2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN) | 978-1-6654-0402-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/IPIN51156.2021.9662582

On the Use of Lookahead to Improve Wi-Fi Fingerprinting Indoor Localization Accuracy

Lu Shi, Nader Moayeri, Chang Li
Advanced Network Technologies Division
National Institute of Standards and Technology
Gaithersburg, USA
lu.shi, nader.moayeri, chang.li@nist.gov

***Abstract*—Causality is a basic concept in system theory. In this paper we introduce the notions of causal and non-causal indoor localization. A non-causal localization system uses future signal and sensor measurements, in addition to past and present ones, to estimate the location of a person or an object at the present time. We provide example use cases where non-causal localization could prove useful.**

The main contribution of the paper is the development of an indoor localization system based on Wi-Fi fingerprinting and the Viterbi Algorithm that could be used in both causal and non-causal modes. Our proposed method finds the best "path" in a building matching a time series of Wi-Fi scan results made by the mobile device carried by the person to be located over a period of time. Our system improves localization accuracy by using the knowledge of building floor plans and the fact that humans do not move faster than a certain speed.

We evaluate the performance of the causal and non-causal versions of our system and compare them against the basic Wi-Fi fingerprinting localization system based on the KNN algorithm in an office building. Our empirical results show that both the causal and non-causal versions of our system outperform the basic fingerprinting system, with the non-causal version yielding significantly higher accuracy than the basic fingerprinting system.

***Index Terms*—Indoor localization, Wi-Fi fingerprinting, Viterbi algorithm, dynamic programming, real-time localization, causal localization, non-causal localization, human mobility model, building floor plans, spatiotemporal constraints, lookahead**

I. Introduction

Most researchers regard (indoor) localization as a process through which the location of an entity of interest is estimated in real-time. That is, the location $\underline{L}(t)$ is estimated using past and present samples of one or more random processes (observations) through time t, and the location estimate $\hat{\underline{L}}(t)$ is made available to the system user instantly. This is what we'd like to call *causal localization*. The random processes are signals received and/or sensor measurements. Admittedly, the boundary between signals and sensors is hazy. Typically, signals include radio frequency (RF) signals, ultrasound, infrared, and visible light. Sensors commonly used include accelerometer, gyroscope, magnetometer, and altimeter. The entity of interest, which is equipped with an electronic device to facilitate localization[1], can be a person, an asset, a robot, etc. The best example of using sensor measurements prior to time t to estimate $\underline{L}(t)$ is a localization system that uses dead reckoning based on an Inertial Measurement Unit (IMU), which includes an accelerometer and a gyroscope. In such a system, the past history of measurements does indeed matter, as $\underline{L}(t)$ cannot be estimated from the instantaneous accelerometer and gyroscope measurements.

More generally, a localization system estimates $\underline{L}(t)$ based on available observations, which may include observations made after time t. Sometimes, one is interested in estimating $\underline{L}(t)$ on demand, such as when one wishes to know the location of a stationary asset on a factory floor. In other applications, one is interested in estimating $\underline{L}(t)$ in a periodic manner over a set of discrete time indices $\Lambda = \{1, 2, \ldots, T\}$. This paper focuses on the second case with the entity of interest being a person.

A *non-causal localization* system is one that uses observations made after time t to estimate $\underline{L}(t)$. This necessarily means that the system has to wait and make more observations beyond time t before it estimates $\underline{L}(t)$. Therefore, $\hat{\underline{L}}(t)$ will be made available some time after time t. We provide two use cases where non-causal localization could prove useful. One is a surveillance application, where the system uses the signals received over Λ by the Wi-Fi Access Points (APs) in a building from the smartphone of a person of interest to determine which places in the building the person visited during Λ, at what times, and how much time the person spent at each location. A second example is when a person spends several hours at a trade show and sometime later wishes to know at which booths he/she spent most time and hence which technologies and products were of most interest to him/her. This can be done using Wi-Fi signals received by the person's smartphone from Wi-Fi APs at the trade show. It may also be done using technologies other than Wi-Fi.

Bahl and Padmanabhan [1] developed the RADAR system based on Wi-Fi fingerprinting. Their paper has had a large impact since its publication in the year 2000. Many researchers have developed indoor localization systems using fingerprinting, not just based on Wi-Fi, but also based on Bluetooth, ZigBee, magnetic signals, or combinations thereof. In addition, many companies have developed commercial indoor localization solutions based on fingerprinting. Wi-Fi fingerprinting has an offline phase and an online phase. A fingerprint catalogue/database, also called a radio map in more

[1] This paper does not address device-free localization.

recent papers, is built in the offline phase. Each fingerprint measurement involves placing a mobile Wi-Fi device at the fingerprint location and making Received Signal Strength Indicator (RSSI) measurements at either the mobile device or at the M Wi-Fi APs deployed in the building. The online phase is when the system is used for localization, where either the mobile device or the APs do a Wi-Fi scan and record the RSSIs. The resulting M-tuple of RSSIs is then compared with the fingerprints in the radio map using the K-Nearest Neighbor (KNN) algorithm, which is often used to solve classification problems. $\hat{\underline{L}}(t)$ would then be computed as a combination of the resulting K fingerprints. In this paper, we use $K = 1$, which means the fingerprint "closest" to the RSSI scan result is selected in our implementation of RADAR.

