# Android OpenCV Project

Fei Xue

Yuan Xiquan

Li Jingya

# Outline

- Introduction
- Method
- Result
- Conclusion

# Introduction

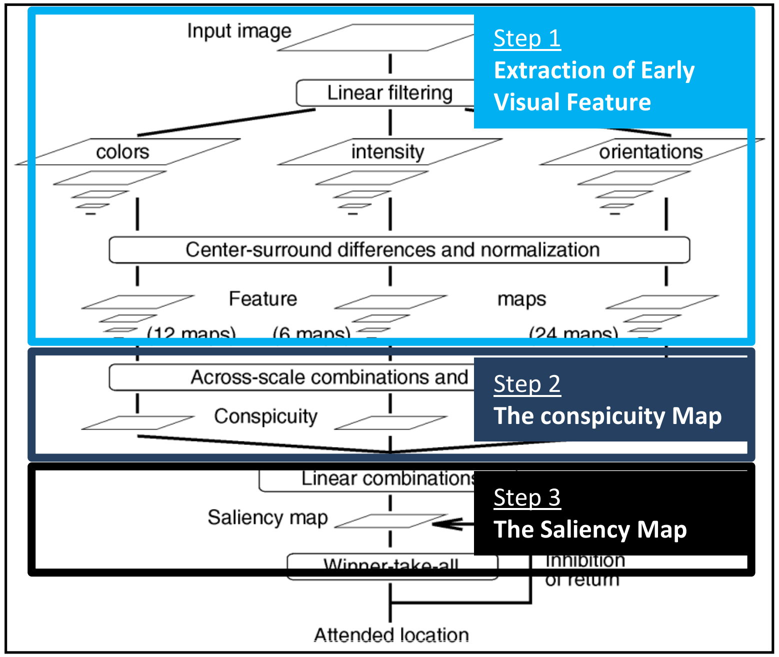
This project aims to realize saliency map on Android and measure OpenCV performance using profiling

- Work environment
  - Platform-Nexus 7 2012
  - Language-Java
  - Library-OpenCV
- Saliency map
  - Represent visual saliency of an image. The most salient location would be a good candidate for attentional selection
  - This project is based on the most influential attempt made by Koch and Ullman (1985)
- Profiling
  - Measure performance of code by tracing time consumption of each part
  - optimize code and measure OpenCV performance using profiling tool “DmtraceDump”

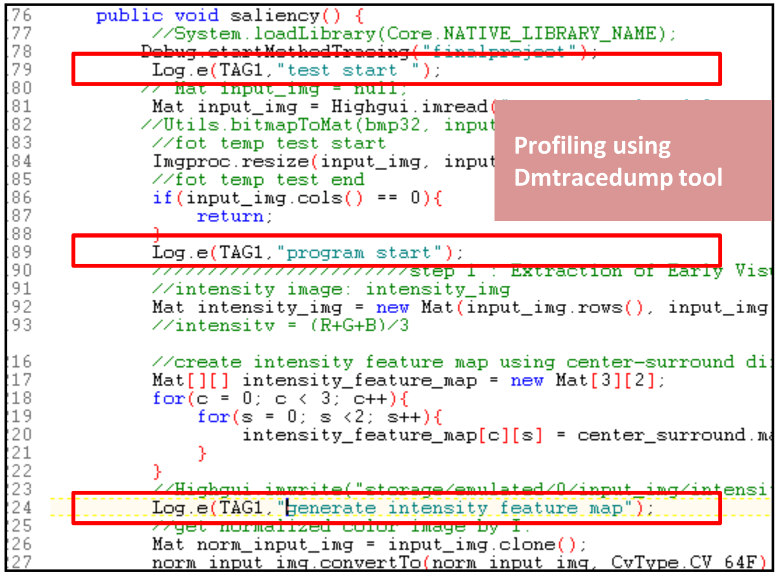
# Method

- The saliency map
  1. <u>Feature maps</u> (color, intensity and orientation) are generated based on extraction of visual feature
  2. <u>Conspicuity maps</u> are generated by applying across-scale addition and “map normalization operator” to features maps
  3. <u>Saliency map</u> is generated by linear combination of conspicuity maps

- Profiling
  – Produce resultsets for OpenCV functions on Android platform
  – Measure time consumption of each step and optimize code based on the result



