GGCS: A Greedy Graph-Based Character Segmentation System for Bangladeshi License Plate

Rashik Rahman
Computer Science and Engineering
University of Asia Pacific
Dhaka, Bangladesh
17201012@uap-bd.edu

Tanmoy Sarkar Pias*
Computer Science and Engineering
University of Asia Pacific
Dhaka, Bangladesh
tsr@uap-bd.edu

Tanjina Helaly
Computer Science and Engineering
University of Asia Pacific
Dhaka, Bangladesh
tanjina@uap-bd.edu

Abstract—**A License Plate Recognition (LPR) is a technique to automatically identify and recognize the license plate of a vehicle. In today's world where we want to do everything in no time, automated LPR opened a new door in Intelligent Traffic System. With the help of ALPR, we can implement many smart systems starting from the parking facility to crime investigations. For effective License Plate Recognition, accurate License Plate Detection and License Plate Segmentation is crucial. Many existing algorithms are using different image processing and machine learning techniques for detection, segmentation, and recognition. These algorithms are highly computational extensive and somewhat unreliable. In our paper, we propose a computationally light graph-based greedy algorithm for the segmentation of the Bangladeshi LPR system which doesn't require any detection as a pre-process. We tested the algorithm against our dataset consisting of 2000 images that were collected from the streets of Bangladesh for the simulations. It can work on both public and private transport LP and we could segment 1995 images successfully from 2000 images. This segmentation technique will help in the recognition part in real-time and real-life scenarios. Our proposed system works with the current road-legal LP with white background, green background, and black text.**

Keywords—License Plate; Segmentation; Image Processing; BFS

I. Introduction

Automated License Plate Recognition (ALPR) is the first step for implementing many smart applications in our Traffic system. A license plate is a unique identifier of a vehicle containing a registration number which gives the owner information as well. Hence, License plate recognition (LPR) can be used to solve different criminal activities related to vehicles such as car theft, kidnapping, drunk & drive, reckless driving, hit & run illegal vehicles, and many more.

Typically, an ALPR system consists of 3 phases: i) License Plate Detection, ii) Character Segmentation and iii) Character Recognition. The correctness of any phase depends on the correctness of previous phases. Many existing algorithms are using different image processing and machine learning techniques e.g. deep learning, neural network, convolutional neural network (CNN), R-CNN, OpenCV, Template matching, Connected Components at each stage of ALPR. And most of these algorithms are computationally heavy and need an expensive computational system to get the result faster.

Our study is about the segmentation section of the LPR system. We propose a computationally light graph-based greedy algorithm that does not require any expensive or high configuration computational system and can work in a cheap and old computational system. This paper is about a robust graph-based segmentation algorithm of the Bangladeshi license plate of vehicles. Segmentation techniques and modules have made great progress so far but still there some limitations; fail to correctly segment the images rotated about 20◦ onwards, pictures taken in low light or night sight, or from a side angle. Our proposed graph-based segmentation algorithm comes to solve this problem. It is a unique and very fast algorithm and will be able to segment not only Bangladeshi license plate but the license plates of most of the countries. This algorithm can be tweaked to one's needs and is understandable by anyone who has at least little knowledge about graph theory. It's a simple yet unique technique without using any heavy computational demanding modules, framework.

The major contributions of this paper are:

a. We introduce a greedy graph-based segmentation algorithm that can segment the LP of Bangladeshi vehicles.
b. GGCS algorithm can segment the LP from a rotated image (0-360 degrees).
c. GGCS algorithm can segment the LP from a horizontally tilted image (0-85 degrees).
d. GGCS algorithm can segment the LP from a vertically tilted image (0-84 degrees).
e. GGCS algorithm can be used to segment any connected components from an image.

The rest of the paper is organized as follows: Literature review is discussed in section **II**. Overview of the Bangladeshi LP is described in this section **III.** Information on our dataset is given in this part of the paper **IV.** Overview, description, working flow, and breakdown of our algorithm are given in this section **V.** Result of our algorithm are provided in this part of the paper **VI**. The conclusion and discussion of our paper are given in section **VII.** Future work based on our paper is given in section **VIII**. Rest of the part of the reference.

