EZPark

Automated Parking Garage Payment System

Constanza Figuerola
Samyukta Lanka
Utkarsh Shah
Max Tromanhauser
Advisor: Boon Thau Loo

April 25, 2016

Today, garage parking systems are annoying, expensive and needlessly complicated for both garages and drivers. Drivers are forced to carry around easy-to-lose tickets, wait in parking lines (especially at large events), and must manually use their credit cards. Garages have to purchase and maintain expensive payment machines, employ parking staff, and lack useful analytics.
The EZPark system was designed to solve these issues. Drivers sign up once in order to easily park in every garage in our system. When a driver enters a garage, cameras send images of the car to a central server which recognizes the plate number and keeps track of how long they stay in the garage. Upon exiting, the associated users credit card is automatically charged and the parking gate is opened. All of this data is aggregated and available to the user and garage. This solution provides convenience and ease of use for drivers, while also simplifying the parking system for garages.

1 Introduction

1.1 Background

Cars are the ubiquitous transport mechanism for the modern world. We use them in our daily lives to travel to work, school, or simply to the grocery store. By necessity, when cars are unused, they need to be stored. In urban environments or locations like sports stadiums that attract a large number of visitors, the solution to this storage problem is the parking garage.

Parking garages/lots are typically owned and operated by private organizations, who purchase a plot of land, optimize it to fit as many cars as possible, and charge an established rate to customers who park there. At its most basic, the task of charging drivers is simple. The entry and exit times for the driver are recorded, and the driver pays the established rate for that time. However, the implementation of this system varies dramatically.

One implementation is the parking booth solution. A booth is placed at the entry/exit of the garage, where a garage attendant works. Upon entry, the attendant gives drivers a time stamped ticket, and upon exit, he compares the ticket to the current time, calculates the rate, and collects payment from the driver.

Another implementation type is the validation solution. Here, drivers still collect a ticket upon entry, but rather than paying at an exit booth, they pay before returning to their vehicle, at validation booth(s). They present the validated ticket at the exit booth as proof of payment.

In both of these implementations, human-operated parking and validation booths can be replaced with machines of varying levels of quality.

There are many systems that currently use license plate recognition in practice. These systems include toll bridges that record the license plate of each car and send the bill through the mail, along with law enforcement that uses plate recognition systems as a way to track down tagged cars[1][2]. These existing implementations and established software convinced us that ALPR could be used to help solve the problems in parking garages.

1.2 Problem Statement

With or without machines, both of these implementations provide a poor parking experience for both drivers and garages.

For drivers, the issues with both of these systems are payment processing and tickets. At some point in the process, it becomes necessary for drivers to take out their wallets and swipe their credit cards. This is a time consuming and annoying process. Since people are slow to retrieve the necessary items, and payment processing takes time, this also can cause excessive lines, either at an exit booth or at a validation location. Tickets are equally frustrating since they are easily lost, resulting in high fees.

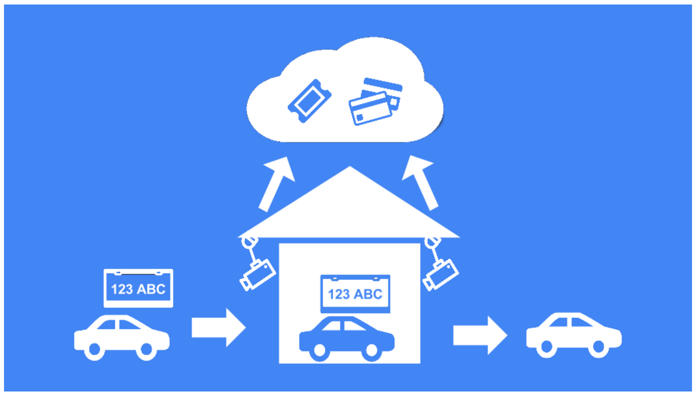
For garages, the issues revolve around cost management. With human-powered implementations, garages have to pay wages to a significant number of personnel in order to avoid long lines and maintain useful hours. With machine-centered implementations, garages have to purchase expensive payment processing devices that understand how to read and charge a credit card safely and efficiently. These machines also tend to have high maintenance costs.

Finally, these current implementations fail to provide both garages and users with useful analytics about their parking status.

