電腦科學系
Department of
Computer Science

CS4185 Multimedia Technologies and Applications Course Project

GAO Ziliu
54381356

17. Nov 2019

Summary

In a general sense, this program consists of three main modules, a graphic user interface(GUI) implemented by Microsoft Foundation Classes(MFC) a backend logic module interact with OpenCV utilities and a MFC controller that draw the UI and invoke the utilities we implemented in the backend.

To be more specific, each module consists of two files, header file and source file. *CvvImage* module are the computer vision methods we implemented to conduct the required tasks. *MFCApplication1Dlg* module is the MFC controller and the MFCApplication1 module is where we initialize our MFC instance and the managers. Inside *MFCApplication1Dlg.cpp* , function *imageretrieval()* are responsible to invoke OpenCV methods to finish task 1 and *objectdetection()* are responsible for task 2.

Features

Overview

We employ C++ Object Oriented Programming(OOP) technique. We encapsulate every utilities into a C++ object and use the MFC singleton to as our main program. We further use the separation of interface and implementation. We declare the interface in the header files and implement them in the source files. This makes the structure of my program more clear, which avoids make everything in-line, and ease the build of the program, allowing separate compilation.

Image Retrieval

- RGB to HSV
- HSV histogram
- SIFT local feature extraction
- Gaussian filter
- Hough transform

The first thought is use HSV color range space to calculate the histogram. It's more accurate than RGB or GRAY color range space since HSV is closer to human vision. Therefore we first transform RGB to HSV and separate channels and keep H channel and S channel to calculate the histogram graph. Different values of bins in the histogram are experimented during the implementation. At last 15 are used. Till now, the average precision and recall are both 0.3.

Using SFIT to extract local features is considered as well. However, the result is not good.

Then we use hough transform to find the circle. This is because all footballs are of the shape of circle. Considering Houghcircles algorithm are sensitive to the noise, we apply a Gaussian Filter to smooth the picture.

Finally we notice that football are all of black and white. Pixels of black and white will have very close RGB value. We further select the circle base on this assumption. **We reach the**

average accuracy 0.6 and recall 0.6

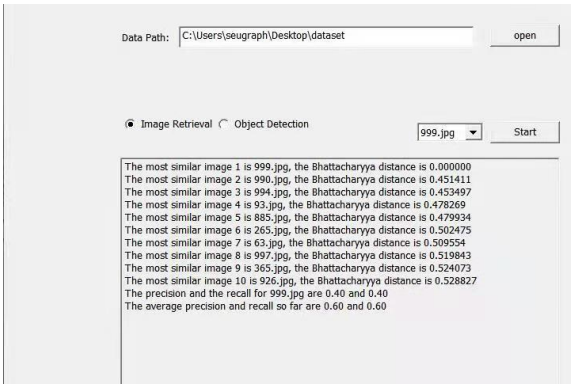
Object Detection

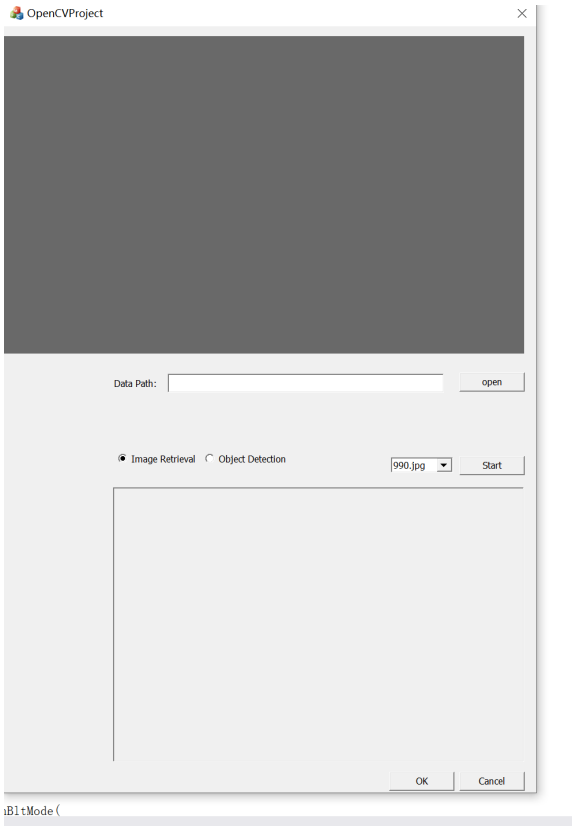
- Hough transform
- sliding window

This task looks very similar to the template matching task. Naturally the first thought will be use template matching method. Then we find that template matching functions of OpenCV is not suitable for resizable(object of different sizes). However, the size of the football is very different in dataset2.

Then we consider Hough transform again. We divide top-10 into 8-2. The first eight will keep the result of sliding window while the last two will use the result of houghcircles. This way we improve the IOU to a great extent. **The accuracy is 0.8 and IOU is 0.63 at last**.

Screenshot





Result

Image Retrieval

filename	baseline precision	mymodel percison	baseline recall	mymodel recall
990	0.10	0.80	0.10	0.80
991	0.20	0.60	0.20	0.60
992	0.10	0.50	0.10	0.50
993	0.20	0.60	0.20	0.60
994	0.10	0.30	0.10	0.30
995	0.10	0.70	0.10	0.70
996	0.10	0.80	0.10	0.80
997	0.20	0.70	0.20	0.70
998	0.10	0.60	0.10	0.60
999	0.30	0.40	0.30	0.40

The average accuracy 0.6 and the average recall is 0.6.

Object Detection

filename	baseline best IOU	mymodel best IoU	mymodel average best IoU
1	0.491859	0.660779	0.602314
2	0.033624	0.890244	0.602314
3	0	0.953065	0.602314
4	0	0.596737	0.775206
5	0	0.661883	0.752542
6	0	0.652367	0.735846
7	0.478485	0.789474	0.743507
8	0.377778	0	0
9	0	0.818594	0.669238
10	0	0	0

The accuracy is 0.8 and the average IoU is 0.63.

Task Distribution

I hereby declare that I finish this project all on my own.