Abstract – Our project aims to allow mobile users to submit their pedestrian movement data, then receive real-time updates of the current traffic conditions. We look at the requirements and categories for a crowdsourcing framework to gather and utilize pedestrian traffic data, including an application prototype and backend server. We looked at the possible integration of a cloud framework into our system pipeline, parallelizing the traffic processing, reducing the computation time. Finally, we assessed the UI presentation of pedestrian traffic conditions based on user surveys.

I. INTRODUCTION

Navigating through a huge crowd, where thousands of people are walking in different directions can be a daunting experience. We probably all have experienced this at some points, whether it is exiting the football stadium after a game, leaving a crowded public lecture, or walking anywhere at the Minnesota State Fair. Unless we have an aerial view of the whole area, our limited field vision and lack of spatial traffic information can prevent us from making the preferable choice of walking routes. With the wide spread of smartphones usage, plus their increasing computational power and connectivity, there is a good opportunity to make use of collective pedestrian GPS data, process/predict the traffic condition and distribute this back to the users in real time. This information can also be used to revise structural designs.

II. CONTEXT & MOTIVATION

The study of pedestrian movement is relatively unexplored but very important for building design, particularly for cases of evacuations. This information can also aid in planning for evacuation to most efficiently control the pedestrian traffic in a way that minimizes the time required to remove people from the dangerous areas. It is often the case that certain exits are underutilized and by evaluating how to best route people to utilize all the available capacity greater speed and efficiency can be achieved. [1] (Zia et al 2011). This information can also be used at the design level by strategically placing building exits in a way that naturally balances load. A situation where this is a common problem to be solved is the case of sports or other performance venues; buildings which often have evacuation-like occurrences as nearly all occupants leave the area at approximately the same time. Therefore one of our goals is to look at the use of collected pedestrians data to generate trajectories that can be used to better understand behavior. The use of trajectories has been often utilized to understand vehicular traffic however it has seen relatively little use to understand pedestrians.

As discussed in the first section, mobile phones, with their increasing capability and usage, are great tools for collecting data. Several popular mapping services such as Google Maps, Waze make use of crowdsourced user data to generate real-time traffic updates. This motivates us to investigate a similar approach for pedestrian traffic. However, in order to have useful results, the system will require a lot of data for a more accurate assessment of traffic conditions. A real-time map of the current traffic congestion will be beneficial for users in the mentioned scenarios, encouraging even more users to participate in submitting their pedestrian data, which in turn will improve the results.

III. PROJECT OVERVIEW

A. Problem Statement

Our main objective is to allow mobile users to submit their pedestrian movement data, then receive real-time updates of the current traffic conditions.

B. Challenges

The main spatial challenge in our general problem is that pedestrian movements are less constrained
Another challenge for our specific investigation is the process of collecting real world data for experimental purpose. Due to the time constraint and lack of public traces, we were only able to collect limited pedestrian movement trace, and had to generate synthetic data for the rest to fill in our application scenario. This proved difficult since there are no predetermined rules or patterns that we can base our workload generator on.

C. Project Goals & Contributions

For this project, we investigated the requirements and categories for a crowdsourcing framework to gather and utilize pedestrian traffic data. We built an Android application prototype that collects user’s GPS trace, which is submitted to a backend server that analyzes the traffic congestions by areas, then send this information back to the user mobile devices to display. We looked at the possible integration of a cloud framework into our system pipeline, in order to parallelize the traffic processing, reducing the computation time. Finally, we assessed the UI presentation of pedestrian traffic conditions based on user surveys.

IV. RELATED WORKS

The Mobile Millennium project [2] is a pilot traffic-monitoring system that uses GPS in smartphones, taxi cabs to gather traffic information. The system processes this crowdsourced data and distributes traffic conditions back to users’ mobile phones in real-time. They make use of machine learning, the Expectation-Maximization algorithm, to infer traffic congestions on arterial road links. This is an iterative algorithm that is both computationally and memory intensive, so they leverage the cloud to perform this traffic estimation process. The system was deployed on their Berkley campus, then expanded to the San Francisco Bay area, Sacramento and Stockholm. However, their main limitation is that the system only works with vehicle GPS traces on arterial road networks.

Another project looking at crowdsourcing data was studied by Calabrese et al. in 2011 [3]. This paper analyzes the Localizing and Handling Network Event Systems (LocHNESs) system developed by Telecom Italia. The LocHNESs system works by triangulating user locations when they connect to the cellular network such as to make a call or send a text message; this is accomplished using signal strengths and time stamps related to the connection between the mobile device and the cellular towers. This system has the advantage of not requiring special capabilities of the data sources such as special applications or other reporting services. It is, however, rather coarse as limited accuracy is possible using the method; currently for urban areas they aggregate the data to a 50m grid in urban areas. Particularly for pedestrian traffic, this inaccuracy can be problematic given the low speeds as small errors in location can give very large errors in speed, one of the more important measurements related to flow.

V. DESIGN & IMPLEMENTATION

A. Offline vs. Online Definition & Requirements

We first investigated the requirements needed for our crowdsourcing frameworks. When considering how to best gather and store data, we first need to determine what the desired data is. While this will obviously depend on the specific application, there are two major categories that these applications will fall under: either an online application or an offline application. An online application is one that utilizes the data in real-time or near real-time. An example of this would be a pedestrian traffic service that allows users to see crowded areas (where movement is restricted) in order to plan their route; for example, to take a different exit when leaving a stadium. For an online application, one of the biggest concerns will generally be the speed at which data can be processed and provided to end users. Additionally a system should be adaptable to many variations throughout the processing pipeline as data sources are gained or lost, processing capabilities change, end-user demand changes, etc.

