UNIVERSITY OF BORDEAUX

INTERNSHIP REPORT

MASTER OF SOFTWARE ENGINEERING (2013 - 2015)

Design and programming of automatic classification methods applied to biological images

---

Student:
LE Van Linh

Supervisor:
Prof. Marie BEURTON-AIMAR

August 19, 2015

Acknowledgements

First of all, I would like to express my deepest gratitude to my supervisor, Prof. Marie BEURTON-AIMAR for her agreement, guide and support during the planning and development of my internship. I would like to thank the staffs, students in LaBRI, who helped, supported for technique and gave me a professional working environment.

I would also like to thank all professors in University of Bordeaux and PUF-HCM, who imparted a lot of knowledge to learning and researching. I would also like to thank the dean of IT-DLU, who allowed me joined in this course. Finally, I would like to thank my family and colleague for their support and encouragement through my study.

Abstract

Image processing is a field that has many application in life. It can be from the usual application to the application in medicine or cosmology. To obtain the best result, all most of applications must follow two processes: Firstly, we involve primitive operations such as reduce noise, contrast enhancement or image sharpening. Secondly, we can apply the segmentation, description the objects to a form suitable for application process and classification of individual object.

The goal of project is built a program with full functions about processing base on the biological images. During my internship at LaBRI, my tasks are developing the algorithm to preprocessing image and programming of automatic classification methods applied to biological images. After finsihed, I integrated it into the IMP tool, which was developed by NGUYEN Hoang Thao. Besides, we also debug the previous code and write the documentation for the next development.

Contents

List of Figures

Chapter 1

Introduction

1.1 Pôle Universitaire Français

The Pôle Universitaire Français (PUF) was created by the intergovernmental agreement of VietNam and France in October 2004. With ambition is building a linking program between the universities in VietNam and the advanced programs of universities in France. There are two PUF's center in VietNam: Pôle Universitaire Français de l'Universite Nationalé du Vietnam - Ha Noi located in Ha Noi capital (PUF-Ha Noi) and Pôle Universitaire Français de l'Universite Nationalé du Vietnam - Ho Chi Minh Ville located in Ho Chi Minh city (PUF-HCM).

1.1.1 PUF-Ha Noi

PUF-Ha Noi is regarded as a nursery for the linking program, it support on administrative procedure and logistics for the early year of program. About administration, PUF-HN directly under Institut Francophone International (IFI), which was created by VietNam National University at HaNoi in 2012.

1.1.2 PUF-HCM

PUF-HCM[1] is a department of VietNam National Univeristy at Ho Chi Minh city. From the first year of operations, PUF-HCM launched the quality training programs from France in VietNam. With target, bring the programs which designed and evaluated by the international standards for Vietnamese student. PUF-HCM always strive in our training work.
So far, PUF-HCM have five linking programs with the universities in France, and the programs are organized into the subjects: Commerce, Economic, Management and Informatics. In detail:

- Bachelor and Master of Economics : linking program with University of Toulouse 1 Captiole
- Bachelor and Master of Informatics: linking program with University of Bordeaux and University of Paris 6.

The courses in PUF-HCM are provided in French, English and Vietnamese by both Vietnamese and French professors. The highlight of the programs are inspection and diploma was done by the French universities.

[1] http;//pufhcm.edu.vn

1.2 Laboratoire Bordelais de Recherche en Informatique

The Laboratoire Bordelais de Recherche en Informatique (LaBRI)[2] is a research unit associated with the CNRS (URM 5800), the University of Bordeaux and the Bordeaux INP. Since 2002, it has been the partner of Inria. It has significantly increased in staff numbers over recent years. In March 2015, it had a total of 320 members including 113 teaching/research staff (University of Bordeaux and Bordeaux INP), 37 research staff (CNRS and Inria), 22 administrative and technical (University of Bordeaux, Bordeaux INP, CNRS and Inria) and more than 140 doctoral students and post-docs. The LaBRI's missions are: research (pure and applied), technology application and transfer and training.
Today the members of the laboratory are grouped in six teams, each one combining basic research, applied research and technology transfer:

- Combinatorics and Algorithmic
- Image and Sound
- Formal Methods
- Models and Algorithms for Bio-informatics and Data Visualisation
- Programmiing, Networks and Systems
- Supports and Algorithms for High Performance Numerical Applications

1.3 The Internship

The internship is considered a duration to apply the knowledge to the real enviroment. It shows the ability synthesis, evaluation and self-research of student. Besides, the student can be study the experience from the real working. My internship is done under the guidance of Prof. Marie BEURTON-AIMAR in a period of six months at LaBRI laboratory.

