PATHSIM:

CONTACT TRACING WITH REALISTIC PATH SIMULATION USING CELLPHONE LOG DATA

by

Michael Mark
BSE Industrial & Operations Engineering

A Thesis Presented in Partial Fulfillment
of the Requirements of the Degree
Master of Science in Computer Science and Information Systems

2021
SAGINAW VALLEY STATE UNIVERSITY
MICHIGAN, USA.

SAGINAW VALLEY STATE UNIVERSITY

MAY 10, 2021
Date

We hereby recommend that the thesis prepared under our supervision by

Michael Mark
BSE Industrial & Operations Engineering

entitled

Pathsim:

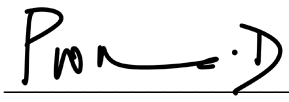
Contact Tracing with Realistic Path Simulation

Using Cellphone Log Data

be accepted in partial fulfillment of the requirements for the Degree of **Master of Science in Computer Science and Information Systems**

AVISHEK MUKHERJEE, Ph. D. ASSISTANT PROFESSOR
Thesis Advisor
DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION SYSTEMS

POONAM DHARAM, Ph.D. ASSOCIATE PROFESSOR
Thesis Reader
DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION SYSTEMS



KHANDAKER ABIR RAHMAN, Ph.D. ASSOCIATE PROFESSOR
Program Coordinator
MS-CSIS

ABSTRACT

Digital contact tracing is being explored in all areas of the world now, even more so recently due to the deadly and widespread COVID-19 pandemic. Currently, new vaccinations are being developed and pushed out rapidly, but cases where infection is spreading are still occurring. Therefore, contact tracing is still a much-needed transmission prevention method in this critical period. Long afterward, contact tracing shall remain an important role in human society to prepare and combat the spread of even deadlier pandemics that may arise in the future.

We propose Pathsim, a highly efficient contact tracing method that utilizes path generation and analysis to determine user proximity based on the existing cellphone LTE log data. Pathsim builds on the foundations laid out in by Mukherjee *et al.* called Vecsim, which proposes a signal model and a method to compare two points and determine their proximity without knowing the actual user locations. While Vecsim was very effective, there were several shortcomings including comparing points receiving signal data from different cell-phone towers as well as locations where the signal was highly discontinuous. Pathsim aims at designing a better system that looks at path information that is automatically logged by a network carrier to build a more robust system, capable of real-world use. In addition, Pathsim preserves user privacy and requires very little infrastructure cost to implement. Our results show that Pathsim exhibits a 75% improvement over Vecsim, when it comes to identifying locations that are connected to different base stations and also improves the overall distance estimation accuracy over its predecessor.

APPROVAL FOR SCHOLARLY DISSEMINATION

The author grants to the Melvin J. Zahnow Library at Saginaw Valley State University the right to reproduce, by appropriate methods, upon request, any or all portions of this Thesis. It is understood that "proper request" consists of the agreement, on the part of the requesting party, that said reproduction is for his/her personal use and that subsequent reproduction will not occur without written approval of the author of this Thesis. Further, any portions of the Thesis used in books, papers, and other works must be appropriately referenced to this Thesis.

Finally, the author of this Thesis reserves the right to publish freely, in the literature, at any time, any or all portions of this Thesis.

Author 

Date May 10, 2021

TABLE OF CONTENTS

ABSTRACT iii

APPROVAL FOR SCHOLARLY DISSEMINATION iv

LIST OF FIGURES vii

LIST OF ABBREVIATIONS ix

ACKNOWLEDGMENTS x

CHAPTER 1 INTRODUCTION 1

CHAPTER 2 INTRODUCTION 4

2.1 Contact Tracing by Governments 4

2.2 Contact Tracing by Bluetooth 6

2.3 Other Contact Tracing Methods 8

CHAPTER 3 Background on LTE networks 9

CHAPTER 4 Prior Work with Vecsim 14

4.1 Distance Estimation 14

4.2 Coping with Signal Discontinuity 16

4.3 Vecsim Accuracy 18

4.4 Limitations of Vecsim 18

CHAPTER 5 PATHSIM 20

5.1 High Level Idea 20

5.2 Distance Estimation in Pathsim 21

5.3 Data Generation 23

5.4 Realistic Path Simulation 25

5.5 Path Generation Techniques and Development 27
5.5.1 Weighted Buildings Method 27
5.5.2 Shortest Path using Binary Matrices 29
5.5.3 The A* Method 31
CHAPTER 6 EVALUATION 33
6.1 Path Generation Tests 33
6.2 Pathsim Test Results 34
6.2.1 Paths with common source 34
6.2.2 Paths with common destination 35
6.2.3 Paths that meet in the middle 35
6.3 Final Pathsim Test Results and Comparison with Vecsim 37
6.3.1 Proximity Path Generation 37
6.3.2 Effectiveness of using adjacency matrix 38
6.3.3 Improvement over Vecsim 40
CHAPTER 7 DISCUSSION, CONCLUSIONS, FUTURE WORKS 44
7.1 Discussion 44
7.2 Limitations 44
7.3 Conclusion 45
7.4 Future Work 46
REFERENCES 47

LIST OF FIGURES

Figure 3-1: A typical example of an MRO report vector. 10

Figure 3-2: Distance estimation between 2 UEs. 11

Figure 4-1: An illustration of the initial estimate based on *Tadv*. 15

Figure 4-2: Distance Estimation of close pairs. 18

Figure 5-1: Pathsim neighbor comparison matrix between two paths. 21

Figure 5-2: 3D rendering of a simulated city block. 23

Figure 5-3: 2D map of detailed building information. 24

Figure 5-4: The birds eye view of the generated signal field. 25

Figure 5-5: Focus area of our study. 26

Figure 5-6: A close-up visual of the focus area. 26

Figure 5-7: Assigning weights to roadways. 28

Figure 5-8: Example paths generated from Weighted building matrices. 29

Figure 5-9: Example paths created with *bwdistgeodesic*. 30

Figure 5-10: Invalid path using bwdistgeodesic. 30

Figure 5-11: Example paths generated using A* Search Algorithm 32

Figure 6-1: Example path pairs with common source. 35

Figure 6-2: Example path pair with an intersection point. 36

Figure 6-3: Choosing random starting/ending locations near each other. 37

Figure 6-4: Paths with similar routes. 38

Figure 6-5: Location pairs with no similarity scores. 41

Figure 6-6: Comparison of estimation accuracy between Vecsim and Pathsim (without DPD).....42

Figure 6-7: Comparison of estimation accuracy between Vecsim and Pathsim (with DPD).....43

LIST OF ABBREVIATIONS

MRO	Mobility Robustness Optimization
GPS	Global Positioning System
CCTV	Closed-circuit television
CDC	Centers for Disease Control and Prevention
eNB	Evolved Node Bs
UE	User Equipment
Tadv	Timing Advance
AoA	Angle of Arrival
RSRP	Reference Signal Received Power
PCI	Physical Cell ID
nPCI	Neighboring PCI identifier
nRSRP	RSRP from Neighboring PCI
LOS	Line of Sight
DPD	Discontinuity Pairs Database

ACKNOWLEDGMENTS

For the development of this thesis, it was not possible without the help from my teammates, Anthony Galea, Beata A. Hejno, and Morgan B. Simmons. Their skills and knowledge helped bring this thesis to fruition. A special thank you also to Professor Avishek Mukherjee who guided us every step of the way.

CHAPTER 1

INTRODUCTION

The COVID-19 pandemic has exacerbated the need for a robust contact tracing method that works better than current strategies and is secure to protect a user's private information [1]. Current systems, especially those deployed by some governments across the world, do very little to protect user privacy, and often requires users to interact with an application on their mobile devices to start the contact tracing process. With Pathsim, we eliminate the need for user intervention, by only looking at the LTE log data that is automatically collected by cellphone network carriers for every connected device on its network. To be more specific, Pathsim relies on data collected in the Mobility Robustness Optimization (MRO) file [2] that is used by network carriers for network management purposes. The MRO file logs connected device data every 5.12 seconds and the data is a high dimensional signal vector containing parameters such as the length of the signal propagation path, the angle of arrival and the signal strength between the device and the surrounding cell phone towers. By using this information, we developed Pathsim, a novel contact tracing approach which:

- Runs on the carrier side and does not require user interaction.
- Protects user privacy by not requiring or recording the user location.
- Estimates the proximity between two users with greater accuracy than most of the existing contact tracing solutions.

The main challenge with proximity estimation is that in urban areas, the wireless propagation model is extremely complicated. The signal undergoes several physical phenomena such as reflection, refraction, diffraction off buildings and other tall structures, which can introduce irregularities and discontinuities in the signal field. This means even if two locations are close to each other, the signal field may look very different. Pathsim addresses these challenges by designing a multipronged approach to tackle signal discontinuity.

First, Pathsim uses the same fundamental signal model introduced by Mukherjee *et al.* called Vecsim [2]. Vecsim generates a similarity score as an estimate of the distance between two points by looking at the signal vectors between these two points. It also tackles the problem of signal discontinuity by performing lookups on a signal discontinuity database to identify signal pairs that are discontinuous. The primary weakness with Vecsim is that if two points are relatively close but are connected to different cellphone towers, Vecsim cannot accurately estimate their distance. The reason for this is because their signal parameters from the two different towers may differ vastly and might not even have any correlation since the signal propagation paths are completely different. Pathsim was designed to primarily tackle this problem of signal discontinuity at neighboring points either due to different network parameters or due to irregularities in the signal itself. This is especially important as newer methods and technologies are being developed to adjust for cellphones [3] near the boundaries of multiple cellphone towers' ranges.

Instead of only looking at the two neighboring points, Pathsim takes into account the walking paths of two users, which can be easily extracted from the MRO file, and proposes a new distance estimation technique that also considers the neighboring points on the two paths to provide a better estimate the distance between two points. This method

overcomes the problem of two nearby locations being connected to different cellphone towers, since it is highly likely that some of the neighboring points on either path will be connected to the same cellphone tower. In addition, since multiple points are being compared, we expect Pathsim to also improve the accuracy of the distance estimation that was proposed in Vecsim by combining the distance estimate at multiple points to provide a better representation of the signal field.

We evaluated Pathsim by generating a simulated map area containing multiple cellphone towers at different locations. Since Pathsim relies on realistic walking paths, we also designed a novel path generation solution that simulates paths very close to real world walking paths taken by users.

Finally, it is important to note that Pathsim relies on proximity detection and not localization which is a much harder problem to solve by solely using the signal data in the log file. Specifically, it determines how close two users are by looking at the changes in the received signal at the two user locations, rather than localizing the users through triangulation or some other techniques, which may infringe on user privacy.

The rest of this thesis is organized as follows. Chapter 2 discusses the related work. Chapter 3 discusses the preliminaries and background of LTE networks. Chapter 4 explains fundamentals of Vecsim and the discontinuity pairs database generation. Chapter 5 introduces Pathsim and its distance estimation methods. Chapter 6 shows the results of our evaluation of Pathsim. Chapter 7 concludes the thesis.