978-1-7281-9090-7/20/$31.00 ©2020 IEEE

II. Literature Review

Many types of research have been done in Automated License Plate Detection/Recognition in the last few years. But, not much progress in Bangla License plate recognition. The first step of ALPR is LP detection. Several methods e.g. boundary edge detection, connected component, color feature, etc. have been proposed and implemented for LP detection. The second stage is segmentation which is a crucial step as the accuracy of final stage recognition depends on the accuracy of proper segmentation.

Du et al. [1] reviewed different techniques of ALP and summarized the techniques used at different stages. They also characterized different algorithms according to different techniques involved at each stage (Detection, Segmentation, and recognition). According to their study, Character segmentation can be done using pixel connectivity [2], projection profiles [3], prior knowledge of characters, character contours [4] [5], and combined features [6].

G. R. Gonc in [7] reviewed ALPR papers and summarized the different segmentation techniques: J. Tan in [8] proposed segmentation methods based on non-linear clustering, D. C. Ciresan in [9] used CNN, P. P. Roy in [10] used Connected Component and Dynamic Programming to segment touching characters. V. Shapiro in [11] proposed an adaptive iterative threshold approach to binarize the image and then using a connected component analysis, segment the characters. There are other techniques e.g. A. P. Nagare in [12] used Gabor transform and vector quantization, J.-M. Guo in [13] used a machine learning approach to segment the characters

G. R. Gonc in [7] also proposed an iterative technique for segmentation. The algorithm is composed of two steps: binarization and find connected components. For binarization, a set of different values for thresholds are used. Starting from a minimum value of 10, they increase the threshold at each iteration of binarization until they found the number of connected components equals to the number of license plate characters.

All these previous papers are for License Plate in English or other languages. Some works are also done for Bangla License Plate Recognition and as part of those License plate character segmentation. R. A Baten in [14] proposed a segmentation algorithm using the connected component where they used the feature called “Matra” and then recognized using template matching. The system would work under the assumption that the digits are aligned and are of the same height.

P. Dhar in [15] used 3 steps recognition similar to many other algorithms. The detection was done using the RGB color information and shape verification. The character segmentation was done using the horizontal symmetric property of the plate and bounding box parameter. The recognition was done using the Adaboost classifier. Nazmus Saif, in [17] uses YOLO for detection and both [16] [17] used CNN for recognition. Recently, smart sensors are also being used for vehicle recognition tasks [18].

III. Bangladeshi Standard License Plate

To drive a vehicle on the Bangladeshi road, the vehicle must have a license plate verified by BRTA (Bangladesh Road Transport Authority). BRTA is the only authority to give license plates or to give road permits. BRTA approved license plates started in 1973 previously it looked different but in 2012 BRTA started giving a new type of license plate called the digital license plate. The public transport vehicles have black text on a green background and private transport vehicles have black text on white background in the digital license plate.

We can divide the license plate into two rows. In the first row, the first segment represents the city name, the second one is always metropolitan and the third character represents the vehicle category. In the second row, the characters represent digits in the Bangla language. Six digits are divided into two segments. The first segment consists of two digits which means the class number of the vehicle and the second segment has four digits separated by a dash (-) from the class number. These four digits are the vehicle number. We will try to segment city number, metropolitan, vehicle number, and all six digits.

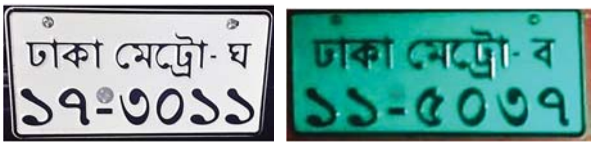


Figure 1. Private and Public Transport License plate

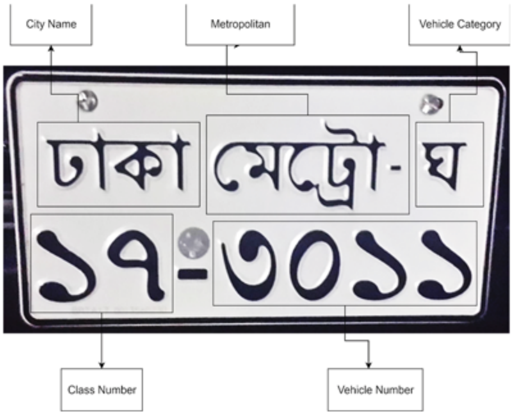


Figure 2. Bangladeshi Standard License Plate

IV. Dataset

We collected 2000 images of public and private transports from streets with the necessary permission to publish the dataset. The images are taken in different situations like at dawn when sunlight is not that bright, in the morning, at noon, and in the evening when there is less sunlight and at night so that we can capture all possible situations based on sunlight availability. Moreover, the images are taken from different angles and perspectives. And for testing our algorithm we took the same license plate image varying the angle from 0 to 85 degree both horizontally and vertically.

