Toward the Creation of a Novel Electric Bike Rental Program to Ease University Congestion

Jacob Barnes
The University of Mississippi
Oxford, Mississippi, USA
jbarnes2@go.olemiss.edu

Barry Muldrey
The University of Mississippi
Oxford, Mississippi, USA
muldrey@olemiss.edu

Charles Walter
The University of Mississippi
Oxford, Mississippi, USA
cwwalter@olemiss.edu

ABSTRACT

At many large universities, parking and moving about campus can be difficult, in part due to high student foot traffic and large distances between parking lots and university buildings. While the universities often encourage alternative means of commuting on campus (bikes, skateboards, etc.), these often see limited adoption and, in some highly-trafficked areas, pose a safety risk. To combat these issues, we have begun creating a new, in house electric bike rental program, dubbed SharkCycles, with the goal of providing students with the ability to move around campus safely and quickly. However, there needs a way to manage the bikes to limit pedestrian collisions, keep track of the bikes on campus, and examine traffic patterns of students utilizing the bikes. In this paper, we will discuss a new system to manage electric bikes using geofencing that can be configured for and orchestrated to each bike. This method ensures reliability and safety by storing and processing location data with onboard embedded computers instead of external servers while still allowing for quick response to policy changes.

CCS CONCEPTS

• **Computer systems organization** → **Embedded and cyber-physical systems**; • **Hardware** → *Communication hardware, interfaces and storage.*

KEYWORDS

Geofencing, Internet of Things, GPS Tracking

ACM Reference Format:

Jacob Barnes, Barry Muldrey, and Charles Walter. 2022. Toward the Creation of a Novel Electric Bike Rental Program to Ease University Congestion. In *2022 ACM Southeast Conference (ACMSE 2022), April 18–20, 2022, Virtual Event, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3476883.3520210

1 INTRODUCTION

Location services have become ubiquitous across computers and mobile devices today. Cell phones, specifically, have the ability to use the data from onboard GPS receivers for many purposes, including GPS-based directions through apps like Google Maps and Waze. Most of these services work by sending data from the cell phone to an external service for processing. In this scenario, the phone acts primarily as a sensor by finding a location, sending that to other systems for processing, and then waiting for a response to tell it what to do based on its location [4]. While this process works well for apps focused on providing directions to users, issues arise when the user needs real-time responses to GPS location information.

Self-driving cars from companies like Google's Waymo [7] have faced similar issues. The cars must be able to react quickly to changing road conditions so that they can follow the road and avoid collisions. Today, these issues are solved by computing the data on high-performance computers that are on-board the vehicles [11]. This allows them to react in real time to pedestrians and changes in road conditions.

Recently, new developments have been made in the use of GPS data for virtual fence systems using the location data from a mobile device [12]. This concept has been termed *Geofencing*. Most of these systems, however, rely on external servers to process the location data and then receive a response with commands to execute. This solution shares many of the disadvantages of direction apps, namely the lack of real-time response and reliance on a consistent internet connection.

In an effort to solve the congestion and parking issues on large college campuses, our University Department of Parking and Transportation (UDPT) is in the process of creating a program to allow students, faculty, and staff to rent electric bicycles. Electric bikes have two primary advantages. The first is that they assist the rider so that they do not have to work as hard to pedal the bike. This is particularly useful for steep hills that must be regularly traversed by students. However, students can ride much faster on an e-bike than on a regular bike, making them potentially unsafe for pedestrians nearby if ridden on a busy walkway or sidewalk. This leads to an important question: "How are we going to keep students safe both on and off these bikes?"

In this paper, we discuss the initial development of a system to allow the monitoring and control of rental electric bikes. Thanks to the ever-decreasing cost, size, and power consumption of mobile CPUs, computer engineers can now create small portable devices with enough processing power to compute their location on their own. Using that data, the devices can execute certain actions based off that location data.

