A Midterm Report on Researching and Implementing a Method for Image Object Recognition Catering to License Plate Applications

To:
Professor Mohit Gupta
CS 534

From:
Carter Steffen
csteffen3@wisc.edu
NetID:csteffen3

Ben Farley
bjfarley@wisc.edu
NetID: bjfarley

Chan Hyeok Yun
cyun4@wisc.edu
NetID: cyun4

Computer Sciences Department
University of Wisconsin, Madison
September 25, 2018

License plates are a staple of the modern car and find many uses in modern society. There are currently 272.1 million vehicles on the road today in the United States [1]. Every single one is legally required to have a license plate. This makes it easy for law enforcement to identify each individual car, who owns it, and to verify that it is on the road legally. Outside of the US government, other groups make use of the license plate. Insurance companies use the license plate to track the vehicles which they are insuring. Hotels, landlords, and businesses often use it to mark which cars are allowed to park in their private spaces.

The ability to easily identify the license plate is crucial to obtaining the information associated with it. Doing this electronically from an image or video feed, would be even more beneficial. It would allow for the data associated with the plate to be accessed instantaneously. This could give the receiver the ability to identify the owner of the car, check if it was stolen, or determine if it is allowed to be in its current spot.

The problem we intend to solve then, is to be able to read the numbers and letters on a license plate from a digital image. It is our goal to do this with many different types of images, where the license plate is at different locations, angles, and sizes. This allows for a larger variety of use cases and would be more robust than many current solutions to the problem.

Video and photo identification software for license plates already exists, however it is often very narrow in its use. The majority of the software has been built for specific applications. The most well known, and potentially most disliked, is to identify a car on a tollway and to charge the owner a fee for passing through. For Illinois tollways, the company responsible for the project is J. A. Watts, Inc. [2]. The setup of their project generates an image where the license plate is at roughly the same location, size, and position in each image. From that, they pull the data necessary to charge the owner of the car. In cases where the plate cannot be read, the image is flagged and sent to a human reviewer. Specific and with failsafe human involvement, this is a proven use case for the technology.

Many other companies currently exist making money solving the problem of license plate recognition. One company, with perhaps more robust image recognition technology, is gtechna. The company creates and sells products for a variety of uses. Their software is able to recognize plates when mounted on moving law enforcement vehicles, for parking operators, and in a variety of other settings [3]. Other major companies involved in this business include Raytheon, PlateSmart Technologies, and Genetec.

Outside of the private sector, many solutions also exist for public use. In testing several of the free codes available, we found the existing approaches worked to varying degrees of success. Many of the solutions worked well for a few images but would incorrectly classify the license plates in tougher images. Often, when the license plate was less clear, tilted at an angle, or set against an odd background, the software struggled. These different distortions of the license plate need to be taken into account. For example, a fast moving car may create some blur in the image. Or perhaps a prominent edge of a car next to its plate may look like the letter “I.” These specific cases will arise, and a software capable of handling them will be more successful.

The main reason that the existing approaches are failing is because they aren't able to adapt to the noise inside the image. Different environments and orientations of the license plate, or even just specks of dirt and edges around the car, can throw the software off. This realization came as we tested various open source code with different images. As we stepped through the code, we displayed a subplot of the image at each point it was manipulated. Figure 1 below serves as an example of some of what we saw. Note, we modified this code with some of our own code, but the premise still holds: the open source software struggled to eliminate much of the noise in the original image.