It is well-known that the basic Wi-Fi fingerprinting localization scheme described above suffers from "jumps" in $\hat{\underline{L}}(t)$ over short times. For example, when a user is walking around in a building and $\hat{\underline{L}}(t)$ is tracking $\underline{L}(t)$ fairly well, the system may suddenly produce a location estimate quite far from the user's true location followed by another one that may be reasonably close to it. The same phenomenon is observed when locating stationary objects. These jumps are annoying and should be prevented from happening at all cost. Gentile and Klein-Berndt [2] used a Markov chain model for a person's movements to prevent the possibility of such jumps. Their scheme basically limits the "next" user location estimate to fingerprint locations in the neighborhood of the "present" location estimate, which itself is a fingerprint location. The system developed in [2] is a causal localization system.

In this paper, we generalize the concept introduced in [2] in two ways. First, we not only prevent location estimate jumps, but we also prevent consecutive location estimates from being on opposite sides of a wall. These location estimate candidates may be very close to each other, but a person cannot move from one side of a wall to the other in a short time span. To prevent such transitions, the system needs to know where the walls and doors are, and hence it needs to have access to the building floor plans. Second, we allow non-causality in our system by using Wi-Fi scan results made after time t to estimate $\underline{L}(t)$. We show later how the solution to this problem is reduced to searching for the path in a trellis diagram most similar to the set of Wi-Fi scan results over Λ. We use the Viterbi algorithm (VA) [3], [4] to solve this problem, thereby increasing the accuracy of the localization system. Specifically, we leverage the abovementioned spatiotemporal constraints to develop an accurate non-causal localization and tracking system based on Wi-Fi fingerprinting.

Our key contributions include:

- We design our system based on a minimum number of Wi-Fi fingerprints taken at each fingerprint location, thereby significantly reducing the fingerprint surveying effort and the memory requirements for the fingerprint database. This also means that we cannot compare our system to that in [2] which uses 100 measurements at each fingerprint location.
- The positions along the path are computed in real-time and corrected dynamically over time.
- The proposed algorithm is evaluated repeatedly on two smartphones of different brands using five test scenarios in a typical office environment. The results show that our algorithm achieved an error as low as $0.7\,m$ on the average, and the 95-percentile point on the probability distribution of the errors was $1.9\,m$.

The rest of the paper is organized as follows. Section II reviews related work on solutions based on Wi-Fi fingerprinting. Section III describes our algorithm in detail. Evaluation results are presented and analyzed in Section IV. Concluding remarks and insights obtained in this study are presented in Section V.

II. Related Work

The Viterbi algorithm (VA) [3], [4] is a dynamic programming algorithm for finding the most likely sequence of hidden states that results in a sequence of observed events. Modeling location estimation over a time window as a maximization problem, the VA has been applied in a few indoor localization papers. While a sequence of hidden states models the user's unknown path in a building, the time series of Wi-Fi scan results (RSSI measurements) represents the observed events. Different probabilities may be assigned to transitions from the current state/location to the next possible states/locations, which may include the current state/location.

Kohri [5] developed a ML estimator of the user's path using the VA. Instead of using Wi-Fi signal strengths, his method is based on comparing the range estimates from a number of fixed nodes in the building (typically called anchor nodes) to the user's location with the fixed distances from anchor nodes to a set of reference locations in the building. One can think of the reference locations as playing the same role as fingerprint locations in a fingerprinting method. The user's path in the building is estimated as a sequence of reference locations.

MapCraft [6] expresses the user path estimation problem as finding the most likely sequence of states in an undirected probabilistic graphical model, i.e. Conditional Random Field (CRF), using the VA. Instead of maximizing the joint probability, computed as the product of state priors and conditional probabilities of observations given states, the conditional probability of state variables given the observations can be directly maximized. Fusing Wi-Fi measurements with data collected from accelerometer, magnetometer and gyroscope on the mobile device, the 97-percentile error of MapCraft varies from $2.37\,m$ to $4.53\,m$ at different indoor test sites.

Trogh et al. [7] proposed to use the sum of squared errors between RSSI measurements and reference values in the radio map as the cost associated with each possible path in the location graph, which is modeled as a trellis diagram. A network planner was utilized to predict the RF path loss between each Wi-Fi AP and each fingerprint location. This method yielded a mean error of $2.2\,m$ in experiments in an office test bed.

Wawrzyniak et al. [8], [9] introduced a sequence detection method for indoor tracking at zone level using a minimum squared error metric for the overall path. Specifically, the transition from zone m to zone n at time t depends on the dissimilarity between reference RSSI data for zone m and RSSI measured at time t, and the dissimilarity between reference RSSI data for zone n and RSSI measured at time t. With a $2\,m \times 2\,m$ zone size, they achieved a mean error of $1.4\,m$ in their experiments, assuming the user was in the center of the zone in which he/she is located.

III. Algorithm

In this section, we describe our algorithm that estimates the mobile device's location along the user's trajectory in real-time. Our algorithm has the property that it continuously updates location estimates at all earlier times. Therefore, by the time it processes RSSI scan results at time T, which is the end of the period for which RSSI scan results are available, we not only have the real-time location estimate for time T, but also non-causal location estimates for any time $t \in \{1, 2, \ldots, T-1\}$, with t referring to the time the t^{th} Wi-Fi scan is completed.

