Applications of Geospatial Data in Digital Communication

William Svoboda
Advisor: Michael Freedman
Department of Computer Science, Princeton University

April 2022

Abstract

Visual learning presents an opportunity to more effectively teach computer science fundamentals. However, existing solutions are unable to both facilitate user interaction and focus on implementation. This paper describes an assignment and programming framework for teaching graph traversal algorithms. Visualization tools are contextualized as a way to improve the learning experience, and an evaluation of the project with real students is discussed.

1 Introduction

With roughly 4.5 billion users worldwide [1], the scale of social media is testament to the importance of digital communication today. At the same time, humans are social animals. It is here in the physical world that we spend most of our time and have our closest interactions with each other. While 99% of users access social media through mobile devices [1], our interaction with these services is often still independent of the physical space around us.

Previous work [2] also suggests that digital interaction does not necessarily correspond to an improved social experience. This is especially interesting in light of the ongoing COVID-19 pandemic and the move towards remote work. Research [3, 4] points to a reliance on digital communication and social media as increasing social isolation.

There is much potential, however, in exploring the intersection between digital communication and physical space. Facebook, Twitter, and Instagram—among the largest social media platforms [1]—now incorporate geospatial data into their products. Other platforms, like Yik Yak, even place location at the core of the user experience [5].

The goal of this paper is to explore how geospatial data affects our perception of digital communication. My hypothesis was that tying interaction to physical proximity would increase user engagement. To this end, I present a novel communication app that gates the sending and receiving of messages to the user's location.

2 Background

2.1 Geospatial Data

Geospatial data is data that is related in some way to a specific geographic position. Such data typically combines location information in the form of coordinates, attribute information about the object or event in question, and temporal information connected to the time this data existed [6]. In this way, geospatial data is able to connect digital information to a very real sense of time and place.

One of the principal uses of geospatial data is in geospatial analysis, empowered through geographic information systems (GIS) and related tools. Geospatial analysis can improve data visualization by providing additional context to traditional data analysis. Of particular relevance to this topic is the flexibility of geospatial data. While originally used in, geology, epidemiology, and the life sciences, geospatial data can be applied to areas as diverse as defense and social science [6].

2.2 Proximity Principle

In social psychology, the proximity principal relates the tendency to form social relationships with the physical distance between people. The phenomenon was first observed by Newcomb [7] and also explored by Festinger *et al.* [8]. Marmaros and Sacerdote [9] surveyed students and recent graduates at Dartmouth College, and found that physical proximity was more influential than any other factor in determining the level of social interaction between people.

3 Related Work

3.1 Geocaching

Geocaching is one of the most popular activities that centers around the use of geospatial data. Participants search for hidden caches by using GPS, attempting to be the first to reach a given cache's location. The coordinates for the cache, as well as clues regarding its surroundings, are posted online by organizers beforehand [10] and can be freely discussed. A similar activity, called waymarking, exists that actually forgoes an actual physical object. The goal instead is to reach a "virtual cache" that represents an interesting trail, vista, or other location of interest [10].

3.2 Social Media

As previously discussed, a number of existing social media platforms already use geospatial data. The extent and purpose of geospatial data varies heavily between different services.

3.2.1 Instagram

Instagram is a photo and video sharing platform that was acquired by Facebook, Inc. (now Meta) in 2012. Like Facebook, Instagram uses geospatial data in the form of geotagging, where geographic information is attached as metadata. Users who click or search for a location can view a feed of other content that was posted nearby. In other words, geospatial data is used to categorize and offer additional context to content.

3.2.2 Twitter

Similarly to Instagram, Twitter supports geotagging. However, this metadata is limited to images that users attach to Twitter messages, or "Tweets." Twitter is also unique in that Tweets are limited in length and are by default text-only [11].

3.2.3 Snapchat

Snapchat is an instant messaging service focusing on "Snaps." Snaps are photo, video, or text messages that by default last only 24 hours. Snapchat supports geotagging for both photo and video messages, and—unique to the platform—also allows messages to be sent to the app's "Snap Map" service. Snap Map, developed by Mapbox, shows a real-time heatmap of Snaps overlaid on the world map [12]. By tapping on part of the map, users are shown a feed of Snaps posted at that location.

