# Wireless Automobile Detection, License Plate Processing, and Data Availability Network Proposal

Kevin Emery, Santiago Gonzalez, Brandon Rodriguez, Taylor Sallee

*Undergraduates, EECS Department, Colorado School of Mines*

March 17, 2014

**Abstract**

This proposal describes a parking lot monitoring system to be used at the Colorado School of Mines (CSM). The system will help campus members enjoy a better parking experience by spending less time trying to find open lots. This is a semester project for the Computer Science graduate course *Wireless Sensor Networks*, taught by Dr. Tracy Camp at CSM. The proposed work will be an extension of a project started by campus computing group ACMx, which uses a wireless sensor network to track the number of cars in specific parking lots. The system will extend the ACMx project by adding components to the sensor network that record the license plate numbers of cars as they enter and exit lots. The information gathered by the sensor network will be displayed on a website that will allow users to survey the state of the lots before arriving, allowing them to make informed parking decisions. The website will also be the result of collaboration between the authors of this proposal and the ACMx group. The proposal provides some background information, describes the proposed system and its components, and discusses implementation strategies. The goal of the project is a real-world deployment on the CSM campus by May 2014.

1

# 1 Project Description

## 1.1 Introduction

The parking lots at the Colorado School of Mines (CSM) can be a source of frustration for students and faculty members. Because space on campus is limited, the lots are often unable to accommodate everyone who needs to use them. Students need to choose a lot before they go to class; however, knowing which lots have free spaces usually requires driving through the lots looking for a space. The extra time spent searching one or many lots can make students or faculty late for class. If campus members could find out which lots had available spaces before they arrived, much time and frustration could be saved. The goal of this project is to design and deploy a working parking lot monitoring system that will be the first step towards a better parking experience on the CSM campus. The system will keep track of both the number of vehicles in a parking lot and the license plate numbers of those vehicles. The information will be available online to system administrators and CSM campus members. There are two main parts to this project:

1. Parking lot capacity monitoring

2. License plate detection and monitoring

The idea behind the first part is that a student can get on their smartphone, tablet, or laptop before heading to class, see a nicely-formatted list/map of the lots on campus, and decide where to park based on how full each lot is. Administrators will monitor the information and make sure it is correct, updating the web application as needed with new statistics, lists and maps as more lots are added to the system. The Association for Computing Machinery group (ACMx) at CSM has already started working on a system for monitoring lot capacities (see section 1.2). We will collaborate closely with the ACMx group, integrating our work with theirs to create and deploy a working system by the end of this semester (May 2014).

The second part of the project is motivated by a desire to test the feasibility of using the small Raspberry Pi Linux computer as a platform for capturing and processing images of license plates. Because the Raspberry Pi is inexpensive ($25-$35 depending on model), it could be useful in a wide variety of applications that require accurate license plate recognition. For now our web application will simply list the license plate numbers for site administrators to view, but we envision that this monitoring system may eventually be used by the campus parking department to detect when vehicles have entered lots without parking passes.

The goal of this project is to work with the ACMx group to create and deploy the system in two parking lots on campus by the end of the semester. See section 2.7 for information about the deployment. The system itself will consist of several subsystems, which are described in detail in section 2.

## 1.2   Related Work

The ACMx group at CSM has designed a system named SmartLots to track the number of vehicles in a parking lot. Their system is a wireless sensor network consisting of sensor nodes and a central base station. The sensor nodes are based on the Arduino Fio micro-controller platform equipped with triaxial magnetometers and XBee Pro radios. The base station is one small, commercially available Raspberry Pi Linux computer. The information collected by the sensor nodes is transmitted to the base station and forwarded to a server running a small web application. The system is described in [1] and [2]. [1] describes an implementation of a system to detect automobiles entering and exiting a parking lot using the Arduino Fio and the Raspberry Pi. This system uses the IEEE 802.15.4 communications standard to communicate automobile detection data from the sensor nodes to the base station. [2] describes the ongoing status of the ACMx project. At the time of writing this proposal, the ACMx team has created a website that will display the status of the CTLM upper and lower lots once the monitoring system has been deployed. [2] is up-
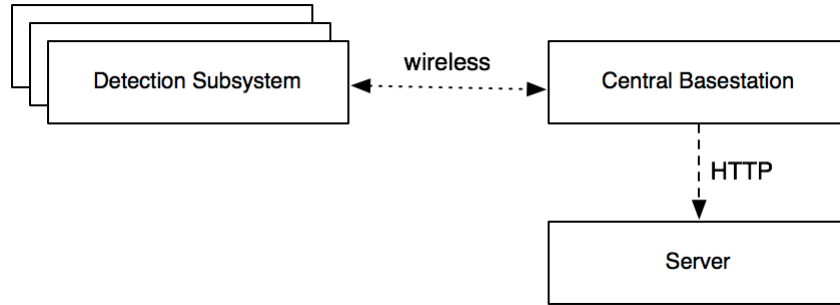
Figure 1: Overall System Architecture

