

## TP Option Analyse d'images (Image analysis practical sessions)

Luce Morin, Marie Babel, Bertrand Coüasnon, Yann Ricquebourg, Éric Anquetil

### Presentation of the project

This project will run during the s7 semester (4 sessions of practical work) and s8 semester (all sessions of practical work) of the option Image Analysis. The goal of the project is to build a pattern recognition application, and to implement it through the 3 following steps:

- Pre-processing and image processing (4 sessions in s7);
- Feature extraction (3 sessions in s8);
- Classification (4 sessions in s8).

The project will be carried out in groups of 3 to 4 students. All groups will work on the same data, and they will aim at obtaining the best classification scores.






At the end of first semester (s7, after the 4 first sessions) and at the end of the second semester (s8, end of project), each group will make an oral presentation of their results and chosen methodology. Each course evaluation will be based on the obtained results and the oral presentation. Methodology, justification of choices, analysis of results will be key points taken into account in the evaluation. A special focus should be done on the methodology used for the evaluation of the recognition quality.










### Project objectives

The data that will be used are extracted from the Niclcon data base.

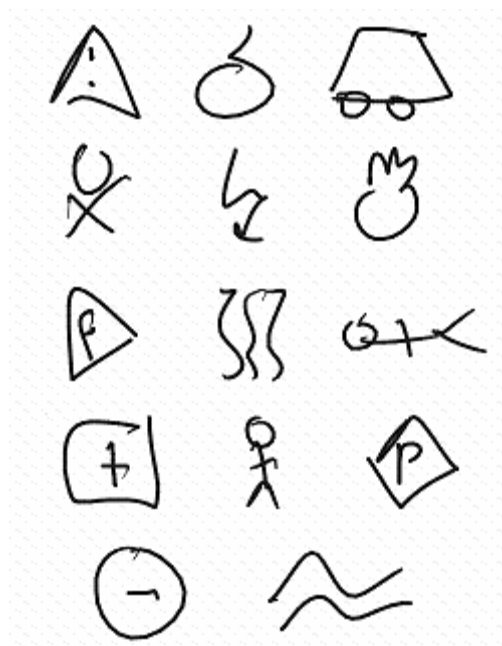
<http://unipen.nici.ru.nl/Niclcon/> (no more available). A copy of the dataset is available on Moodle.

It is a set of 14 icons which can be used by emergency services in a crisis situation (see figure below).

Name	Example	Description
Accident		a triangle with exclamation mark inside
Bomb		a circle with fuse
Car		side-view of a car
Casualty		a circle with an 'X' cross below it
Electricity		a "lightning" arrow with head downwards

Fire		a gesture with flames
Fire brigade		a triangle with character 'F' inside
Flood		two horizontal curly lines
Gas		three vertical curly lines
Injury		a person laying horizontal
Paramedics		a square with a '+' cross inside
Person		a vertical (standing) human figure
Police		a diamond with a 'P' inside
Road block		a circle with a '-' sign inside

These icons allow fast communication to headquarters with tablets, by transmitting information on the accident and thus asking for specific emergency material. Here are some typical examples of transmitted drawings.



### Objective

The goal of the project is to realize an original method for automatic identification of these icons and to evaluate the performances of the proposed method.

### Data Base

The database was obtained by asking 35 volunteer scripters to reproduce each of the 14 icons, in 3 different sizes (large, medium, small).

Each scripiter filled 22 forms. On a form, each row contains: first a printed model of the target icon to be reproduced, then a set of 5 boxes where the icon has to be reproduced by hand. For each scripiter, the first and second forms (numbered 0 and 1) are used for training: they contain each of the 14 icons, without a specification of size.

In the forms numbered 2 to 21, icons are randomly ordered, and the specified size is also random. The order is thus different for each scripiter.

The data base contains:

- on-line data: provides position of the pencil as a function of time,
- off-line data: provides only scanned forms.