The first step in the offline phase of developing any fingerprinting localization system is to select a set of N fingerprint locations. It is common practice to lay a point lattice over the service area for the localization system, and that's what we do in this paper. However, that is not the only choice, and the techniques presented in this section are equally applicable in the situation where the fingerprint locations do not follow the regular structure of a point lattice. Let δ, f_s, and $v_{\max}$ denote, respectively, the distance from each lattice point in the left part of Figure 1 to its closest neighbors, the frequency at which Wi-Fi scans are made, and the maximum speed of a person moving around in a building. $v_{\max}$ is typically assumed to be about $1.4\,m/s$, which corresponds to roughly $5\,Km/h$. This suggests that the user cannot move by more than $v_{\max}/f_s$ meters during one period of Wi-Fi scanning. Comparing this maximum possible displacement to δ determines to which lattice points in the neighborhood of a given lattice point the user may move to by the time the next Wi-Fi scan is completed. For example, if the user is at lattice point 5 in the left part of Figure 1 at time t, then at time $t+1$ the user can be at any of the lattice points to which a transition with a red dashed line is shown, including lattice point 5 itself if the user does not move at all. Transitions to lattice points 3, 6, 8, and 9 are not feasible, even though they are still within maximum possible displacement from lattice point 5, because the user cannot breach the wall. (If the user enters the room through the door, it will take longer than $1/f_s$ seconds for him/her to get to those points.) However, a transition from lattice point 5 to lattice point 7 is feasible, because the total length of the piece-wise linear red dashed line from 5 to 7 is less than $v_{\max}/f_s$. Essentially, one has to look at the shortest feasible path (not going through walls) from lattice point i to lattice point j to decide if j is reachable from i. Note that the notion that the radius of the circle that encloses the lattice points reachable from a given lattice point has to be $v_{\max}/f_s$ is not 100% correct. Lattice point 5 may be the closest lattice point to the user location at time t, but the user may not be precisely at lattice point 5 at that time. This affects which lattice points in the vicinity of lattice point 5 are reachable from the true user location. All that can be assumed is that lattice point 5 is a "candidate" estimate for user's location at time t. This issue is further discussed at the end of Section IV.

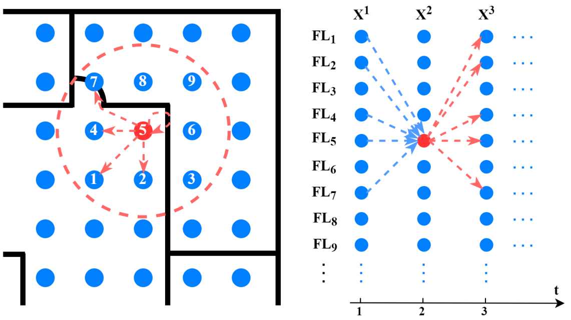

Fig. 1. Possible transitions from one fingerprint location to its neighbors and itself

In the right part of Figure 1, we have shown the transitions from lattice point 5 to all lattice points (fingerprint locations) reachable from lattice point 5 as well as transitions from the same reachable set to lattice point 5. Note that if a transition from lattice point i to lattice point j is possible, then so is a transition from lattice point j to lattice point i. The horizontal axis depicts discrete time. Also, note that if the wall going through the circle did not exist, transitions to/from all eight neighbors of lattice point 5 and lattice point 5 itself would have been possible. Figure 2 depicts a trellis diagram that shows all possible state transitions from time 1 to time $T=5$, stage by stage. The set of hidden states at each stage of the trellis is the set of all N lattice points in the service area, where fingerprint measurements are made in the offline phase. Stage t of the trellis is marked by X^t, which is the RSSI M-tuple resulting from the Wi-Fi scan by the mobile device at time t. For any $t \in \Lambda$, if one approaches the non-causal localization problem in a brute force manner, one would think there are N^t possible sequences of location estimates for the user from time 1 to time t. However, such a large set of sequences include sequences with jumps and/or wall breaches. The trellis prevents such jumps and wall breaches, and hence the actual number of possible paths in the trellis, which represents feasible sequences of estimates for user's location from time 1 to time t, is far smaller than N^t. Therefore, all that is needed to estimate the user's location from time 1 to time t is to search the trellis up to time t for the path whose fingerprint labels are "closest" to the sequence of Wi-Fi scan results from time 1 to time t. We use the VA to solve this problem. We next describe the fingerprinting process and precisely define what we mean by "closest".

Let M denote the number of Wi-Fi APs whose signals can be received in the localization system service area. While the fingerprinting process used in some papers in the literature

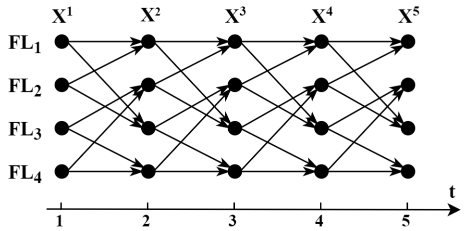

Fig. 2. Example of a trellis with $N = 4$ hidden states

TABLE I
TERMINOLOGY USED IN THIS PAPER

Symbols	Descriptions
X^t	Online Wi-Fi scan results at time t
$Y_{i,\phi}$	Fingerprint at location i and direction ϕ, where $\phi \in \{E, N, W, S\}$
Y_i	$(Y_{i,E}, Y_{i,N}, Y_{i,W}, Y_{i,S})$
F^t	Set of binary flags $f[t][i]$ indicating whether state i is reachable at time t
P^t	Set of minimum-cost t-tuple paths $p[t][i]$ for each reachable state i at time t
C^t	Set of minimum costs $c[t][i]$ associated with paths in P^t
$\hat{l}^t$	Real-time estimate of user's location at time t