1.3.1 Objectives

1.3.2 Overview about my task

1.3.3 Origianization of the document

The all report mainly have five chapters. In the chapter 1, this is the short introduction about my university, mainly information about the lab, where I do the internship and the objectives of my internship. In chapter 2, we talk about the necessary preliminaries in image processing field. In the chapter 3, I propose the algorithm to preprocessing image, with the aim is decrease the noise in the input and increase the effective of the classification methods. In the chapter 4, I mention to the classification process. Finally, I present about the implementation of the preprocessing image algorithm and classification methods.

[2] http://www.labri.fr

Chapter 2

Background

2.1 Overview about image processing

We have a lot of programs what used to edit the photos (e.g. photoshop, gimp, paint,...). By apply some technique, we can effectively some property to change the image such as: scaling image, blurring image, rotating image,.... We also know that, an image is a set of pixels. Each pixel have a value that present for the color at this location, and its location was indicated by coordinates in two-dimension.. When combine the value of all pixels, we have the image as we can see in the real word. The changing on image really changing the value on each pixel in image. Behind the techniques in the programs are mathematical operations and the field using mathematical operation on an input image, called *image processing*. The output of image processing may be either an image or a set of characteristics related to the image. And most of image processing technique are performed on two-dimensional image.

2.2 Smoothing filters

Smoothing filters are used for blurring and noise reduction. This technique is used in preprocessing steps, such as remove some small object unexpected from input image, or bridging of small gaps in lines. Noise reduction can be done by blurring with a linear filter or order-statistics filter.

2.2.1 Linear filter

The idea behind this filter is replacing the value of every pixel in the image by the average of the gray levels in the neighborhood defined by the filter mask. By this work, this filter sometime are called averaging filter. The result of this process is an image with reduced the sharp edges in gray level, it also reduce the noise because the noise is typically and random in the image. The mask is a matrix useful for blurring, sharpening, or edge-detection,.... The output image is accomplished by convoluting between a mask and an image.

2.2.2 Order-Statistics filter

By ordering the pixels in the image and then replacing the value of the center pixel with the value determined by the ranking result. This is the idea of the median filter is the best example used this technique.

2.3 Image transformation

2.4 Histogram

Histogram is a representation about distribution of data on the regions (we called bin) in the data range. The bins are the number of sub-range when we divide the entire data range into several small interval (i.e. With the range from 0 - 255 and the size of each sub-range (bin) is 16, the number of bins is 256/16 = 16 bins. The first bin range is 0 - 15, the second range is 15 - 30, and so on). The value at each bin is the numbers of data which have value belong to this bin. Normally, histogram represented by the columns chart with x-axis represented for the number of bins, and y-axis represented for the value of each bin.
Histogram can be used effectively for image enhancement, also useful in many image processing applications, such as image compression and segmentation.

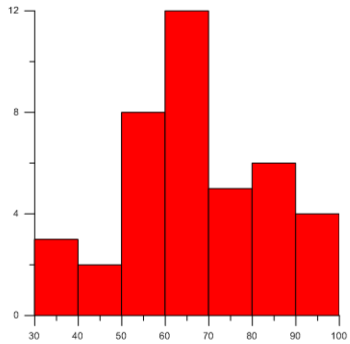


Figure 2.1: An example about histogram

2.5 Segmentation

Segmentation subdivides an image into its regions. The size of regions is depend on the problem being solved. This mean, segmentation should stop when the regions of interest in application have been detected. In the real, the segmentation was applied into many fields such as machine vision, medical imaging, object detection, etc. The most of segmentation algorithms are based on the basic properties of intensity values: discontinuity and similarity. In the first case, the segmentation based on abrupt changes in intensity. The second case, the image segmentation based on a predefined criteria. It means the image was segmented into regions that are similar according to a set of criteria. And, we have many the method to segment an image such as thresholding method, region growing, clustering method, histogram-based method, etc.

