License Plate Recognition Using Raspberry Pi & Amazon Rekognition OCR Engine

Joseph Rodrigues
Central Michigan University
Department of Engineering
Mount Pleasant, United States
Rodri2ja@cmich.edu

Wei Kit Wong
Waterford Institute of Technology
Internet of Things
Waterford, Ireland
20075628@mail.wit.ie

Daemon Macklin
Waterford Institute of Technology
Internet of Things
Waterford, Ireland
20075689@mail.wit.ie

Yu Zhang
Central Michigan University
Department of Engineering
Mount Pleasant, United States
Zhang14y@cmich.edu

***Abstract*— This paper discusses a robust and efficient method to optimize current police to driver interaction using ocular character recognition (OCR) in combination with Internet of Things. Using a camera combined with OCR, the querying of license plates can be automated and therefore the task is taken off the police officer.**

Keywords—Police, technology, computer vision, raspberry pi, ocular character recognition.

I. INTRODUCTION

Ever since the recent mobile technology revolution, technology has become more and more a part of everyday life. This has brought with it many good things and many bad things. For instance, many police cars have laptops built in used to access information on potential criminals. With this it has become easier for the police to catch criminals, but having laptops and phones in a car has proven to be a massive distraction for drivers, showing evidence of a relationship between technology use and accidents [15]. Our project will eliminate this distraction, automating the process of searching a license plate to see if the police need to stop them, completely removing the need for police offices to manually, find and record a license plate. Allowing them to spend their time more productively and focus on the road, making their job less dangerous.

II. SYSTEM DESIGN & APPLICATION

A. Optimizating a Method Already in Use

In a typical police officers daily shift, they do many things. Respond to calls, enforce traffic laws, surveillance for people breaking other laws, etc. Among these tasks, most of their time, position dependent, is spent driving around in their police car. While driving it is common for them to manually search the license plate number of cars they see through a police database, retrieving information on the owner. This information depending on the state can include the registration, national crime information center data, wanted status, driver information, bail conditions, felon indicators and prior in-house incidents with the owner. Traffic stops conducted with this method are often referred to as “bingo” stops. This is already a tried and tested process but has room for optimization.

The proposed method of optimization is through the use of a camera placed on the front of a police car with software capability of reading multiple license plates and then checking the reports tied to the vehicle owner. Once the report is pulled by the software, the report is analyzed automatically and if any flags appear such as a warrant or suspended license for example, the officer is given a notification through a sound prompt. Once the officer is notified, they will then manually check the license plate to assure no mistake was made by the software and if a match occurs, they can then intervene however deemed appropriate.

Using this system, the officer will no long have to spend the time manually typing plate numbers into the computer and will have the ability to give full attention to driving or making sure other drivers are adhering to the laws. In combination with relieving the officer of distraction, this system would also optimize the capabilities of using the police database to check license plates. Rather than having to manually check each license plate one at a time, the software would be able to optimize this task, increasing the number of plates read in a given amount of time, as well as being able to read plates at times the officer wouldn’t be able to [4].

B. Data Analytics

This system would also extend past the use as a device to pull drivers over. This system could also be used to collect and analyze data. For example, the police database already supplies information from the National Crime Information Center. This includes information from 21 files including: stolen items,

people enquiries (sex offender, gang affiliation, missing person) as well as pictures of the offenders to help law enforcement identify them [1]. Using this information, locations of interactions with or even spotting's of said person can be logged into a database and further referenced creating crime statistics helping police further do their jobs.

C. Hardware Design

For our implementation of this system, our tentative plan is to use a Pi camera module connected to a Raspberry Pi. The Raspberry Pi was chosen as its hardware capabilities were adequate for data and image processing, with a web search worth of previous computer vision-based projects already implemented. In combination with this, it also has the ability to connect to the internet, a must have in an IOT project. To simplify the process of connectivity with the Raspberry Pi, the Pi Camera Module V2 was chosen as a viable camera source, but the same results can be achieved with any USB camera. For the real system itself, other SoC (System on a Chip) and camera devices may be used, as the former hardware were chosen for its ease of accessibility.