We utilize Geofencing as a means of ensuring safety for both riders and pedestrians on campus. Using Geofencing, the bicycle's onboard computer can detect if it is inside predefined areas on campus, and then dynamically adjust certain settings, such as maximum speed and pedal assist. We also use this technology to detect

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
ACMSE 2022, April 18–20, 2022, Virtual Event, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-8697-5/22/04…$15.00
https://doi.org/10.1145/3476883.3520210

other scenarios, like if a bike has left campus, and alert department administrators or automatically immobilize the bike until it can be retrieved.

This also provides us a way to record the location data from a user's ride and provide analytics to them, showing information such as the distance, average speed, and path taken. Furthermore, the recorded data would be useful if a bike is ever vandalized or damaged. In such an event, the data could easily be provided to law enforcement to help find the persons responsible.

2 RELATED WORK

Our university is not the first organization to create an electric bike rental program. However, while others have built similar systems, almost all of them track the bike and combat theft or other malicious activity through some kind of out-of-band means. Usually this is accomplished by using the renter's cell phone [8]. Other systems have different requirements to keep users from stealing the bikes. CitiBike infamously makes their bikes overly heavy and once required that the bikes be checked in to a station every 30 minutes to ensure that they were not stolen. If a rider missed this deadline, fines would be charged to the renter. This has recently been changed and you can now use the bike for up to 48 hours before it is deemed "lost" [2].

In addition to rentable bicycles, rentable e-scooters have made their way onto the market. These systems use onboard GPS devices, but send their data to servers in the cloud for processing [9]. Systems like this are cheaper and much more widely available, but have their own issues. These GPS trackers are prone to failure if the servers were ever to go down, making the e-scooters unable to be ridden since the security features are also tied in with the online services. They can also be easily bypassed, as numerous news articles have shown, leading to mass theft and vandalism [5][6]. Furthermore, these e-scooters lack the more advanced safety features of our system, which can detect issues without having to report back to the central control system.

Aside from rentable vehicles, almost all geofencing arrangements (and most applications of GPS technology in general) use off-device processing. Even when not using GPS for localization, processing is almost always performed remotely. Wang et al. [13] utilized accelerometer and magnetometer data collected using a mobile app on a smartphone for indoor localization. This data was then sent via a network connection to a realtime database. A separate server was set up that monitored the database for changes and once found, ran calculations to compute the movement of the device. This movement data was written back to the database so that the smartphone app can read the new location. This would work well in an area that always has good network connectivity but when the GPS receivers are used in the real world, this is not always the case.

Our solution aims to solve many of the issues with these systems. By processing all the location data on the device, we eliminate the need for backend servers that could cause reliability issues. All geofencing processing happens on the device so even if the communication channels fail, or if our servers stop responding, the bikes can still protect users by checking their location against the local configuration data. Servers are still necessary in our use case because they are used to orchestrate settings to all of the bikes and record information to be displayed on the web portal.

This management system, where configurations are orchestrated to each bike and stored locally on internal storage, was inspired by Dell's Wyse Thin Client products [10]. Dell created a piece of management software called the Wyse Management Suite (WMS) that has the ability to orchestrate settings to up to one million devices [3]. This software has a web portal where settings can be configured, along with a backend server that the thin clients communicate with to receive their configuration data.

When the Wyse thin client devices start up, they connect to the WMS server and check for any changes in their configuration. If no changes are found, they continue with their boot-up process, but if changes are found, they request the full configuration from the server and save it to the device's local storage. For the thin clients, these configurations could include hundreds of settings including server connections, desktop backgrounds, networking configurations, and much more. This management scheme allows the thin clients to function as normal if the server is offline due to the fact that the configuration is stored on the local device. The WMS system is very similar to how the SharkCycles bikes receive their settings, checking in and then downloading the configuration only when there are changes available.

3 PREVIOUS EFFORTS

Our electric bike rental system, SharkCycles, originally began as an Electrical and Computer Engineering master's project, where several graduate students worked together to create a prototype bicycle (Figure 1). This prototype was built from off-the-shelf parts, purchased from an electric bike vendor - including a battery, motor, and other control circuitry. In addition to the kit, the students built their own hardware that made use of a Raspberry Pi and an onboard LTE cell modem. This prototype hardware had some basic reporting functionality, connecting to a server that was hosted by campus IT. Every three to five seconds, the Raspberry Pi would send a message to the server to report the current location and battery level. Based on Barbeau et al. [1], we can expect battery life of around 50,000 seconds with this rate, which should ensure power throughout a normal day of operations. It also had the ability to report other information about the state of the bike, such as if it was currently being ridden or if there were errors.

