Android OpenCV Report

Li Jingya, Yuan Xiquan
School of Computer Engineering
50 Nanyang Avenue Singapore 639798
[S140033,XYUAN004]@e.ntu.edu.sg

FeiXue
School of MAE
50 Nanyang Avenue Singapore 639798
Xfei001@e.ntu.edu.sg

Abstract

This project aims to generate saliency map based on multiple image features on the platform of Android. OpenCV library of Java is utilized for generation of image features. After that, method of profiling is adapted to measure time consumed by each function and identify the number of OpenCV function calls. We identify that the fully implemented saliency java program can achieve highaccuracy in saliency area recognition.The computation time to process an individualpicture is around 5 minute in android tablet. Color Feature Map Generation takes 60% of the overall computation, since it involves some self-developed java codes. If we implement it in the native C++ code, the processing time may be reduced. After that, we build anandroidapplication based on the saliency code. Once the user takes a photo, the saliency area will be identified automatically. It demonstrates the feasibility of saliency Technology deployment in our daily life.

1. Introduction

Bottom-up saliency map approach encodes topographically for local conspicuity over entire visual scene. It understands bottom-up attention and the underlying neural mechanisms which does not take into account the internal states such as personal experience or goal of the organisms at the time. Therefore it is possible to analyze which part of the scene is selected by neural system for further detailed processing to solve the problem of information overload of perception. The most influential work is attempted is made by Christ of Koch and Shimon Ullman[1]. In this research work, 3 visual features (intensity, color, orientation) are combined into single topographically oriented map which is the final saliency map.

This project works on realization of the saliency map based on the work of Christ of Koch and Shimon Ullman. The platform is Android on Nexus 7. Programming language is java using library of OpenCV. We have also profiled code to compute full saliency stack and analyze overall time consumed and percentage of time taken of each function from the tracing information. The result will be used to optimize code and measure OpenCV performance using profiling tool DmtraceDump.

2. Motivation

One of the most important issues of perception is information overload. Continuous afferent signals are generated by peripheral sensors and the computation cost is high taken into account all the information over the entire time. Thus selection and ordering process of prioritized stimuli, which is called selected attention, is important. Therefore it is motivated to introduce bottom-up saliency map to understand bottom-up factors, which determines stimuli selected or discarded by attentional process.
The platform of generating saliency map is Android using library OpenCV. The number of Android devices has increased significantly over past few years. Android operating system is applied from smart phone to robotics. Statistics suggest thatAndroid devices are becoming the dominant computing platforms. Therefore, we plan to study on the OpenCV using Android before the performance of android devices surpasses that of laptops and desktops.

3. Method

The saliency map is achieved by two main stages. The first stage takes input image and generates three sets of feature maps of color, intensity and orientation. The second stage combines feature maps through across scale addition and map normalization operator.

In the first stage, intensity feature map has an input of Gaussian pyramid of intensity image and center-surround difference is calculated between various Gaussian pyramid scale orders. Color feature map has an similar process as intensity feature map except that the inputs to Gaussian pyramid is from normalized and pre-processed color channel. RGB channel are pre-processed by normalizing it by intensity map to decouple color from intensity. Besides that, two color feature maps are generated based on "color double-opponent" system. Orientation feature map has a different approach from intensity feature map. It takes oriented Gabor pyramids as input and performs center

surround differences process over 4 directions. The Gabor filter is performed based on each output of Gaussian pyramid scale to achieve better results.

In the second stage, a special "map normalization operator" is first performed over feature maps and the results are summed up into 3 conspicuity maps of color, intensity and orientation. This map normalization operator calculate square of difference between local and global maxima and multiply the map by this amount. The purpose is to enhance the contrast when the map includes outstanding active location and suppress the difference when there is nothing unique inside the image.Afterwards, these three conspicuity maps go through the same "map normalization operator" and linearly summed up into the final saliency map.

Besides saliency map, logging tools are inserted into the code to trace the timeline of the code during running time. It can identify the most time-consuming areas in the code. We have taken advantage of these propertiesand optimizing code where cost overwhelming time. Besides that, some profiling tools can be used to produce graphical call-stack diagramsandtime allocation information. In this way it is possible to take a look inside OpenCV and measure the performance.

4. Experiments

The Android code is implemented based on the work of Koch and Ullman. The number of lines regarding core Java codes is around 300. At first place, the Saliency Java code is simulatedin laptop, which would reacha faster speed than android tablet. Therefore the development using laptop is a time-efficient way.After that the Saliency java code is ported into Android developmentenvironmentby adjusting some part of the code. On top of that, the Saliency code is linked to the Android file system to obtain the input images and generate outputs. We have tested two types of inputs. One is pre-stored images and the other is pictures taken from tablet camera. Both images are the color pictures and the output is the black and white, where high intensity value indicates stimuli selected by human attentional process.