In the project, you will only use off-line data.

## Work to be done

The steps of the project correspond to the 3 parts of the course:

### 1.Pre-processing and image processing

- Constitute a ground-truth from the provided data.
  - ⇒ Build a data base containing isolated drawings and associated information: icon, size, scripiter, form id, form location.

### 2.Feature Extraction

- Choose and extract features from each drawing
  - ⇒ Several types of characteristics will be extracted from the drawings. Some of them will be mandatory and you will choose other characteristics that seem meaningful. You will use the WEKA software to display the extracted features, to help selecting the set of most relevant features.

### 3.Classification

- Develop a classifier based on the chosen set of features.

### 4.Evaluation

- Evaluate the performances of the developed classifier.
- Analyze obtained results

## OpenCV library

To facilitate the implementation of image processing steps, you will use the OpenCV library. Version 4.1.2 is installed on Linux.

Documentation:

- Reference manuals are available online:  
<https://docs.opencv.org/4.1.2/>

You will find here tutorials ([https://docs.opencv.org/4.1.2/d9/df8/tutorial\\_root.html](https://docs.opencv.org/4.1.2/d9/df8/tutorial_root.html)), examples (<https://docs.opencv.org/4.1.2/examples.html>)...

You will program in C++ with CLion IDE.

## Part 1: Pre-processing and construction of ground-truth data

### OpenCV and first processing steps

#### Ex1: Test a project for histogram computation with OpenCV

To create your programs in a multi-platform way using OpenCV, you will use a project with CMake. Download a project example for histogram computation on Moodle: "Projet OpenCV-CMake.zip". Unzip it on your home directory.

You will develop on Linux with CLion IDE. To do so you only need to import the project in CLion, then compile and run it.

The provided program allows to:

- Read and display an image
- Reduce the size of the image and display the reduced version
- Extract the RGB components of the image, compute and display the histogram for each of them.

Complete the program to:

- Perform HSV decomposition of the image and display each H,S,V component (see `cv::cvtColor()` on <http://docs.opencv.org>)
- Apply a threshold on one of the components (see `cv::threshold()`)
- Extract a sub-image (ROI, Region Of Interest) and save it in a file (see `cv::imwrite()`)

#### Ex2: Test examples of programs coming with OpenCV

Run some examples from the library (*contours2*, *convexhul...*) you can find in:

<https://docs.opencv.org/4.1.2/examples.html>

You can directly copy paste the source code from the web page and use it to replace the code in the main.cpp file in the histogram project.

You can also simply define a new project from the "Projet OpenCV-CMake.zip".

Some of them require an input image and should be run from the command line.

#### Ex3: Realization of part 1 of the project (Pre-processing and image processing)

- Specify tasks to be realized for step 1
- Propose methods and implement them
- ...

## Appendix: Files to be produced for the end of Step 1

- 1 file READ\_ME  
Explains the method used + performance results: rate of success, quality of extraction, cases of failure...
- 1 data directory containing the set of all snippets (mini-images of extracted drawings) from the base
  - For each snippet: 1 .png file + 1 .txt description file

### Files names:

image file: iconID\_scripterNumber\_pageNumber\_row\_column.png

description file: iconID\_scripterNumber\_pageNumber\_row\_column.txt

### Contents of description file:

# free comment (group name, year...) (other comment lines allowed)

label <labelName>

form <formNumber=scripterNumberpageNumber>

scripter <scripterNumber>

page <pageNumber>

row <rowNumber>

column <columnNumber>

size <small/medium/large (or nothing if size not extracted)>

### Example

fire\_000\_02\_1\_2.png

fire\_000\_02\_1\_2.txt containing

# .....

label fire

form 00002

scripter 000

page 02

row 1

column 2

size medium

### Notes

- Do not keep the square border of the drawings
- Keep all information from the original drawing: snippet = sub-images extracted from non-modified original color images.