1.3 Solution

For this project, we made the assumption that we would only be dealing with drivers that are registered users. From now on, when talking about a driver

Figure 1: User Flow



using an EZPark garage, it can be assumed that the driver is registered with our service.

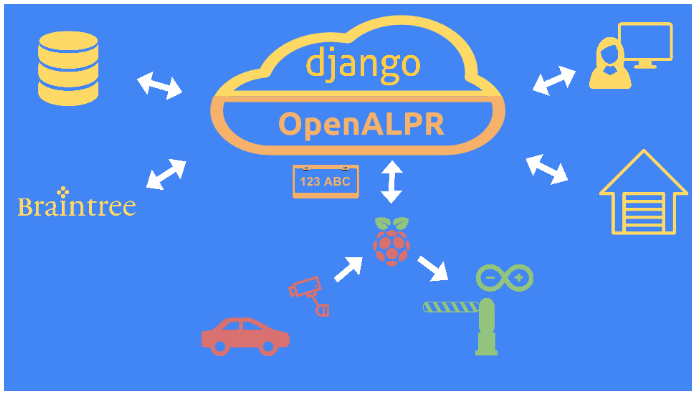
When a car drives up to the gate to enter a garage, instead of taking a ticket, our system would record the license plate. A camera would detect that an object has entered the frame, send the image up to our server for analysis and once the user associated with the license plate is found, the gate is opened. Upon exiting, the amount that the driver needs to pay is calculated and their pre-registered credit card is automatically charged.

1.4 Benefits

The EZPark system benefits both the drivers and the garages involved. For drivers, there are many points of annoyance with current parking garages. First, wait times to pay and exit a garage can range anywhere from a few minutes to an hour depending on the traffic in the garage. There have been many cases of hour long lines to exit garages after large events like football games or concerts[3]. We solve this issue because our system takes only a few seconds (see Results Section). Second, there are many cases in which users have lost tickets and are forced to pay for the entire day. Our system eliminates the need for tickets and thus reduces a little of the burden for drivers. Finally, our system provides an easy way for drivers to track their spending and parking habits.

The main benefits for garages end up being monetary. They are able to reduce costs because they no longer have to maintain expensive payment machines. In addition, our system removes the need for parking attendants. Our target parking garage would be one that has parking attendants processing tickets and payments. A normal sized parking garage would require around two parking attendants. Since our system is able to operate 24/7 we would be replacing two attendants working all day every day. The average hourly pay for a parking attendant is $10.83[4]. Depending on the hours of operation for the garage, this adds up to savings between $124,761.60 and $189,741.60 per year.

Figure 2: EZPark System Diagram



Finally, garages can greatly benefit from the analytics our service provides. All the information provided about the most and least popular parking times can help a garage change and perfect its strategy.

2 Technical Approach

Before going further in depth into the technical details of the approach taken, it is important to understand how the different modules in the system interact with each other. Our server, which interfaces with license plate recognition, client dashboard, database and payment service, must also interact with the hardware in the garage. In order to do so, a Raspberry Pi is set up that connects to a camera taking pictures of entering and exiting cars as well as an Arduino microcontroller that opens and closes the gate.

When a car enters the garage, the system detects that a picture should be sent up for analysis. The Raspberry Pi sends an image of the car to the license plate recognition software. If a license plate is detected, the server checks to see if it is a registered user and logs the entrance in the database. It tells the Raspberry Pi to signal the arduino to open the gate. In the case of an exit, the server calculates the amount to be paid, and charges the associated users credit card before responding to the Raspberry Pi.

2.1 Motion Detection

The camera in the garage takes a picture every 0.5 seconds so that the system can detect an approaching car as quickly as possible. However, if it sent every image taken to the server for analysis, there would be a significant delay for results. Because the license plate recognition software takes about 1.3 seconds on average to determine the license plate, it would not be practical to send an image every half second.

As a result, the raspberry pi runs a script to check if the current image has changed enough from the previous image to indicate that something has entered the cameras frame. However, comparing every pixel of the image would take 14.7 seconds, even longer than before. Instead, we take a randomized sampling of 2.5% of the image and compare it to the previous image. We determined that if about 10% of the pixels sampled have changed, then it is reasonable to assume that a large enough object has come into the cameras frame. This means that we can look at a much smaller number of pixels, and still accurately determine when a car enters the frame. Only these images are sent for analysis, which significantly eases the burden on the server.