Our University's Department of Parking and Transportation liked the prototype but had one major problem: they needed to be able to keep riders from hurting others by riding the bikes quickly through busy areas on campus. In addition, the department also wanted to prevent riders from stealing the bikes or participating in other malicious activities with them. To mitigate these issues, we began work on a system to track and control the bikes remotely. This system needed to be able to dynamically change the bike's configuration based off location data from the bike by checking whether it was inside a geofence set up by the administration.

4 GEOFENCING FOR ELECTRIC BIKES

Our bike management system is split into 3 major parts: the bike hardware, the bike server, and the web portal. The bike's hardware contains a GPS receiver and a network modem, as well as

Figure 1: Initial Prototype Bike

the components needed to provide pedal assist. The "bike server" handles bike check-ins and configuration updates, in addition to bridging the gap between the bikes and the web portal. Finally, the web portal allows users and UDPT staff to interact with the system. Both the bike server and web portal are connected via a single SQL database. Figure 2 shows the design of the system.

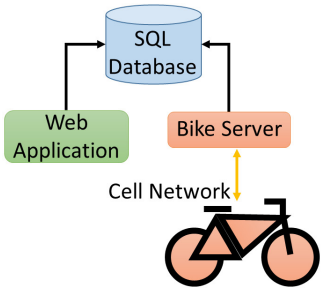

Figure 2: System Layout Diagram

4.1 The Bike Hardware

Each bike is equipped with a custom-designed and low-powered computer that has hardware similar to a Raspberry Pi. These devices contain a GPS receiver to find their location and a LTE modem to allow them to connect to the internet. Each device also has software that communicates with the bike server using a custom protocol, similar to HTTP, that was developed for this project. In addition, the devices have onboard storage to hold configuration data, which is downloaded periodically from the bike server. This stored configuration data is used when the device checks its location to detect if it is inside a configured geofencing scheme.

When initially designing this project, we realized there was a need for some very small geofenced areas and that many areas on campus may require a faster response to GPS data than is possible when awaiting server responses. This necessitated shifting the decision making process from the server to the bike itself. As mentioned in Section 1, this paradigm is very different from similar systems that use external devices (ex. the user's phone) or an onboard device, both of which report the location data to external servers for processing. The main reasoning for this is to make the bikes more responsive when entering or leaving a specified zone and to prevent tampering with the location data. Even though the bikes do all location processing using the onboard computer, some data is still reported back to the bike server so that it can be displayed on the web portal.

When a user starts a ride, the bike begins continually checking its location (accurate to ±5 feet) using the built-in GPS receiver. It uses that location data to see if the bike is inside any geofencing zones that are specified in the saved configuration. If a match is found, the bike executes any actions that have been configured through the web portal, giving UDPT control over various settings (e.g. maximum speed, maximum pedal assist). While geofencing can restrict users from riding the bike off campus to steal it, what would happen if someone were to stick the bike in their car and drive away? Another advantage of on-device processing is that we can always know where the bike is at all times. Most bike rental systems use the renter's mobile device to record the location of the rented bike during their ride, but this can be problematic because users can use software on their phones to manipulate the device's reported location easily. In our system, a scenario like this would have no effect because users cannot manipulate the physical hardware on the bike to change the detected location. Furthermore, if the hardware was tampered with, this would be easily detectable because it would no longer be checking into the server.

4.2 The Bike Server

The bike server is mainly responsible for bridging the gap between the bikes and the web portal. It does this by communicating with each bike to receive status information and send configuration data. All of this data is stored in the shared SQL database as shown in Figure 2.