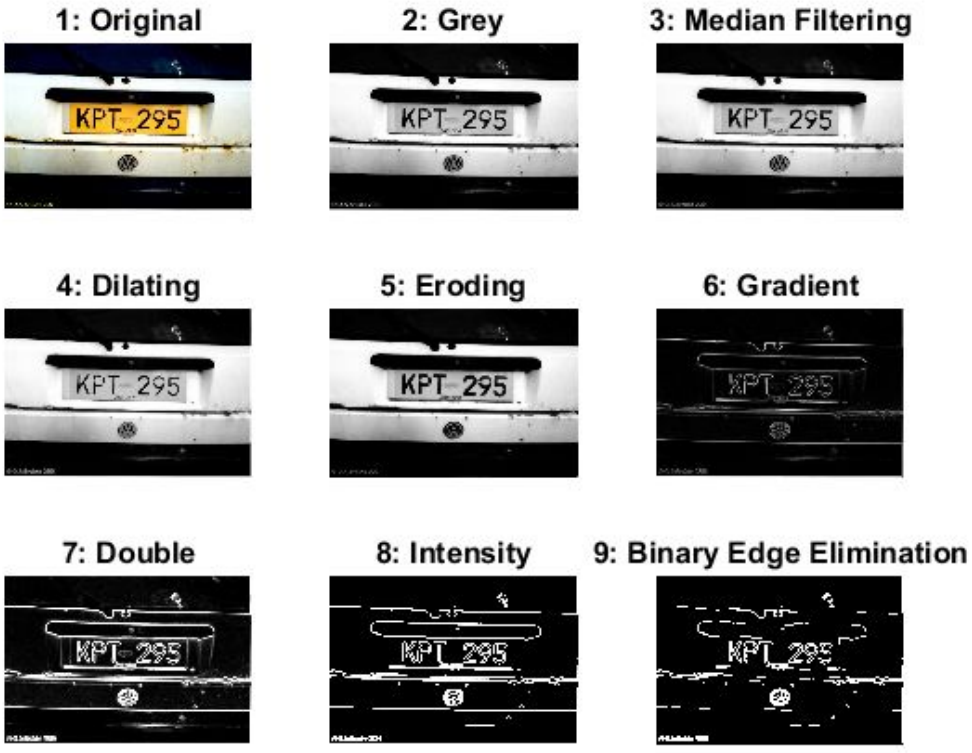


Figure 1: Some of the steps an open source license plate recognition software took an image through.

The approach in Figure 1 converts the image to grayscale, and eventually gets the intensity image of each pixel. However, the intensity image is difficult to read as there is a lot of noise within the image. Where the actual letters and numbers of the license plate exist there are just outlines. Edges remain and serve as confusing points for the letter recognition to deal with. From these manipulations, it is still very difficult to determine the letters and numbers.

As shown this task is very difficult, so we are beginning by trying to implement an existing approach. The approach we've started with uses the image segmentation algorithm, mean shift, from class. While computationally expensive mean shift was a fairly simple algorithm which did a good job of segmenting the image. Thus, from the segmented image we can extract the license plate by looking for rectangular shaped objects with numbers and letters inside of it. If we have more time, we could try the other segmentation algorithms discussed in class, k-means clustering and minimized cost of normalized cut, but we felt from the notes that the mean shift did the best job reducing the additional noise of the background. If we have time we will try other segmentation algorithms to compare the results. It also uses the erosion and dilation steps done in the walkthrough for hw2 to help remove noise and extract the license plate from the rest of the image. Once we have a semi-successful program from these general techniques, we will try to create our own novel approach.