dated periodically with new information about project progress. The project described in this proposal will augment and extend the ACMx project while collaborating closely with Stillwell (ACMx president and author of [1]), with the goal of a real world deployment by May 2014.

## 2 Proposed Work

### 2.1 Overview

The project we are proposing will be an extension of the ACMx SmartLots project. We will collaborate closely with Stillwell and the ACMx group to create and deploy the first working parking lot monitoring system on the CSM campus. Our system will consist of several components, each of which will be either an extension of a current SmartLots system or an entirely new subsystem that will be integrated into the existing system. Figure 1 shows the overall architecture of our system.

The system will rely on three main components:

1. Detection Subsystem

2. Central Base Station

3. Server

The detection subsystem will be made up of a small number of sensor nodes, placed strategically at the parking lot entrances. Each sensor node will contain an Arduino Fio equipped with a triaxial magnetometer that will be used to detect when a car passes the sensor. The work to determine the hardware configuration and software for the Arduino Fios has already been done by the ACMx group. Our contribution will be to also equip each node with a Raspberry Pi mini-computer and camera, which will take pictures of the license plates of cars as they pass by. When a car passes a sensor node, the magnetometer will detect the car, and the Arduino Fio will send an interrupt to the Raspberry Pi via the XBee radio, waking it up so it can take a picture of the license plate.

The central base station will be a Raspberry Pi placed in an upper window of the CTLM building, which will receive all the data from the sensor nodes via transmission from the XBee Pro radios attached to the sensor nodes. This base station Raspberry Pi will also have a camera attached that will take periodic images of the lots from above. The base station will forward the data to the ACMx server.

The server will be running a web application with an HTTP interface that will both receive the data from the base station and route it to the correct part of the application (database, image processor, etc.). This interface will be the method of communication between the server and the base station. When data is received from the base station, the application will process it, store it in a database, and display it appropriately on the website. The following sections describe the architecture and function of each of the aforementioned subsystems in detail. The detection subsystem is broken up into two parts: Automobile Detection (Section 2.2) and License Plate Image Acquisition (Section 2.3). The server subsystem is also broken into two parts: Server Processing (Section 2.5) and Web Application (Section 2.6).
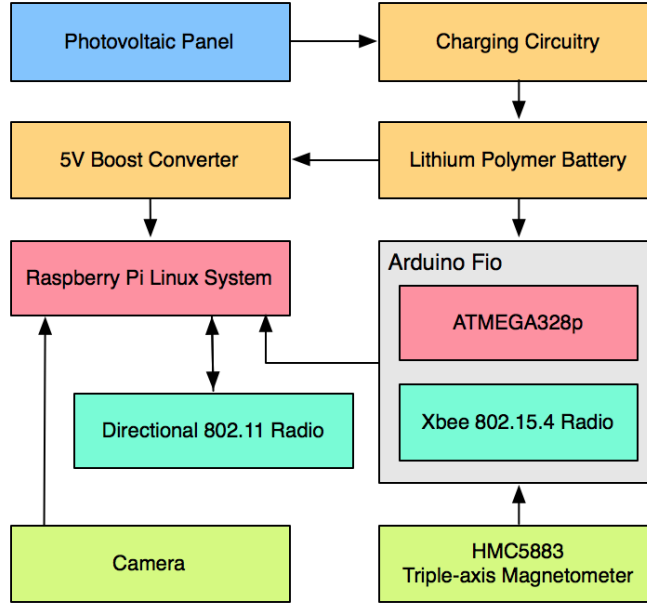
Figure 2: Automobile Detection Subsystem Architecture

