ISSN: 2350-0328

International Journal of Advanced Research in Science, Engineering and Technology

Vol. 7, Issue 1 , January 2020

Automatic License Plate Localization and Recognition Using C++ Programming Language and OPENCV Library

Kudaybergenov Jabbar Kadirbergenovich, Uteuliev Nietbay Uteulievich

Assistant teacher, Department of Software engineering, Tashkent University of Information Technologies – Nukus Branch, Nukus, Uzbekistan

ABSTRACT: Automatic License Plate Number Recognition (ALPR) system is a real time embedded system which automatically recognizes the license plate numbers of cars. This paper presents an alternative method of implementing ALPR systems using C++ programming language and the Open Computer Vision Library.

KEYWORDS: License plate, Computer Vision, Pattern Recognition, C++, OpenCV.

1. INTRODUCTION

In these days the world is deploying research in intelligent transportation systems which have a significant impact on peoples' lives. ALPR is a computer vision technology to extract the license number of vehicles from images. Typical ALPR systems are implemented using proprietary technologies and hence are expensive. In this paper we are going to show implementing such system and development of the system. One of the most important contributions of the open source community to developing such computer software Intel's researches in Computer Vision so called Open Computer Vision (OpenCV) library, which is dedicated to computer vision development and pattern recognition [1].

In Uzbekistan, especially, there are four kinds of license-plates, black characters in white plate and black characters in yellow plate with rectangular and quadratic shape. The first one is for private vehicles and second one is for public service transports and trucks. Our algorithm is only to address the first categories of number plates.

The image of the car is captured using a high resolution photographic camera. But in real situations an Infrared (IR) camera is used. We do not use IR cameras, instead we use regular photographs captured with high resolution photographic cameras.

Steps to locate number plate. In the Fig. 1. we can see the main algorithm steps, plate detection and plate recognition. First we have to detect the plate in the image. To do this task, we divide it in two steps: segmentation and segment classification. Segmentation is the process of dividing an image into multiple segments [1]. This process is to simplify the image for analysis and make feature extraction easier. We will apply a Gaussian blur 5 x 5 and remove noise.

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 7, Issue 1 , January 2020

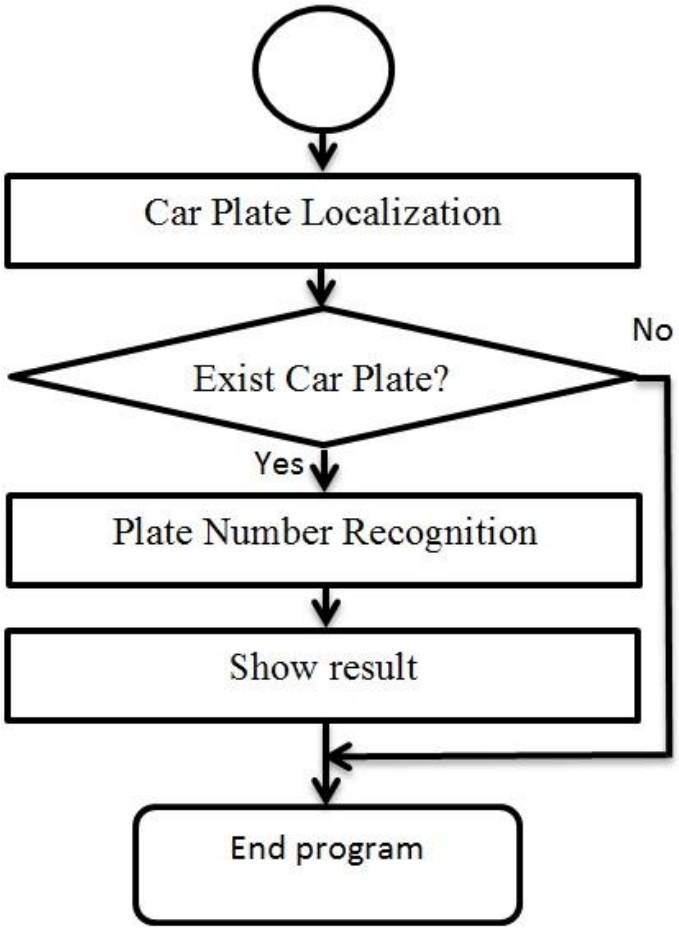

Fig. 1. Algorithm steps for localization and recognition

If we don't apply a noise-removal method, we might get a lot of vertical edges that produce a failed detection.

```
Mat image;
cvtColor(input, image, CV_BGR2GRAY);
blur(image, image, Size(5,5));
```

To find the vertical edges, we will use a Sobel filter and find the first horizontal derivative. The derivative allows us to find the vertical edges on an image. The definition of a Sobel function in OpenCV is:

void Sobel(InputArray src, OutputArray dst, int ddepth, int xorder, int yorder, int ksize=3, double scale=1, double delta=0, int borderType=BORDER_DEFAULT)

Here, *ddepth* is the destination image depth, *xorder* is the order of the derivative by x, *yorder* is the order of derivative by y, *ksize* is the kernel size of either 1, 3, 5, or 7, scale is an optional factor for computed derivative values, delta is an optional value added to the result, and *borderType* is the pixel interpolation method.