Figure 2 shows the promisingexperimental results for the android implementation.The input images are pre-stored images with contents of different levelregarding visual salience. Output saliency map is unsupervised and gives a explicit differentiation of the most outstanding objects by assigning high intensity values. These results indicate that we have successfully generated saliency map for visual scenes in which the most salient location would be a good candidates for attentional selection. We have also achieved similar outputs for inputs from tablet camera as shown in Figure 2.

Figure 1 Example of saliency map result. Left: pre-stored input image. Right: corresponding saliency map.

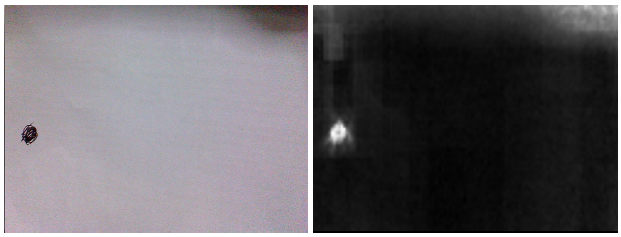

Figure 2 Example of saliency map result. Left: input image from tablet camera. Right: corresponding saliency map.

The main limitation is that the saliency code requiring large amounts of computing power to accomplish saliency picture. The processing time is around 5 minutes for android tablet and 10 seconds for laptop.

The processing times of feature maps generation are listed in the Figure 5. The Color Feature MapGeneration uses60%

of the execution time. After investigate the color feature map generation, the broadly-tuned color is the most time consuming part. We implemented this part using Java, rather than Native C++. This may be the reason, which causes the program slow.It has some room to improve the efficiency if we use some optimization method from OpenCV.Apart from broadly-turned color, the rest of the program still takes few minutes in android tablet.

Step	Module	Processing time (ms)
1	Generate Intensity Feature Map	13740
2	Generate Color Feature Map	43450
3	Generate Orientation Feature Map	2340
4	Generate Intensity Conspicuity Map	1260
5	Generate Color Conspicuity Map	2590
6	Generate Orientation Conspicuity Map	5210
7	Generate Saliency Map	80

Figure 5 The processing time for each module

Dmtracedump is a tool, which can generate graphical call-stack diagrams and measure exclusive elapsed times from trace log files. The trace log file is created during running time. Figure 3 shows the record of an execution of the program. In Figure 4, Mat.nGet, Mat.get and Mat.put functions were executed many times. In the future, we can spend more time to optimize the MAT class. At same time, the Highgui.imread function, which is responsible for image IO, also consumes a lot of computing power. Figure 5 illustrates that relationship between OpenCV function calls.

Usecs	self %	sum %	Method
484974	22.06	22.06	/core/Mat.nGet
384785	17.50	39.55	/core/Mat.get
264359	12.02	65.32	/core/Mat.put
171044	7.78	73.10	/highgui/Highgui.imread
165717	7.54	80.63	/core/Mat.nPutD
124068	5.64	86.28	/core/Mat.type ()
104369	4.75	91.02	/imgproc/Imgproc.resize
69408	3.16	94.18	/core/Mat.n_type
62348	2.84	97.01	/core/CvType.channels
153	0.01	99.89	/core/CvType.<clinit>

Figure 4 Top 10 OpenCV functions based on Exclusive elapsed times

5. Related Work

Visual salience is the distinct subjective perceptual quality which makes some items in the picture stand out from their neighbors and immediately attracts our attention. As a research topic, visual saliency theory has evolved rapidly to produce a wide range of methods. However, their computational cost remains significantly high for real-time applications that require execution at full frame rate (> 25 frames per second (fps)).However, computational modeling of this basic intelligent behavior still remains a challenge. Most of the visual saliency models can be categorized into two main groups, a) biological models and b) computational models. There are many applications built on the saliency technologies. Google Glass adapts color saliency function to assist color blind people. Although Google glass still has some constraints, the initial laboratory tests of Google glass demonstrate the feasibility of a smart glass to use the color salience for colorblind individuals in a variety of real-world activities [1]. Apart from the real time application, saliency technologies can be applied for image data image. Recent NASA missions like the Lunar Reconnaissance Orbiter (LRO) have produced vast archives of surface imagery that must be analyzed to locate landmarks with distinctive visual features, like craters, which provide important information about geologic history and potential mineral resources. Saliency-based landmark detection algorithm can help NASA and private-sector scientists and engineers analyze surface imagery for mission planning, resource prospecting, and scientific research[2].There is a clear need for better tools to tackle the overwhelming amount of imagery data collected by NASA missions in the past, present, and future. This study has demonstrated the potential of an image data mining system based on a content-based image retrieval framework.