involves making hundreds [10] or even thousands [11] of Wi-Fi scans at each fingerprint location, we make only four Wi-Fi scans at each location. While holding a smartphone in our hands, we stand at the location facing E, N, W, and S and we make a single Wi-Fi scan at each of those directions. Most Wi-Fi APs are dual-band and transmit and receive signals in both $2.4\,GHz$ and $5\,GHz$ Industrial, Scientific and Medical (ISM) frequency bands. Each of these bands are further divided into smaller sub-bands or channels. The center frequencies of these channels range from $2.412\,MHz$ to $2.484\,MHz$ and from $5.180\,MHz$ to $5.825\,MHz$. Therefore, there may be as many as $2M$ RSSI samples for each fingerprint and each of the four directions. We say as many as $2M$ samples, because one may not see all M APs and signals in both frequency bands at a given fingerprint location and direction. Hence, the fingerprint database or radio map can be represented by a $4N \times 2M$ matrix. Note that matrix entries corresponding to the signals that were not received at given locations and facing certain directions were set to $-100dBm$, corresponding to the smartphone Wi-Fi receiver sensitivity. The 2D or 3D coordinates of the fingerprint locations are stored in a separate table.

The terminology used to describe the way we apply the VA is provided in Table I. The smartphone carried by the user initiates a passive Wi-Fi scan every $1/f_s$ seconds. Upon the receipt of beacons that are periodically transmitted by Wi-Fi APs, the phone extracts and records information about the beacons received in both ISM frequency bands from each Wi-Fi AP within the wireless communication range. The information recorded includes the RSSI for each beacon received by the phone. This results in a sequence of Wi-Fi scan results $X^1, X^2, \ldots, X^T$. The details of our algorithm that sets binary flags F^t equal to 1 for each state reachable at time t and computes the minimum-cost paths P^t for those states and the corresponding minimum costs C^t, given F^{t-1}, P^{t-1}, and C^{t-1} are given in Algorithm 1. The cost function, which is a measure of dissimilarity between X^t and a fingerprint Y_i is defined as:

$$c(X^t, Y_i) = \sqrt{\sum_{\phi \in \{E,N,W,S\}} [d(X^t, Y_{i,\phi})]^2}, \qquad (1)$$

where

$$d(X^t, Y_{i,\phi}) = \sqrt{\sum_{m=1}^{2M} (x_m^t - y_{i,\phi,m})^2}, \qquad (2)$$

where x_m^t and $y_{i,\phi,m}$ are the m^{th} components of X^t and $Y_{i,\phi}$, respectively.

The algorithm described above is an incremental algorithm that computes real-time estimates of the user's location one sample at a time. It is applied from time 2, after an initialization step at time 1 is done. In some applications, the user location at time 1 is known and assumed to be one of the fingerprint locations, say $k \in \{1, 2, \ldots, N\}$. In that case, F^1 is an all 0s sequence of length N except for a 1 in the k^{th} position, P^1 is a sequence of length N, all whose elements are set to (0) except for the k^{th} element that is set to (k), and C^1 is an all-infinity (or the largest number the smartphone can represent correctly) sequence of length N, except for the k^{th} element which is set to $c(X^1, Y_k)$.

In other applications, the user location may not be known at time 1. In such cases, it is modeled as being any of the N fingerprint locations. In that case, F^1 is an all 1s sequence of length N, $P^1 = \{(1), (2), \ldots, (N)\}$, and

$$C^1 = \{c(X^1, Y_1), c(X^1, Y_2), \ldots, c(X^1, Y_N)\}. \qquad (3)$$

The computational complexity of Algorithm 1 is comparable to that of the KNN algorithm used in basic Wi-Fi fingerprinting localization systems. At each time t, both have to compute N terms of the form $c(X^t, Y_i)$ and they have to find the minimum among N numbers. The complexity of the first part is $\mathcal{O}(MN)$. Algorithm 1 has to additionally compare, for each trellis state i at stage t, the minimum costs associated with all trellis states j at time $t-1$ that have a transition to i. If each trellis state has on the average transitions to $\overline{N}$ states at the next stage, then this additional cost will be $\mathcal{O}(N\overline{N})$ comparisons of a pair of real numbers. Note that $N\overline{N}$ is actually the total number of transitions in one stage of the trellis. Therefore, the computational complexity of Algorithm 1 is $\mathcal{O}(N(M + \overline{N}))$.

All the experimental results presented in Section IV are based on the first assumption. That is, the starting location of the user is assumed to be known in all test scenarios.

IV. Testing and Performance Evaluation

In this section, we describe the experimental work we did to assess the effectiveness of the causal and non-causal indoor localization and tracking system we developed, and we present our empirical results along with an analysis of the results.