2.2 ALPR

To recognize the license plates of incoming cars we build upon an open source project called OpenALPR[5]. The recognition software uses computer vision and trains itself on license plates from every state in the US.

To get an idea of the systems accuracy with pictures from our own camera, we took pictures of more than 50 cars at a few angles each. Through this process we got an accuracy of 94.4% with pictures where the license plate was straight on. This was the first test we had to do to make sure that license plate recognition was a viable option for this system. Given the existing systems that use license plate recognition and our accuracy with OpenALPR, it was a reasonable assumption to make that license plate recognition could be used in a parking garage setting.

There are a few differences between the systems that we described and the environment of an actual garage. The main difference is that it is difficult to get a straight up image of the license plate since it would be difficult to place a camera straight behind or in front of the license plate. Instead a more reasonable set up is to have the camera at an angle. To test the accuracy of this we used the original angled pictures we took of license plates and ran it through the system. We ran these pictures through OpenALPR and got a accuracy of 35.0%. This accuracy was way too low to be a viable option. As a result we looked into different ways to configure the pictures with the given angle. The best way to do this was to pre-warp the images that we were given from the camera to make the license plate appear as if it were straight on. Figure 3a is an example of the original angled image. We warped the image by rotating the x,y and z axes to make it look more straight on like in figure 3b.

2.3 Garage Manager

There is one part of the server that interacts with both OpenALRP and the database. The logic for logging a car in and out, calculating the amount to be paid and responding to the raspberry pi live here. Since our license plate recognition does not have 100% accuracy, we had to do a little more to boost the results. When sent an image, OpenALRP returns a set of possible plates with confidence values. If the top plate result does not have a corresponding

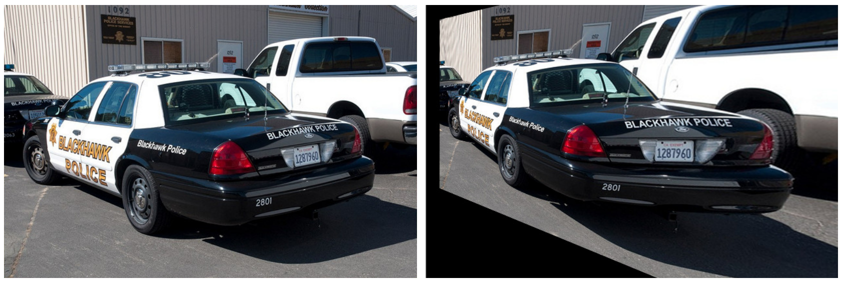

(a) Before (b) After

Figure 3: Warping an Angled License Plate

user, we search through the other plates returned and see if any of those plates match a user. This particularly helps our system in cases of angles, where the plate recognition is not as accurate. Additionally, sometimes the raspberry pi will send up multiple images of the same car when entering or exiting a garage. To prevent logging and/or charging the driver multiple times, we have set up a time delay so that if a car is seen entering or exiting multiple times within one minute, it does not log the additional entries.

2.4 Client Dashboards

In order to provide drivers with a centralized location from which they could register for the EZPark service and view their parking history and statistics, we created a website which displays this information for them.

The client dashboard provides transaction history and basic statistics at a single glance. It also provides drivers with the ability to register multiple cars, edit their payment methods, and see detailed parking history.

2.5 Garage Dashboards

Like the client dashboard, the garage dashboard provides garages with a centralized location to view the statistics and history. In a single glance, they can see how many customers are in their garage, how much they are making in parking fees, and the average time a customer stays in the garage.

Additionally they can view their earnings trend, so they can see if they are doing better or worse as the month progresses. They can also view their daily car volume distribution so that they can learn more about what times of the day the garage is most popular.

2.6 Payments

In order to charge clients for parking, EZPark combines a variety of systems. First, EZPark requires new users to create a payment method. This is done via the Braintree API. Braintree, a subsidiary of PayPal, allows vendors to safely charge customers without having to store sensitive financial information[6]. To do this, EZPark follows strict protocols to manage API keys. Our registration page requests a Braintree key from our server, allowing it to host a Braintree payment form. When drivers fill out this form, Braintree creates a customer object and stores their payment information in their BrainTree Vault.