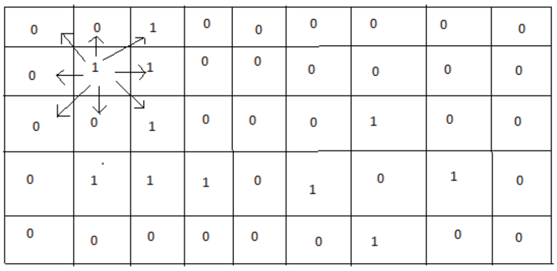
Moreover, we have a plan to build a complete Bangladeshi license plate detection and recognition system. At first, we are building a license plate detection system by using YOLO3. So, we have also added YOLO3 files to our dataset. In other words, our dataset can be used to train YOLO3. Initially, our dataset has 2000 images but we are working to enrich it and make it ten times larger. Figure 3 is an example image from our dataset.

Figure 3. A sample image from this dataset

V. Proposed Methodology

After selecting a 24-bit RGB color image of a license plate, it is first converted into an 8-bit grayscale image. We don't need to preserve the color information of the license plate as it has no contribution to our segmentation algorithm. Then the grayscale image is converted into a 2D array of integers ranging from 0 to 255. After that, the grayscale image is converted into a binary image with an adaptive threshold. The pure black and white binary image has only two unique values. One of them is 0 which represents the black region of the image and the other value is 255 which represents the white portion of the image. But for simplicity, we mapped the 0 and 255 to 1 and 0 respectively. Now we have a binary image that has values 0 and 1 representing the white and black regions respectively.

As the Bangladeshi standard license plate has a white background and black characters. So, the background region is represented by the value 0 and all characters are represented by value 1. Let's call it license plate binary matrix and in short LPB matrix. Figure 4 represents the 8 neighbors from one pixel.



0	0	1	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0
0	1	1	1	0	1	0	1	0
0	0	0	0	0	0	1	0	0

Figure 4. 8-neighbors of one pixel

Our proposed algorithm is a greedy version of standard BFS traversal algorithm. The image is converted into a binary matrix. So, we traverse from start to end of the matrix. To prevent from visiting the same pixel of the binary matrix we create a Visited matrix initially values are set to 0 being the same size as the LPB matrix.

Our algorithm starts from the left upper corner and traverses from top to bottom in a column then moves to the next column. When it gets a 1, it checks the corresponding cell of the Visited matrix and if it is 0, it replaces that the corresponding cell of the Visited matrix with 1. Then it starts BFS from that cell. As we know BFS enqueues the adjacent node into the queue. As we are only concerned with the pixel value 1, all the unvisited neighbor pixel locations (coordinates) with value 1 are pushed into a queue.

Then it pops one pixel from the queue and again uses the modified BFS according to the rules previously mentioned until the queue is empty. When there's no 1 in the neighboring pixel, the queue becomes empty thus the BFS will stop. It means that one segment of the picture is visited. From the visited matrix, min_row, min_col, max_row, and max_col are calculated. The (min_row, min_col) and (max_row, max_col) represents two pixels by which the segment will be boxed. In other words, if we draw a rectangle considering (min_row, min_col) the upper left vertex and (max_row, max_col) the opposite vertex, we will get the segment in the rectangle. So, these two coordinates are stored in the segment list. The algorithm restarts from the next pixel.