Thresholding is a simplest method of image segmentation. Thresholding use a particular threshold value "t", we split the image into two parts: the first part includes pixels which have the value greater than t, and the second part contains the pixels vice versa. With this technique, thresholding can be used to create an binary image from a gray scale image. In fact, we have many type of threshold, as follows:

- *Global thresholding*, when t is a constant over an entire image
- *Variable thresholding*, when t changes over an image

- *Local or regional thresholding*, is variable threshoding in a region of an image
- *Dynamic or adaptive thresholding*, if t depends on the spatial coordinates.
- *Multiple thresholding*, thresholding on 3 dominant modes (color image)

2.6 Color processing

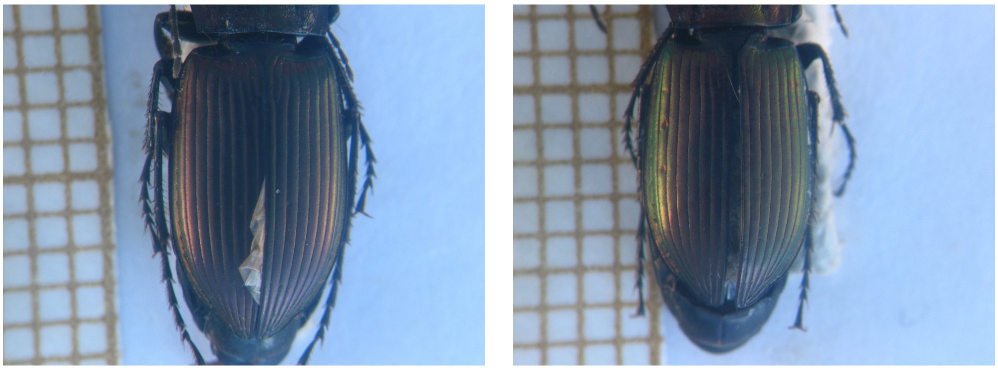
The use of color in image processing do not just identify or extract an objects from scene, it also a factor for image analysis. Choosing the color model solves depends on respective issues

Chapter 3

Preprocessing image

3.1 Problem

The propriety of an algorithm or a program often based on a good input set. To obtain the good result when applying the automatic classification methods. In this chapter, we suggest the algorithm preprocessing image. With the input images contains the parts of insect and an unexpected object, specifically yellow grid (figure 3.1), we need remove the yellow grid to have only insect and just keep the insect.



(a) The yellow gird on the left of insect (b) The insect overlap the yellow grid

Figure 3.1: The input images with yellow grid

3.2 Analysis

Each input image contains the two objects: the part of insect and the yellow grid (called grid). Addition, the grid always stayed in the left of image, and the insect can either overlap the grid or not. About the color, we can see three main groups color: the background color, the yellow color of grid and the color of insect. The image is presented in BGR model. So, the color at each pixel must be combine among three values (blue, green, red). If we process the image in BGR model, the algorithm may be complex. While, the HSV model just has a channel to present the value of color and each color has a clear range. We can apply this property for detecting and removing the gird. So, in this stage, we suggest that we convert the image to HSV model where has a clear range for each color and try to remove the yellow grid.
The analysis system is constructed from two main stages: finding the limiting and replacing points, replacing the yellow point from the begin to the limit point.

3.2.1 Finding the limiting and the replacing point

Browsing all of pixels to checking its color and replacing it if the color of pixel is yellow, we must process on all image. If we do that, it will be waste time. To decreasing the browsing time, in this step we find the limit points. These are the points which located on the right of grid and gird closest.

Finding the limit points will solve the above problem. Instance of checking on all pixel, we just check the pixels stay on the left of limit points. As we know, the width of grid usually less than a two-thirds of width of image. So, to reduce the time to finding the limit point, we also check from the begin of image to two-thirds of image. The result of this step is the limit points, these used for limiting the length when we check the pixels on yellow grid.