## 2.2 Automobile Detection

The automobile detection subsystem provides a means for detecting ingress and egress of vehicles from a parking lot. This subsystem is to be placed on the side of the road next to each parking lot entrance and exit. Automobiles will be detected using a magnetometer which perceives the induced change in the local magnetic field as the metallic structure of the vehicle passes by, as described in [1]. The automobile detection subsystem will be based around the commercially available Arduino Fio 16-bit platform which utilizes the ubiquitous Atmel ATMEGA328p microcontroller. When an automobile is detected by the Arduino Fio, a packet is sent through the onboard XBee Pro radio to the central base station (described in Section 2.4). The packet will contain the direction of the passing vehicle. At the time of detection, a hardware interrupt is also sent to the license plate recognition system to awaken it from sleep.

The hardware to be used for automobile detection will be contained in the same enclo-

sure as the license plate image acquisition hardware for simplicity. Both components will share a unified power supply based on a high-capacity lithium polymer battery which is charged by a relatively small photovoltaic panel as seen in Figure 2. A 5V boost converter will be required to interface with the Acquisition Raspberry Pi since the lithium polymer batteries only provide 3.7V.

The use of a magnetometer ensures that automobiles and other roadway vehicles such as motorcycles are detected, while pedestrians and particulate deposits, such as snow, are ignored. As in [1], we shall use the commercially available HMC5883 triaxial magnetometer as it provides an adequate balance of sensitivity and low power consumption.

### 2.2.1 Tests and Analyses

Because the ACMx group has already written the necessary software and designed a working configuration for detecting vehicles with the Arduino Fios, the bulk of our work on the automobile detection subsystem will involve a variety of tests and analyses to ensure system and data integrity. Power consumption analyses will be conducted on the automobile detection and license plate image acquisition subsystem to identify power usage per main component (Figure 2) while the mote is in different states, such as sleeping and detecting vehicles. Such an analysis will allow us to allocate how much time the mote is allowed to spend in each state and determine the required lithium polymer battery capacity.

Analyses will also be performed on the timing of different actions within the network, including:

- Automobile detection lag

- Imaging of vehicle

- License plate recognition and extraction

- Packet transmission

This will help determine whether certain portions of the image data processing may need to be performed on a server and whether it would be feasible to funnel the image data through the XBee instead of a dedicated directional IEEE 802.11 system.

The efficacy of the automobile detection subsystem will also be evaluated through several tests. A variety of different sized vehicles will be passed through the system at different velocities to ensure high detection accuracy. We will also analyze the end results of deployment after a certain period of time, to determine how often the automobile count needs to be reconciled.

## 2.3   License Plate Image Acquisition

Kevin From Brandon: I called out your section in my section to talk about the determination of cropping and image manipulation on the server versus the Acquisition Raspberry Pi.

## 2.4   Central Base Station

The central base station will be the point of contact between the sensor network and the server. The data gathered by the Arduino Fio and Raspberry Pi sensors will be routed to the base station via XBee Pro radios. The base station, which will itself be a Raspberry Pi, will forward the data to the ACMx server using HTTP requests. Forwarding data is the main function of the base station. The base station will run the Raspbian operating system [3], a lightweight version of Linux. Because Raspbian is optimized for the Raspberry Pi, It should be easy to write some shell scripts that will take care of forwarding the data via HTTP to the server. A second function of the base station will be to take periodic overhead photos of the parking lots to help reconcile the sensed data with the actual state of the lots. System administrators will be able to look at the photos to see if the number of cars reported in the lot by the sensors is correct, and fix any errors. See section 2.6 for more detailed information about the error correction functionality of the web application.

8

## 2.5 Server Processing

Once an image has been determined to contain a license plate, it will be transmitted from the Acquisition Raspberry Pi to a server online via the base station Raspberry Pi. The server will then be responsible for using recognition software technology to extract the digits of the license plate from the image. Upon successful completion, the textual representation of the license plate will be stored in a database residing on the server to be later integrated into the front-facing web application.