CHAPTER 2

INTRODUCTION

Contact tracing is becoming increasingly adopted by governments, especially in Asia. The demand of finding and tracking infected individuals is high during these times of stress, but the current contact tracing methods adopted by several governments may be questionable and ineffective. Some methods may not be very accurate or may require users to manually install and activate applications on their devices, which may not always be a realistic expectation, and may even intrude on the users' privacy. In this Chapter, we look at some of the related work in this area and identify some weaknesses in each of these solutions.

2.1 Contact Tracing by Governments

Given the above demand of information on a mass scale, and the requirements for users to cooperate well together, it is easy to see why contact tracing is still in early stages of development and why governments are still slow to accept any official implementation. However, some governments have already adopted various forms of digital contact tracing despite infringing upon user privacy and technical issues. South Korea, China, and Singapore have already seen moderate success [4] with their methods of digital contact tracing.

South Korea uses a combination of technologies, including accessing patient medical records, Global Positioning System (GPS) data, user card transactions (i.e., credit cards), and

closed-circuit television (CCTV) cameras to fully analyze contact tracing. At all times, a user is subject to these tracing methods, almost completely removing their privacy whenever they step outside [5]. Given the high population of some cities in South Korea, such as Seoul, this method is highly necessary, and even more so in China, where population density is at its highest.

China has already made an exception to the virus and will be accepting any form of prevention. This includes extreme measures such as lockdown of major cities, acceptance of any kind of digital contact tracing disregarding user privacy, and mandatory quarantine for any international travelers [6]. In addition, China's contact tracing efforts include mandating the scanning of a QR code with the user's phones, in malls and other public spaces, to create a digital footprint of every person [7]. Clearly, the accuracy of this approach is the same as the size of the public spaces.

Singapore [8] and India both use a combination of automated contact tracing through apps, which record location history and manual contact tracing by conducting phone interviews [9] with exposed individuals. Regardless of a government mandate to download a cellphone app for contact tracing, which only a portion of individuals will do, it is found that even then a smaller chance will arise that any two randomly chosen individuals who cross paths will both have the app – and in Singapore's case only 4% [10].

In the United States, previous contact tracing efforts have mostly been manual, where infected individuals will be contacted by health professionals by phone call, text, or email to interview them and check if they have been in contact with anyone with the disease. Manual contact tracing relies on the users' memories to recreate their paths and visited locations, and then it requires constant communication with trained professionals in the health division to track infected users. To increase the effectiveness of this method, the

infected individual needs to update their information every 24 hours, and this information is monitored by the Centers for Disease Control and Prevention (CDC) [11] [12]. Exposure alerts in the form of text messages will be sent out to individuals who may be at risk, and self-isolation practices are being recommended daily.

Several states are starting to implement their own innovative solutions and turning to digital tools to complement manual contact tracing. For example, in North Dakota [13], the government has developed a Bluetooth-based application (or 'app') to track and detect the users' proximity to other cellphone users. A few states have also implemented similar strategies, and some have announced multistate collaboration policies to better handle the current pandemic [14]. However, without a consistent approach, these efforts have not seen much success yet. Further, while some of these apps do try to preserve user privacy, their data is less effective since they are unable to collect important information regarding the users' names or location data [15].

2.2 Contact Tracing by Bluetooth

With the onset of the recent pandemic, many private enterprises have also developed applications for contact tracing that rely on Bluetooth [16] [17], such that when two phones are close, the Bluetooth modules can discover each other, with varying degrees of accuracy. Cleary the success of these applications not only depend on user adoption of these apps, but also rely on multiple platforms willing to collaborate and share user information with each other. Recent studies have also shown that these applications are susceptible to a range of network attacks [18], which reduce user confidence in these methods.

The above contact tracing is called Bluetooth-based contact tracing and requires users to download, install an app on their devices, and finally, give the app the necessary permissions to use the Bluetooth interface. These methods are decentralized and make use

of the networking capabilities of these devices to report their user location. However, its effectiveness is the key issue in why many governments have not already adapted to it yet. The benefits of Bluetooth-based contact tracing currently do not outweigh the costs and risks, especially when relying on user intervention and accurate self-knowledge of a user's health, which may not always be a realistic option.

A study and questionnaire were completed in a contact tracing thesis that showed that although the mass population of Jordan accepted contact tracing, there were many concerns of privacy issues. The method that A. Abuhammad *et al.* [19] proposed an app that users would download, that can pull the users' data regarding their location and the paths they have taken in the past. The app was not created for the purpose of their thesis, but the app could also send information regarding the age, gender, and other pertinent information regarding the user.

It should also be noted that the implications of using these applications can vary based on population density. Users in certain sparsely populated regions may not require robust contact tracing solutions compared to others. For example, a recent study [20] shows that the average number of people that can test positive for COVID-19 can be mathematically quantified by the average number of people a person infects (R_0). The study found that when R_0 is 3.5, more than 90% of contacts need to be traced and should practice self-isolation techniques to reduce the spread of COVID-19. However, when R_0 falls lower to 2.5 or 1.5, then contact tracing coverage would only need to be observed at 70% and 50%, respectively, to be effective at controlling the spread of the disease [21]. It is recommended to utilize a wide range of various testing, isolation, tracing, and social distancing strategies to effectively combat the disease.

2.3 Other Contact Tracing Methods

While government mandated contact tracing and applications using Bluetooth form the bulk of current contact tracing efforts, there are other methods that have been relatively successful to identify exposed individuals. Some of these methods of contact tracing include the following:

- *Social media contact tracing* [22], which uses user data social networks like check-ins, geo-tagged photos to identify peers who may have been in close contact with the infected user. Privacy issues aside these methods require the user and all their peers to be active on social networks, which may not be a reasonable assumption.
- *Contact tracing using Wi-Fi MAC addresses* [23], which uses the wireless MAC address for identification, useful as an indoor option. This method is complementary to Pathsim which uses cellphone network data to perform contact tracing.
- *Contact tracing via wearable devices* [24], which require users to have a wearable device on them and is primarily used in factories and workplaces where there is a high risk of user infection to track proximity to other devices in the same area. This method uses proprietary technologies to identify users and report to some central management system.

Unlike the other contact tracing methods mentioned in this Chapter, the proposed contact tracing method, Pathsim, does not require users to wear any sort of manual device, is able to cover large outdoor areas, and does not require user invention, making it have a clear advantage over the above methods. Pathsim can serve as a strong alternative or addition and has the potential to offer much higher accuracy while still preserving user preservation.

CHAPTER 3

BACKGROUND ON LTE NETWORKS

LTE networks are widely adopted across many countries today, and this already established system is a great starting point to develop a global, highly targeted contact tracing strategy. Before beginning however, we must cover the background on the fundamentals and terminology of cellphone LTE networks.

An LTE network consists of many Evolved Node Bs (eNB), where each eNB represent the basic element in any LTE access network and may consist of one or multiple towers. Each tower typically mounts three base stations, each associated with an identification number called Physical Cell ID (PCI). A base station may have one or multiple antennas, although this data is from base stations with one antenna. The antennas are directional and the relative strength of signal at a point in space is determined by the radiation pattern as a function of the azimuth and tilt angle, which can be learned from the antenna manufacturer and must be considered by our algorithm. Typically, each antenna faces a sector around 120 degrees, and may be tilted by a few degrees for better coverage. These towers typically provide network coverage for User Equipment (UEs) which may be cellphones and other user devices that are subscribed to the carrier's network. The wireless signal may reach the UE directly, but more often than not will reflect off a surface, penetrate a surface, diffract at the edge of a building, or scatter at small objects [25]. After hitting a surface, a fraction of the wireless energy may be absorbed, reflected, or penetrate the surface

and reach the next layer of the material. The exact fraction of energy in each category is a function of the incident angle, the surface material, and the signal polarization, according to the Fresnel's equations [26], and can be highly complicated, especially for mixed materials [27].

The eNB periodically generates log files for all its base stations. The log files include the MRO file and the MRE file. The MRO file logs the data for all connected UEs. An entry for a UE is generated every 5.12 seconds, which is a high dimensional vector, called the signal vector.

UE_ID	Time	PCI	RSRP	Tadv	nPCI	nRSRP
33617902	43310080	84	48	17	7	30	5	8	176	41	20	175	46	10						
33656218	43310080	84	51	25	1	40	10	1	128	17	11	159	13	5	176	13	8			
33657577	43310080	84	36	18	17	29	7	0	65	21	13	130	28	22	166	21	17	167	20	
33691380	43310080	84	49	25	4	30	22	0	167	17	17	176	12	9	86	32	5	175	25	
167785169	43310080	84	55	25	17	44	23	-5	24	43	15	25	40	9	176	38	9	26	40	
167793806	43310080	84	56	20	7	40	28	8	161	38	16	176	45	19	175	50	11			

Figure 3-1: A typical example of an MRO report vector.

Figure 3-1 shows some examples of the information collected in the MRO file. Each vector contains several attributes. The UE_ID and the PCI identifier have already been explained above. The other fields include:

- *Timing Advance* (*Tadv*): This represents the upper limit of the length of the signal from the base station to the user. This value is proportioned between 0-10 with a granularity proportion of 78 meters.
- *Angle of Arrival* (*AoA*): Also denoted by θ, this is the angle from which a signal is received from the top of the tower, which is a real value between [0, 2π]. This

can be visualized as the angle of the signal from the cellphone observed at the base station.

- *Reference Signal Received Power* (*RSRP*): This value is a measure of the signal strength between the UE and the connected base station.
- *nPCI* and *nRSRP*: This represents the PCI identification, and the signal strength of the neighboring towers close to the UE. While any UE will only be connected to one base station, it can still receive packets from neighboring towers. This mechanism is built into LTE, such that if the signal strength of a neighboring base stations becomes stronger than the current *RSRP* value, the UE can switch connections to the other PCI, thus optimizing the network coverage. There is only one *RSRP* but potentially multiple *nRSRP* values.

The above information can be visually represented as shown in **Figure 3-2** which shows two users and the wireless signal propagation path from the same base station. The distance estimate between any two points can be visualized as follows.

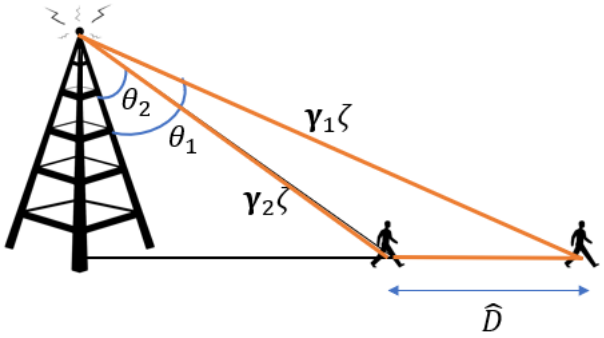


Figure 3-2: Distance estimation between 2 UEs.