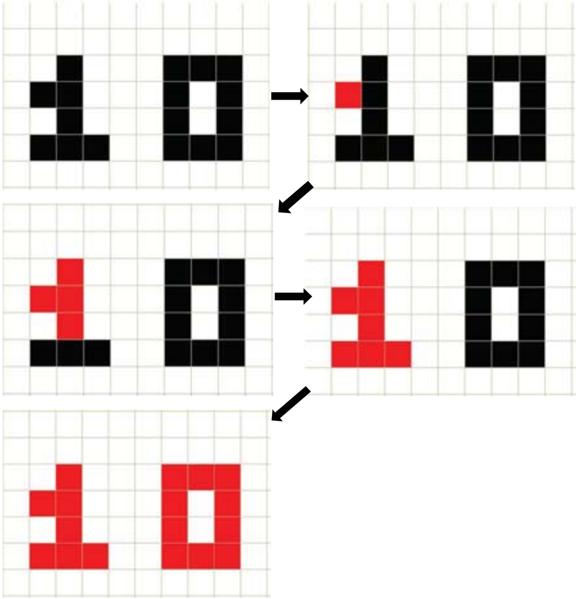


Figure 5. Algorithm steps

In figure 5 the unvisited 2D matrix contains 0 (unvisited) and 1 (visited) values. From this figure we can see that the segmentation algorithm starts traversing from top to bottom then left to right as an example we visit all the rows from top to bottom of the same column then we move on to the next column then start again from top to bottom. While traversing if it gets an unvisited black pixel, it makes that pixel visited by making it red (replacing 0 with 1 in the visited matrix) then store its coordinates in a queue

Now we have <queue> = [(3,1)], We store the row and column in two different lists where currently row=[3] and col=[1]. The algorithm runs until the queue is empty. First, we dequeue the top element from the <queue>. In this case, we get coordinates (3,1). Now it checks it's 8 neighbors to find unvisited black pixels. When it gets unvisited black neighbor pixels, it marks them visited and stores the coordinates in the row and col list, and push the coordinates in the queue. The status of the lists and queue is in the following

row=[3,2,3,4]
col=[1,2,2,2]
<queue>= [(2,2), (3.2), (4,2)]

It dequeues one element from the <queue> and gets (2,2) but as we can see there are no unvisited black pixels of (2,2). A similar situation arises for the next dequeued coordinate (3,2). So again it dequeues one element from <queue> and gets (4,2). As it has three unvisited black neighbors, the coordinates are pushed into the <queue> marking these elements visited and storing the coordinates in the row and col list. Now we have

row=[3,2,3,4,5,5,5]
col=[1,2,2,2,1,2,3]
<queue>= [(5,1), (5.2), (5,3)]

Now again we pop from <queue> and get (5,1) but it has no unvisited black neighbor pixels. A similar case arises for (5,2) and (5,3). Now as the <queue> is empty then the while loop breaks and we get the min_row, min_col, max_row, max_col.

min_row=min(row)=2
max_row=max(row)=5
min_col=min(col)=1
max_col=max(col)=3

So now we get the first segment that is from the matrix (2, 1) to matrix (5, 3). This is the first segment of the 2D matrix. Then we store this segment in a list

Previously the while loop started from (3, 1) so the traversal will resume from (3, 1) and try to find an unvisited black pixel if we find one then the whole process repeats. After (3, 1) we find (5, 1) a black pixel but as it's already visited so we ignore it then when traversal of this column is down we go to the next column and find (2, 2) a visited pixel so we ignore it then we get (3, 2), (4, 2), (5, 2), (5, 3) respectively but all of these are visited pixel so we ignore it. Then it goes to the next column and if it finds another unvisited black pixel then the whole process repeats.Our proposed algorithm is given below. It starts by taking a color image. Then, it converts the image into an 8-bit binary image. After that, the binary image is mapped into 0 and 1 where 0 represents the white region or the background, and 1 represents the black regions or the characters. Then, our greedy algorithm starts working on it to segment the characters.

Algorithm 1: Greedy Graph-based Segmentation

```
Input 24-bit RBG color image of the license plate
Convert the image into an 8-bit grayscale image
Convert the grayscale image to a binary image
composed of 0 and 255
Map 0 with 1 and 255 to 0. 1 means black pixel and 0
means the white pixel
For c in column
    For r in row
        If image[r][c] == 1 and visited[r][c] == 0:
            Visited[r][c] = 1
            Push (r, c) into queue Vertex, r into list R,
            and c into list C
            While queue Vertex!= empty:
                Check for unvisited black from adjacent
                8 neighbor of the pixel (r, c)
                If found push the coordinate (r,c) into the
                queue Vertex, r into R, and c into C
            min_c = minimum of C
            min_r = minimum of R
            max_c = miximum of C
            max_r = miximum of R
            Clear C, Clear R
            Push (min_c, min_r, max_c, max_r) into a
            list character_box
For each element of character_box:
    Crop the rectangular segment using (min_c, min_r)
    and (max_c, max_r) which represents the top left
    corner and bottom right corner of a rectangle.
```

VI. Result

We used our algorithm on our dataset and got very promising results. As it is a segmentation algorithm we only focused on the segments which represent the digits and characters of the license plate. The other segments which don't represent digits or characters are filtered out. Those can be filtered out by using machine learning but in our case, as we are primarily interested in segmenting all the important characters of the license plate we filtered out the other segments or noises manually. We ran our algorithm on 2000 images of our dataset. It successfully extracted characters from 1995 images

We also analyzed the horizontally and vertically tilted plates. In the paper [20], they could segment a maximum angle of 75 degrees. But we introduce a huge improvement as our segmentation algorithm can segment up to 84 degrees.