The subsystem encapsulating server processing will require a review of recognition software and literature. There exists a market for Automatic License Plate Recognition (ALPR) software that relies on Optical Character Recognition (OCR) engines to extract characters. [4] describes the various methods and features used to extract characters from a license plate. Once an ALPR solution is chosen, it will be necessary to collect a set of test data using the Acquisition Raspberry Pi to tune the workflow of character extraction.

### 2.5.1 Considerations for ALPR Software

For the scope of this project, three ALPR solutions will be evaluated to determine which is the most suitable for our purposes. An ideal ALPR software needs to be currently maintained and capable of reading a license plate if it is skew in an image, taken in poor lighting conditions, and taken with low resolution.

Q-Free Intrada ALPR is a license plate recognition software that offers a C++ API, as well as a cloud-based service. Both the API and cloud-based service can be utilized from a Linux server. Q-Free's software is used in countries around the world for traffic management and toll collection. The wide-ranging geography of its use mean that Intrada ALPR is likely to be reliable and accurate for all types of license plates. Use of the Intrada suite is dependent on Q-Free granting an educational use license for this semester.

OpenALPR is another C++ library for use with both North American and European plates. This library relies upon two underlying technologies: OpenCV (an open-source

computer vision library) and Tesseract OCR (an OCR engine being developed and maintained by Google). The Tesseract OCR Tool is self-sufficient, and relies on included training data. [5] goes into greater detail how Tesseract extracts data from images. The open-source nature of this project make it appealing as it can be immediately integrated without needing to obtain an educational license first. Additionally, the last updates were pushed to the source repository in January. While it is desirable for the software we consider to be relatively up-to-date and maintained, this repository was started only four months ago (November, 2013). Use of such a young library may create a less stable solution than what is practical to work with throughout the course of the semester.

JavaANPR markets itself as an Automated *Number* Plate Recognition library (hence ANPR instead of ALPR) using Java's built-in libraries. This software is also open-source, and therefore offers the ability to be quickly integrated into our testing environment. The documentation of the software has not been updated since 2007, and so this option may be the weakest of the three because it is not clear whether or not it is still being actively developed.

### 2.5.2   Collection of Test Data

Since two of the three proposed ALPR softwares do not require connecting to an external server for computation, it may be necessary to provide training data to improve the accuracy of the system. This would be in addition to the training data that comes with the libraries.

The majority of the plates used in our application can be assumed to be Colorado licenses since it is planned to be deployed in the Denver area. Depending on the amount of training the underlying OCR engines have with Colorado plates, the accuracy of the ALPR software may be able to be improved provided additional images of known license plates.

Collection of test data would require using the Acquisition Raspberry Pi to take images,

as this would simulate the real-world use of the application.

For web services such as Q-Free's Intrada solution, [6] shows a glimpse behind their OCR technology. The Intrada software already has a reliable sum of data from its real-world deployments.

### 2.5.3   Interface with Base Station Raspberry Pi

The recognition and extraction of license plate characters will not directly interface with the Arduino Fio or Acquisition Raspberry Pi that takes the images of automobiles. Instead, it is assumed all communications will come through the base station Raspberry Pi, and that the base station Raspberry Pi will upload received images to the ACMx Linux server (See section 2.6.1 about the HTTP interface).

Upon upload of the image to the ACMx server, a script will be invoked to process the image. The ACMx server will then run redundancy measures to ensure the image uploaded does contain a license plate. Upon a successful identification, the server will invoke the selected ALPR strategy to extract characters from the license plate image. Cropping and image manipulation required by the selected ALPR strategy will be done before invoking this script. The amount of cropping done on the server and on the Acquisition Raspberry Pi is something to be determined throughout the course of this project. Section 2.3 goes into more detail about this subject.

If the chosen ALPR strategy is self-sufficient, all calculations will be completed on the ACMx server. If the chosen ALPR strategy uses a centralized cloud computing strategy, a request will be made to the appropriate server to extract the characters, and the returned result will be used. The cloud computing strategy will also introduce an asynchronous workflow as the ACMx server will perform other tasks while waiting for the result.