Algorithm 1: Estimate user location at time t in real-time

```
Input  : $X^t$, $F^{t-1}$, $P^{t-1}$, $C^{t-1}$
Output: $F^t$, $P^t$, $C^t$, $\hat{l}^t$
foreach $i \in \{1, 2, \dots, N\}$ do
    $c[t][i]$ = LARGEST POSSIBLE NUMBER ;
    $f[t][i] = 0$ ;
    foreach state $j$ that has a transition to state $i$ do
        if $f[t-1][j] = 1$ then
            if $c[t-1][j] < c[t][i]$ then
                $p[t][i] = p[t-1][j]$ appended by $i$ ;
                $c[t][i] = c[t-1][j]$ ;
                $f[t][i] = 1$ ;
                Update $F^t$, $P^t$ with $f[t][i]$, $p[t][i]$ ;
            end
        end
    end
    $c[t][i] = c[t][i] + c(X^t, Y_i)$ ;
    Update $C^t$ with $c[t][i]$ ;
end
$\hat{l}^t = i$ associated with the smallest $c[t][i]$ ;
```

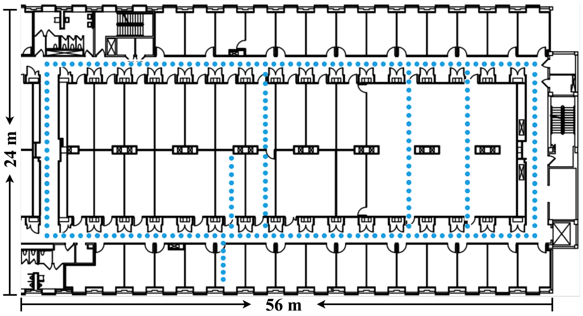

Fig. 3. Test points deployed in the office and laboratory building

A. Evaluation Methodology

We applied our algorithm to real-world settings in an office environment.

Test Site: We used an office environment in a multi-story office and laboratory building, which has brick exterior walls and a combination of drywalls and metal walls inside. As shown in Figure 3, 230 points, spaced 3 feet apart, were laid on the floor over half of the second floor that served as the service area for our indoor localization and tracking system. This covers an area of 56 $m \times 24$ m with corridors, offices, laboratories, and a few offices with cubicles. We used these points as locations for fingerprinting as well as performance testing. The 2D coordinates of these test points were computed relative to a test point at one corner of the service area. There are 44 Wi-Fi APs deployed in this building and 33 of them were used in the radio map. Signals from the remaining 11 APs could not be received in the service area. We performed all our experiments during regular working hours when people were in the building and busy with their daily activities. One important feature of this test site is its "one-dimensionality". In most cases, each test point (or fingerprint location) has only two immediate neighbors. Only eight points out of 230 have three immediate neighbors and two points have only a single neighbor. Many papers in the literature have used this type of test site.

Devices: We used two Android smartphones of different brands, referred to as Phone A and Phone B hereafter, in our experiments. We used Phone A running on Android OS 7.1.1 for offline data collection and the online testing phase. In addition, we used Phone B running on Android OS 9 in the online testing phase to check our algorithm's robustness to the use of different brands and models of phones. It does not matter exactly which brands and models we used, as phones change every few months.

Wi-Fi Fingerprint Database: As stated earlier, we used Phone A to build the fingerprint databases. Standing on top of each fingerprint location, the subject held the smartphone in the hand and prompted it to do one Wi-Fi passive scan while facing each of the four cardinal directions of East, North, West, and South.

Test Scenarios: A test scenario is defined as a trajectory at the test site along which a test subject walked at an average speed of approximately $1.4\, m/s$ while visiting a number of test points in a certain order and holding a smartphone in the hand. The test point sequence is planned prior to testing and the distance between two consecutive test points in the sequence might vary to realistically capture real pedestrian motion rather than artificial, constant speed trajectories. At each test point on the trajectory, we made only one Wi-Fi scan and had the phone record the results in the same format as had been used in the offline fingerprinting phase. A real-time location estimate at each test point on the trajectory was computed using Algorithm 1 based on the Wi-Fi scan made at that test point and all others made before reaching that test point. Algorithm 1 might backtrack and correct previous location estimates when the phone has collected additional Wi-Fi scan results. The proposed algorithm was evaluated using five different scenarios, each of which includes 93 test points and was repeated five times. Starting and ending locations of the trajectories varied, covering different areas of the test sites or walking in different directions. Some of the test points were visited more than once. For example, two or more consecutive test points along a trajectory might be the same, because the test subject stayed at that point for a while. Sometimes the test subject returned to a previously visited test point.

For performance evaluation purposes, we estimate $\underline{L}(t)$ at a finite number of discrete times $\Lambda = \{1, 2, \dots, T\}$. To estimate $\underline{L}(t)$ at any $t \in \Lambda$, we consider four possibilities:

1) Use Wi-Fi scan at t only. This is how the basic RADAR scheme [1] works. This method has been dubbed KNN in the performance figures that follow and in Table II.

2) Use Wi-Fi scans at times $1, 2, \dots, t$.

3) Use Wi-Fi scans at times $1, 2, \dots, t + \tau$, where τ is a fixed positive integer called the system lookahead. In this case, we do not insist on having the estimate for $\underline{L}(t)$ right after the Wi-Fi scan at time t is made. Rather, we are willing to wait

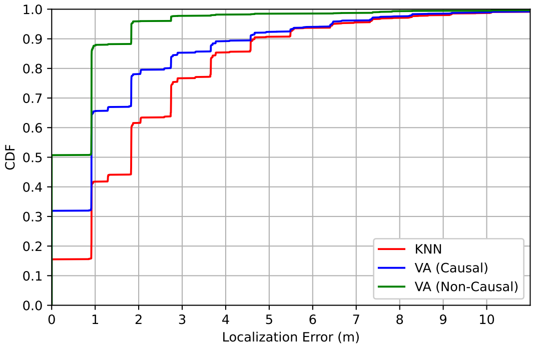

Fig. 4. CDF of localization error in the office environment

until τ additional Wi-Fi scans are made before we estimate $\underline{L}(t)$ based on this extended set of Wi-Fi scans.