When designing the bike server, we had two main goals in mind: reliability and simplicity. The server needed to be reliable because we could potentially injure riders or people near them if something went wrong, and simple because we wanted to be able to serve large numbers of bikes without requiring large amounts of server resources. Reliability was achieved by doing as much processing on the bike's onboard hardware as possible. This ensures that even if there was a server outage or network issue, the bikes would still operate correctly. In an attempt to simplify the server (and clients) as much as possible, we decided to forego implementing a full HTTP stack and instead implement a custom protocol because it would require less resources to process requests. This also allows us to implement our own authentication methods and message verification processes to ensure that the messages are legitimate.

Every three to five seconds, the bike's computer checks in to the bike server and sends the current location, battery percentage, and status information. There are four main steps to the check-in process, also known as a heartbeat. Those steps are: (1) transmission of status information to the bike server, (2) validation and storage of the data, (3) updating of the configuration on the bike (if necessary), and (4) executing any pending actions (like unlocking the bike).

When the bike checks in, the submitted data is validated by the server for legitimacy and recorded in the shared database for the web portal. This is done by checking the authorization header that is passed as part of the request (see Figure 3) and validating the format

of the data. The server then calculates the current configuration version and returns that, along with any pending control commands, to the bike. After the bike computer receives the configuration version, it checks the current cached configuration to see if it is older than the version reported by the server. If the configuration version from the server does not match the version stored on the bike, the bike will request the latest configuration from the server and the server will reply with the current configuration for the bike to store. Finally, after any updates have completed, the bike executes any pending commands that were sent in the initial response such as unlocking the bike for a rider to use it.

Each message (Figure 3) starts with a header that defines the verb (whether the client is receiving data or sending data) and the command. The message contains request headers that can pass information like the type of data that is being sent and the authorization header to authenticate the bike. Finally, any other data being sent to the server is included at the bottom and the message is ended with a blank line.

```
POST Heartbeat/Invoke
Authorization 00000000-0000-0000-0000-000000000000
Content-Type JSON
{"latitude": 123, "long...}
```

Figure 3: Example Bike Message

Due to the high frequency with which these messages are being sent, the bike server was designed to be run in a distributed configuration. Multiple instances of the bike server can be run simultaneously, with a network load balancer spreading the requests across them all. This will allow us to easily scale our operations as more bikes come online across the network.

The primary reason that the bikes continuously poll the server is to ensure the bike will unlock promptly after a user has rented it. The unlock command is sent via the response from the heartbeat command. In order to keep users from waiting a long time for the bike to unlock when it is rented, we have chosen keep the polling rate quite short. Additional investigation is needed to determine an optimal heartbeat timing for both battery life and user experience.

In the case of a heartbeat message (similar to that shown in Figure 3), the data is saved in the shared database so that it can be viewed by the web portal. If needed, the bike server can send configuration data to the clients. Configuration data is also saved in the shared database so that it can be read and edited by the web portal.

4.3 The Web Portal

Each setting that is saved in the database can be managed through the web portal's administrator interface. While the bike server records some data (bike location, battery percentage, etc.) during the heartbeat process, the web portal is the "face" of the project and is the primary way that users (of all kinds) interact with the system. The system is designed to have two users, normal users, who primarily interact with the web portal to rent bikes, and administrators, who can set system settings and geofencing zones.

Normal users are shown a map that displays the current location of all active and available bikes, but any bike that is currently in use will be hidden from the normal user view. To begin a ride, a bike can be selected from the map or by entering in a code printed on the bike. The portal records that the bike is ready to start a ride so that the next time that it checks in, the unlock command will be sent. The user can ride the bike to their destination and when they have completed their ride, they can end the ride through the website. When the "end ride" button is clicked, the web portal sets a flag in the database to lock the bike and calculates the user's final fare. After the ride is completed, the rider can view their account data, such as the rental price, past rides, and account balance. They also will be able to view analytics about their past rides, allowing them to see the path they took, total distance, average speed, and more.