Once the ACMx server has a plain text representation of the license plate characters, they will be stored in a MySQL database, along with a timestamp and lot id, where they can be retrieved and processed by the Web Application subsystem described in Section 2.6.

11

## 2.6  Web Application

As mentioned in Section 1.2, the ACMx group has already created a web application that will display the data collected from the sensors in the system. The site is currently accessible online at `http://acmxlabs.org/parking/`. Since the current site is essentially just a shell with no data, one main goal of this project is to make the main pages of the site fully functional. Another main goal is to create an administrator side to the site, where data can be viewed and updated by site administrators. The main focus of this subsystem will be to provide interfaces for both general users and administrators that make the data we collect both useful and easy to access. Since the ACMx team already has some protocols in place for collecting and displaying the data from the magnetometer Arduino Fios, the ACMx team will continue development on this interface, and our group will focus on other parts of the application, as described below. The web application itself will be written with PHP on the back end and HTML5/CSS3/JavaScript on the front end. The following subsections describe the various pieces of the site that we aim to complete during this project.

### 2.6.1  HTTP Interface

The base station Raspberry Pi will need some way to communicate its data with the web application/MySQL database. The simplest way to meet this need is to create an HTTP interface on the server. We will need this interface to handle requests from the front end of the web application anyway, and so it makes sense to also allow the base station to communicate with our server through HTTP requests. We will create a RESTful HTTP interface that corresponds closely to the database schema and the front-end of the webpage, and the base station will send all its data to the application via the same interface. For example, if an image of a license plate is detected, the base station would send an HTTP POST request to 'acmxlabs.org/parking/licenseImage' (actual urls may change when we start designing the interface), and the server would receive the request

12

and send the image to the ALPR software, as described in section 2.5. When the web application needs to get the data associated with a license plate, it will send an HTTP GET request to 'acmxlabs.org/parking/licensePlate/123ABC'.

### 2.6.2 Viewing License Plate Data

As mentioned in section 2.5, the license plate images will be translated from an image into a string of plain text, and stored in the MySQL database on the ACMx server. One of the main functions of the web interface will be to display this data to administrators. The page that displays the license plate data will have a simple interface which will allow administrators to filter the data by date/time, lot, and license plate number. For example, an administrator should be able to type in a specific license plate number and see all the dates/times that plate has been seen, and in which lot. Alternatively, an administrator will be able to enter a date range and lot(s) and see all the license plates that entered/exited the lot(s) during that date range. For now we are only interested in a proof-of-concept type interface that shows we have the capability to display the collected plate data in a useful format; in the future it could be incredibly useful to cross reference our data with the campus parking department's database to check if a car has entered a lot without a permit.

### 2.6.3 Validating and Adjusting Lot Capacity Data

We expect that our first deployment of this system may not be 100% accurate at detecting all vehicles that enter and exit lots. Therefore, as mentioned in section 2.4, we will be collecting periodic overhead images from a camera mounted on the base station Raspberry Pi. These images will serve as a visual representation of how many cars are actually in the parking lot at the time the image was taken. Administrators should be able to periodically compare this image to the collected data from the sensor network to make sure the numbers being computed by the network accurately represent the number of cars

in the lot. As previously mentioned, the deployment of this project will only be on two campus parking lots, and we may only have one overhead camera to start off with, although ideally we would have two - one for each lot. The web interface will allow administrators to pull up the image for any lot with an overhead camera and see the numbers for that lot right next to the image. If any discrepancies are found, the administrator should be able to overwrite the count in the database to reflect reality. This interface will be extremely useful in helping to calibrate the system. For example, if the sensors are consistently reporting less cars in the lot than the image shows, we can fine-tune the collection process to make sure fewer cars slip through unnoticed. Future work in this area will include an automated version of this process, where a computer program will compare the image to the sensed data and make any necessary corrections.

