Reliable Geofence Activation with Sparse and Sporadic Location Measurements

Kien Nguyen
Department of Computer Science
University of Southern California
Los Angeles, CA, USA
kien.nguyen@usc.edu

John Krumm
Microsoft Research
Microsoft Corporation
Redmond, WA, USA
jckrumm@microsoft.com

***Abstract*—Geofences are a fundamental tool of location-based services. A geofence is usually activated by detecting a location measurement inside the geofence region. However, location measurements such as GPS often appear sporadically on smartphones, partly due to weak signals or privacy preservation. Users may restrict location sensing, or conserve energy, because sensing locations can consume a significant amount of battery. These unpredictable, and sometimes long, gaps between measurements mean that entry into a geofence can go completely undetected. In this paper we argue that short term location prediction can help alleviate this problem by computing the probability of entering a geofence in the future. Complicating this prediction approach is the fact that another location measurement could appear at any time, making the prediction redundant and wasteful. Therefore, we develop a framework that accounts for uncertain location predictions and the possibility of new measurements to trigger geofence activations. Our framework optimizes over the benefits and costs of correct and incorrect geofence activations, leading to an algorithm that reacts intelligently to the uncertainties of future movements and measurements.**

***Index Terms*—location-based services, geofence, sporadic locations, location prediction, decision theory, payoff matrix**

I. Introduction

Geofences are virtual geographic regions used to trigger certain actions upon entry or exit. A typical example is a region near a store where an advertiser may want to deliver ads to the phones of people in the region. When someone with a location-sensitive device enters a geofence, some action is automatically triggered.

One problem with geofences is they may fail to activate if there is no location measurement taken inside the geofence, even if the person passes through. This problem can occur whenever the gaps between measurements are large enough to miss a geofence, including when a person is walking or in a vehicle. With a high enough sampling rate, this is not a problem. However, geofence applications may not able to proactively trigger new measurements. One reason is when the location signal is too weak, e.g., inside a tunnel or surrounded by high buildings. Another reason is that user may only allow such applications to monitor location readings after a time delay due to, e.g., privacy concerns. Yet another reason is that sensing location, especially with GPS, drains a phone's battery. For instance, Liu *et al.* [1] estimate that running a GPS receiver continuously will drain a phone's battery in about six hours. Therefore, some power management programs of the device may restrict, e.g., the frequency of location sensing. This leads to conservative sensing, where measurements are often relatively far apart in time. As an example, Figure 1 shows a histogram of time spans between measurements for a random sample of 1000 users (described in detail later in Section IV). It shows that over half the points are separated by five minutes or more, making it easy to miss a geofence depending on the geofence's size and the user's speed. Our results in Section V show that the current practice of waiting for a measurement to trigger a geofence performs poorly.

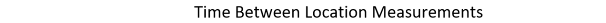


Fig. 1: The time between location measurements can be long enough to miss a geofence.

Short term location prediction can help alleviate this problem by giving the probability that the person will enter the geofence in the near future. However, there are two issues: (1) probability by itself may not be sufficient to make the optimal, benefit-maximizing decision of activating the geofence or not, and (2) a new measurement could arrive at any time, wasting prediction computations that go beyond that time.

In this paper we develop a novel framework that uses short term location prediction to solve the problem of triggering geofences with sparse and sporadic location measurements. Our framework takes into account both the predictive uncertainty for decision making and the potentially wasted computation issues. The fundamental approach is to use decision theory as a principled way to manage the trade-off between costs and benefits of acting or waiting. Decision theory is enabled by computing the probability that a user will intersect the geofence before the next measurement is available using a probabilistic location prediction method. This is a new approach for deciding whether or not to trigger a geofence, reflecting richer, more subtle reasoning than the traditional method of passively waiting for a point to appear inside the geofence. In addition, when we use location prediction, decision theory helps bridge the gap between the accuracy of location prediction and the benefit/penalty of geofence triggers in a principled way. We further reason explicitly about the temporally sporadic nature of location measurements by modeling their arrival times as Poisson distributed. This reasoning gives us a principled way to stop the prediction process, thus avoiding redundant predictions. We show how our approach is superior to the baseline technique of using only the given measurements without considering the possibility of missing a geofence. To the best of our knowledge, in the context of geofences, neither probabilistic location prediction, nor decision theory, nor reasoning about sporadic measurements has appeared in the research literature before.