The staff and administrators at UDPT use the web portal to manage overall system settings. Some of these settings are simple, such as user accounts and pricing, while others are more complex, like the configuration on each bike and the geofencing zones. Using the app's visual geofencing editor (Figure 4), schemes can be created that allow an admin to define areas and then apply actions to that area. Actions include slowing down, reducing pedal assist, or completely stopping the bike.

For each scheme, these actions can be configured for when they enter or leave a zone. Staff can also define a schedule for when the schemes will be in effect, which is necessary for safe operations during high-attendance, on campus events (home sports games, graduation, etc.). All of the configuration data is saved in the shared database so that it can be read by the bike server when it is requested.

The web portal has been designed with privacy and security in mind. Log-in is through the university central authentication system, bike server communications are encrypted with a complex shared secret key, web requests to the portal require HTTPS, and data is encrypted in the SQL database to limit the risk of data leakage. Our servers are also configured to use best practices to further ensure secure data storage. Data shown through the admin portal is limited to bike location and if the bike is rented, with no direct connection between bikes and current users.

4.4 Testing

At the time of writing, the project is still in active development with multiple groups working on different aspects of the project. Until the hardware teams finish designing and manufacturing the control PCBs and other components, we are unable to fully test our system. However, the teams have had some small-scale tests that were quite promising. A demo at The University of Mississippi's "E-Day", an academic preview day for high-school students, was extremely successful. The software on the original prototype bicycle was modified to accept the new message format, and it was successfully able to continuously send messages to the bike server. The only mild issues seen were with the network connection, due to the prototype hardware only supporting WiFi at that time.

Request capacity for the bike server is something that the software teams have been keeping an eye on during the design, testing, and implementation phases. During our recent test, we did not see CPU and RAM loads become an issue, with a lot of capacity remaining to process other requests. Should either the capacity for the server become overloaded, or the batteries on the bike lose charge

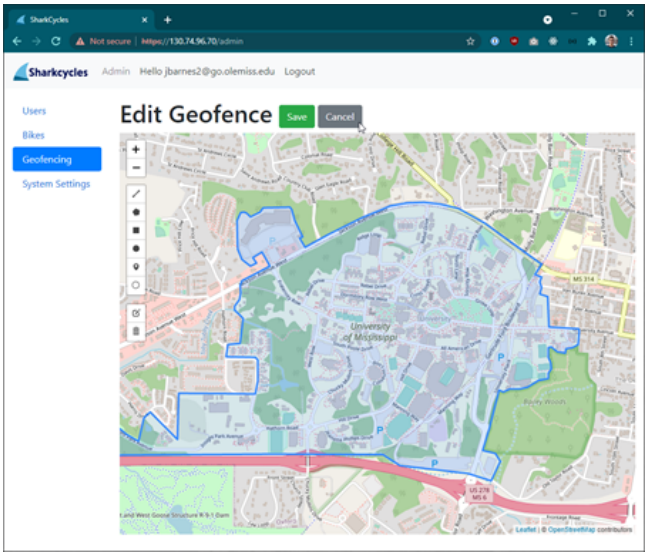

Figure 4: Visual Geofencing Editor

quickly, the interval for server communication could be increased, as recommended in [1].

5 CONCLUSION

Today's college campuses can be difficult for students, faculty, and staff to navigate due to the large distances between buildings on campus and parking structures for those buildings. Campuses can also be crowded, leading to heavily-used sidewalks and paths with students and faculty hurrying to their next classes. To alleviate some of these issues, a new electric bike rental program called "SharkCycles" is in development at our university. These bikes will be rentable through a website, accessible from the internet, so that students can use them to get to class and other locations on campus more quickly.

In the planning of the project, the Department of Parking and Transportation of our university was concerned about accidents happening between bike riders and pedestrians. They wanted a system to be able to manage the bikes, in addition to being able to define areas on campus where the bikes were speed-restricted to prevent collisions with pedestrians. We have developed a system using geofencing technology that allows department administrators to configure restricted areas on campus through a web-based interface. This data is sent, and saved to, low-powered computers that control each bike. These computers can detect, in real time, if the bikes are inside of one of these restricted zones and apply speed policies accordingly.