From figure 6 we can see that all the necessary segments or characters have been extracted successfully. The noise and the dashes are removed as they don't carry any important meaning. In figure 7 we have applied our segmentation algorithm on a full-scale frontal image of a private vehicle. Surprisingly, we segmented all the necessary characters. Though we got a lot of noises we filtered out them by considering the size. In figure 8 the characters of the license plate are segmented from public transport. In figure 9 we segmented the characters from a blurry image.

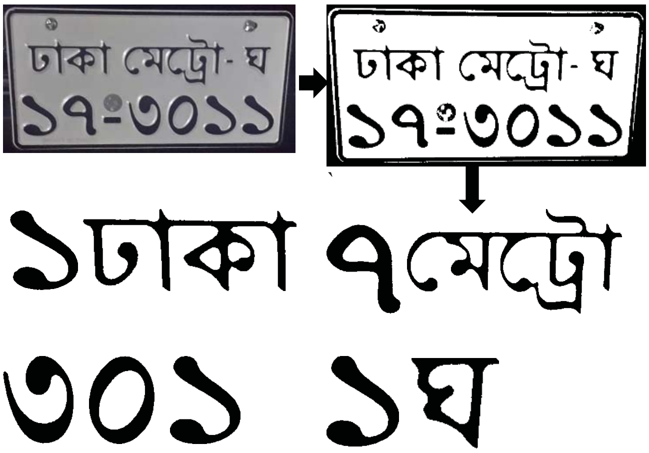


Figure 6. License plate segmentation

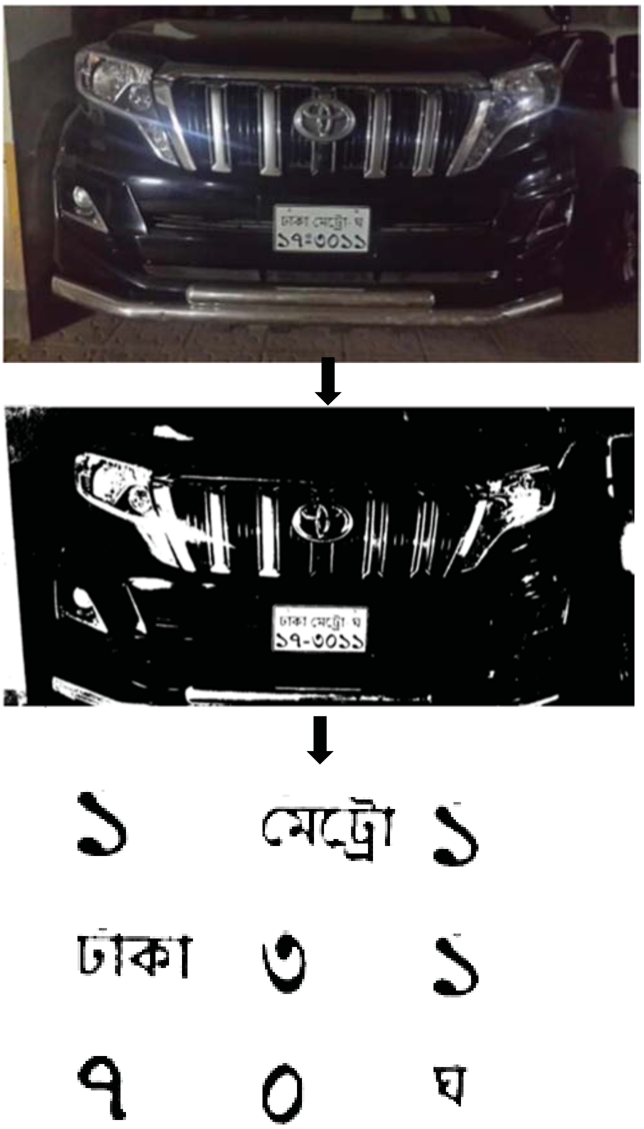


Figure 7. License plate segmentation from a private vehicle in natural condition

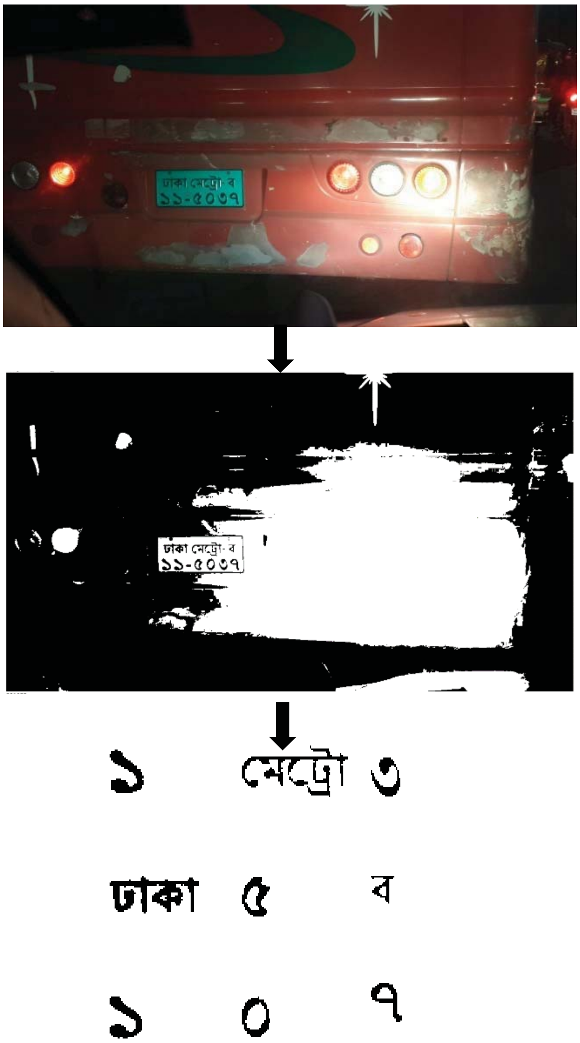


Figure 8. License plate segmentation from a public vehicle in natural condition