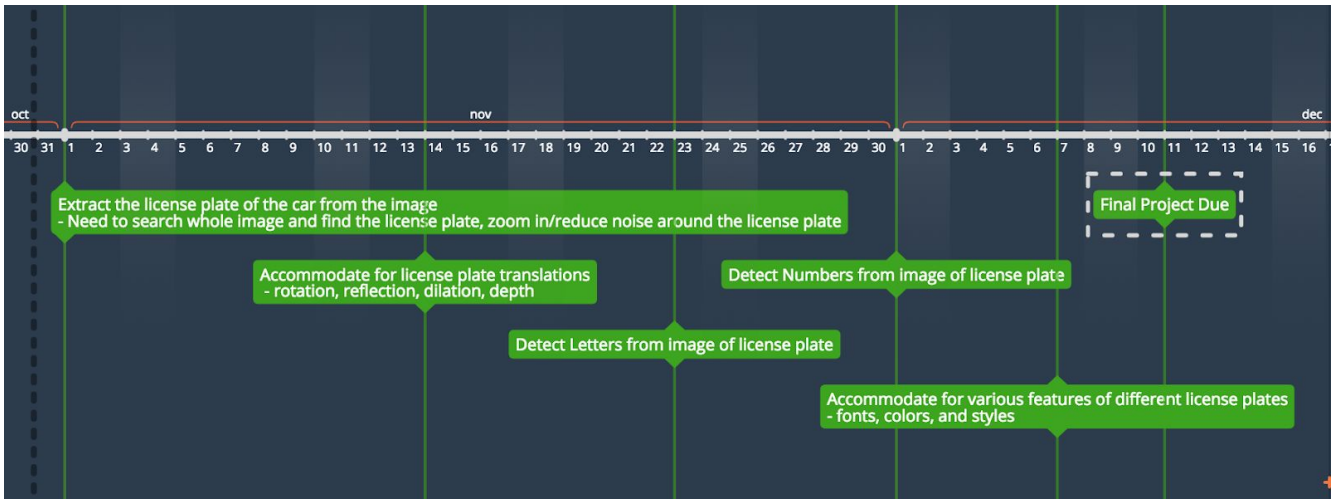
There a few different things we would like to implement to make the code more robust. We will try to create our own algorithm to manually trace through the image pixels and extract the license plate rather than calling a method to do it for us. We will study in depth how the algorithm works and hopefully we can create a representation that is equal or better at segmenting the license plate from the rest of the image. We will also try implementing filters to use on the image after segmentation to reduce the noise of the letters and numbers. This will help make it easier to separate things close in nature, for instance distinguishing the letter "Z" from the number "2." When extracting the license plate, we want to blow up (linearly transform) the extracted image. This will create more distinct edges and lines between the pixels such that we can clearly see the letters and numbers.

In implementing these steps, we believe that our solution will work better than the other solutions. From these additions the software will be attempting to process the letters and numbers from a cleaner image. The image itself will be a zoomed in image of the extracted license plate to easily distinguish the prominent edges and lines in the image. If a human can do this simple job of extracting the letters and numbers from the images, then a computer should be able to do it to a similar degree of success.

We will evaluate the performance of our code by extracting the license plate numbers and letters from a variety of images of stationary cars first. Perhaps from images of cars in a parking lot. We will try to figure out the farthest distance, the most blur, and lowest resolution we can scan the plate. From those three tests, we can determine what area we need to better improve within the algorithm. Then, we will measure the time we consumed to scan those license plates, and compare it to the time taken by a human. Hopefully, we can create something comparable in duration to what a human can do. Lastly, and most importantly, we will compare our code against the other license plate recognition codes we found online. We will feed each implementation the same images and compare which algorithm gets more letters and numbers on the license plates correct. The images will vary in where the license plate is placed, how blurry it is, etc., but the evaluation will be the same: the more letters and numbers it gets correct, the

better the algorithm. We hope that our novel implementation can compete, if not beat, the code samples we found online.

Accomplishing this over the next few weeks will take a good amount of hard work. To help pace ourselves, we have made the timeline shown below.



Work Cited

[1] "U.S. - Vehicles in Operation 2018 | Statistic." *Statista*, 2018, www.statista.com/statistics/859950/vehicles-in-operation-by-quarter-united-states/.

[2] "ISTHA Automatic Number Plate Recognition System." *JWI*, 26 Mar. 2018, www.jwincorporated.com/projects/the-i-94us-41-smart-highway-project-mentor-protege-program/.

[3] "Our Law Enforcement Systems." *Gtechna Integrated ECittion Software*, www.gtechna.com/products/.