Considering a line-of-sight scenario, an initial estimate between two locations can be derived as the following in **Eq. 3-1**:

$$\widehat{D} = [\gamma_1^2 + \gamma_2^2 - 2\gamma_1\gamma_2 \cos(\theta_1 - \theta_2)]^{\frac{1}{2}} \zeta \qquad \textbf{Eq. 3-1}$$

where,

$\widehat{D}$ is the distance between the two users,

γ_1, γ_2 are the *Tadv* values reported by the two users,

θ_1, θ_2 are the Angle of Arrival values reported by the two users, and

ζ is 78 meters, the granularity proportion of the MRO file given by the base station.

While this is a very simple and accurate estimate, the issue with this initial estimate is that the above formula assumes that the users are in Line of Sight (LOS) with the tower (**Figure 3-2**). The distance estimate will not be accurate in cases where the signal reaches the UE using a reflected or diffracted path. Still, it serves as a good starting point for Pathsim. As established earlier, the signal model for non-LOS paths grows exponentially in complexity, especially in urban areas. It is impractical to design a signal propagation model that will work well in the general case. Instead, we decided to implement a data-based approach that analyzes the log data to learn the signal field. Given the high-dimensional and massive log data, the structure of the signal field, although complicated, may be eventually revealed, because it is still governed by the laws of physics and number of unknowns in the system becomes relatively small with the incoming continuous stream of log data. However, it is still highly challenging to use the log data for signal field analysis, mainly because of the following reasons:

- *The log data does not contain the location information about the UE*, as the UE is not required to measure and report its location. In other words, the log data is unlabeled. Therefore, the field cannot be learned by simply mapping each log record to a physical location. Existing triangulation methods for phone

localization may have large errors and cannot be used for fine-grained localization.

- *The detailed environment information can be difficult to cope with*. Signal propagation is highly related to the environment. While coarse-level geographic data about the building location and dimension can be obtained with some effort, to achieve higher accuracy than existing ray tracing methods, the detailed building surface layout and material should also be considered and handled in some manner by the algorithm. However, information like the locations of the glass windows and concrete walls, as well as the building materials which may consist of multiple layers, can be difficult or even impossible to obtain. Even if the information is available, it might be overly complicated to consider all such details, e.g., every glass window, in the algorithm. Higher accuracy can likely be achieved with more data; however, the computation complexity also grows with the data size.

CHAPTER 4

PRIOR WORK WITH VECSIM

Vecsim stands for Vector Similarity and was built as an initial solution to contact tracing using LTE log data. It uses the same model described above to estimate the distance between two user devices. While Pathsim was designed to be a significantly better solution over Vecsim, it still uses some of the fundamental techniques outlined in Vecsim. In this Chapter, we go over some of the salient features in Vecsim, as well as outline some of its weaknesses.

4.1 Distance Estimation

The core heuristic of Vecsim lies in its simple but accurate distance estimation method between two user devices that are connected to the same base station. The initial distance estimate denoted by $\widehat{D}$ is given by the same equation described below in **Eq. 4-1**.

$$\widehat{D} = [\gamma_1^2 + \gamma_2^2 - 2\gamma_1\gamma_2 \cos(\theta_1 - \theta_2)]^{\frac{1}{2}} \zeta \qquad \textbf{Eq. 4-1}$$

Clearly, this estimate is correct when the signal propagation paths to both locations are in LOS, in which case the distance is simply the length of an edge in a triangle, where the other two edges are of length $\gamma_1\zeta$ and $\gamma_2\zeta$, respectively, with angle $\theta_1 - \theta_2$ in between. In urban areas, where the LOS assumption is not true, the initial estimate may still give a reasonable estimate in certain cases. **Figure 4-1** shows an example in which the signal paths have undergone a reflection, such that the initial estimate is basically the distance between

locations A and B in the figure, which are mirrors of the actual cellphone locations. However, fortunately, the distance between the mirrored locations is exactly the actual distance between the cellphones.

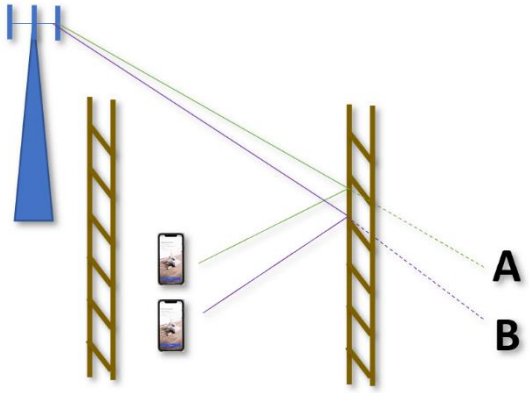


Figure 4-1: An illustration of the initial estimate based on *Tadv*.

While **Eq. 4-1** is valid in rural areas, the initial estimate will likely deviate more from the actual distance in more complicated propagation environments with more reflections. It is however still a good starting point in many cases, because a cellphone connects to a base station likely because there exists a strong path, and strong paths typically do not undergo many reflections. To tackle the signal field in non-LOS environments, Vecsim uses two additional estimates to further improve on the distance estimation:

- *Maximum Likelihood Estimate based on the Initial Estimate*: The initial distance estimation is converted into probabilities of $N(=8)$ discrete distances starting from 25, 50, … up to 200 meters. Distances beyond 200m are treated as 200m for the purposes of contact tracing. The idea is given an actual discrete distance the probability of the discrete initial estimate is found using maximum likelihood principles. For example, suppose the actual discrete distance is 5 and the initial

estimate is also 5, then it is highly likely that this estimate is likely to be correct. On the other hand, if the actual discrete distance is 5 and the initial estimate is 6, a lower probability value is assigned to the score, meaning that it is less likely but still possible.

- *Maximum Likelihood Estimate based on the Unique Base Stations*: The second likelihood estimate is based on the PCI of the unique base stations. A unique base station as the base station which is unique to either of the two MRO records. The idea is that if the two records are far apart, there is a greater chance that a UE will record the *RSRP* of base station that the other UE cannot hear. Thus, given the actual discrete distance, we also determine the probability of the measured *RSRP* value.

The final estimate uses the product of these two likelihood functions to find the best distance estimate from 1 – 8 that represents distances values 25m, 50m, 75m, 100m … 200m. This is defined as the similarity score.

4.2 Coping with Signal Discontinuity

Vecsim uses one additional feature to tackle the discontinuous nature of the signal. It was observed that there can be locations in cities with many buildings, where Vecsim reports a large distance estimate even when the actual locations are quite close. For example, if a person is emerging from a small alley onto a major roadway, the signal shows a large change even though the two locations are actually very close to each other. Vecsim tackles this problem by learning the discontinuity of the signal field using the massive LTE log data available. To be more specific, Vecsim creates a database of Discontinuity Pairs, called the Discontinuity Pairs Database (DPD), which keeps track of such locations. While this may

seem challenging, it is important to note that cellphone data is being collected from millions of people walking in a city every day. By simply logging these walks, it is possible to find most of the discontinuous pairs in an area.

A discontinuity pair is defined follows. If two locations A and B are within 25 meters of each other and their discreate distance estimates is larger than 4 then A and B form a discontinuous pair and is added to the DPD. These can be identified by looking at the log data for a single user who is mobile. If the MRO vectors between two consecutive timestamps on the user's path show a significant difference, then it is likely these are discontinuous pair, since it is improbable that the user covered more than 25 meters within 5 seconds. These can be learned over time using the MRO log data and logged to a database for lookups. The key idea is that when a person is walking and their adjacent points on their path form a discontinuity pair, the MRO records located at A and B will show a large difference but will be only a few timestamps of each other. These pairs are then added to a database that is indexed by the *Tadv* and PCI values to allow for fast lookups. In the Vecsim study, over 90% of DPD pairs were learned after 500,000 paths logged. While this may be a substantial amount of information to keep in the database, Vecsim does account for this by using a compression technique to reduce the size of the DPD by removing similar pairs.

4.3 Vecsim Accuracy

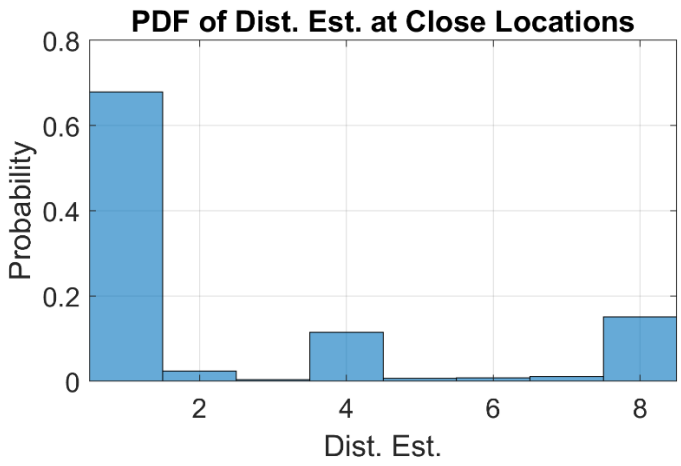


Figure 4-2: Distance Estimation of close pairs.

Vecsim was found to be more than 95% accurate in alerting cellphones within 50 meters of infected individuals after factoring in discontinuity pair checking. **Figure 4-2** shows the probability distribution of distance estimations at close locations. Vecsim also alerts less than 4.5% of cellphones beyond 150 meters. Exploiting the MRO log data, Vecsim is not only able to be universally appealing, but also preserves user privacy beyond what they have already submitted to the cellphone carriers.

4.4 Limitations of Vecsim

While Vecsim is an effective low-cost solution, it suffers from some drawbacks. The biggest drawback of Vecsim is that it cannot estimate distances between pairs of locations that are connected to different base stations. Since Vecsim only relies on the MRO data between two points, the signal data looks very different for locations connected to different towers, as the signal propagation parameters vary significantly for different base stations. While the accuracy of Vecsim is very good, it can be further improved and is a critical factor

in measuring the effectiveness of contact tracing solutions. The random walk simulation explored in Vecsim to learn the discontinuous pairs is not a realistic simulation and may not exactly match the walks in the real world.

Overall, Vecsim provides a good theoretical solution to the problem of contact tracing. It preserves user privacy and requires almost no infrastructure to implement. However, it does suffer from some limitations as outlined above, which makes it difficult to implement it in practice. Pathsim tackles the above limitations of Vecsim and improves its overall accuracy, making it an ideal contact tracing solution in the real world.

CHAPTER 5

PATHSIM

In this Chapter we introduce Pathsim, or short for Path Similarity, our current attempt at estimating the distance at two user locations using the MRO log data collected by cellphone network carriers. Pathsim is built on the theoretical foundations established in Vecsim but attempts to mitigate the drawbacks outlined in the previous Chapter. The core idea behind Pathsim is based on the idea that the path information of two users can used to compute a better distance estimate between two locations. Our work on Pathsim has two major contributions. First, we design a random walk solution that creates realistic walking paths that users may take to bring Pathsim closer to a real-world simulation. Second, we revise our distance estimation, based on the discrete estimates at neighboring points between two paths to achieve greater accuracy.

