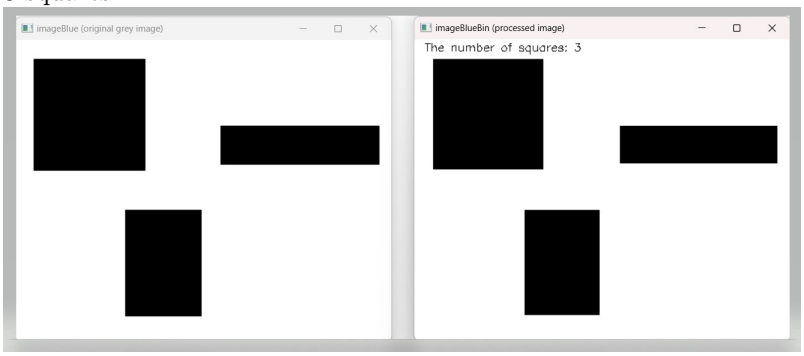
2.5 setting of showing image results

```
string  s = " The number of squares: "; // declare and set the string
                                        //for showing the number of squares
string res = s + to_string(Squarecount); // combine the showing string and
                                         //variable to a string

imshow("imageBlue (original grey image)", imageBlue); // show the original grey image

putText(imageBlueBin, res, Point(5, 15), FONT_HERSHEY_SIMPLEX, 0.5, Scalar(0, 0, 0), 1);

imshow("imageBlueBin (processed image)", imageBlueBin); // shwo the processed image

waitKey(0);
return 0;
```

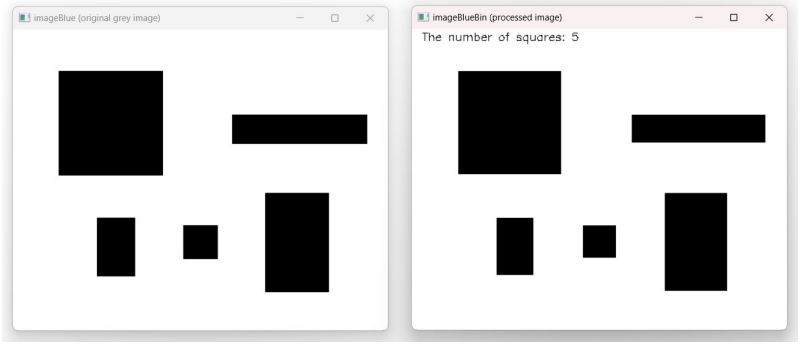
Chapter 3

Results

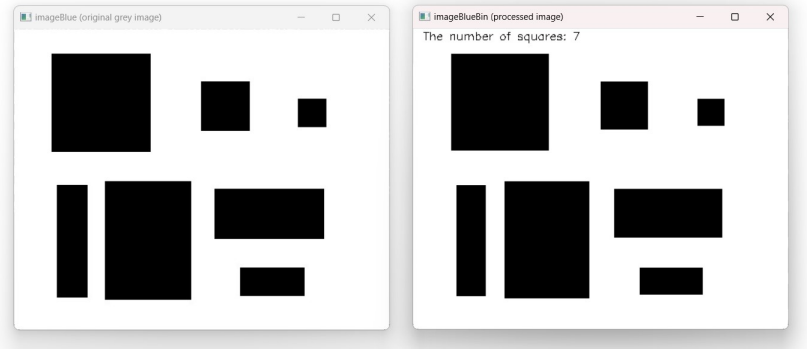
1. 3 squares



2. 5 squares



3. 7 squares


imageBlue (original grey image)
imageBlueBin (processed image)
The number of squares: 7