Upon entering a garage, a Transaction object is created and stored in the EZPark database. This includes the entry time and license plate of the vehicle. Upon exit, the Transaction object is fetched from the database, the exit time is populated, and the total charge is calculated based on the garages predetermined rates. At this time, the server uses a server key to ask Braintree to charge the relevant customer for the calculated amount, completing the transaction.

This system allows EZPark to safely charge customers for parking in a garage, since it ensures that even if EZParks data was leaked, all customer credit card information remains secure with Braintree.

2.7 Gate

To ensure that all drivers must still pay for their time in the garage, the system has a gate that only opens after it has recorded the license plate and charged the users credit card. The gate uses a servo motor to open and close the arm because those motors are designed to rotate a certain amount in both the clockwise and counterclockwise directions. An arduino is directly connected to the raspberry pi and waits for input before opening the arm.

3 Results & Measurements

There are two main metrics that we used to determine the results of the system: the accuracy of the license plate recognition and the amount of time it takes from the moment a car pulls up to the gate to when the gate opens. Below we describe a few areas that were bottlenecks in achieving the desired accuracy and timing.

3.1 Motion Detection

When figuring out when to send an image up for analysis, we initially would compare every single pixel in the image, which turned out to take an average of 14.7 seconds. While the results were very accurate in determining whether something entered the cameras frame, the computation time was infeasible for our system. Instead, the randomization allowed the calculation to happen in 0.37 seconds on average. In making the switch to randomization, we have to ensure that even if there is a loss in quality of motion detection, it still can always detect when a car enters the frame. Figure 4 shows the results of analysis on the motion detection. If more than 13% of the image changes, then we are able to detect that a significant change happened 100% of the time. If less than 10% of

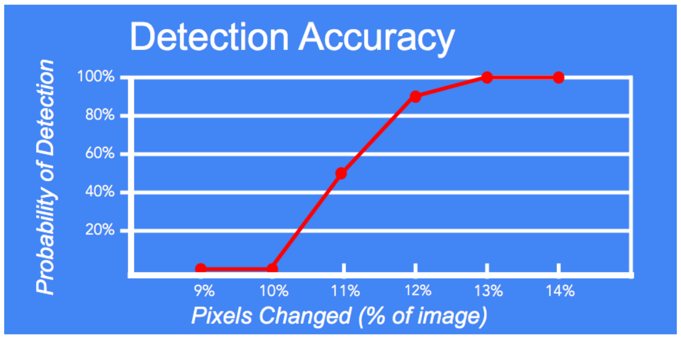


Figure 4: Motion Detection Accuracy Graph

the image changes, the system reliably does not send the image up. In between 10% and 13%, there is some undefined behavior, but with proper setup of the system, a car entering the garage will definitively take up more than 13% of the image, and thus its image will always be sent up for analysis.

3.2 ALPR Warping

To test out the warping we went to a local garage and took pictures of license plates at approximately the same angle (45 degrees) each time. Using these pictures we were able to create a warping configuration that made the angled images look like they were straight on. When we tested the recognition system with these new configurations we were able to achieve a 92.5% accuracy. This is high enough that it would be a viable strategy for the real system. This would require the system to be setup at each location that it is being used. However, once it is configured correctly given the angle, there is no more setup and the camera can be used for any car that comes in.

Accuracies can be found in Table 1.

3.3 Overall System Timing

The overall system has four main components: the motion detection, license plate recognition, web application and the gate. The motion detection takes about 0.37 seconds to determine when to send an image up for analysis. The OpenALRP analysis of image takes on average 1.3 seconds on a wired connection to respond with the license plate. The requests to, calculation in and response

Straight	Angled	Warp
94.4%	35.0%	92.5%

Table 1: ALPR Accuracies

from the web application to log the car takes about 0.8 seconds. Finally, the gate takes 0.1 seconds to open once sent the signal. The system takes on average 2.6 seconds from the time a car pulls up to the gate to when the gate opens.