For both privacy and energy conservation, our approach is designed to be simple enough to run on the user's local device rather than transmitting any location data. Likewise, it does not trigger any new measurements, relying instead on opportunistic measurements that are made available by other processes on the device.

Specifically, our new research contributions are:

- Probabilistically predicting geofence intersections
- Modeling costs/benefits of geofence activations with decision theory
- Reasoning about temporally sporadic location measurements with a Poisson distribution
- Extensive experiments over different settings for location measurements and algorithmic parameters

II. Related Work

While it seems obvious to use location prediction with geofences, there is surprisingly little research on the topic. In [2], Zimbelman *et al.* characterize the problem of trying to detect when a moving geofence (*e.g.* around a person) intersects a static location. Fattepur *et al.* [3] present a state transition algorithm for a GNSS chipset that does simple reasoning about geofences, but does not use location prediction. The work most closely related to ours is from Nakagawa *et al.* [4]. Their solution to the problem of missing geofences is to adaptively increase the location sampling rate when the user is getting closer to a geofence.

While there has not been much research on geofence intersection detection, there is a large literature on location prediction, which is one of the central components of our approach. A survey of location prediction approaches appears in [5]. In the realm of short term, probabilistic location prediction, a classic example is the Kalman filter [6], which creates a Gaussian-distributed prediction as part of its measurement update algorithm. The particle filter [7] also has a prediction step, as does the unscented Kalman filter [8].

III. Decision Theory with Location Predictions

A. Decision Theory

A geofence triggers some action when a user enters. However, a user's location typically has some uncertainty due to measurement noise or prediction error. This leads to uncertainty about whether or not the user is inside the geofence. We represent a user's two-dimensional location coordinates at time t as the vector $\boldsymbol{x}(t) = [x(t), y(t)]^T$, distributed according to the probability distribution $P_{\boldsymbol{X}(t)}\big(\boldsymbol{x}(t)\big)$, as in Figure 2. If the geofence region is represented by $\mathcal{R}$, the scalar probability that the user is inside the geofence is

$$p_{\mathcal{R}}(t) = \int_{\mathcal{R}} P_{\boldsymbol{X}(t)}\big(\boldsymbol{x}(t)\big) d\boldsymbol{x} \tag{1}$$

Based on $p_{\mathcal{R}}(t)$, the geofence can be programmed to either act or wait: acting means the geofence triggers some action, e.g. the delivery of an advertisement (ads) to the user, while waiting means nothing is triggered. A payoff matrix captures the value $V(t)$ of acting or waiting depending on whether or not the user is inside the geofence, shown in Table I. The value of acting when the user is inside $\mathcal{R}$ is β, which would normally be positive, reflecting the intended functioning of the geofence. The value of the two error conditions are acting when the user is outside (δ) and waiting when the user is inside (α). Both of these values would likely be negative. Waiting while the user is outside is the correct decision, but the payoff in this case would normally be zero. The exact values of the elements of the payoff matrix depend on the scenario. For ads, the values would depend on the cost of delivering an ad and the response rate. These values are normally proprietary and beyond the scope of this paper, although this would be an interesting extension of our work. We explore different payoff matrix settings in our experiments. Using a payoff matrix instead of raw prediction accuracy has two advantages. First, it allows us to express and evaluate the true costs of mistakes and successes. Second, by using costs, it lets the algorithm optimize for cost rather than raw accuracy, which can lead to different decisions.

		user state	
		in	out
decision	wait	α	0
	act	β	δ

TABLE I: The payoff matrix gives values of decisions depending on the state of the user.

The expected payoff values, given an activation decision, can be computed from $p_{\mathcal{R}}(t)$ and the payoff matrix:

$$\mathbb{E}\big[V(t) \mid \text{wait}\big] = \alpha p_{\mathcal{R}}(t) + 0(1 - p_{\mathcal{R}}(t)) \tag{2}$$

$$\mathbb{E}\big[V(t) \mid \text{act}\big] = \beta p_{\mathcal{R}}(t) + \delta(1 - p_{\mathcal{R}}(t)) \tag{3}$$

$$\mathbb{E}\big[V(t)\big] = \max\Big(\mathbb{E}\big[V(t) \mid \text{wait}\big], \mathbb{E}\big[V(t) \mid \text{act}\big]\Big) \tag{4}$$

The decision to wait or act corresponds to which has the larger expected value. This changes with time depending on $p_{\mathcal{R}}(t)$. In our scenario, after the first "act" decision, the geofence is deactivated, disallowing any acts for that user for some time.