Some other bike rental systems (such as CitiBike [2]) use the renter's cell phone (or other similar means) to track the bike as it is being ridden. In our use case, using an external device, or external services, for GPS processing is not feasible because our bikes need to be able to respond quickly to changes in their location by dynamically updating the configuration of the onboard electric motor. To solve this problem, we have decided to do all location processing using the built-in computer system on the bike. This gives us much better reliability and responsiveness when moving through different areas on our campus.

6 FUTURE WORK

In the future, we plan to run large-scale testing of the web portal and bikes on campus to investigate how students use the bikes and determine an optimal heartbeat timing for both battery life and user experience. Additionally, we plan to expand the system to allow for bikes to be placed in different groups, each with their own configurations. This would allow us to have sets of bikes to operate only in sections of campus where there are dormitories. The bikes could also be configured to operate only on one of our remote campuses, giving us the flexibility to configure areas on those campuses also.

The data from rides could also be used for research in the future. Due to the fact that location data from each ride is stored in the shared SQL database, this can be extracted and used at a later date in exercise research or research into the movement patterns of students on campus. This is only possible with our system because of the amount of analytics that each bike directly recods during a ride.

Finally, showing advanced ride statistics is another feature that can be implemented in the future. Right now, our application shows basic stats about past rides, like the distance and time, but a more advanced view could be created using the data. Popular workout apps show users lots of information about their rides, which would be useful for any renters that use the bikes for exercise.

REFERENCES

[1] Sean Barbeau, Miguel A. Labrador, Alfredo Perez, Philip Winters, Nevine Georggi, David Aguilar, and Rafael Perez. 2008. Dynamic Management of Real-time Location Data on GPS-enabled Mobile Phones. In *2008 The Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*. 343–348. https://doi.org/10.1109/UBICOMM.2008.83

[2] CitiBike. 2019. *Citi Bike Rental Agreement.* https://assets.citibikenyc.com/rental-agreement.html#section_7

[3] Dell. 2021. *Wyse Thin and Zero Clients Managing Software Suite | Dell USA.* https://www.dell.com/en-us/work/shop/wyse-endpoints-and-software/wyse-management-suite/spd/wyse-wms

[4] Sergio Ilarri, Eduardo Mena, and Arantza Illarramendi. 2010. Location-dependent Query Processing: Where We Are and Where We Are Heading. *ACM Comput. Surv.* 42, 3, Article 12 (March 2010), 73 pages.

[5] Miles Klee. 2019. Inside the Lawless New World of Electric Scooter Hacking. *Mel Magazine* (2019). https://melmagazine.com/en-us/story/inside-the-lawless-new-world-of-electric-scooter-hacking

[6] Sasha Lekach. 2018. E-scooters Can Be Hacked. Here's What Companies Are Doing About It. *Mashable* (Dec. 2018). https://mashable.com/article/e-scooter-hacks-bird-lime

[7] Waymo Localization. 2021. *Waymo – Waymo.* https://waymo.com/

[8] Motivate International, Inc. 2018. *Motivate Privacy Notice.* https://assets.citibikenyc.com/privacy-policy.html

[9] NIU Electric Scooters. 2020. *NQi Series: App | NIU Electric Scooter.* https://www.niu.com/en/n-series/app/

[10] Dell Support. 2021. *Support for Wyse 5010 Thin Clients / D10D/D10DP/D90D7 | Overview | Dell US.* https://www.dell.com/support/home/en-us/product-support/product/wyse-5010tc-series/overview

[11] B.V. Tiulkin and N.V. Kulabukhova. 2018. Convolutional Neural Networks for Self-driving CARS on GPU. In *CEUR Workshop Proceedings*, Vol. 2267. RWTH Aahen University, 611–614.

[12] OPEN TEXT SA ULC. 2021. System and Method for Geofencing.

[13] Yunshu Wang, Lee Easson, and Feng Wang. 2021. Testbed Development for a Novel Approach Towards High Accuracy Indoor Localization with Smartphones. In *Proceedings of the 2021 ACM Southeast Conference*. 79–86.