Current parking garages take a lot longer for their systems to run. For entry, a car must pull up to the gate, stop, and the driver must roll down the window and take a ticket from the machine. On average, this takes about 7 seconds before the car is able to drive through the gate. About 1 second to pull up and stop, 2 seconds to roll down the window, 3-4 seconds to reach and take the ticket, and another second before being able to drive the car forward. For exiting, the process takes a lot longer. In general, it will take 2 seconds to hand the ticket to the booth attendant if the driver has their ticket handy, quite possibly more if the driver is not prepared. Another 2 seconds before the attendant says the amount to be paid. Taking out a credit card and handing it over will take at least 5 seconds, another 3-5 seconds before the card has been charged, and another 5-10 seconds to put your card away and drive through. That means that in current systems, it takes at the very least, 15 to 20 seconds per car. When there are a lot of drivers leaving at the same time, these wait times add on to one another. Our system can save 13-18 seconds per car.

4 Ethical/Privacy Concerns

This project, due to images of license plates being taken, has obvious ethical and privacy concerns. The two main areas of concern are that garages will have access to private information and the storing all these images of license plates in EZPark databases.

To alleviate the first concern, we make sure that garages do not have access to any more private information than they would have by walking around the garage. They are only ever able to know the cars currently in their garage, and they are not allowed access to any of the images. In particular, this means that they have no record of license plates that used to be in the garage. Through the garage dashboard, the data is aggregated for them in easy to understand ways, but there is no more raw data than they already have in existing parking garages. In terms of storing images, we do so only for the piece of mind of customers. This way, they are able to challenge any charge on their card and there will be proof whether or not the charge is valid. About a month after the transaction happened, the image is erased from our system and no one has access to it.

5 Discussion

The process of building a large system with numerous components lead to many different draft implementations and a keen sense of what works and what doesnt.

Through our process of trial and error, we refined our project to include strengths particular to the issues we faced. Once such strength is the efficiency of our motion detection algorithm. In the environment with which we were working, the initial 14.7 second process time was unacceptable. Most people could take out their wallet easily within that time. Being able to cut that down to less than half a second was a solid accomplishment that put the system back into a competitive time frame. Another strength of our final product is the robustness of the web application. With only the bare-bones implementation of our solution, the garages tangible effects are simply removing existing drawbacks. The inclusion of analytics solidified a tangible benefit for the garages, and gives a unique differentiation for our solution over other cost cutting measures.

Our final draft is far from perfect, and through the many pitfalls we ran into, we learned what better resources and more time could add to EZPark. There were some choices we made for the demo that could be strengthened for a practical application. For the demo, we used a single camera that looked forward from the gate to the front of our demo car. In many states, Pennsylvania included, a front licence plate is not required by law. An additional set of cameras for the back licence plate and a way to aggregate the analysis from both cameras could have add some reliability to the system. In a similar problem, our demo camera is not securely mounted and often looks close to straight on to the plate. In a garage setting, this is unlikely, and a more robust solution could compensate for that by using the prewarping images before analysis by OpenALPR.

Generally, the weak links in our project that could be fixed with more time or resources involved the onsite hardware. During demo, our internet connection was through a team members tethered phone, which gave spotty connections at time. Having a wired connection could dramatically improve the speed with which the hardware communicates with our servers and would reduce the uncertainty of the entire system. In a similar vein, the demo gate was put together using the parts from a toy and so is quite small and wouldnt put up with the rigors of a garage setting. The EZPark system could work better with its own native, professional garage gate, or could be built to integrate with existing property, either of which would be an improvement over ours.

Finally, on our server, improvements could be made on existing processes. Our licence plate recognition software communicates several possible licence plate numbers along with their confidence, and we currently do a somewhat naive check for whether there are any in the database. With more time, a deeper selection could be written, where existing plates in the area could be selected despite having one character off (such as by replacing 3 with 8). The analytics could also be easily built on, as there is plenty of information to be taken from the raw data. Discussions with people within the industry would help reveal what the most helpful information would be.

Notes

[1]http://goldengate.org/tolls/
[2]http://www.theiacp.org/Portals/0/pdfs/IACP_ALPR_Policy_Operational_Guidance.pdf
[3]http://www.bostonmagazine.com/news/blog/2013/01/11/gillette-stadium-parking/
[4]http://www.bls.gov/oes/current/oes536021.htm
[5]https://github.com/openalpr/openalpr
[6]https://www.braintreepayments.com/products-and-features/data-security