The Frame Processing Rate of the OpenCV for Android is faster than that of self-implemented algorithm using Android library with a total average ratio of 0.41. Meanwhile the power consumption per frame test consumes less power than that of self-implemented algorithm using Android library with a total average ratio of 0.39 [3].The OpenCV functions are written in C++ and compiled, but Java is the programming language for Android. In order to run OpenCV, Java Native Interface enables the Java code running in a Java Virtual Machine (JVM) to interact with OpenCV. However, the Java Native Interface adds extra overhead, which results in performance degradation. The overhead is incurred for frame-Based Processing. the more OpenCV functions called per frame, the bigger the cumulative performance degradation [4].Apart from image processing, openCV provides

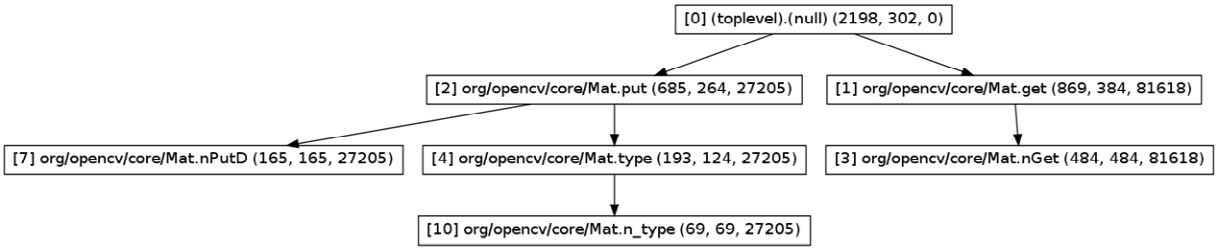

Figure 5 the graphical call-stack diagrams

physics and graphical computation. Bouncing balls game is classic method for performance test. The SDK drops its performance drastically with more balls, while the NDK have almost linear performance degradation with more balls [5].Some OpenCV functions can be finished within 5 milliseconds (e.g. color conversion). In statics, sample size mainly contributes the robustness of the algorithms. The Sample size is determined by camera frame rate. The camera frame rate of Nexus 7 is 30 fps. In some cases, Nexus 7 is able to handle more than 30 frames. Therefore, camera frame rate is a considerable factor for OpenCV performance.

6. Conclusions

In this project, we have achieved 2 main targets. First, weimplement full saliency stackusingOpenCV on Android platform. We develop the java code using laptop at first place, becausethe laptop is faster than android tablet. Once the Java code become stable, we port the Java code into Androiddevelopenvironment.Porting process is straightforward, sinceportability is an important feature of Java language.Second,we used some profiling methods to measure performance of Android devices through overall processing time and each function. The overall processing time for anindividual picture is about 5 minutes. In the saliency stack, the colorfeature map generation takes most of processing time because it contains some self-developed Java codes. If we implement the color feature map generation using Native C++, the overall processing time may be reducedto 3 minutes. We test some pictures and verify the resultsvisually. The accuracy of the saliency stack is acceptable.

[1] Itti, L.; Koch, C.; Niebur, E., "A model of saliency-based visual attention for rapid scene analysis," Pattern Analysis and Machine Intelligence, IEEE Transactions on , vol.20, no.11, pp.1254,1259, Nov 1998 doi: 10.1109/34.730558 for Rapid Scene Analysis,IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 20, NO. 11, NOVEMBER 1998

[2] Santner, K.; Fritz, G.; Paletta, L.; Mayer, H., "Visual recovery of saliency maps from human attention in 3D environments," Robotics and Automation (ICRA), 2013 IEEE International Conference on , vol., no., pp.4297,4303, 6-10 May 2013

[3] Saipullah, K. M. (2012). “OpenCV based real-time video processing using android smartphone.” Int. J. Comput. Technol. Electron. Eng. , 1 (3), 1–6. Read More: http://ascelibrary.org/doi/ref/10.1061/%28ASCE%29CP.1943-5487.0000377

[4] Xi Qian, Guangyu Zhu, Xiao-Feng Li, Comparison and Analysis of the Three Programming Models in Google Android, First Asia-Pacific Programming Languages and Compilers Workshop (APPLC), in conjunction with PLDI 2012, Beijing, China, June 14, 2012

[5] Eric Gregori (31th January 2013) Developing OpenCV computer vision apps for the Android platform, Available at: http://www.embedded.com/design/programming-languages-and-tools/4406164/Developing-OpenCV-computer-vision-apps-for-the-Android-platform (Accessed: 12th November 2014).