5.1 High Level Idea

Pathsim is a novel contact tracing method, that aims to overcome the drawbacks of the Vecsim algorithm. The main idea is that Pathsim looks at the user's walking path to calculate the user proximity instead of only looking at the two user locations, as in Vecsim. By looking at neighboring points on a path, Pathsim can better account for irregularities in the signal field and finds a solution to the problem of comparing two neighboring points where the recorded signal field is from different base stations. The rest of this Chapter

discusses our revised distance estimation method and our efforts at simulating real world walking paths.

5.2 Distance Estimation in Pathsim

Pathsim revises the distance estimation method for estimating the proximity between two points. Consider two users (say A and B), walking in close proximity to each other. Also, suppose their paths are denoted as P_A and P_B, where P_A and P_B consist of a list of co-ordinates that represent the locations visited on each path, respectively. If we want to measure the distance estimate $\widehat{D}$ at two locations a and b where $a \in P_A, b \in P_B$, instead of only comparing the MRO vectors at the two user locations a and b, Pathsim also compares the distance estimation between the neighboring points on each path. To be more specific, Pathsim generates an adjacency matrix of similarity scores between K neighboring points from the original points on each path. That is, if a and b represent the i^{th} and j^{th} index in P_A and P_B respectively, Pathsim computes a series of scores between $P_A[i-K, \dots i, \dots i+K]$ and $P_B[j-K, \dots j, \dots j+K]$.

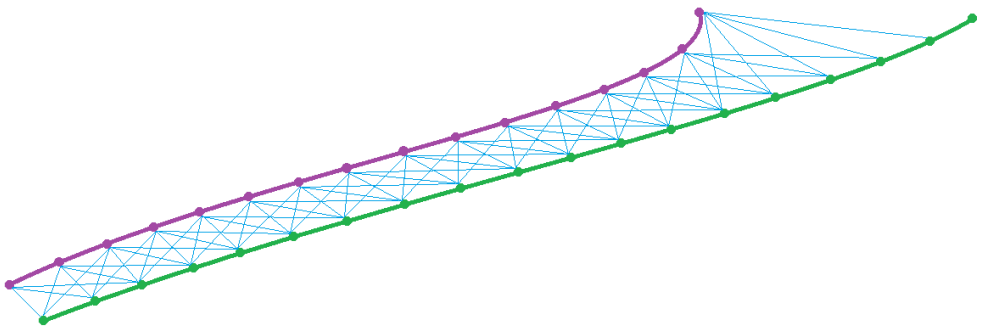


Figure 5-1: Pathsim neighbor comparison matrix between two paths.

A visual representation of the above method can be seen in **Figure 5-1**. In our current evaluations, the value of K is set to 2. We found only comparing 2 neighboring points strikes a good balance between improving our accuracy while not increasing the complexity of Pathsim too much over Vecsim. The result is a 5x5 path adjacency matrix that contains the discrete estimate between neighboring points on each path as shown below.

	a-2	a-1	a	a+1	a+2
b-2	$\widehat{D}_{b-2}^{a-2}$	$\cdots$	$\cdots$	$\cdots$	$\widehat{D}_{b-2}^{a+2}$
b-1	$\vdots$	$\cdots$	$\cdots$	$\cdots$	$\vdots$
b	$\widehat{D}_{b}^{a-2}$	$\cdots$	$\widehat{D}_{b}^{a}$	$\cdots$	$\widehat{D}_{b}^{a+2}$
b+1	$\vdots$	$\cdots$	$\cdots$	$\cdots$	$\vdots$
b+2	$\widehat{D}_{b+2}^{a-2}$	$\cdots$	$\cdots$	$\cdots$	$\widehat{D}_{b+2}^{a+2}$

The above representation of the distance estimation has two distinct advantages. First if the distance estimate at $\widehat{D}_{b}^{a}$ does not exist, an accurate distance estimate can also be learned by looking at the discrete estimation values at other points in the adjacency matrix to fill in the gaps that arise from neighboring points being connected to different base stations. Since these neighboring points are physically close to the original pair of points, it is very likely that some of these points on different paths may still be connected to the same base station and a valid similarity score can be computed for these points. Obviously if the paths are far apart, it is likely that neither of the location pairs will share a common base station, which works out well, since Pathsim will determine that these location pairs to not be in proximity.

The above adjacency matrix of scores can be represented as a probability density function and the final distance estimate is computed by calculating the mode of the

distribution. To be more specific, we look at the most frequently occurring similarity score. For multimodal distributions, we compute the final estimate by taking either a random selection or the average of the most frequently occurring elements in the distribution, which we will compare the results in the Evaluation section of in Chapter 6.

5.3 Data Generation

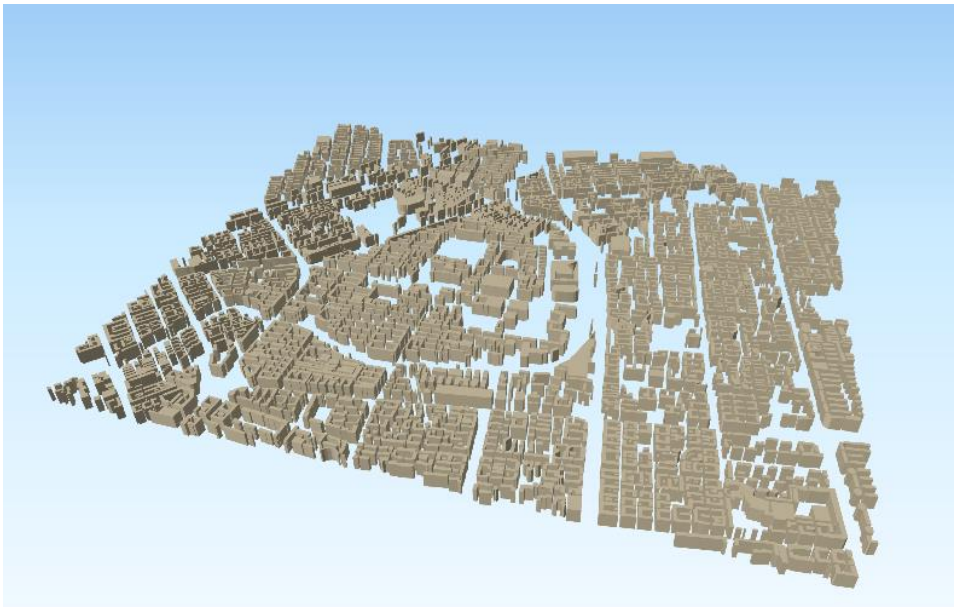


Figure 5-2: 3D rendering of a simulated city block.

We used Wireless Insite [28], a ray-tracing software program, to generate the signal field in an urban area. **Figure 5-2** shows a 3D render of a 4 km by 4 km urban area that we analyze in our study. **Figure 5-3** shows a 2D view of the same model. It can be seen that the simulated map contains rich information of roads and building information that closely resembles a real-world environment.

Figure 5-3: 2D map of detailed building information.

The signal field was generated by simulating a total of 13 towers throughout the city, which contains a total 39 base stations. As the signal decays fast, to limit the computation complexity, for each base station, only the 2 km by 2 km area centered at the base station is simulated. For each base station, the Wireless Insite software is configured to calculate the rays received at locations on 4 m by 4 m grids. The bandwidth is 20 MHz. **Figure 5-4** shows a birds-eye view of the signal field in where the field of different base stations are shown in different colors. Note that the field of a base station may only be partly shown if it has some overlap with another base station plotted afterwards.

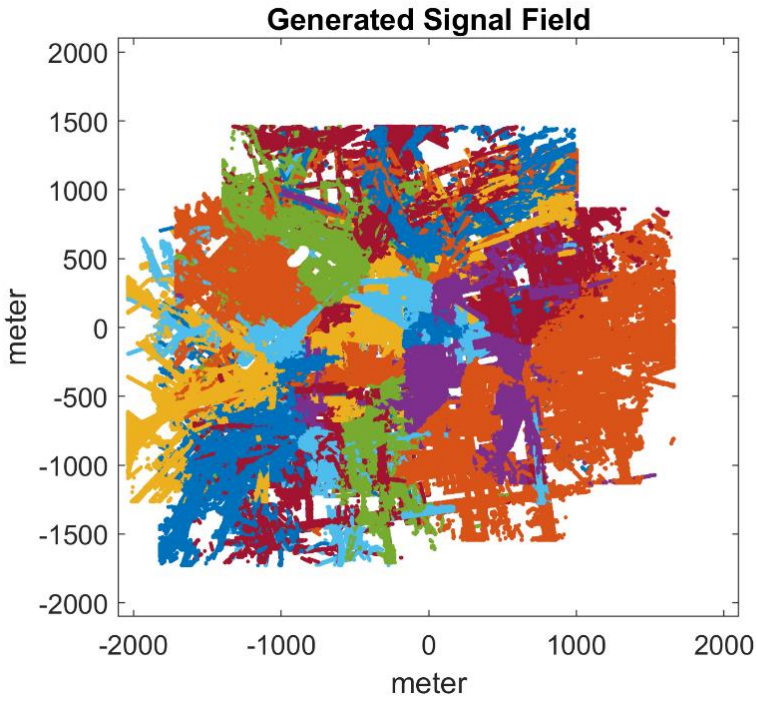


Figure 5-4: The birds eye view of the generated signal field.

Since we do not have the information of the building materials, the building is modeled as concrete blocks in the software. We therefore do not have indoor data and limit this study to outdoor areas only.

5.4 Realistic Path Simulation

In the development of Pathsim, simulating an individual's realistic walking path has been a primary consideration. In this subchapter we outline the different techniques used to simulate walking paths that are similar to patterns found in actual user movement, such as walking closer to buildings, using sidewalks and alleys. By generating these walking paths and analyzing them, Pathsim can be evaluated in a setting close to what can be seen in a real-world environment.

To reduce the complexity, we identified a focus area inside our original map and ran our path generation simulations within this area. As shown in **Figure 5-5**, the 2 km by 2 km area in the center of the map is used as the study area. **Figure 5-6** shows a close-up view of the focus area.

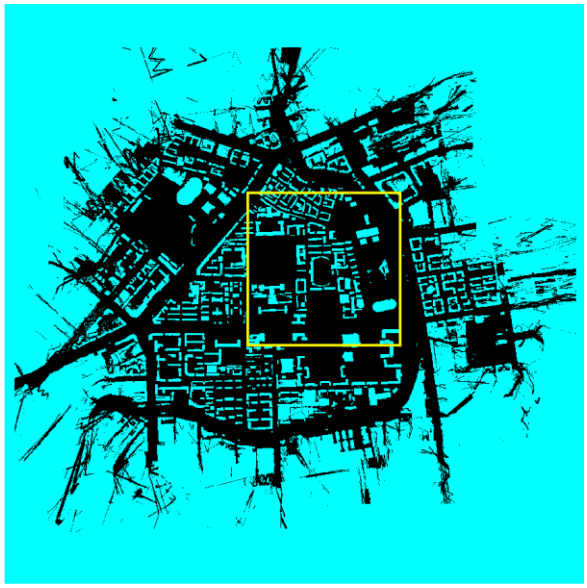


Figure 5-5: Focus area of our study.