4) Use Wi-Fi scans at times $1, 2, \ldots, T$. In this case, we essentially estimate $\underline{L}(t)$ for *all* $t \in \Lambda$ based on all Wi-Fi scans at times $1, 2, \ldots, T$.

The first two methods are examples of causal localization, the last two examples of non-causal localization. In what follows, we compare the performance of these four methods. We compute the mean localization error, root mean square (RMS) error and the standard deviation of the localization error. Since all the test points are on the same building floor, we focus on horizontal performance of the algorithm. Along with these commonly used performance metrics, we also include the 95-percentile and 50-percentile points on the Cumulative Distribution Function (CDF) of circular error, i.e. CE95 and CEP, respectively, according to the guidance from the international standard ISO/IEC 18305, Test and evaluation of localization and tracking systems [12].

B. Performance Comparisons

Table II compares the performance of three localization and tracking systems in the office environment. These systems are based on the KNN algorithm (with $K = 1$) corresponding to the first of the four methods described earlier, the second method, and the fourth one. These methods have been dubbed KNN, VA (Causal), and VA (Non-Causal). The unit for all numbers in the table is meters. Figure 4 shows the Cumulative Distribution Functions (CDFs) of localization error for the three approaches in the office environment. Both proposed approaches based on the VA achieved remarkable improvement over the KNN on almost all performance metrics. The improvement, however, is more pronounced for VA (Non-Causal). As for VA (Causal), the overall mean localization error reached $1.584\ m$ and the CEP was $0.915\ m$, which are 30% and 50% smaller than corresponding performance figures for KNN, respectively. The VA (Non-Causal) system, which continuously makes corrections to all location estimates along the user's trajectory, further reduced the overall mean localization error to $0.7\ m$, while its RMS error and the standard deviation of its error were both less than $1.5\ m$. It is worth noting that only 5% of the errors were larger than $1.83\ m$, and at least half of the location estimates generated by the system matched the test points perfectly, leading to a CEP of $0\ m$.

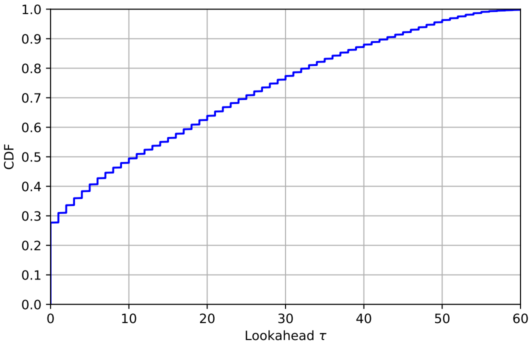

Fig. 5. CDF of $\tau_{\min}$ in the office environment

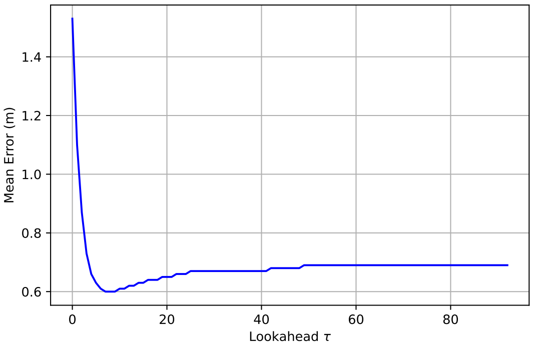

Fig. 6. Mean localization error as a function of lookahead τ

When we look at the performance results of the two phones, we can see that KNN worked better on Phone A than on Phone B in terms of all metrics. This can be explained by the fact that the Wi-Fi fingerprint database was built using Phone A. The RSSI measured by the two phones at the same time and location differ. The same is true for RSSI measured by the same phone at the same location, but at different times. Regardless of whether we look at VA (Causal) or VA (Non-Causal), the performance figures on both phones were enhanced with the use of the VA. However, it is surprising that our algorithm had even higher performance on Phone B than on Phone A for most metrics.

For any time $t \in \Lambda$ and any lookahead $\tau \in \{0, 1, \ldots, T-t\}$, let $\hat{L}_{t,t+\tau}$ denote the estimate of user location at time t computed by the VA based on Wi-Fi scan results through time $t + \tau$. Also, let $\tau_{\min}$ be the smallest number in the set $\{0, 1, \ldots, T - t\}$, such that $\hat{L}_{t,t+\tau} = \hat{L}_{t,T}$, for any $\tau \geq \tau_{\min}$. As defined, $\tau_{\min}$ is the minimum lookahead beyond which the location estimate $\hat{L}_{t,t+\tau}$ converges to $\hat{L}_{t,T}$. Figure 5 shows the CDF of $\tau_{\min}$ in our experiments. The mean of $\tau_{\min}$ is 16 time steps, and the 50- and 80-percentile points on its CDF are 11 and 33 time steps, respectively. In the context of the two use cases described in Section I (after the fact path estimation in the surveillance application and for the person visiting a trade show), it does not matter whether $\tau_{\min}$ is large or small.