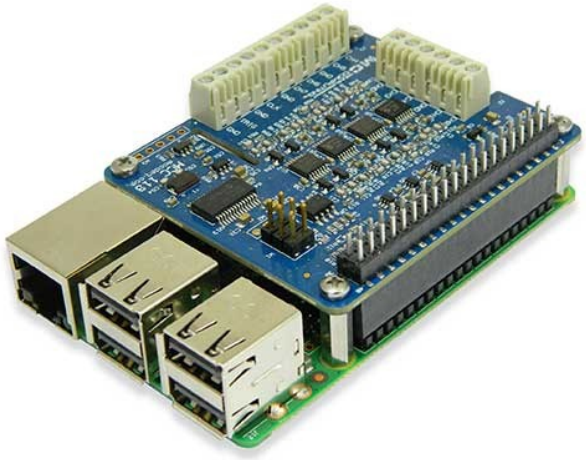

Fig. 1. Picture of Raspberry Pi 3 Model B+

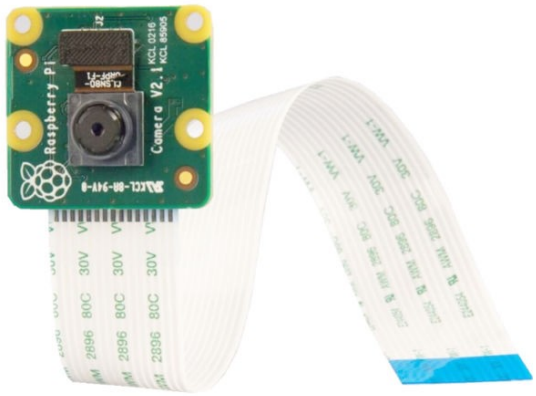

Fig. 2. Picture of Pi Camera Module V2

D. Software Research

The software implementation for this project is using a computer vision system that can interpret the text of a license plate to search through a database. The image processing will be divided into two parts; plate detection, and character recognition. This is because it can be slow to recognize a character from the image, so to combat this, the processes are divided. Localizing and segmenting the license plate are key steps of the license plate recognition system. Because the license plate image is a sub-region that is very characteristic in the original image, and that is concentrated in a special part. Its gray value differs to that of periphery areas, thus there will be an obvious boundary, which will facilitate segmentation of image by edge detection.

The initial proposed method of computer vision is using the open source OCR engine, Tesseract, which was first developed by HP but more recently developed and used by Google [13]. This method begins the process by localizing the text, creating a region that is snipped from the overall image to be further analyzed. Next the orientation and number of the text is found using a method called "line fitting". Line fitting in this context will be simple because a license plate number is always horizontally oriented as well as only one line. Next, the spacing between characters is found. This is done by finding the number of pixels between the letters. Known information about license plates, the spacing between numbers/letters as well as colors will aid in this process [2]. Finally, the characters are identified. This is done by having the software classify the text or numbers based on their distinct shape [3]. This is possible due to license plates uniform text color as well as distinct edges.

After initial testing and further research, it was decided Tesseract OCR would be replaced with a more thorough user-

friendly API that Amazon offers. This system is called Amazon Rekognition. Amazon Rekognition is an API that takes an image, detects text within it and converts it into a machine-readable text [16]. In more technical discussion, Amazon Rekognition uses deep learning technology to analyze images, continually training the system on new data to enhance its abilities and accuracy. As with any image processing software, Amazon Rekognition faces the same issues depending on variables relating to the image [16]. Things like image resolution, color scales, text size and font must be considered as these are all properties that may lower OCR accuracy. These properties are relevant because they all relate to the distinguishing properties of characters within an image. If the resolution is too low, text is difficult for even humans to read as shape becomes blurred. If the color scale is jeopardizing, character borders become more difficult to distinguish. If the text size is too small, once again even for humans the text can be difficult to read and the same applies for OCR application as the shape becomes difficult to distinguish. And lastly, unfamiliar fonts can complicate the OCR process as text recognition is made capable by analysis of fonts already known. Unique fonts might not match the characteristics of these known fonts and could mislead the OCR process. However, with these challenges, Amazon Rekognition still proves to be a robust system and along with its robustness it is also a simple system to implement, with its API being integrated within the chosen cloud infrastructure WIA.io, which will be discussed later in the text.

The image to text conversion is conducted within the cloud interface WIA.io. WIA is an online service that allows hardware and software integration, all using an web interface [17]. By connecting a Raspberry Pi to WIA, images taken can have the license plate text successfully recognized and outputted where further action can be taken. Another resource WIA provides is its ability to send webhooks. In this case, the license plate number can be sent in the form of a webhook, querying it in a database to pull specified information. Another useful service is WIA's ability to run custom scripts of multiple programming languages. This applies to this project in the situation of checking the text outputted from Amazon Rekognition, verifying its length and that it includes both numbers and letters; all specific characteristics of license plates that help distinguish the readings from other possible texts that could be picked up.