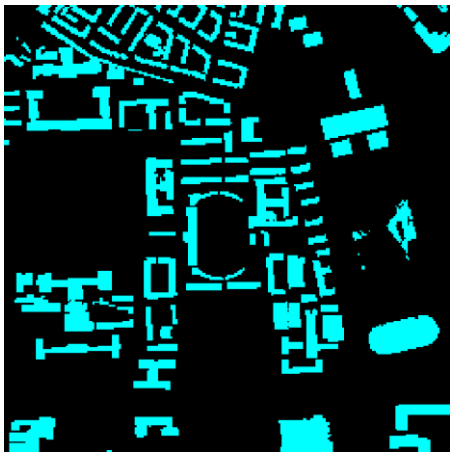


Figure 5-6: A close-up visual of the focus area.

Since the signal field is not generated inside buildings the map above can be represented as a binary matrix that of 1s and 0s. 0s indicate buildings or points that do not contain signal field and users cannot walk through these locations. 1s indicate roads and alleyways where the users can walk through. It is important to note that while our final evaluation focuses on the study area shown above, we tested our path generation algorithms on the entire city area to evaluate its robustness.

5.5 Path Generation Techniques and Development

In this subsection, we discuss the different methods considered for our realistic path simulation. The primary focus was to use techniques that produced realistic walking paths but also keeping the complexity relatively simple such that these paths can be generated within a reasonable amount of time. The methods considered included:

- Weighted Buildings Method (Dijkstra)
- Shortest Path using Binary Matrices
- A* Method

5.5.1 Weighted Buildings Method

Weighted buildings method was our first attempt in generating paths. The idea was to treat the entire map as a connected graph and assigning higher priority weights to road locations right next to buildings. These would simulate sidewalks on the binary map. We then used Dijkstra's method [29], a modified breadth-first search algorithm, to simulate a walking path between a randomly chosen source and destination. By assigning a lower numeric cost to roadways near the edges of buildings, as shown in Figure 9 we enable Dijkstra to choose paths closer to buildings that simulate a user walking on a sidewalk when travelling from one location to another. This also enables us to provide a more challenging

scenario for Pathsim, as the signal at points close to building tend to be more discontinuous than those in open spaces. **Figure 5-7** shows some examples of paths generated using the Weighted Matrix method.

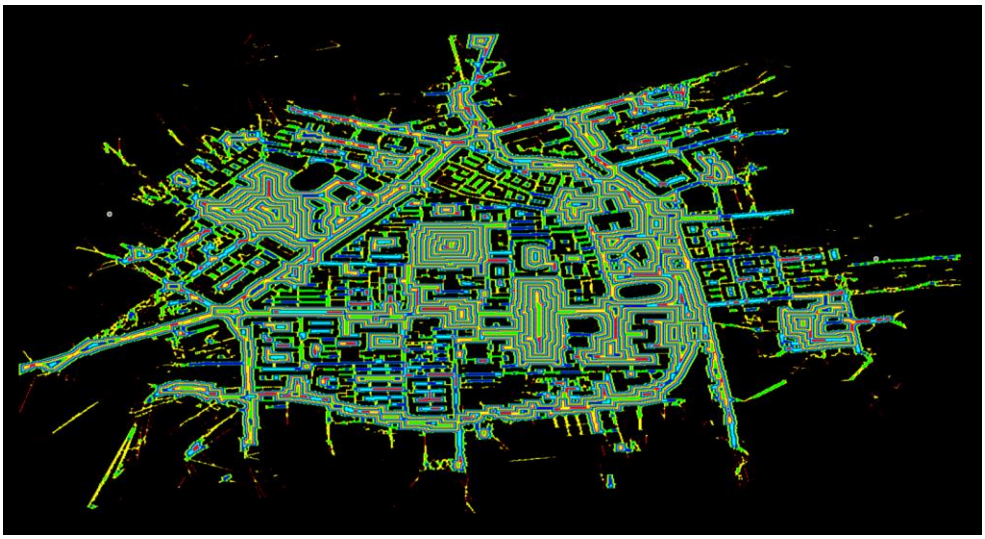


Figure 5-7: Assigning weights to roadways.

While the above method generated realistic paths, the complexity was quite high. Generating a single path was time consuming and was not a feasible solution to generate paths in bulk. Further some of the path generation runs would fail since the randomly chosen source and destination did not have a connected path between them. As mentioned in the next Chapter, we ran Pathsim on a small set of paths generated using the above method, and the results were very encouraging.

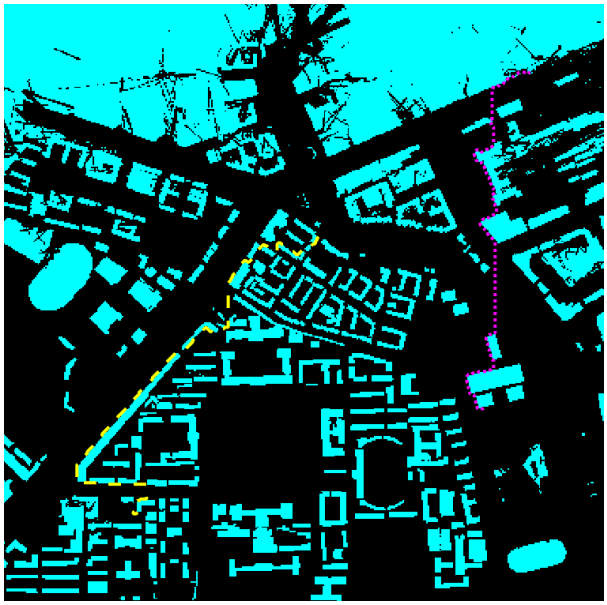


Figure 5-8: Example paths generated from Weighted building matrices.

As seen in **Figure 5-8**, the paths followed the buildings, which were assumed to be sidewalks. However, when crossing the roads and deciding on which sidewalks to take, there was still room for improvement in the algorithm. It is hard to accomplish this with any algorithm without an advanced AI algorithm or even greater weighted map with correct sidewalk crossings set in place, both which require even more time and effort to develop. Therefore, we sought to use more simpler scripts that would generate paths quickly.

5.5.2 Shortest Path using Binary Matrices

Our second attempt at realistic path generation was using image processing to compute distance transforms on a binary image. To be more specific, the matrix of 0s and 1s can be considered as a binary image and we leverage tools like *bwdistgeodesic* [30] within MATLAB to find the shortest path between two points on the image [31]. This function in MATLAB will compute the Euclidean length of the distance from any 0 to 1 in a matrix, while taking into account of objects (such as buildings in our case). When used on both the

starting and ending matrix, then can take the sum of those two matrices to show all the possible shortest paths. The result is a path region that outlines a region of the map representing the shortest path(s) between two points as shown in **Figure 5-9**. The gray areas in the plot indicate multiple paths that could be a possible path, which can be thinned down to one single line shown in yellow.

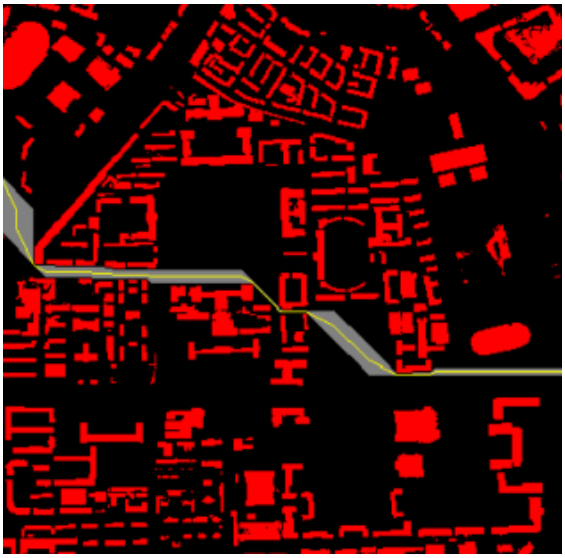


Figure 5-9: Example paths created with *bwdistgeodesic*.

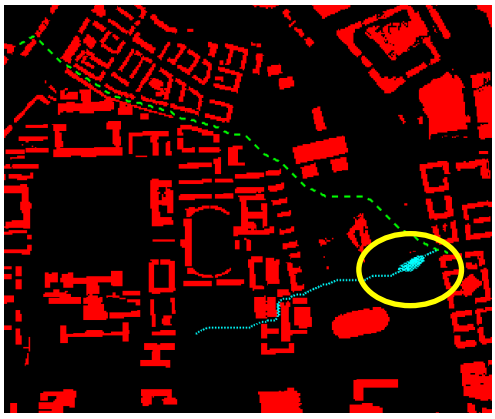


Figure 5-10: Invalid path using bwdistgeodesic.

While this generation was significantly faster than its predecessor, the paths were not completely realistic, and would often result in invalid paths, containing loops, as shown in **Figure 5-10** that were not desirable for our purposes. We ended up discarding the paths using this method, but still mention it here for academic purposes.

5.5.3 The A* Method

Our final path generation involved using the A* (A Star) method [32], which is a best-first search algorithm that improves on Dijkstra's method by using heuristics at a step-by-step analysis to provide a faster and more accurate path generation analysis [33]. Usually, Dijkstra is sufficient for calculating the shortest path by considering many combinations of paths up until the shortest path is found. However, when dealing with a map on a very large scale, this is impractical since we do not want to consider so many paths out of the way. The A* search algorithm considers a lot less points versus Dijkstra's algorithm [29].

A* requires processing heuristic functions of the actual distance the current unit is from the target. These heuristic distance functions are as follows:

$g(n)$ – distance away from the target node

$h(n)$ – distance away from the starting node to the target node

$f(n)$ – the sum of the two above distances $(g(n) + h(n))$

Some of these functions (like $g(n)$ and $h(n)$) are also computed in the previously mentioned binary matrix method. Thus, the A* algorithm combines elements from the previous two methods to make a more intelligent path finding algorithm.