TABLE II
PERFORMANCE COMPARISONS IN THE OFFICE ENVIRONMENT

Phone	Method	Mean Error	RMS Error	STD of Error	CE95	CEP
Phone A	KNN	2.014	2.899	2.084	6.394	1.824
	VA (Causal)	1.678	2.845	2.297	6.470	0.915
	VA (Non-Causal)	0.798	1.653	1.448	1.831	0.912
Phone B	KNN	2.484	3.345	2.241	7.316	1.830
	VA (Causal)	1.490	2.457	1.954	6.371	0.915
	VA (Non-Causal)	0.608	1.250	1.093	1.871	0.000
Overall	KNN	2.249	3.130	2.177	6.473	1.829
	VA (Causal)	1.584	2.658	2.134	6.444	0.915
	VA (Non-Causal)	0.703	1.466	1.286	1.833	0.000

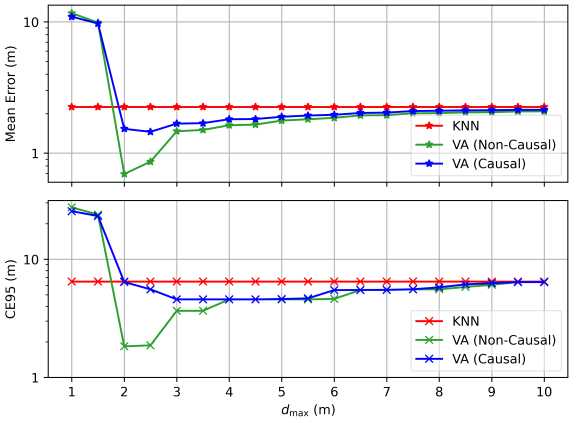

Fig. 7. Performance vs. $d_{\max}$ in the office environment

One can afford to wait and work with the entire time series of Wi-Fi scans from time 1 to time T to estimate the entire user path. However, there may be use cases where one is not allowed to use an arbitrarily large lookahead. In such cases, it is not desirable for $\tau_{\min}$ to get too large. Figure 5 has a number of implications. We discus two of them. First, it implies that 28% of the time, the VA (Causal) algorithm generates the same location estimate at time t that VA (Non Causal) does after observing all Wi-Fi scans through time T, i.e. $\hat{L}_{t,t} = \hat{L}_{t,T}$. Second, it implies that we need to use a lookahead of 49 time steps if we wish to ensure with 95% probability that we can generate a location estimate that is the same as $\hat{l}_{t,T}$.

Figure 6 shows the performance of the third method described at the end of the previous subsection. It shows that mean localization error is minimized for lookahead values $\tau = 7, 8, 9$ and then it starts increasing beyond $\tau = 9$. That is, some lookahead would be good, but too large a lookahead may not be as good. This may appear counter-intuitive at the first glance based on the results presented earlier that showed VA (Non-Causal) is superior to VA (Causal). The explanation is that method 3 with a fixed lookahead τ may result in a sequence of $\hat{L}_t$'s that does not correspond to any legitimate path in the trellis diagram. It may result in lower mean localization error than what is achievable with the second or fourth method, but we know that a path with jumps and wall breaches is not realistic.

In Section III, we suggested that points in a square lattice that are approximately within distance $v_{\max}/f_s$ of a given lattice point should be regarded as reachable from that lattice point during one Wi-Fi scanning period $1/f_s$, unless they are on the other side of a wall. The radius of the circle drawn with dashed red line in the left part of Figure 1 determines connectivity in our trellis diagram. It is not a hard and fast rule that the radius of this circle should be $v_{\max}/f_s$. It is worthwhile to study the effect of the radius of that circle on the accuracy of our proposed method based on the VA (Causal or Non-Causal). Let $d_{\max}$ denote the radius of the circle. It turns out that $d_{\max}$ is an important parameter in the operation of our VA-based system. If $d_{\max}$ is set too small, our proposed method (causal or non-causal) cannot keep up with the user's movements, because transitions to the neighboring points of a lattice point may be severely limited. This results in large localization errors. On the other hand, if $d_{\max}$ is on the order of the diameter of the service area for the localization system, the performance of the VA-based algorithms approach that of the KNN algorithm, which uses the "closest" fingerprint to Wi-Fi scan result X^t as $\underline{\hat{L}}(t)$. (Actually, it is safer to say that "if $d_{\max}$ tends to infinity, ...". The reason is that if the distance between two lattice points is close to the diameter of the service area, the presence of walls may make the shortest path between those points considerably longer than the diameter of the service area. In other words, crow's flights are not possible.) As $d_{\max}$ gets larger, so does the connectivity in the trellis diagram and the computational complexity of our method (causal or non-causal). Eventually, we reach a situation where every lattice point is reachable from all others. That's when the performance of our method (causal or non-causal) will be the same as that of the KNN. The use of a large $d_{\max}$ would make transitions to far away points possible, defeating the whole point of using the VA to prevent jumps. Therefore, there has to be a happy medium for the value of $d_{\max}$ that works best. This happy medium is typically a bit larger than $v_{\max}/f_s$. In the empirical results we presented earlier in this section, we used the same value of $d_{\max}$ for all experiments. The values of $d_{\max}$ used in the office environment was $2\,m$.