### 2.6.4 User Accounts

As the site currently stands, there is no way to restrict certain data to administrators. We would like to create a simple user log-in system that allows general users to view the main parts of the site (maps of lots, state of each lot, stats and trends) without having to log in, and only allows access to the administrator side of the site (2.6.1 and 2.6.2, among other parts) if the user has a valid administrator username and password. This part of the project will also include an administrator interface that allows top administrators to create, update, and delete other administrators.

## 2.7 Deployment

The system is planned to be deployed at the CTLM Upper and Lower parking lots at CSM. These parking lots were chosen because of their accessible location, unobstructed visibility from a campus structure, and wide use. Both parking lots have two entryways, each of which will be equipped with an automobile detection subsystem as shown in Figure 3. All entryways also benefit from having a clear view of the sky for increased photovoltaic
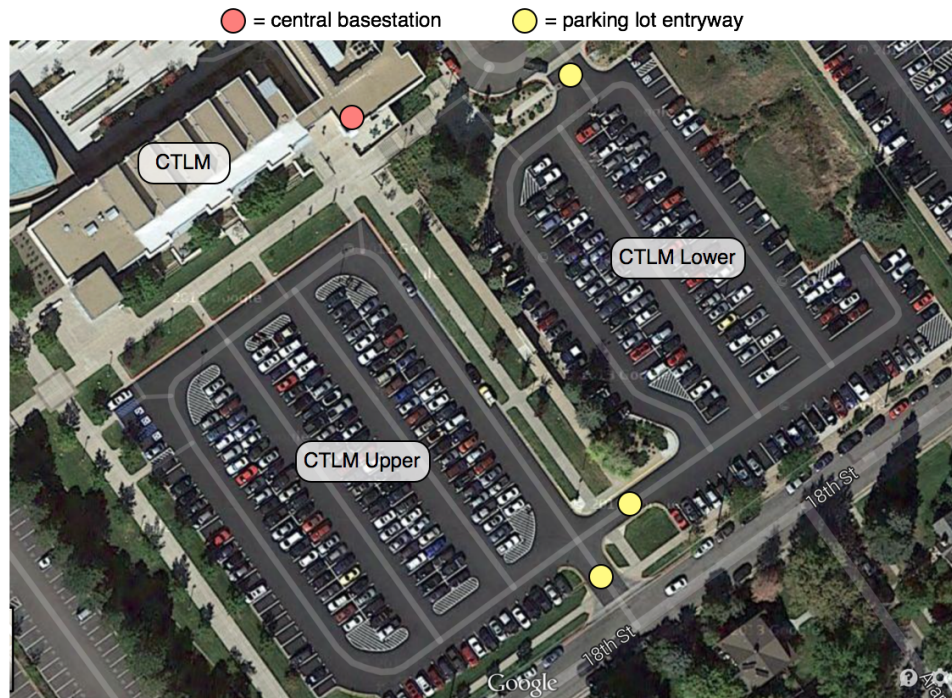
Figure 3: Deployment Location

efficiency.

The central base station, as described in Section 2.4, will be located on the second level of the CTLM building. Several potential locations exist which allow for a clear line-of-sight to all automobile detection subsystems, one of which is detailed in Figure 3. These locations also provide internet access to the base station Raspberry Pi through ethernet ports.

# 3    Summary

A summary [7] Talk about the motivation behind this project. Why it is important.

# References

[1] R. Stillwell, A. Wilson "Magnetometer Parking Sensor," *EGGN 383 Final Project, Colorado School of Mines.* December 12, 2013.

[2] R. Stillwell. (2014). *Parking Sensor Wiki* [Online]. Available: `http://github.com/ColoradoSchoolOfMines/parking_sensor/wiki`

[3] *Raspbian* [Online] Available: `http://www.raspbian.org/`

[4] S. Du et. al., "Automatic License Plate Recognition (ALPR): A State-of-the-Art Review," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 23, no. 2, Feb. 2013.

[5] C. Patel et al., "Optical Character Recognition by Open Source OCR Tool Tesseract: A Case Study," *Intl. Journal Computer Applications* vol. 55, no. 10, Oct. 2012.

[6] Q-Free ASA. (2014). *OCR Technology* [Online]. Available: `http://www.q-free.com/product/ocr-technology/`

[7] X. Johnson