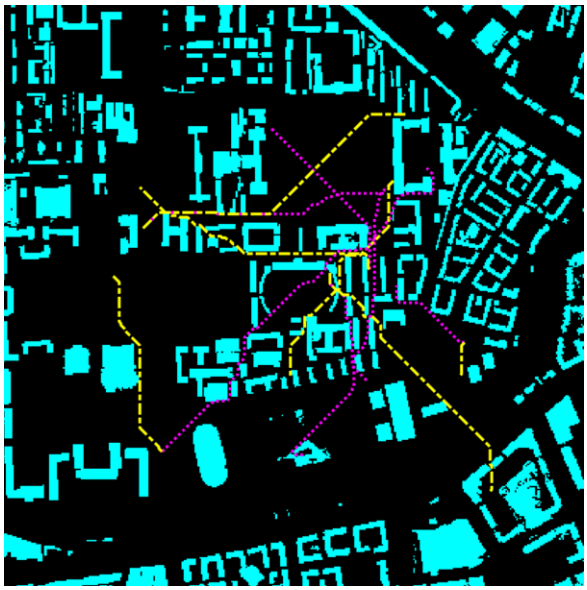


Figure 5-11: Example paths generated using A* Search Algorithm

The A* method produced accurate paths and mitigated most of the drawbacks of the previous method. It has a lower complexity than Dijkstra's method and failed less than 1% of the time. **Figure 5-11** gives an example of some of the path pairs (with the same starting points) that was generated using the A* method.

CHAPTER 6

EVALUATION

6.1 Path Generation Tests

The data generation for the Pathsim experimental setup is outlined in Chapter 5.3. We used Wireless Insite [28], a radio propagation software for analyzing wireless communications systems using 3D ray tracing. The 2 km by 2 km focus described in the previous chapter was taken at the center, and we avoided the edges of the map where the signal strength may be poor or unreachable in some places. We then used the data generated from Wireless Insite and inserted it into a MATLAB cell array which contained all the MRO vectors in a 2D plot of the map.

The A* search algorithm [33] was incorporated into several scripts and functions in MATLAB, which was then used to generate four different test case scenarios with varying degrees of path differences. These tests produced roughly a thousand path pairs each. To simplify the timing values for the paths, we consider the index of each point to serve as the relative timestamp value for the path.

1. The first set of tests generated path pairs where the starting point was the same for both paths. The end point was selected at random.
2. The second set of tests generated path pairs where starting point was selected at random, and the ending point was the same for both paths.

3. The third set of tests generated path pairs where the starting and endpoints were selected at random, but it was ensured that the paths intersected somewhere in the middle. The timing was adjusted such that both paths had a similar index value at the point of intersection.
4. The fourth set of tests generated path pairs with similar starting and ending points. This ensured that all points on both paths were close to each other from the source to the destination. This posed a harder challenge for Pathsim and was used to compare its performance with Vecsim.

The first three test suites were used to check the correctness of Pathsim using simulated paths. The results are discussed in the subsequent subchapters.

6.2 Pathsim Test Results

In this Chapter we report the result of running Pathsim on the first three test suites outlined above.

6.2.1 Paths with common source

Two paths (A and B), with the same starting point, but different random ending points.

This test suite aims to compare paths that initially start at the same point and might follow the same path, but usually the two paths will veer off into different directions. This test should yield similarity scores of varying degrees of all ranges. Some example path pairs can be seen in **Figure 6-1**, where the yellow and magenta paths form a pair with same source location. It was found that in 97% of cases the starting location pairs yielded a similarity score of < 4, indicating they were close together.

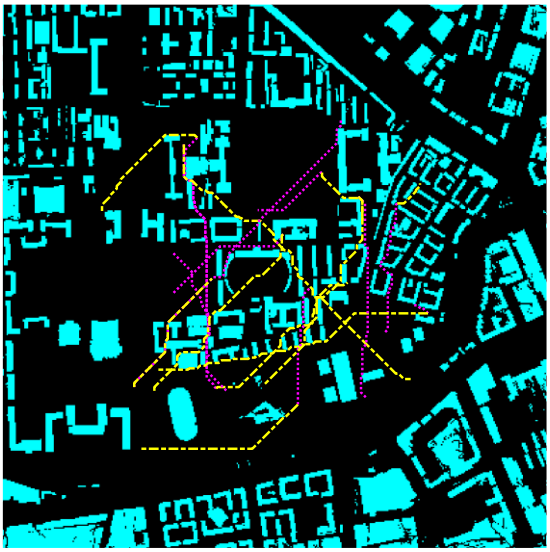


Figure 6-1: Example path pairs with common source.

6.2.2 Paths with common destination

Two paths (A and B), with common ending coordinates, but random starting coordinates.

This test suite aims to compare two paths that initially start at some random point and eventually converge at a common destination. Since the length of these paths can be quite different an adjustment was made to truncate the length of the larger path, such that both paths were roughly equal in length. This was done so that the indexes at which these paths converge can be kept relatively the same to remain consistent with our timing index. Pathsim was highly efficient and yielded a similarity score between 1-4 near the destination locations of both path pairs in all test cases.

6.2.3 Paths that meet in the middle

Two paths (A and B), that meet somewhere in the middle but have random starting and ending points.

This test suite evaluates the performance of Pathsim on path pairs that initially start off at random locations, then converge to an intersection point before finally diverging to different random ending points. These paths were generated by generating the paths in two

steps. The first step was selecting some of the paths from the previous test suite. Basically, a path pair was selected with different starting points but a common ending point. Then a second pair of paths were generated by re-using the ending point of the previous pair as the starting point for the next step. The two path pairs were then concatenated together to simulate a path pair that has a common intersection point. As with the previous test suite, the length of the paths obtained in Step 1 were adjusted such that the timestamp index at the intersection point remains the same for both pairs. An example of a simulated path can be seen in **Figure 6-2**.

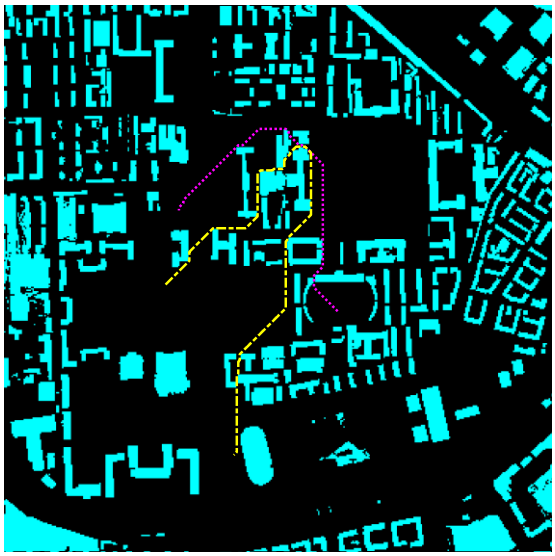


Figure 6-2: Example path pair with an intersection point.

Running Pathsim on this test suite yielded a similarity score of < 4 when comparing path pairs near the point of intersection in 98% of cases.

It should be noted that running Pathsim on the above cases resulted in similarity scores of < 4 in some other path pairs as well. This is normal since some of these paths may have intersected multiple times due to the random nature of our path generation algorithm.

6.3 Final Pathsim Test Results and Comparison with Vecsim

One of our main considerations when evaluating Pathsim, was to test its improvement over our prior evaluations with Vecsim. Vecsim is already efficient but suffers from some drawbacks as outlined previously in Chapter 4. In this Chapter we report our comprehensive test results when comparing the two methods.

6.3.1 Proximity Path Generation

Two paths (A and B), with similar starting and ending coordinates, within a given radius of 5 units.

Our test suite for this Chapter, first uses a radius random point function to select a random starting and ending point location on the road (not building) that has a valid signal value, as seen in **Figure 6-3**. We then generate a second path where the starting and ending locations are within a 5-unit radius form the starting and ending points of the first path, respectively. Some examples of these paths are shown in **Figure 6-4**, where it can be seen that these paths will often follow very closely to each other, sometimes overlapping in the middle. These form ideal test suites as we expect the similarity score to remain low throughout the length of both path pairs setting up a fair comparison with Vecsim. We generated 925 of these path pairs for our evaluation.

Figure 6-3: Choosing random starting/ending locations near each other.

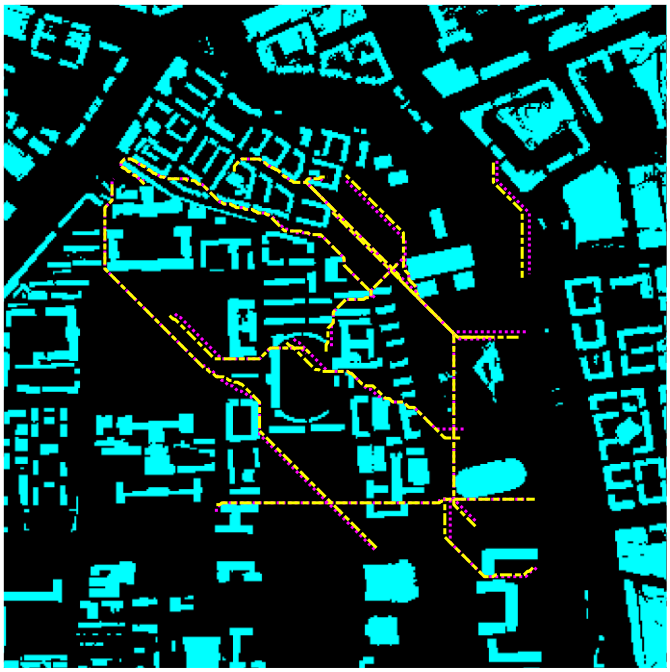


Figure 6-4: Paths with similar routes.

6.3.2 Effectiveness of using adjacency matrix

Before looking at the overall test results, it is important to look at some individual test cases to see the improvement of using an adjacency matrix of scores when comparing two user paths.

The matrix below shows a typical case where Vecsim fails to produce a valid result when comparing two close locations, A and B, that belong to different paths. In this case, the UEs are connected to different base stations and thus Vecsim cannot establish a discrete value for their distances. However, as Pathsim also computes the similarity score between neighboring locations of the original pair, we can clearly see that these points are close together as most of their nearest neighbors (at most 2 units apart) report a similarity score of 1, which means it is highly likely that A and B are also neighboring points.

	B-2	B-1	B	B+1	B+2
A-2	1	2	2	2	1
A-1	1	N/A	1	1	1
A	1	1	N/A	1	1
A+1	1	1	1	1	1
A+2	1	1	2	2	2

It is important to note that neither Vecsim nor Pathsim maintains the actual user location information. The neighbor information that is used in Pathsim is solely derived from the fact that by logging the users' path continuously, we can safely assume that the MRO data reported by the same UE in consecutive timestamps are close to each other, which is then used to determine the neighboring points.

Another scenario where Pathsim holds a clear advantage over Vecsim is shown in the following matrix that was selected from the test results.

	B-2	B-1	B	B+1	B+2
A-2	1	1	2	2	2
A-1	1	2	2	2	3
A	1	6	6	1	2
A+1	1	2	1	1	2
A+2	1	1	1	2	2

In this case due to irregularities in the signal field, Vecsim returns a similarity score of 6, even though A and B are relatively close to each other. However, with Pathsim, the adjacency matrix of similarity scores represents a probability density function, and the mode

can be obtained as the overall representation of the similarity score. In this case, as there is a tie between the frequency of scores of 1 and 2 a rounded average value is used to represent the final score.