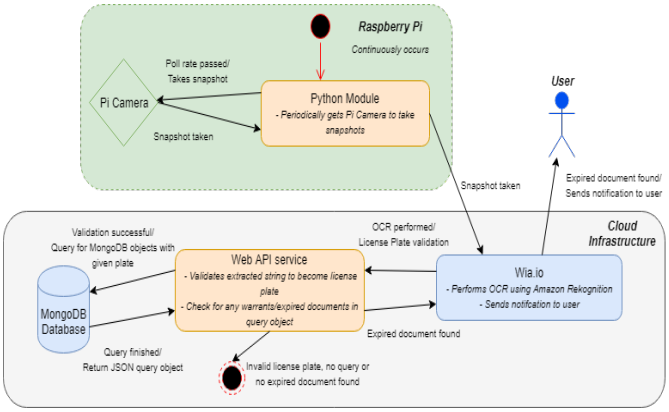

Fig. 3. Flow chart of system process.

III. Previous Work

Previous work relating to this project starts with a text recognition to speech program that uses tesseract, the proposed method in this project to recognize characters. The process that was used begin with image capturing and preprocessing. A still image was taken and changes such as: noise reduction, image enhancement and binarization to help with later image processing to give more accurate results [13]. Next, the text in the image was segmented for further analysis. This was done to separate the text to be recognized from the rest of the image which becomes irrelevant. The next step, and the most important was to recognize the text using the open source engine tesseract. Tesseract was chosen due to its better performance when matched against other open source engines [13].

In another project, Tesseract was used to automatically detect vehicles based on their license plate and visual features. The first portion of this project is relevant as it uses the same character recognition engine to complete the same task. Once again, Tesseract was chosen due to its open source availability in combination to its high accuracy [14]. To achieve the final goal of receiving a plate number from an image input, the same steps as previously discussed were taken. First the license plate in the image was localized and preprocessed for character recognition. This included the same steps of binarization and image enhancement as well as segmentation. Next, using Tesseract, the characters are recognized using both adaptive classification and repeating recognition [14]. Using this information combined with vehicle characteristic recognition, their project would verify if the vehicles identity. This is where our projects differ. Instead, in our project, the outputted license plate number is then searched through a cloud database where various information is supplied to the police officer.

IV. Implementation

For the physical implementation of this project a Raspberry Pi 3 Model B+ is equipped with a USB camera or the specific Pi Camera Module and attached to the front of said police car with the camera facing forward with a field of view of 60 degrees (hardware specific to the camera) as shown below in Fig. 2.

Fig. 4. Image showing camera orientation and horizontal field of view.

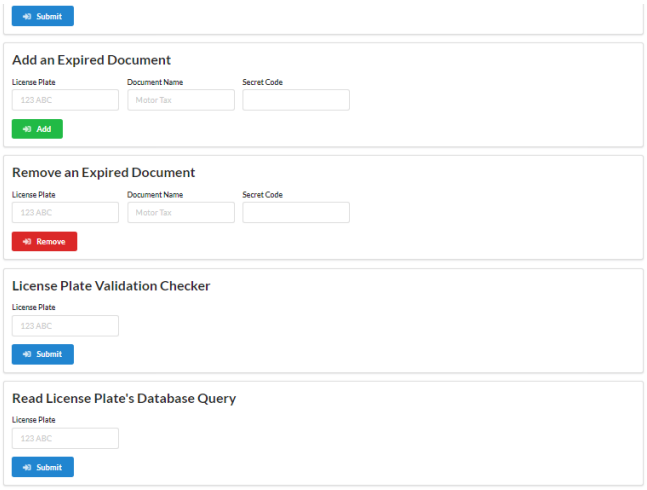

Fig. 5. Picture of web API used to interact with MongoDB.