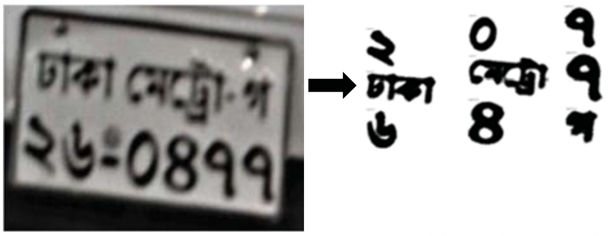


Figure 9. Segmentation from a blurry license plate

In a real-life scenario, the CCTV camera will be placed at one side of the street like the traffic signal post. As a result, we won't be available to capture images of the license plates from a perpendicular perspective. The images will be tilted horizontally and sometimes even vertically. Let's measure the angle with the primary axis which is perpendicular to the license plate. The images figure 6, figure 7, or figure 8 are taken with a 0-degree angle. The results of different orientations are shown in tables I, II, and III.

TABLE I. ROTATION

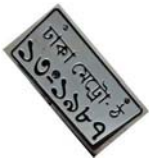
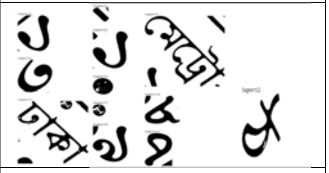
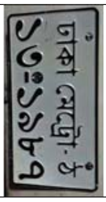
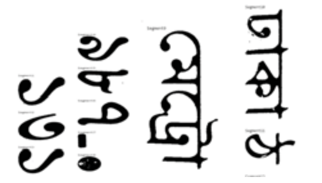
Angles in degrees	*Input License Plate Image*	*Segmentation Result*
50		
90		