The algorithm to find the limit points are followed:

Data: inputImage: The input image (contains the insect and grid)
Result: The coordinate of limit point

```
Declare the variables;
Convert image from BGR to HSV;
Split HSV image into several channel Set up initial limit_point and assign with the
left-top corner.;
Declare a variable yellow_count to count the number yellow points on each columns
when processing. An column become a limit line if the number of yellow points on this
column less than a constant value.;
for j ← 10 to inputImage.columns do
    if H value at (5, j) > 100
    ||(H value at (5, j) > 70
    && H value at (5, j) < 100
    && S value at (5, j) < 10
    && V value at (5, j) > 175 ) then
        limit_point.x ← j;
        limit_point.y ← 0;
        yellow_count ← 0;
        for i ← 1 to grayImage.rows * 2/3 do
            if H value at (i, j) <= 38 then
                yellow_count + +;
                if yellow_count >= 8 then
                    limit_point.x ← 0;
                    limit_point.y ← 0;
                    break;
                end
            end
        end
        if limit_point.x! = 0 then
            break;
        end
    end
end
if limit_point.x == 0 then
    limit_point.x ← inputImage.columns/3 + 200;
    limit_point.y ← 0;
end
```

Algorithm 1: Algorithm to find the limiting points

Now, we indicate which is the color used to replace the yellow points. Hence, we choose the points having the value nearest with the background color. The histogram is ideal for choosing the position to replace, but we also have some conditions to obtain a good value.
The algorithm to find the replacing points are followed:

```
Data: inputImage: the input image
Result: The coordinate of replacing point
Convert image to gray scale image;
Calculate the histogram on gray scale image and mean of histogram;
Find the limit point:;
for i ← 0 to grayImage.rows do
    for j ← 0 to grayImage.columns do
        if value at (i, j) >mean of histogram
        && H value (i,j) > 90
        && H value (i,j) > 130
        && S value at (i,j) > 50
        && V value at (i,j) > 215 then
            return this position ;
        end
    end
end
```

Algorithm 2: Algorithm to find the replacing point

3.2.2 Replacing the grid

After having the limit points. By processing on all rows of image. At each row, we replace the pixels which have the color value stay in the range of yellow by another value. But the grid is not only created by the yellow point, it contains more the pixel have the value stay in the same range with background. But the brightness of these pixels is less than the background. So, we needs to replace it obtained the good image. In each row, this work repeated until meeting the limit points or a "special point" (called "break" point). It can be a point stayed on the insect or a point belong to background.
For each part of the insect, the color on insect or the background also have the difference value. So, we establish the difference values for each part. Based on the file name of image, we can classify it.

Data: filePath: the file path of image
Result: Which part of insect in image

```
QString temp ← filePath.toLower();
if temp contains "ely" then
    return ELYTRE;
end
if temp contains "md" then
    return MDROITE;
end
if temp contains "mg" then
    return MGAUCHE;
end
if temp contains "prono" then
    return PRONOTUM;
end
if temp contains "tete" then
    return TETE;
end
return ELYTRE;
```

Algorithm 3: Algorithm to get the parts of insect

Data: inputImage: the input image; limit_point: the limit point; part: part of insect; minBrightness: minimum of brightness; rpoint: replacing point

Result: The image after replace the yellow grid

```
for i ← 0 to inputImage.rows do
    for j ← 0 to limit_point.x do
        if part is ELYTRE then
            if value at (i, j + 50) satisfy breaking condition then
                break;
            end
        end
        if part is MDROITE or MGAUCHE then
            if value at (i, j + 50) satisfy breaking condition then
                break;
            end
        end
        if part is PRONOTUM then
            if value at (i, j + 50) satisfy breaking condition then
                break;
            end
        end
        if part is TETE then
            if value at (i, j + 50) satisfy breaking condition then
                break;
            end
        end
        if H value at (i, j + 50) in yellow range then
            replace value at this point by the value at replacing point;
        end
        else if V at (i, j + 50) > minBrightness then
            replace value at this point by the value at replacing point;
        end
        ;
    end
end
Merging three channel of HSV;
Convert the image from HSV to BGR;
```

Algorithm 4: Algorithm to replace the yellow grid

3.3 Summary

In this chapter, we propose a method to remove the grid in the image. In short, the algorithm have steps followed:[1]

1. Converting the input image to HSV model
2. Splitting the image (in HSV) to get the individual channel
3. Finding the limit points

[1]The algorithm is combined from the algorithms in each step, which was described above.

4. Choosing the replace point (calculating the histogram and mean value)
5. Getting the type of input and establish the break conditions.
6. Finding and replacing the yellow points and the "miss brightness" point.
7. Merging the channels of HSV
8. Converting the HSV image to BGR image

Chapter 4

Classification methods

In previous chapter, we introduce a method to remove the unexpected object. In this chapter, we will introduce a method to obtain the features that we are interested in and the landmarks on the insect. The method was proposed by Palaniswamy[1].