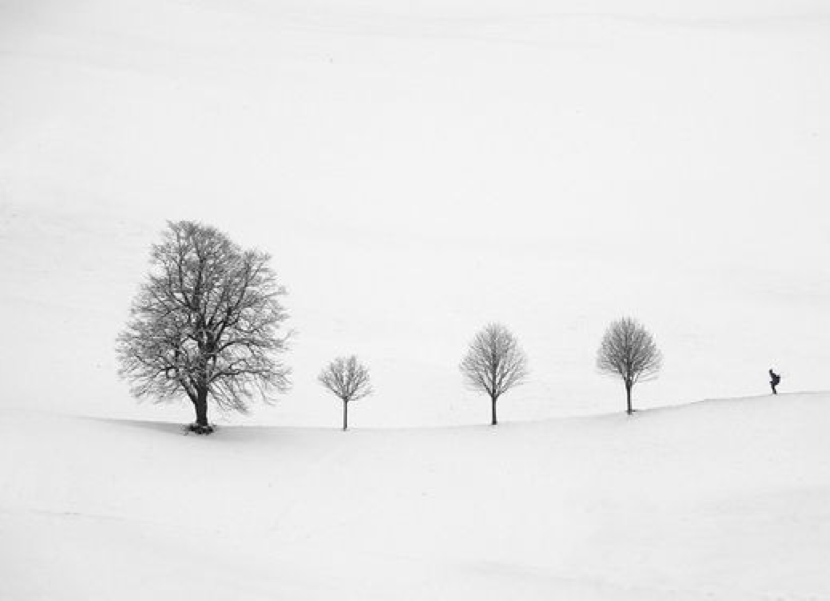
General architecture of saliency map model


```
    public void saliency() {
        //System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
       Debug.startMethodTracing("finalproject");
        Log.e(TAG1,"test start ");
       // Mat input_img = null;
        Mat input_img = Highgui.imread(
       //Utils.bitmapToMat(bmp32, input
        //fot temp test start
        Imgproc.resize(input_img, input
        //fot temp test end
        if(input_img.cols() == 0){
            return;
        }
        Log.e(TAG1,"program start");
        ///////////////////////step 1 : Extraction of Early Vis
        //intensity image: intensity_img
        Mat intensity_img = new Mat(input_img.rows(), input_img
        //intensitv = (R+G+B)/3

        //create intensity feature map using center-surround di
        Mat[][] intensity_feature_map = new Mat[3][2];
        for(c = 0; c < 3; c++){
            for(s = 0; s <2; s++){
                intensity_feature_map[c][s] = center_surround.ma
            }
        }
        //Highgui.imwrite("storage/emulated/0/input_img/intensi
        Log.e(TAG1,"generate intensity feature map");
        //get normalized color image by I.
        Mat norm_input_img = input_img.clone();
        norm_input_img.convertTo(norm_input_img, CvType.CV_64F)
```

