Digital Image Processing, Report Car-Plate Recognition

Jhony Herrera - Daniel Ramirez

***Abstract*—The digital image processing has many applications, the application that will be presented in this document is the car-plates recognition, which combines techniques of image processing as filtering and detection of contours in specific regions of the image with algorithms such as k-means with the aim of identifying the alphanumeric characters that make up the mentioned plate. In addition some of the utilities that this technology can have in different areas such as security and medicine will be mentioned.**

I. Introduction

The digital processing of images is a set of techniques that allow to improve or look for information in a digital image, with a wide range of applications that can be useful for the simple improvement of an image as for pattern recognitionand also detection of diseases in many fields of medicine.

The car-plate recognition is a new optical tecnology that involves the techniques of digital image processing, its a process by which computers can recognise license plates. This can be useful in many scenarios for example:

- Automatically opening gates/ doors for authorised vehicles
- Alerting on unrecognised vehicles parking in your car spot or driveway
- Alerting on unrecognised vehicles parking in reserved spots
- Traffic monitoring and recognition
- Car Park usage monitoring

The system of automatic recognition of plates is not only focused on parking, but can be used in all those facilities that need to control, monitor and have a record of all vehicles that pass a certain access. Examples are the private garages of companies, shopping centers, tolls, hospitals.

In this document an application that has been developed in OpenCV with the python language will be presented, the objective is that the application recognizes the characters presented in an image that corresponds to the plate of a car; for this, the image will be digitally processed with filtering and object recognition algorithms and techniques based on neural networks for the algorithm to improve its accuracy.

II. Prior and related work

For the realization of this project it was necessary a previous investigation for the acquisition of the knowledge as base for proposed work.

For the realization of the project it was necessary to learn the basic use of python as well as the management of basic lists and those managed by the numpy library. For k-means, it was necessary to investigate the operation of the OpenCV library and the parameters it receives and returns.

Below are the concepts and techniques applied that were used in the development of the application.

Digital processing of images: [1] It is the set of techniques encompassed within the image preprocessing whose main objective is to obtain, from an origin image, another final whose result is more suitable for a specific application by improving certain characteristics of it that makes it possible to carry out processing operations on it.

The main objectives that are pursued with the application of filters are:

- **Smoothing the image:** reduce the amount of intensity variations between neighboring pixels.

- **Elimitate noise:** eliminate those pixels whose intensity level is very different from that of their neighbors and whose origin can be both in the process of image acquisition and transmission.
- **Enhance edges:** highlight the edges that are located in an image.
- **Detect edges:** detect the pixels where there is a sudden change in the intensity function.

Therefore, filters are considered as operations that are applied to the pixels of a digital image to optimize it, emphasize certain information or achieve a special effect in it.

Car-Plate Recognition: [2] This technology of optical recognition is being used in the reading of vehicle license plates, with the aim of identifying and contrasting the identification elements to inspect the total number of vehicles passing through the streets and avenues of the cities. The system also allows automating access through the use of license plate lists authorized to reduce situations of car theft or internal theft in parking lots.

K-means clustering: [3] [4] is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells.

Theresholding: [5][6] The simplest thresholding methods replace each pixel in an image with a black pixel if the image intensity $I_{i,j}$ is less than some fixed constant T (that is, $I_{i,j} < T$), or a white pixel if the image intensity is greater than that constant. In the example image on the right, this results in the dark tree becoming completely black, and the white snow becoming completely white.

III. Description of algorithm

For the plate recognition algorithm, the python language and the OpenCV library were used for the image processing and the use of the clustering algorithm already implemented in the library called k-means, the operation of the implemented algorithm is described below:

Choose the range of images that you want to use in the program where the first position corresponds to the test image and the others that will be used for the kmeans algorithm.

It is called the function ”image_processing()” which receives two parameters, the list that contains the numbers of the images to be loaded and a Boolean value corresponding to if we want to create the txt file that will be used to recognize the letters and numbers that were recognized in the image manually by the user or load a file already done.

Within the function image_processing() there are four variables that will be used for each of the images to be processed, which are:

- **npaFlattenedImages:** It stores the images corresponding to the letters and numbers that were extracted from the original image which will be stored inside npaFlattenedImages once its matrix has been flattened to a vector.
- **intClassifications:** If the function processing_imagenes() as the second parameter is passed the value of ”True” this list will be used to store the ASCII value corresponding to the letter or number that the person entered when the images of the plates are being recognized.
- **primera_imagen_pass:** This default variable has the value of ”False” because the first image that is loaded will be the image to test the algorithm and this value will serve so that the first image is not used when the person is identifying the numbers and letters of the license plate.
- **cantidad_primera_imagen:** It is used to know the number of characters that were recognized for the first image that will be used as proof.

Using a for are loaded all the images to be used and stored in a list, once finished is proceeded by another cycle the path of each of the images within the list, if the image was not loaded with Success will proceed to show a message where you will be informed and the execution of the program will be

stopped.

If the image was successfully loaded, a gray scale transformation was made and a Gaussian filter will be applied to erase as much noise as possible. Once this is done, the ”AdaptableThreshold()” function of the OpenCV library in the image blurred (figure 1) in gray scale so that the resulting image is one where the letters and numbers that make up the image are blank and the rest in black (figure 2).