On the Raspberry Pi, a program was written to take pictures at a specified frequency. Depending on the camera used, taking the picture is accomplished either by using the picamera Python library, where a picture can be taking using the capture() command. Or if a USB camera is being used, the fswebcam library is installed and a ran within the Python script using the subprocess function. The subprocess function must be used as fswebcam can only be run through the terminal and subprocess runs a terminal script through Python. These pictures are then sent to the WIA cloud interface through internet connection. Within WIA ocular character recognition through Amazon Rekognition is used to identify and recognize the text within the images taken, outputting it as a machine usable string. Theses strings are then checked, specifying whether they satisfy the characteristics of United States license plates including; length of 6-7 characters and the inclusion of both numbers and letters. If the string satisfies these characteristics, the license plate number is then sent through a webhook where it is queried from a web API through a simulated database. In this case it is a MongoDb hosted through the website mLab. If the license plate queried matches a number within the database, its information connected to the plate is sent back to the web API and sent once again back to WIA.io where the information is analyzed, deciding whether it is relevant to send to the police officer in the form of a notification.

V. Testing & Results

Current tests have shown the proof of concept. In the first stage of the project, the Pi camera was successfully programmed to take pictures at a specified polling rate. The next stage was achieved by sending these pictures to WIA.io where the OCR was conducted successfully and accurately as shown below in Fig. 5.

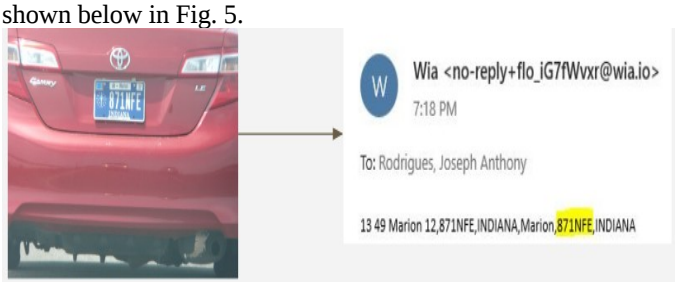

Fig. 6. Picture displaying results in early stage OCR.

In Fig. 5. above it can be seen the text in the image is successfully read by Amazon Rekognition and sent the output in an email where the license plate number of the car is highlighted. In the early stage of the project before the webhook is implemented and functioning, WIA is simply programmed to send the OCR results in an email. Within the WIA cloud interface, a check is run, assuring the text recognized in the images meets a certain length, as earlier specified standard issue United States license plates consist of 6 to 7 characters varying depending on the state. From these implementations, the largest challenges have been overcome; successful OCR and successful cloud interface connection.

The next steps are to connect WIA to the web API specifically created for this testing and the web API to a MongoDB created to simulate a police database where license plate numbers will

be stored with varying information attached to each. Once fully set up the entire system will successfully represent the system diagram displayed earlier in the text in Fig. 3.

After both the web API and the MongoDB are implemented, the system functions as a whole. License plates can be added, along with relevant information related to them by using the web API hosted by glitch.me. For testing purposed, multiple faux license plates were added to the database with varying details added to them. Then, faux license plates (paper with large numbers matching those added to the database) are scanned using the camera. For testing purposes, the camera is triggered by simply running the Pi python script, rather than by a timer which is the plan during actual implementation. Now, once a license plate with relevant information attached to it is recognized, expired insurance documents for example, an event is sent from the web API back to WIA where it is then sent as a notification in the form of an email to the user. This is pictured below in fig. 6.

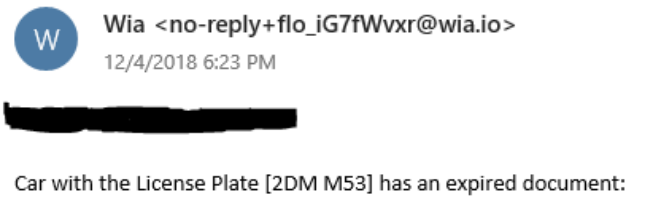

Fig. 7. Picture displaying notification sent to user if a plate is scanned with an expired document.

After multiple tests, it was found that the text orientation of the faux license plates is important. Error occurred during testing due to space between the letters unintentionally being to wide. Because the OCR must recognize this space as it is important when interpreting real license plates, it can't be ignored and causes the system to interpret the plate as a number that it isn't. This problem wont transition to actual implementation of the system but was something to be considered during testing.