Example code of profiling in Android OpenCV platform

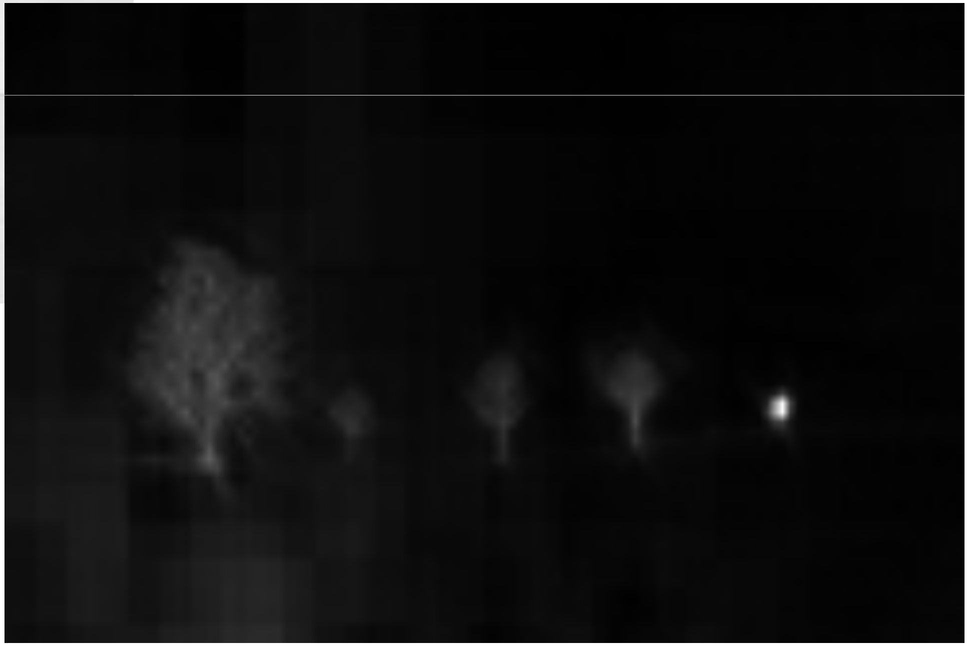
# Result



Input image from internet

Photographer: **Peter Svoboda**

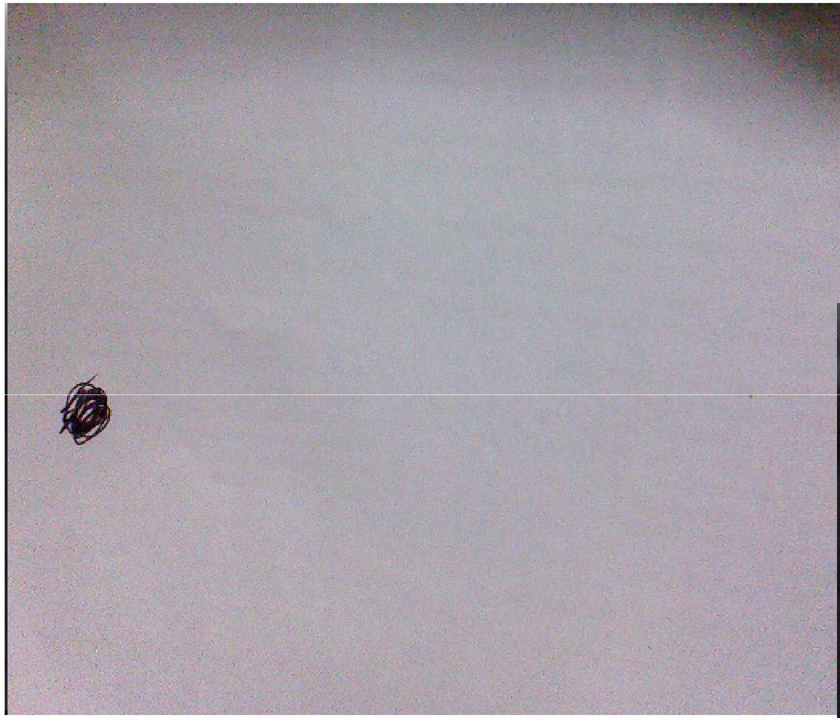
Saliency map generated from pre-stored image. High intensity value indicates stimuli selected by human attentional process



Saliency map

# Result

Saliency map generated from camera input. High intensity value indicates stimuli selected by human attentional process

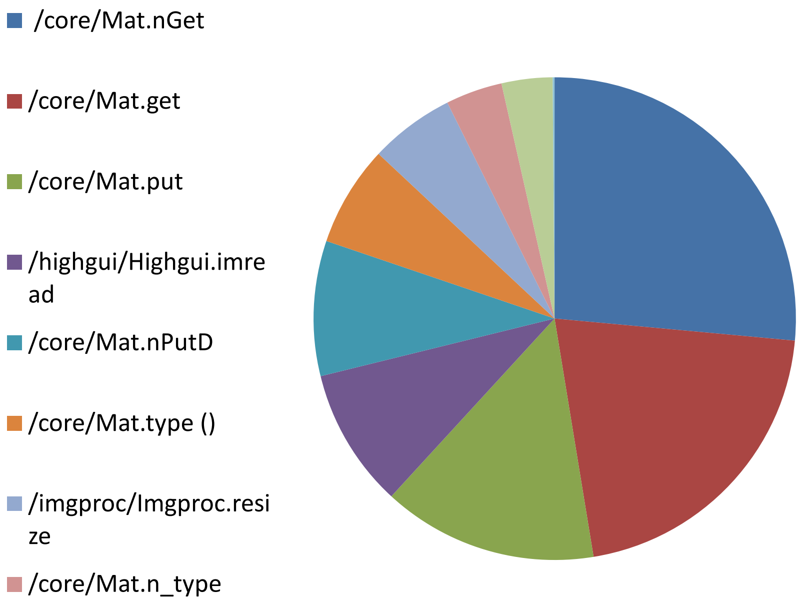


Input image from tablet camera

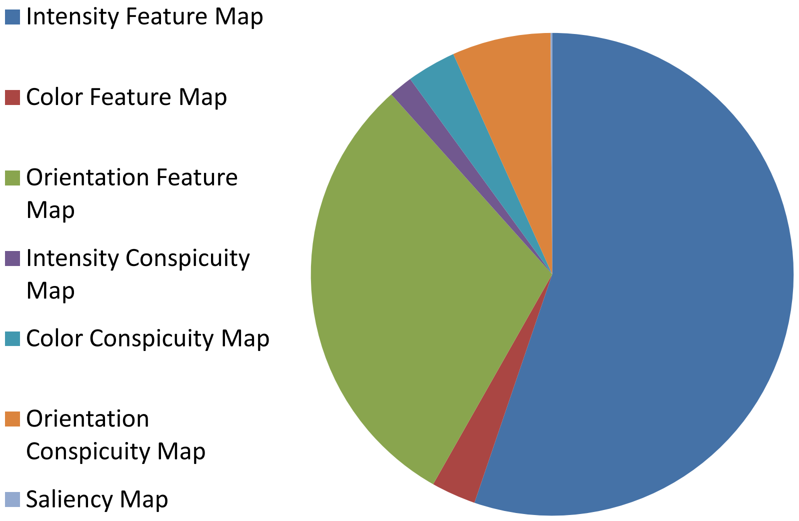
Saliency map

# Result

Profiling results of OpenCV performance measured by elapsed time of core functions (left) and time consumption of generating maps in each step (right)



Top 10 OpenCV functions based on Exclusive elapsed times



Processing time of generating feature maps, conspicuity maps and saliency map

# Conclusion

- Successfully generated saliency map using image features on Android platform
- Measured OpenCV performance and optimized code using profiling method
- Code optimization could be further improved by continuing modifying data structures based on profiling results

Thank you

Q&A