TABLE II. HORIZONTAL TILT

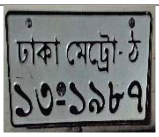
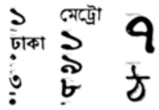
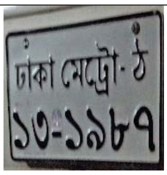
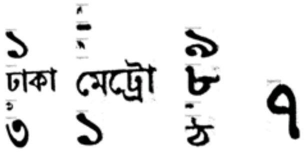
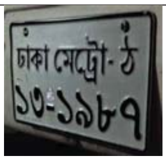
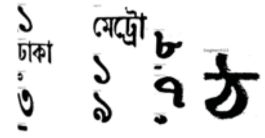
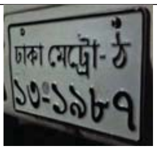
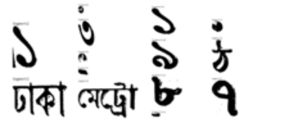
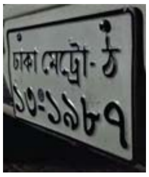
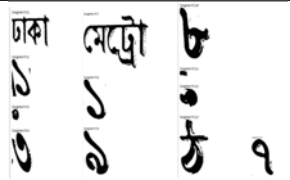
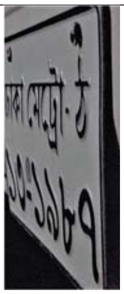
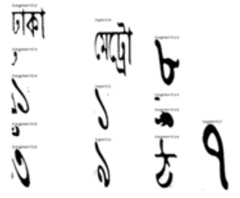
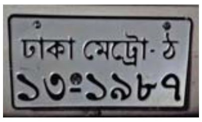
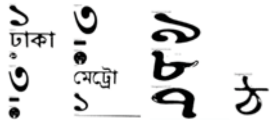
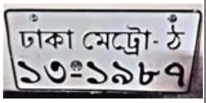
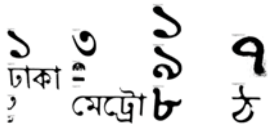
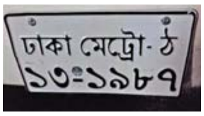
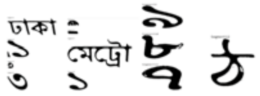
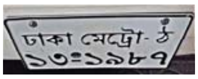
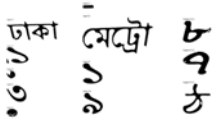
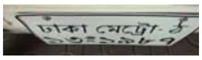
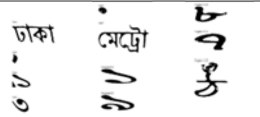
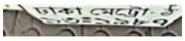
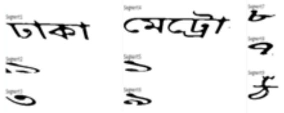
Angles in degrees	*Input License Plate Image*	*Segmentation Result*
0		
20		
40		
60		
80		
80-85		

TABLE III. VERTICAL TILT

Angles in degrees	*Input License Plate Image*	*Segmentation Result*
0		
20		
40		
60		
80		
80-85		

VII. CONCLUSION

Our research is about a greedy graph-based approach to solve some segmentation problems in license plates and we've successfully achieved our goal. However, this algorithm can be applied to other segmentation problems as well. But the limitation of this algorithm is that this can segment non-overlapping characters and cannot segment overlapped segments. Still, it is the state of the art approach for segmentation of Bangladeshi License plate as there is literary no other algorithm with this level of performance. This algorithm can segment any visually recognizable characters from a license plate. This algorithm has already been used in many license plate recognition systems [19].

VIII. FUTURE WORK

We will use image processing techniques to enhance the blurry images to get a better segmentation. Moreover, this paper deals with the segmentation of the characters from the license plate. So, we will be focusing on the end-to-end system for Bangladeshi license plate recognition in the future. We will propose a complete system from the detection of the license plate, segmentation of the characters from the license plate, and then recognition of the characters. We are planning to detect the license plate with YOLO3, segment the license plate with our segmentation algorithm, and then we want to use the CNN model to recognize the characters. We can even improve the recognition part. In short, we will propose a complete Bangladeshi license plate recognition system shortly.

ACKNOWLEDGMENT

We would like to thank the Institute of Energy, Environment, Research, and Development (IEERD, UAP) and the University of Asia Pacific for financial support.

REFERENCES

[1] Du, S., Ibrahim, M., Shehata, M. and Badawy, W., 2012. "Automatic license plate recognition (ALPR): A state-of-the-art review," IEEE Transactions on circuits and systems for video technology, 23(2), pp.311-325.