After applying these functions, we have regions in the image that could contain a plate; however, most of the regions will not contain license plates. These regions can be split with a connected-component analysis or by using the findContours function [2]. This last function retrieves the contours of a binary image with different methods and results. We only need to get the external contours with any hierarchical relationship and any polygonal approximation results:

```
vector <vector<Point>> contours;
findContours(img_threshold, contours,          // a vector of contours
CV_RETR_EXTERNAL,   // retrieve the external contours
CV_CHAIN_APPROX_NONE); // all pixels of each contour
```

For each contour detected, extract the bounding rectangle of minimal area. OpenCV brings up the minAreaRect function for this task. This function returns a rotated rectangle class called RotatedRect. Then using a vector iterator

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 7, Issue 1 , January 2020

over each contour, we can get the rotated rectangle and make some preliminary validations before we classify each region:

```
vector <vector<Point>>::iterator itc= contours.begin();
vector <RotatedRect> rects;
while (itc!=contours.end()) {
RotatedRect mr= minAreaRect(Mat(*itc));
if( !verifySizes(mr)){   itc= contours.erase(itc);   }
else {  ++itc;   rects.push_back(mr);   } }
```

Today there is a lot of computer software developed to traffic control. Automatic license plate recognition (ALPR) is a very complex process due to diverse effects such as of light and speed. In many cases, systems like ALPR are having been developed using programs like Matlab. In this paper we are going to show how to implement an automatic license plate recognition algorithm in using OpenCV library [3].

There are different approaches and techniques based on different situations, for example, IR cameras, fixed car positions, light conditions, and so on. We can proceed to construct an automatic number plate recognition (ANPR) application to detect automobile license plates in a photograph taken between 2-3 meters from a car, in ambiguous light condition, and with non-parallel ground with minor perspective distortions of the automobile's plate.

The main purpose of paper is image segmentation and feature extraction, pattern recognition basics, and two important pattern recognition algorithms Support Vector Machines and Artificial Neural Networks [1].

Classification. After we preprocess and segment all possible parts of an image, we now need to decide if each segment is (or is not) a license plate. To do this, we will use a Support Vector Machine (SVM) algorithm. A Support Vector Machine is a pattern recognition algorithm included in a family of supervised-learning algorithms originally created for binary classification. Supervised learning is machine-learning algorithm that learns through the use of labeled data. We need to train the algorithm with an amount of data that is labeled; each data set needs to have a class. The SVM creates one or more hyperplanes that are used to discriminate each class of the data.

OpenCV has an easy way to manage a data file in XML or JSON format with the FileStorage class, this class lets us store and read OpenCV variables and structures or our custom variables [2]. With this function, we can read the training-data matrix and training classes and save it in SVM_TrainingData and SVM_Classes:

```
FileStorage fs;
fs.open("SVM.xml", FileStorage::READ);
Mat SVM_TrainingData;
Mat SVM_Classes;
fs["TrainingData"] >> SVM_TrainingData;
fs["classes"] >> SVM_Classes;
```

We then create and train our classifier. OpenCV defines the CvSVM class for the Support Vector Machine algorithm and we initialize it with the training data, classes, and parameter data:

```
CvSVM svmClassifier(SVM_TrainingData, SVM_Classes, Mat(), Mat(),
SVM_params);
```

Our classifier is ready to predict a possible cropped image using the *predict* function of our SVM; this function returns the class identifier *i*. In our case, we label a plate class with 1 and no plate class with 0.

```
vector<Plate> plates;
for(int i=0; i< possible_regions.size(); i++)
{
Mat img=possible_regions[i].plateImg;
Mat p= img.reshape(1, 1);//convert img to 1 row m features
p.convertTo(p, CV_32FC1);
int response = (int)svmClassifier.predict( p );
if(response==1)
plates.push_back(possible_regions[i]);
}
```

First, we obtain a plate image patch as the input to the segmentation OCR function with an equalized histogram, we then need to apply a threshold filter and use this threshold image as the input of a Find contours algorithm; we can see this process in the next figure:

ISSN: 2350-0328

International Journal of Advanced Research in Science, Engineering and Technology

Vol. 7, Issue 1 , January 2020

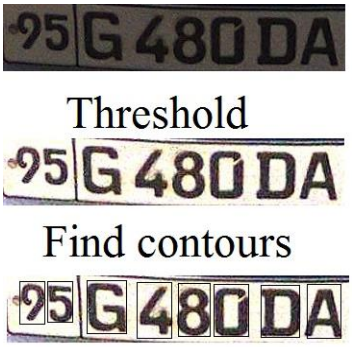

Fig.2. Finding of contours

II. CONCLUSION

The application returns the output command-line error ratio for each sample size. For a good evaluation, we need to train the application with different random training rows; this pro duces different test error values, then we can add up all errors and make an average.

REFERENCES

[1] Abhay C., Nidhi S., Automatic License Plate Recognition System using SURF Features and RBF Neural Network. International Journal of Computer Applications. Volume 70 - No.27, May 2013.
[2] Ankush R., Debarshi P., Number Plate Recognition for Use in Different Countries Using an Improved Segmentation. Volume 3, Issue 6, June 2013.
[3] Duda R., Hart P. Pattern recognition and scene analysis. MIR, Moscow, 1976.