6.3.3 Improvement over Vecsim

We computed the similarity scores between all adjacent points on both paths using both Vecsim and Pathsim. We repeat this for all 925 test cases and accumulate Vecsim and Pathsim similarity scores for over 100,000 unique point pairs belonging to different paths. Our first comparison analyzed these pairs of points to determine how many of these pairs Vecsim was not able to produce a similarity score for. As explained earlier, Vecsim is unable to produce a similarity score for pairs that are close to each other but are connected to different base stations. **Figure 6-5** shows a bar chart comparing the number of point pairs that did not produce a similarity score in either Vecsim or Pathsim. Out of 109,827 unique location pairs, Vecsim failed to produce a similarity score roughly 22% of the time, whereas Pathsim was able to produce a valid score in over 95% of cases. The small fraction of cases where Pathsim also failed to compute a valid score where cases in which the entire 5x5 adjacency matrix resulted in invalid scores. We also note that the performance of neither method improved with introduction of the DPD check.

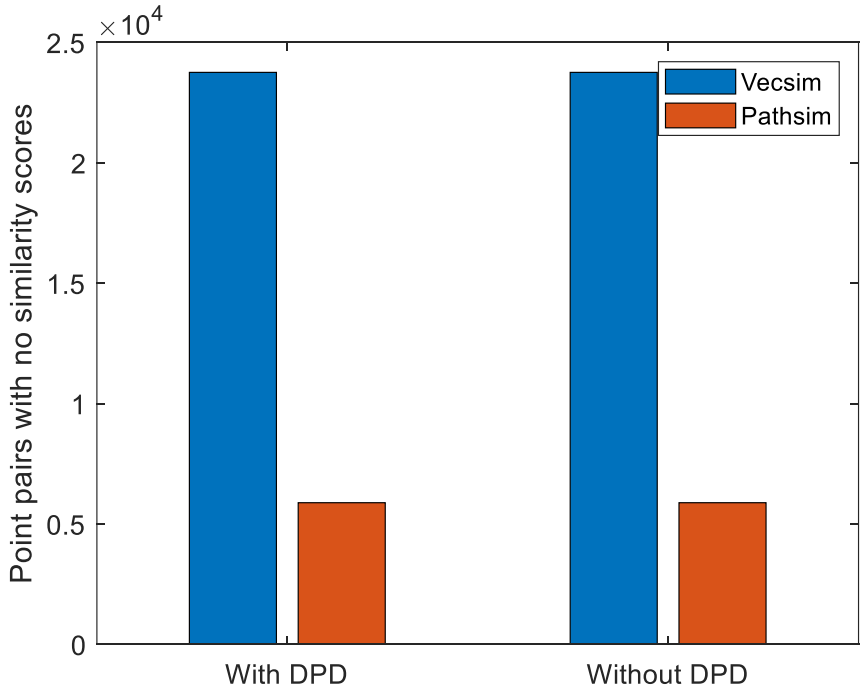


Figure 6-5: Location pairs with no similarity scores.

This represents the biggest advantage of Pathsim over Vecsim. By enabling similarity score computation even when UEs are not connected to the same base station solves one of the biggest limitations of Vecsim.

The second avenue of comparison was to check the distance estimation accuracy when compared to Pathsim. Even though both methods rely on the same signal model, the availability of additional comparisons between neighboring points can produce a modest improvement over the accuracy reported in Vecsim. **Figure 6-6** shows this improvement by generating a Cumulative Density Function plot of the similarity scores obtained using Vecsim and Pathsim. The x-axis represents the similarity score values and the y-axis represents the fraction of cases with the corresponding x-values. Since all of these path pairs were generated using relatively close paths, in theory all of the similarity scores should ideally

be less than 4, indicating a distance of less than 100 meters between any two points. It can be seen that Vecsim reports an accurate result roughly 80% of time when the similarity score is <=4. On the other hand, Pathsim reports a higher accuracy of 84% when using the simple mode method i.e., picking a random mode for multimodal distributions. The accuracy drops slightly when using the averaged mode for multi-modal distributions but is still an improvement over the performance of Vecsim. We note that to keep the comparison fair, we did not include a Discontinuity Pair Database check for either of the two methods.

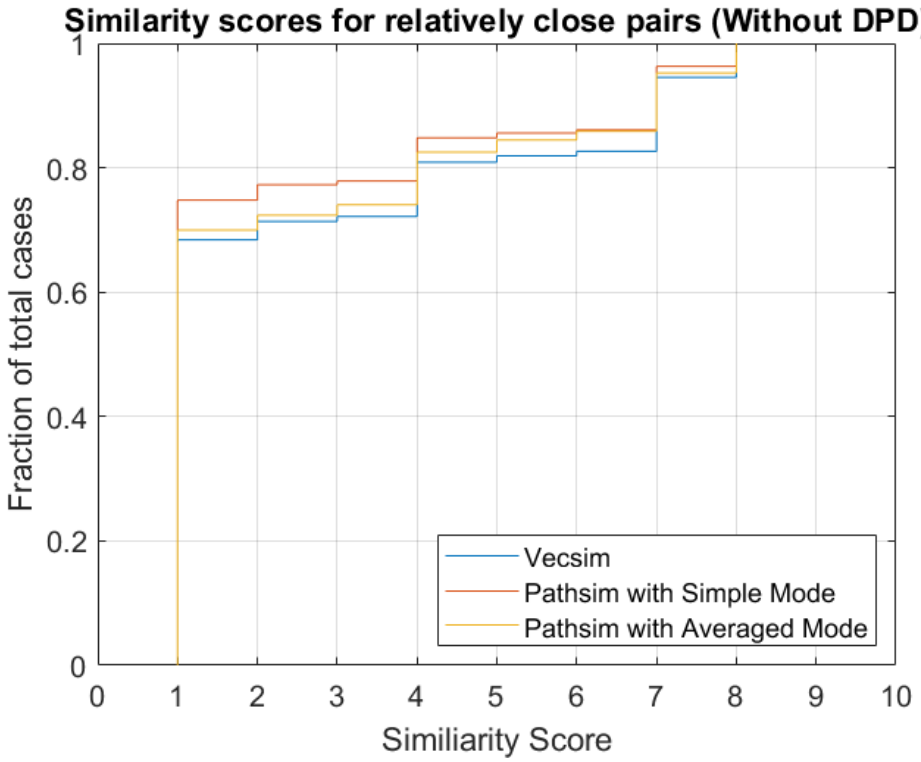


Figure 6-6: Comparison of estimation accuracy between Vecsim and Pathsim (without DPD).

We ran a final set of tests comparing the accuracy improvement of Pathsim over Vecsim when incorporating the DPD (Discontinuity Pair Database, as discussed earlier in Section 4.2) check in our processing code. The results are shown below in **Figure 6-7**.

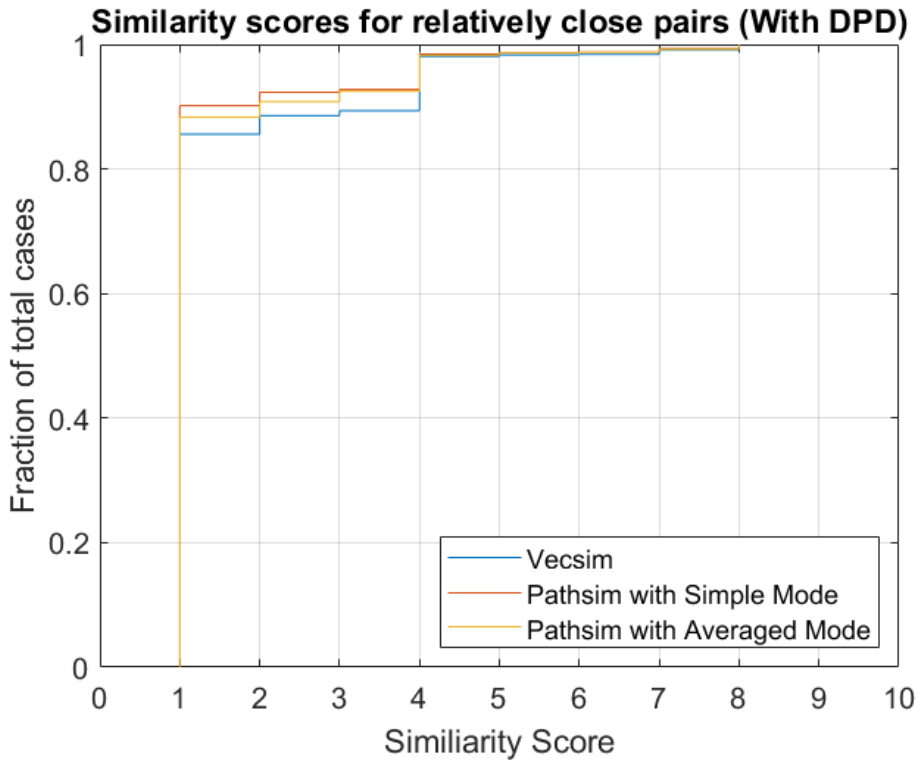


Figure 6-7: Comparison of estimation accuracy between Vecsim and Pathsim (with DPD).

With the introduction of the DPD check, the performance of Vecsim dramatically improves with over 95% of cases reporting a similarity score of <=4. Pathsim also reports a similar number but it can be seen that it reports a higher fraction of similarity scores < 2, indicating a better estimation of the distance between two points.

Finally, using Pathsim, resulted in a small increase in the false positive rate over Vecsim. In roughly 0.5% of cases, Pathsim reported a lower score than Vecsim when random point pairs were considered that were far apart. We note that for an application like contact tracing, a higher false alarm rate can be tolerated as it is less critical than accurately estimating the distance at close locations.

CHAPTER 7

DISCUSSION, CONCLUSIONS, FUTURE WORKS

7.1 Discussion

The above results are highly encouraging and demonstrates the advantage of comparing path information to better estimate the distance between neighboring locations. As mentioned in our theoretical analysis, the biggest advantage of Pathsim is its ability to generate similarity scores for points that are connected to different base stations. While Vecsim discards similarity scores for points with different PCI identifiers roughly 20% of the time, the percentage of discarded pairs with Pathsim goes down to only 5% in our evaluation suite. This represents a significant improvement of around 75%. In addition, Pathsim also reports a modest improvement in the distance estimation accuracy over Vecsim. Finally, the inference from the test results also indicates that the DPD check is much more necessary in Vecsim than in Pathsim.

7.2 Limitations

Due to the sheer amount of data that must be processed, the study of Pathsim is based on a small area of the map developed from Vecsim because of the DPD testing and comparisons between the two methods. Therefore, Pathsim, like Vecsim, is still bound to the map generated and the raw processing power of both the software and hardware that is being utilized.

Although Pathsim has shown that it can overcome several limitations that Vecsim had imposed, Pathsim still has several areas where it can improve. The experimental setup assumes cellphones are all oriented in the same direction, which may not be a realistic assumption. Still, theoretically Pathsim' s performance can only improve if a real-world run were to take place, since Pathsim may be able to access to different sets of MRO data corresponding to multiple orientations of each users' cellphone at the same location. However, this is still untested, and the distance estimation algorithm would need to be tweaked slightly to accommodate this.