Figure 7 shows the performance of the VA (Causal) and VA (Non-Causal) systems as a function of $d_{\max}$ in the office environment. We have also shown the performance of the KNN

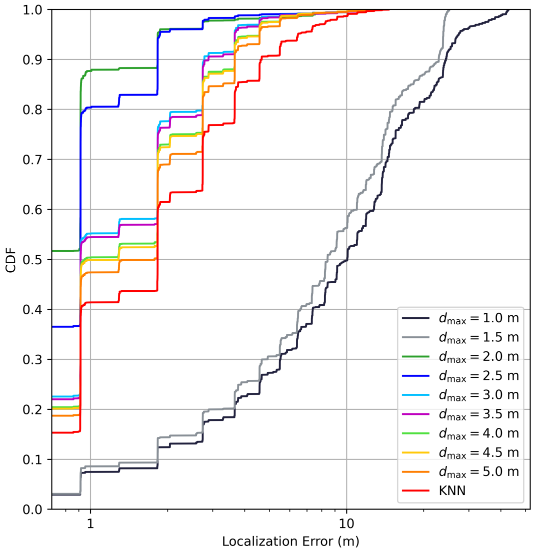

Fig. 8. CDF of localization error for different $d_{\max}$ values in the office environment

algorithm, which is independent of $d_{\max}$. The performance metrics examined in this figure are mean error and CE95. As expected, the figure shows that the performances is poor at small values of $d_{\max}$, but there is a sharp improvement when $d_{\max}$ reaches approximately 2 m. The performance continues to improve as $d_{\max}$ is increased until we get to the happy medium mentioned above. From that point on, the performance degrades as $d_{\max}$ is increased, even though not always monotonically. Figure 8 shows the CDF of localization error of the VA (Non-Causal) system in the office environment for different values of $d_{\max}$. It shows that $d_{\max} = 2\,m$ is the best choice, resulting in the highest curve. For reference purposes, we have also shown the CDF of localization error for the KNN algorithm. Note that $d_{\max} = 1\,m$ and $d_{\max} = 1.5\,m$ are bad choices making the VA (Non-Causal) system inferior to the KNN algorithm.

V. Conclusions

This paper presented the notion of non-causal localization. We developed one such indoor localization system in the context of Wi-Fi fingerprinting. The proposed method uses the VA as well as the knowledge of building floor plans. It is possible to use our system in the causal mode, where location estimates are generated in real-time, with fixed lookahead, and in a mode that uses all observations available through time T. We empirically evaluated the performance of our method in an office environment and compared it with the basic Wi-Fi fingerprinting method using the KNN algorithm as a baseline. Our empirical results suggest that our method, regardless of whether it is used in the causal or non-causal mode, outperforms basic fingerprinting. The improvement in localization accuracy achievable with the VA (Non-Causal) system, in particular, over basic fingerprinting is substantial. We also showed that using a fixed lookahead τ can lead to lower mean localization error than to use all observations through time T. We showed that the computational complexity of our method is comparable to that of the basic Wi-Fi fingerprinting method.

Even though we developed our method in the context of Wi-Fi fingerprinting, we believe the same techniques can be applied to other types of localization systems. In addition, it would be worthwhile to assess the performance of the proposed method in a truly two-dimensional service area covering not just corridors, but also all rooms/offices.

Disclaimer

Certain commercial entities, equipment, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

References

[1] P. Bahl and V. N. Padmanabhan, “Radar: an in-building rf-based user location and tracking system,” in *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 2, March 2000, pp. 775–784.

[2] C. Gentile and L. Klein-Berndt, “Robust location using system dynamics and motion constraints,” in *2004 IEEE International Conference on Communications (IEEE Cat. No.04CH37577)*, vol. 3, 2004, pp. 1360–1364 Vol.3.

[3] A. Viterbi, “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[4] G. D. Forney, “The viterbi algorithm,” *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[5] T. Kohri, “Location estimation method based on viterbi algorithm,” in *2009 IEEE 70th Vehicular Technology Conference Fall*, 2009, pp. 1–5.

[6] Z. Xiao, H. Wen, A. Markham, and N. Trigoni, “Lightweight map matching for indoor localisation using conditional random fields,” in *IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*, April 2014, pp. 131–142.

[7] J. Trogh, D. Plets, L. Martens, and W. Joseph, “Advanced indoor localisation based on the viterbi algorithm and semantic data,” in *2015 9th European Conference on Antennas and Propagation (EuCAP)*, April 2015, pp. 1–3.

[8] P. Wawrzyniak, S. Hausman, and P. Korbel, “Sequence detection of movement for accurate area based indoor positioning and tracking,” in *2015 9th European Conference on Antennas and Propagation (EuCAP)*, 2015, pp. 1–4.

[9] ——, “Area based indoor tracking algorithm based on sequence detection and maximum likelihood metrics,” in *2016 10th European Conference on Antennas and Propagation (EuCAP)*, 2016, pp. 1–3.

[10] M. Youssef and A. Agrawala, “The horus wlan location determination system,” in *Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys ’05. New York, NY, USA: Association for Computing Machinery, 2005, p. 205–218.

[11] K. Chintalapudi, A. Padmanabha Iyer, and V. N. Padmanabhan, “Indoor localization without the pain,” in *Proceedings of the Sixteenth Annual International Conference on Mobile Computing and Networking*, ser. MobiCom ’10. New York, NY, USA: Association for Computing Machinery, 2010, p. 173–184.

[12] “ISO/IEC 18305:2016, Information technology – Real time locating systems, Test and evaluation of localization and tracking systems,” 2016. [Online]. Available: https://www.iso.org/standard/62090.html