[2] K. Miyamoto, K. Nagano, M. Tamagawa, I. Fujita, and M. Yamamoto, "Vehicle license-plate recognition by image analysis," in Proc. Int. Conf. Ind. Electron. Control Instrum., vol. 3. 1991, pp. 1734–1738

[3] C. A. Rahman et al, "A real time vehicle's license plate recognition system," IEEE Conf. Adv. Video and Signal Based Surveillance, pp. 163-166, 2003

[4] K. B. Kim et al, "Recognition of car license plate by using dynamical thresholding method and enhanced neural networks," Comput. Analysis of Images and Patterns, vol. 2756, pp. 309-319, 2003

[5] A. Capar and M. Gokmen, "Concurrent segmentation and recognition with shape-driven fast marching methods," Int. Conf. Patt. Recog., vol. 1, pp. 155-158, 2006

[6] S. Nomura et al, "A new method for degraded color image binarization based on adaptive lightning on gray scale versions," IEICE Trans. Inform. And Systems, vol. E87-D, no. 4, pp. 1012-1020, 2004.

[7] G. R. Gonc，alves, S. P. G. da Silva, D. Menotti, and W. R. Schwartz, " Benchmark for license plate character segmentation, " Journal of Electronic Imaging, vol. 25, no. 5, p. 053034, 2016.

[8] J. Tan, J.-H. Lai, C.-D. Wang, W.-X. Wang, and X.-X. Zuo, "A new handwritten character segmentation method based on nonlinear clustering," Neurocomputing 89, 213–219 (2012).

[9] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber, "Convolutional neural network committees for handwritten character classification," in Document Analysis and Recognition (ICDAR), 2011 International Conf. on, 1135–1139, IEEE (2011).

[10] P. P. Roy, U. Pal, J. Llados, and M. Delalandre, "Multi-oriented touching text character seg- ´ mentation in graphical documents using dynamic programming," Pattern Recognition 45(5), 1972–1983 (2012).

[11] V. Shapiro and G. Gluhchev, "Multinational license plate recognition system: Segmentation and classification," in Pattern Recognition, (ICPR) International Conf. on, 4, 352–355, IEEE (2004).

[12] A. P. Nagare, "License plate character recognition system using neural network," International Journal of Computer Applications 25(10), 36–39 (2011).

[13] J.-M. Guo and Y.-F. Liu, "License plate localization and character segmentation with feedback self-learning and hybrid binarization techniques," Vehicular Technology, IEEE Trans. on Vehicular Technology, Vol 5, No. 3, May 2008.

[14] R. A Baten, Z. Omair, and U. Sikder. "Bangla License Plate Reader for Metropolitan Cities of Bagladesh Using Template Matching." 8th International Conference on Electrical and Computer Engineering. Dec, 2014.

[15] P. Dhar, M. Z. Abedin, R. Karim, F. T. Johora, and M. S. Hossain. "Bangladeshi License Plate Recognition Using Adaboost Classifier". Joint 2019 8th conference on Informatics, Electronics & Vision (JCIEV) & 3rd International Conference on Imaging, Vision and Pattern Recgonition (IVPR).

[16] Rahman, M. M., Nasrin, M. S., Mostakim, M., & Alom, M. Z. (2018). Bangla License Plate Recognition Using Convolutional Neural Networks (CNN). arXiv preprint arXiv:1809.00905.

[17] Saif, Nazmus, Nazir Ahmmed, Sayem Pasha, Md Saif Khan Shahrin, Md Mahmudul Hasan, Salekul Islam, and Abu Shafin Mohammad Mahdee Jameel. "Automatic License Plate Recognition System for Bangla License Plates using Convolutional Neural Network." In TENCON 2019-2019 IEEE Region 10 Conference (TENCON), pp. 925-930. IEEE, 2019.

[18] Pias, Tanmoy Sarkar, David Eisenberg, and Muhammad Aminul Islam. "Vehicle Recognition Via Sensor Data From Smart Devices." In 2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE), pp. 96-99. IEEE, 2019.

[19] Sarif, Md. Mesbah, Tanmoy Sarkar Pias, Tanjina Helaly, Md. Sohel Rana Tutul and Md. Nymur Rahman. "Deep Learning-Based Bangladeshi License Plate Recognition System." In press.

[20] Lin, Cheng-Hung, and Ying Li. "A license plate recognition system for severe tilt angles using mask R-CNN." In 2019 International Conference on Advanced Mechatronic Systems (ICAMechS), pp. 229-234. IEEE, 2019.