VI. Conclusion

The following benchmarks were established at the beginning of the project to evaluate success. Successful OCR of a license plate (real or simulated) with interconnectivity to a cloud system that has the ability to search the scanned license plate through a database which simulates a police database. Based on these benchmarks and the results achieved, this project could be deemed a success. Successful OCR was achieved using Amazon Rekognition and through the use of WIA, the glitch.me web API and the MongoDB, full interconnectivity of the system was achieved. Future additions to this project include a method to better decide when the system takes a picture. Rather than the current method of using a timer, a motion sensor could be used to detect when new objects (cars) enter the FOV and trigger the camera to take a picture. Another approach could be to use the OpenCV library and do real time image processing of a video feed from the camera, analyzing when license plates enter the FOV and triggering a picture to be taken.

References

[1] S. W. Craun and P. J. Detar, "Designated as Armed and Dangerous," *Journal of Criminal Justice*, vol. 43, no. 5, pp. 437–442, 2015.

[2] H. Jiang, T. Gonnot, W.-J. Yi, and J. Saniie, "Computer vision and text recognition for assisting visually impaired people using Android smartphone," *2017 IEEE International Conference on Electro Information Technology (EIT)*, 2017.

[3] J.-H. Seok and J. H. Kim, "Scene text recognition using a Hough forest implicit shape model and semi-Markov conditional random fields," *Pattern Recognition*, vol. 48, no. 11, pp. 3584–3599

[4] J. J. Willis, C. Koper, and C. Lum, "The Adaptation of License-plate Readers for Investigative Purposes: Police Technology and Innovation Re-invention," *Justice Quarterly*, vol. 35, no. 4, pp. 614–638, 2017.

[5] H. N. Do, M.-T. Vo, B. Q. Vuong, H. T. Pham, A. H. Nguyen, H. Q. Luong, "Automatic license plate recognition using mobile device," 2016 International Conference on Advanced Technologies for Communications (ATC), 2016.

[6] S. Dhote, P. Charjan, A. Phansekar, A. Hegde, S. Joshi, J. Joshi, "Using FPGA-SoC interface for low cost IoT based image processing," 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2016

[7] K, Tejas, K, Ashok Reddy, D, Pradeep Reddy, & M, Rajesh Kumar. (2017). Efficient Licence Plate Detection By Unique Edge Detection Algorithm and Smarter Interpretation Through IoT.

[8] Bulan, O., Kozitsky, V., Ramesh, P., & Shreve, M. (2017). Segmentation- and Annotation-Free License Plate Recognition With Deep Localization and Failure Identification. *Ieee Transactions On Intelligent Transportation Systems*, *18*(9), 2351-2363.

[9] S. Venkatraman, K. Fahd, S. Kaspi, R. Venkatraman,"SQL Versus NoSQL Movement with Big Data Analytics", International Journal of Information Technology and Computer Science(IJITCS), Vol.8, No.12, pp.59-66, 2016. DOI: 10.5815/ijitcs.2016.12.07

[10] Ratan, V. and Singh, G. (2016). Raspberry Pi based Implementation of Internet of Things using Mobile Messaging Application - 'Telegram'. *International Journal of Computer Applications*, 145(14), pp.17-21.

[11] T. Sorwar, S. B. Azad, S. R. Hussain, and A. I. Mahmood, "Real-time Vehicle monitoring for traffic surveillance and adaptive change detection using Raspberry Pi camera module," *2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC)*, 2017.

[12] Crawford, M. (2017). *AUTOMATED COLLECTION OF HONEY BEE HIVE DATA USING THE RASPBERRY PI*. Appalachian State University.

[13] "Detecting Text Based Image with Optical Character Recognition for English Translation and Speech Using Android." *Research and Development (SCOReD), 2015 IEEE Student Conference On*, by Sathiapriya Ramiah et al., 2015, pp. 272–277.

[14] Lyu, Hao. *Automatic Vehicle Detection and Identification using Visual Features*, University of Windsor (Canada), Ann Arbor, 2018. *ProQuest*, http://cmich.idm.oclc.org/login?url=https://search-proquest-com.cmich.idm.oclc.org/docview/2015699068?accountid=10181.

[15] Parnell, Stanton, & Plant. (2018). What technologies do people engage with while driving and why? *Accident Analysis and Prevention*, *111*, 222-237.

[16] "Amazon Rekognition – Video and Image - AWS," Amazon. [Online]. Available: https://aws.amazon.com/rekognition/. [Accessed: 20-Nov-2018].

[17] Wia, "Any device. Any service. One cloud.," *Wia*. [Online]. Available: https://www.wia.io/. [Accessed: 20-Nov-2018].