Another limitation that Pathsim has is its assumption that all the MRO data belongs to one carrier. All the MRO data variables we used are in the same format. However, in the real world, this is not the case, as cellphones can originate from a variety of different carriers that Pathsim would need to account for. In order to make the Pathsim method successful, the carriers would need to collaborate and share their data, perhaps even making the data standardized for easy record keeping as well. While it would be beneficial for all contact tracing methods to have access to standardized data like this, the time it would take for the carriers to set up this system may be lengthy given their already established systems.

7.3 Conclusion

In conclusion Pathsim is a highly effective method of comparing user signal given sufficient MRO log data. When utilizing Pathsim as a contact tracing method, it not only preserves user privacy, but also accurately estimates the user proximity 96% of the time. Pathsim is also significantly different from other contact tracing efforts in that it does not require user intervention or localization techniques and is completely transparent to the end user. Pathsim overcomes several of the practical limitations of Vecsim, while still using its sound fundamental signal model. Pathsim sacrifices some of the low complexity techniques

used in Vecsim, and in turn exhibits a higher accuracy than its predecessor. We believe Pathsim can function as a real-world system to drastically improve the quality of current contact tracing efforts.

7.4 Future Work

Pathsim has already shown that the collection of discontinuity pairs is not as needed if we are able to generate the path of the users. Innovative technology would further aid in the development of contact tracing methods such as Pathsim, which may completely remove the need for the DPD. Some examples of these upcoming technologies would be big data, new coding languages (to reduce processing time), and artificial intelligence (for more advanced and accurate path generation). However, advanced technology alone is not the answer to everything, and it is found that even in countries with high fear of the virus, the best method of approach is to use a mix of manual tracing, low technology such as phones and wearables, and high technology as previously mentioned [34]. Continued study in contact tracing methods is vital during this critical period but implementing these methods and starting proposed plans immediately is the key to stopping any virus from spreading rapidly.

REFERENCES

[1] M. Cebrian, "The past, present and future of digital contact tracing," *Nature Electronics,* vol. 4, no. 1, pp. 2-4, 2021.

[2] W. Zheng, H. Zhang, X. Chu and X. Wen, "Mobility robustness optimization in self-organizing LTE femtocell networks," *EURASIP Journal on Wireless Communications and Networking,* vol. 2013, no. 1, pp. 1687-1499, 2013.

[3] N. Eagle, J. Quinn, A. Clauset, T. Hideyuki, M. Beigle, A. Friday and A. J. Brush, "Methodologies for Continuous Cellular Tower Data Analysis," *Pervasive Computing,* pp. 342-353, 11 May 2009.

[4] The Lancet Digital Health, "Contact tracing: digital health on the frontline," 2020. [Online]. Available: https://doi.org/10.1016/S2589-7500(20)30251-X. [Accessed 26 April 2020].

[5] Korea Centers for Disease Control and Prevention, "Contact Transmission of COVID-19 in South Korea: Novel Investigation Techniques for Tracing Contacts," *Osong Public Health Res Perspect,* vol. 11, no. 1, pp. 60-63, 28 2 2020.

[6] P. Boeing and Y. Wang, "Decoding China's COVID-19 'virus exceptionalism': Community-based digital contact tracing in Wuhan," *R&D Management,* 23 3 2021.

[7] N. Gan and D. Culver, "China is fighting the coronavirus with a digital QR code. Here's how it works," 16 April 2020. [Online]. Available: https://www.cnn.com/2020/04/15/asia/china-coronavirus-qr-code-intl-hnk/index.html. [Accessed 30 April 2021].

[8] Singapore Government Agency, "TraceTogether, safer together," 2020. [Online]. Available: https://www.tracetogether.gov.sg/. [Accessed 30 April 2021].

[9] Centeres for Disease Control and Prevention, "Contact Tracing, Get and Keep America Open: Supporting states, tribes, localities, and territories," November 2020. [Online]. Available:

https://www.cdc.gov/coronavirus/2019-ncov/php/contact-tracing/index.html. [Accessed 30 April 2021].

[10] "Show evidence that apps for COVID-19 contact-tracing are secure and effective," Nature, 2020. [Online]. Available: https://www.nature.com/articles/d41586-020-01264-1. [Accessed 1 May 2021].

[11] Center for Disease Control and Prevention, "Case Investigation and Contact Tracing : Part of a Multipronged Approach to Fight the COVID-19 Pandemic," 3 December 2020. [Online]. Available: https://www.cdc.gov/coronavirus/2019-ncov/php/principles-contact-tracing.html. [Accessed 25 April 2021].

[12] C.-E. Juneau, A.-S. Briand, T. Pueyo, P. Collazzo and L. Potvin, "Effective Contact Tracing for COVID-19: A Systematic Review," 2020. [Online]. Available: https://doi.org/10.1101/2020.07.23.20160234. [Accessed 25 April 2021].

[13] North Dakota State Government, "Care19 Alert," [Online]. Available: https://ndresponse.gov/covid-19-resources/care19. [Accessed 5 May 2021].

[14] National Academy for State Health Policy, "State Approaches to Contact Tracing during the COVID-19 Pandemic," 22 April 2021. [Online]. Available: https://www.nashp.org/state-approaches-to-contact-tracing-covid-19/#tab-id-2. [Accessed 1 May 2021].

[15] L. V. Ness, "Contact Tracing Apps Balance Privacy With Effectiveness," 22 March 2021. [Online]. Available: https://www.governing.com/security/contact-tracing-apps-balance-privacy-with-effectiveness.html. [Accessed 26 April 2021].

[16] Apple, Google, "Privacy-Preserving Contact Tracing," [Online]. Available: https://covid19.apple.com/contacttracing. [Accessed 30 April 2021].

[17] E. Yoneki, "FluPhone Study: Virtual Disease Spread using Haggle," *In Proceedings of the 6th ACM workshop on Challenged networks (CHANTS '11),* pp. 65-66, 2011.

[18] R. A. Michelin, W. Xue, S. Ruj, R. Malaney, S. S. Kanhere, A. Seneviratne, W. Hu, H. Janicke and S. K. Jha, "A Survey of COVID-19 Contact Tracing Apps," *IEEE Access,* vol. 8, pp. 134577-134601, 2020.

[19] S. Abuhammad, O. . F. Khabour and K. H. Alzoubi, "COVID-19 Contact-Tracing Technology: Acceptability and Ethical Issues of Use," *Patient preference and adherence,* vol. 14, pp. 1639-1647, September 2020.

[20] J. A. Kucharski, P. Klepac, A. J. K. Conlan, S. Kissler, M. L. Tang and H. Fry, "Effectiveness of isolation, testing, contact tracing, and physical distancing on reducing transmission of SARS-CoV-2 in different settings: a mathematical modelling study," *The Lancet,* vol. 20, no. 10, pp. 1151-1160, 16 June 2020.

[21] N. Lanese, "Why hasn't contact tracing managed to slow the massive surge of coronavirus in the US?," September 2020. [Online]. Available: https://www.livescience.com/covid19-contact-tracing-us-states.html. [Accessed 25 April 2021].

[22] MathWorks, "What Is Artificial Intelligence (AI)?: 3 things you need to know," [Online]. Available: https://www.mathworks.com/discovery/artificial-intelligence.html. [Accessed 26 April 2021].

[23] E. Hernández-Orallo, P. Manzoni, C. T. Calafate and J. Cano, "Evaluating How Smartphone Contact Tracing Technology Can Reduce the Spread of Infectious Diseases: The Case of COVID-19," *IEEE Access,* vol. 8, pp. 99083-99097, 2020.

[24] E. Waltz, "Back to Work: Wearables Track Social Distancing and Sick Employees in the Workplace," 1 May 2020. [Online]. Available: https://spectrum.ieee.org/the-human-os/biomedical/devices/wearables-track-social-distancing-sick-employees-workplace. [Accessed 30 April 2021].

[25] Y. Zhengqing and I. F. Magdy, "Ray Tracing for Radio Propagation Modeling: Principles and Applications," *IEEE Access,* vol. 3, no. 1089-1100, 2015.

[26] E. Hecht, Optics, Edinburgh Gate Harlow, UK: Pearson, 2016.

[27] M.-K. Olkkonen, V. Mikhnev and E. Huuskonen-Snicker, "Complex permittivity of concrete in the frequency range 0.8 to 12 GHz," *2013 7th European Conference on Antennas and Propagation, EuCAP 2013,* pp. 3319-3321, 2013.

[28] Remcom, "Wireless InSite," [Online]. Available: https://www.remcom.com/wireless-insite-em-propagation-software/. [Accessed 30 April 2021].

[29] Wikipedia, "Dijkstra's algorithm," [Online]. Available: https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm. [Accessed 26 April 2021].

[30] MathWorks, "bwdistgeodesic: Geodesic distance transform of binary image," [Online]. Available: https://www.mathworks.com/help/images/ref/bwdistgeodesic.html. [Accessed 30 April 2021].

[31] S. Eddins, "Exploring shortest paths – part 4," 6 December 2011. [Online]. Available: https://blogs.mathworks.com/steve/2011/12/06/exploring-shortest-paths-part-4/. [Accessed 30 April 2021].

[32] P. E. Hart, N. J. Nils and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE Transactions on Systems Science and Cybernetics,* vol. 4, no. 2, pp. 100-107, 1968.

[33] Wikipedia, "A* search algorithm," [Online]. Available: https://en.wikipedia.org/wiki/A*_search_algorithm. [Accessed 26 April 2021].

[34] P. H. O'Neill, "Five things we need to do to make contact tracing really work," MIT Technology Review, [Online]. Available: https://www.technologyreview.com/2020/04/28/1000714/five-things-to-make-contact-tracing-work-covid-pandemic-apple-google. [Accessed 30 April 2021].

[35] W. H. Gates III, "Overview of Windows Architecture: The 32-bit Days," Microsoft Notes, Seattle, 2001.

[36] A. Church, "Lambda Calculus Revisited," *Journal of Advanced Programming,* vol. 15, no. 6, pp. 24-27, 1953.

[37] A. M. Turing, "On Computability," *Advanced Mathematical Principles,* vol. 15, no. 7, pp. 175-183, 1942.

[38] J. L. von Neumann, EDVAC Programming Manual, Princeton: IAS Press, 1948.

[39] J. P. Eckert and J. Mauchly, "Building the First Digital Computer," in *Princeton Computer Conference*, Princeton, 1945.

[40] A. Mukherjee, Y. Zhong, Z.-H. Zhang, T. Zhao and J. Zhang, "Vecsim: Carrier-based, Privacy-Preserving Cellphone Contact Tracing," *In Proceedings of the 18th ACM Symposium on Mobility Management and Wireless Access (MobiWac '20),* p. 47–55, 2020.