3.2.4 Yik Yak

Of the platforms that have already been mentioned, Yik Yak is the only service that actually controls access to content using geospatial data. Yik Yak is a messaging app where individual posts, called "Yaks," are submitted and viewed within a 5-mile radius [5]. This gates user interaction to the immediate community surrounding them.

4 Approach

The design of a product straddles two distinct areas. The first, and most important, is the intended user experience. In order to evaluate my hypothesis, it was necessary to determine what features might be able to produce the desired user behavior. A large part of this process involved deciding how my project would both draw from and differentiate itself from existing platforms. Ultimately, I envisioned an application with the following qualities:

1. A main screen showing the user's current location overlaid on a map of the surrounding area

2. The ability to leave messages tied to one's location, viewable by other users and disappearing after 24 hours

3. The restriction that a message can only be viewed when near the coordinates where it was posted

I drew direct inspiration from Snapchat and Yik Yak in several regards. The idea of a main map screen, similar to Snap Maps, was intended to center the user experience to the world around them. Likewise, the ephemeral nature of messages mimics how individual Snaps last for only a day. The intended effect of this aspect was to keep any message encountered in the world temporally relevant. Because messages are tagged to a physical location they are also spatially relevant in the way Yik Yak messages are.

The defining characteristic of my app, however, is that access to messages is directly gated to physical proximity. While Snap Maps allows users to view Snaps anywhere on the map, and Yik Yak reveals any message sent within 5 miles, my app requires users to be almost within sight of where a message was posted in order to interact with it.

My goal was that this requirement would better connect users with the messages they found by placing them in the same geospatial context.

The second area of design relates to the actual implementation of the product. Given the dominance of mobile devices in social media [1], I decided on an iPhone app as primary interface. This had the added benefit of access to MapKit, Apple's framework for geographic visualization. The actual application follows a distributed client-server model. The iPhone app, acting as the client, handles direct user interaction and communication with the server. In turn, the server handles user content, authentication, geospatial queries, and other centralized functionality.

5 Implementation

The client and server applications were functionally structured as two separate projects, each version controlled with git and hosted online with GitHub.

5.1 Client

Following previous discussion, the choice of a mobile app still left the question of the exact implementation. Because I would be using MapKit, a native iOS application was a natural fit. The app was coded in Swift, a programming language developed by Apple explicitly for native application development on Apple hardware. This was paired with SwiftUI, a new framework also developed by Apple for building user interfaces. Swift and SwiftUI have, respectively, the benefit of interoperability with Objective-C, Apple's earlier programming language, and existing Apple interface frameworks like UIKit.

SwiftUI is a declarative framework, meaning that the user interface is simply described by the developer. This description is then translated by SwiftUI's layout engine at compile-time [13]. As a result, most of the presentation logic is actually handled by the framework itself.

5.1.1 Interface Design

Before any code could be written, however, it was necessary to determine the general design of the user interface and the desired user flow. I wireframed the user interface using Excalidraw, a free and open-source tool for diagram sketches. Each screen in the wireframe was then ultimately translated into a SwiftUI view.

5.1.2 Application Architecture

The use of SwiftUI naturally separates layout and appearance from the actual business logic of an app. From this, I chose the Model-View-ViewModel (MVVM) pattern as the underlying architecture. MVVM decouples presentation logic (*views*) from business logic (*models*) by using intermediate objects called *view models*. The state of each view is bound to the associated view model, which facilitates interaction with models and user input [14, 15].

Each part of the final user interface is composed of one or more SwiftUI views, with view models being set as necessary to handle state management and any dynamic behavior. In any application, it is likely certain logic needs to be shared throughout the overall system. My app needed to access global settings, authentication state, and location services at multiple points. To solve this problem, I wrote different *managers* that represented a shared service layer. When required, these managers were injected into a view model to access the appropriate services.

5.2 Server

Compared to the client app, the server required drastically fewer lines of code. However, the difficulty was in composing the separate technologies that made up the server's basic functionality. The core API is handled by FastAPI, a high-performance web framework written in Python. I chose Redis, an in-memory datastore, as the database for its ease-of-use and included geospatial capabilities.

These core services are packaged together as one Docker application. Docker is used to containerize the server, which isolates the app from its environment. By using Docker, I could develop, test, and deploy the server on different machines while ensuring that each build would behave the same.