4.1 Preprocessing image and feature extraction

Feature extraction stage is duration that we extract essential information from the digital images and retaining only these features that we are interested in. To obtain the good result, before extracting the features in the image, we need to pre-process the image with a appropriate technique to reduce the noise as well as enhance the features that we care. The expect result in this result is list of pairwise lines which use to construct the pairwise histogram in the next step.

4.1.1 Preprocess image

In this application, we use the threshold technique to preprocess the image. With a threshold value "t", we can decrease the noise and obtain the interested in features. The threshold value was indicated by the histogram analysis. Based on the histogram of the image, we compute the mean and median of this histogram. After that, we split the histogram into two parts: the first part begin from the bin 0 to the limit value (the limit value is smallest value between mean and median); the second part, starting from the limit value to the end of histogram. For each part, we find the maximum and minimum value and calculating the mean of it. The value "t" obtained by the mean of two mean value in two parts of histogram.
With the threshold value "t", we apply the threshold technique to pre-process image in the CV_THRESH_BINARY mode.

Data: inputImage: the input image
Result: outputImage: the image after processing

Convert the input image into gray scale image;
Calculate the histogram on gray scale image;
Compute the *mean* value and *median* value of histogram;
$limit \leftarrow (mean > median?median : mean)$;
for $i \leftarrow 0tolimit$ **do**
  $imax1 \leftarrow$ maximal bin of histogram ;
  $imin \leftarrow$ minimal bin of histogram ;
end
$middle1 \leftarrow (imax1 + imin)/2$;
for $i \leftarrow limittoend\_of\_bins$ **do**
  $imax2 \leftarrow$ maximal bin of histogram ;
end
$middle2 \leftarrow (imax2 + imin)/2$;
$middle \leftarrow (middle1 + middle2)/2$;
Apply the threshold with threshold value is *middle*;

Algorithm 5: Algorithm to preprocess image

4.1.2 Feature extraction

In this stage, we apply the Canny algorithm to detect the step edges, which incorporates non-maximal suppression and hysteresis thresholding. But, the Canny algorithm is not aware of actual edges, the edge detecting was based on the Sobel operator, extracted with non-maximal suppression. Canny does not connect pixels into chains or segments. So, after using the Canny, we apply another technique find contours, to get the set of edges. After apply the **findContours**, we will have a set of edges, it was presented by a vector of the edges, and each edge was presented by a vector of the points.

4.1.3 Segmentation edge

Now, in this step, we will segment the edges which obtain from the previous step. It mean, we will separate the edge into many straight line. To do that, we apply an algorithm is similar with algorithm proposed by Lowe.

Data: listPoints: list of points which presented the edge
Result: Queue of "step" points on the edge

```
Set up a straight line between the endpoints of the edges (line d);
Initialization the max value: maxDistance ← −1;
Split point: imax ← −1 ;
for point p on edges do
    distance ← from p to line d;
    if distance ¿ max_distance then
        maxDistance ← distance;
        iamx ← position of p;
    end
end
λ ← maxDistance/length(d);
ratio ← maxDistance/length(p0-imax);
if ratio > λ then
    split the list of points into 2 parts;
    preprocess on each part;
end
if imax = −1 then
    push imax into queue;
    // queue is a variable of class
end
```

Algorithm 6: Algorithm to segment an edge

4.1.4 Algorithm

The process to extract the features can be summerised as follows:

1. Calculating the histogram of image (in gray-scale mode)
2. Computing the mean and median value.
3. Computing the threshold value t.
4. Applying the thresholding technique
5. Applying the Canny and findContours to obtained the step edges.
6. Segmentation the edges.

4.2 Pairwise geometric histogram

4.3 Probabilistic Hough Transform

Chapter 5

Implementation

5.1 Software architecture

about architecture of IMP software....

5.2 Image preprocessing

about clear the yellow grid...

5.3 Automatic classification

about methods

5.4 Result

result...

Chapter 6

Conclusion

about conclusion

Bibliography

[1] Sasirekha Palaniswamy, Neil A Thacker, and Christian Peter Klingenberg. Automatic identification of landmarks in digital images. *IET Computer Vision*, 4(4):247–260, 2010.