The other major category of applications is offline ones, in that data is used at a future time. Research is a common offline application, whether it be to understand pedestrian behavior or to identify problematic areas in a building. Here, we are generally interested large amounts of interrelated information such as speed, surrounding conditions, weather etc. This data can also be added to the database at an arbitrary time, for example when a user submits a trajectory recording. For offline applications, processing speed is generally considered a non-issue since the processed information does not need to be used immediately. What is of concern is storing the
large volumes of data in a way that allows for convenient access to the data subsets of interest in a way that is efficient in terms of storage and the ability to be updated.

B. System Pipeline

Based on the categorization above, we have chosen to focus first on designing the framework for the online applications. This pipeline includes four main stages: data recording, collection, processing and presentation.

![System Pipeline Diagram]

C. System Components

As seen in Figure 1, we look at four main stages of our system. The GPS Recording stage happens on the user mobile phones, we have developed an Android application prototype that use the built-in GPS sensor to record its current geolocation and timestamp.

The Data Collection stage involves the submission of data from multiple mobile devices to the backend server where they are stored based on timestamp. During the Data Processing stage, our current implementation look at the overall snapshot at each timestamp. For the UI evaluation purpose of the next stage, we try two settings: (i) the server divides the area into equal square blocks and computes the congestion level, based on the number of users in each, then express congestion in different colors (ii) the server also divides the area into blocks, finds the centroid of each based on locations reported, express the congestion levels as circle radius size.

In the Data Presentation stage, mobile phones poll the most up-to-date pedestrian traffic map from the distribution server and display this on screen for users. Here they have the option to view traffic conditions in two different UI layouts mentioned above. We surveyed users feedbacks on these settings in our next section.

D. Leveraging the Cloud

We have relatively light computational models in our Data Processing stage, which our single server could handle well. However, for improved accuracy and better traffic estimation results, other more computationally intensive traffic models can be used (such as the iterative machine learning algorithms). Additionally, high volume of participating users can result in hundred thousand data points for each snapshot. This potential increase in CPU and memory demand will require much more resource than a single server machine. We allow our system framework to export the data processing stage to a private cloud or public cloud clusters, then receive and store the computed results. This integration of cloud framework is crucial for the application to scale up and to improve its usefulness.

VI. EXPERIMENTS & EVALUATIONS

A. Experimental Setup & Surveys

To better understand the nature of pedestrian data two trajectories were recorded when leaving and entering Mall of America Stadium during a Vikings football game. Since in our initial experiment, we obviously weren’t able to get other users in the stadium to use our application and to record their walking traces. Therefore, along with this real world data recording, we have written a synthetic workload generator to evaluate our Data Collection, Processing and Presentation stages.

B. Results & Evaluations

One thing of note about the recorded data points is that there are large gaps in the data; these occurred due to the ineffectiveness of GPS indoors. A speed-time plot while departing the stadium was created using the speed recordings determined by the device, as shown in Fig 2:

![Speed vs. Time plot of a pedestrian exiting the stadium.]

While there is a great deal of variability in speed one can observe a generally higher speed after approximately 450 seconds; this time corresponds with a location just
outside the stadium, i.e. it is when travel was no longer indoors and limited by hallways. Another calculation was done to estimate the speed from the times and locations of the data points where the speed was the distance between two points divided by the time difference between them. This plot, along with the speed measured by the device, is shown in Figure 3:

![Figure 3: Speed vs. Time plot of a pedestrian exiting a football stadium, comparing measured speed and speed calculated via distance traveled over time.](image)

While the two plots are generally fairly similar and demonstrate similar trends there are a number of unexpectedly high points which are additionally unreasonable for pedestrian speeds. This helps demonstrate the issue of GPS inaccuracies for pedestrians which can lead to unrealistic results after processing.

For the UI and Data Presentation evaluation, our survey presents to a sample of 57 users the 2 pedestrian traffic conditions maps, we then note their preferences and reasons: Figure 4: Traffic Congestion Representation

![Figure 5.6: Survey results on traffic map representation](image)

The results show that the majority prefer the square blocks representation of the traffic congestion, citing reasons that it is more fine-grained and give them a better sense of which area is more crowded. Whereas the other group prefer seeing the concentration of each area and potential spread of those crowds.

VII. FUTURE WORKS

There are many things we would like to improve, such as implementing a better traffic processing and estimation models for improved results. A potential direction is using a modified EM algorithm to infer blocks congestion. This will also lead to the opportunity to export and parallelize the algorithm on a cloud framework such as Spark. Other important things to consider include ensuring data privacy, and filtering data noise for improved accuracy. Indoors location recording is also a major challenge. Getting the application out there, collecting more users feedbacks and pedestrian traces would be crucial for more detailed analysis.

VIII. CONCLUSION

Overall, our project allows mobile users to submit their pedestrian movement data, receive real-time updates of current traffic conditions. We look at the requirements and distinction between offline vs. online system for a crowdsourcing framework. Our implementation includes a prototype and backend server. We looked at the possible integration of a cloud framework, parallelizing the traffic processing, reducing the computation time, as well as effective presentation of traffic congestion based on a survey from users.
REFERENCES & CODIFIED/TACIT SOURCES