Fig. 1. Gray scale

Fig. 2. Black and white

Once the ”adaptiveThreshold ()” is applied, the contours of the image are searched again with the help of the OpenCV library with the function ”findContours ()”, then the area of all the contours found is searched and only those that have an area greater than 600 pixels and smaller than 9500 pixels will be used, these values were found intuitively performing tests until only the contours of the numbers and letters remained mostly.

Once you have the contours you proceed to use the boundingRect() function of the OpenCV library that receives the selected contours to find the minimum rectangle enclosing said contour and returns the position in X and Y where the rectangle starts and the height and width that has rectangle, these four values are stored in a list and then organized ascendingly with respect to the first value that corresponds to the X position of the rectangle, then each rectangle is compared with the others to detect if any is contained in another rectangle, this it is carried out by comparing the first value corresponding to the position X where it starts the rectangles and the position X plus its width of the first rectangle if the second rectangle is contained in the first one its position in a list is saved.

When you have the list with the positions of the rectangles that should not be counted because they are inside another then by means of a for cycle all the rectangles are saved in a new list except those that should not be considered.

If the first image is being processed, the value of the variable primera_imagen_pass is changed to ”True” and the number of rectangles that were found is saved, so as not to consider the first image to be recognized manually by the user.

Then by means of a cycle For it is extracted from the image resulting from having applied the function ”adaptiveThreshold ()” the images corresponding to the numbers and letters inside the rectangles that were previously calculated, then they are resized so that they all have the same width and If you do not choose to recognize the plates manually, you will save the matrices corresponding to the images of each letter or number in a list by flattening the matrix and if you chose to recognize the plates manually,the original image will be shown on the screen with the first rectangle drawn so that the person presses the key corresponding to that number or letter (in capital letters), then the next rectangle will be drawn until the image is finished and the next one will be used until the end all the images, the pressed key is stored and compared if it is a number from 0 to 9 or a letter from A to Z with its ASCII equivalent to avoid pressing a different character, if it is recognized with a valid entry it is saved in a list , also the matrix corresponding to the image of the letter or number in a list is saved after being flattened,finally, the image_processing() method returns a list that contains all the pixels that make up the numbers and letters of all the images, the list that contains the ASCII value if we proceeded to identify each letter and number of the image, and the amount of numbers or letters that the first image has (which will be used to identify it).

With the list that returns image_processing () corresponding to the images of numbers and letters,

all its values are converted to ”float32” with the help of the numpy library, this because the function kmeans () requires that the test data be a list with its values in that format, then proceed to invoke the Kmeans() function of the OpenCV library by passing the list of images, the number of centroids you want (this is 36 by the number of numbers and letters that exists), the stop criteria, the number of times the algorithm is executed using different labels and the position where the centroids will start will be random.

Once the Kmeans() function is finished, the classification made by the person is saved in a txt file if the person decided to identify the images, then through a cycle to create a matrix with 91 columns where the corresponding labels for each number will be stored and letter of the images, the txt created above is read and its data is saved in a list.

By means of a For cycle each label is saved in the matrix that was created from 91 positions in the position that corresponds to the value of each position of the classification that the person made, this because both the classification matrix and the matrix quantity of images of numbers and letters have the same size.

Once the previous For cycle has been completed, the 91 column array is retraced but this time to identify the number that is most repeated in the internal lists since, since it is the number that is repeated the most, it will be the one that identifies each number or letter that corresponds in ASCII to the position of the matrix of 91 column.

Finally we read the list of labels that returns kmeans() up to the value corresponding to the size of the first image and that label is looked for in the 91-position array and if it is found, the chr () function of python is used to have the number or letters corresponding to the position in ASCII format, if it is not found a question mark is placed in its place and the original image is shown with the numbers or letters that were identified and the value is created has the image.

IV. Experimental results

Below are presented graphically the entries received by the program and their corresponding outputs.

Fig. 3. Algorithm input

The algorithm recognizes the contours and encloses them in rectangles to be processed

Fig. 4. imagen original con las letras/números encerrados salida: A?A?H?3

Fig. 5. imagen original con las letras/números encerrados salida: PF0C1P

V. Conclusions

The combination of artificial intelligence techniques and digital image processing shows good results for the identification of characters on the plates of the cars.

The use of kmeans can be very complex and depends on the heuristic that is chosen for its operation and therefore its efficiency

The advantages of digital image processing can be evidenced in each of its applications as it provides many tools for use depending on the need or problem you want to solve

Because the plates have different types of sources, it is more difficult to recognize the characters that are in them.

The amount of test data for are few and therefore do not generate the necessary information to achieve a good solution

REFERENCES

[1] "Procesamiento digital de imagenes." https://es.wikipedia.org/wiki/Procesamiento_digital_de_imágenes.

[2] "Software anpr." https://noticias.autocosmos.com.co/2016/08/12/sistema-de-reconocimiento-de-placas-vehiculares.

[3] "K-means clustering." https://en.wikipedia.org/wiki/K-means_clustering.

[4] "K-means clustering." https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_ml/py_kmeans/py_kmeans_understanding/py_kmeans_understanding.htmlkmeans-clustering-understanding.

[5] "Thresholding." https://en.wikipedia.org/wiki/Thresholding_(image_processing).

[6] "Thresholding." https://docs.opencv.org/3.1.0/d7/d4d/tutorial_py_thresholding.html.