5.3 Deployment

In order to distribute the client app to users, I relied on Apple's TestFlight service. TestFlight allows developers to beta-test their iOS applications and receive feedback from users. The benefit of this approach was that I did not have to go through the full App Store approval process in order to deploy my app.

The server was deployed on a DigitalOcean Droplet. Each Droplet is a Linux virtual machine (VM) that runs on virtualized hardware. After reserving an instance, I could access the Droplet remotely using SSH and install all required code and tools using git and the system package manager.

6 Evaluation

7 Conclusions and Future Work

References

[1] The University of Maine, *Social media statistics details*, Undiscovered Maine, Sep. 2021. [Online]. Available: https://umaine.edu/undiscoveredmaine/small-business/resources/marketing-for-small-business/social-media-tools/social-media-statistics-details/.

[2] N. Ducheneaut, N. Yee, E. Nickell, and R. J. Moore, "'Alone Together?': Exploring the social dynamics of massively multiplayer online games," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '06, Montréal, Québec, Canada: Association for Computing Machinery, 2006, pp. 407–416, ISBN: 1595933727. DOI: 10.1145/1124772.1124834. [Online]. Available: https://doi.org/10.1145/1124772.1124834.

[3] T. Bonsaksen *et al.*, "Loneliness and its association with social media use during the covid-19 outbreak," *Social Media + Society*, vol. 7, no. 3, p. 20 563 051 211 033 821, 2021. DOI: 10.1177/20563051211033821. eprint: https://doi.org/10.1177/20563051211033821. [Online]. Available: https://doi.org/10.1177/20563051211033821.

[4] S. Mann and L. Holdsworth, "The psychological impact of teleworking: Stress, emotions and health," *New Technology, Work and Employment*, vol. 18, pp. 196–211, Oct. 2003. DOI: 10.1111/1468-005X.00121.

[5] Yik Yak, Inc., *Legal*, archived from the original, Jul. 2015. [Online]. Available: https://web.archive.org/web/20150714161653/http:/www.yikyakapp.com/legal/.

[6] IBM, *What is geospatial data?* IBM. [Online]. Available: https://www.ibm.com/topics/geospatial-data.

[7] T. M. Newcomb, "Some varieties of interpersonal attraction.," in ser. Festschrift for Gardner Murphy. Oxford, England: Harper, 1960, pp. 171–182.

[8] L. Festinger, S. Schachter, and K. Back, *Social pressures in informal groups; a study of human factors in housing.* Oxford, England: Harper, 1950, pp. x, 240–x, 240.

[9] D. Marmaros and B. Sacerdote, "How Do Friendships Form?*," *The Quarterly Journal of Economics*, vol. 121, no. 1, pp. 79–119, Feb. 2006, ISSN: 0033-5533. DOI: 10.1093/qje/121.1.79. eprint: https://academic.oup.com/qje/article-pdf/121/1/79/5230680/121-1-79.pdf. [Online]. Available: https://doi.org/10.1093/qje/121.1.79.

[10] National Geographic Society, *Geocaching*, Resource Library, Jan. 2011. [Online]. Available: https://www.nationalgeographic.org/encyclopedia/geocaching/.

[11] Twitter, Inc., *New user faq*, 2022. [Online]. Available: https://help.twitter.com/en/resources/new-user-faq.

[12] B. Tadesse, *Mapbox helps power snap map*, Mapbox Blog, Jul. 2017. [Online]. Available: https://blog.mapbox.com/mapbox-helps-power-snap-map-4ced4fb3176a.

[13] Apple, Inc., *Swiftui overview*, Apple Developer, 2022. [Online]. Available: https://developer.apple.com/xcode/swiftui/.

[14] D. Britch, N. Schonning, C. Dunn, and J. Osborne, *The model-view-viewmodel pattern*, Microsoft Docs, Aug. 2021. [Online]. Available: https://docs.microsoft.com/en-us/xamarin/xamarin-forms/enterprise-application-patterns/mvvm.

[15] P. Hudson, *What is mvvm?* Hacking with Swift, May 2019. [Online]. Available: https://www.hackingwithswift.com/example-code/language/what-is-mvvm.