This rule for waiting or acting, based on Equation 4, is a principled way to account for the costs and benefits of acting under uncertainty for geofences, and is our main contribution to the problem of geofence activation. The values from the payoff matrix allow the geofence owner (*e.g.* advertiser) to quantify the urgency of delivering a message to someone who should receive it (β), versus the cost of not delivering it to someone who should receive it (α), versus the cost of mistakenly delivering it to someone who should not receive it (δ). The probability $p_{\mathcal{R}}(t)$ accounts for measurement uncertainty, and would be especially applicable for less precise location sensing modalities, e.g. cell towers or WiFi. However, we are interested in further dealing with the problem of low sample rate data, when entering a geofence can be completely missed. For this we can use location prediction, described next.

B. Location Prediction

We can predict whether or not a user will be inside a geofence by predicting the user's location $\boldsymbol{x}(t)$. Referring to Figure 2, we reset the clock to $t = 0$ at the most recent measurement $\boldsymbol{x}_0$, thus $\boldsymbol{x}(0) = \boldsymbol{x}_0$. Then we make probabilistic predictions $p_{\mathcal{R}}(t)$ for $t > 0$. At each $t > 0$, we evaluate Equation 4, and we "act" the first time $\mathbb{E}\big[V(t) \mid \text{act}\big] > \mathbb{E}\big[V(t) \mid \text{wait}\big]$. We call this time $\hat{t}$. Once we have acted on the user (*e.g.* sent a message) for this geofence, we will not act again for that user on that geofence, or we may not act again until some preset time elapses, such as a few hours.

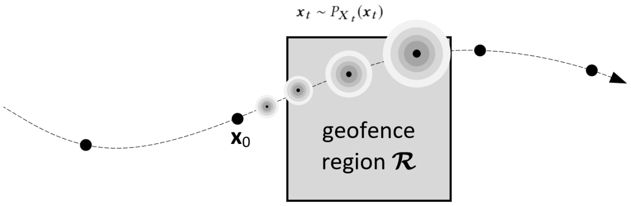


Fig. 2: $t = 0$ is the time of the most recent measurement. Our algorithm makes probabilistic predictions of location to infer the future probability of being inside the geofence.

Our framework accepts any type of probabilistic location prediction. That is, the prediction must produce a distribution $\boldsymbol{x}(t) \sim P_{\boldsymbol{X}(t)}\big(\boldsymbol{x}(t)\big)$ for $t > 0$. However, as we aim for mobile applications, we also prefer a light-weight model that can run on-device, even without sending measurements out of the device. Therefore, for our experiments, we predicted location with Gaussian processes (GPs) [9]. We create independent GPs for $x(t)$ and $y(t)$ to predict two-dimensional location $\boldsymbol{x}(t) = [x(t), y(t)]^T$.

Several choices are required for implementing a GP. We nominally assume a standard deviation of $\sigma_m = 3$ meters as the measurement Gaussian noise. We use a common squared exponential kernel for our GPs. A GP also depends on a mean function $m(t)$ which defines the expected mean values of the data points. Standard GPs are assumed to have a zero mean.

Although a GP can capture the trends of the user's movement to predict future locations, the short-term movement may not follow such trends. Therefore, we propose an adaptation of the GP for short-term location prediction, aiming to make the prediction rely more on the linear extrapolation of the most recent measurements. For this, we utilize the mean function $m(t)$ and set it to be the line going through the two latest points $x(t_{-1})$ and $x(t_0)$. We denote this as *GP + mean func*.

C. Sporadic Location Measurements

In Section III-B, we showed how to compute $\hat{t}$, which is the first time when the expected value of acting is predicted to exceed the expected value of waiting. However, $\hat{t}$ may be very large, especially if the geofence is a long distance from the user, leading to long, possibly infinite, computations of the location predictions for those geofences. In these cases, a new measurement could occur before the computed $\hat{t}$, causing much of the previous computation to become wasted. This section describes a principled way to suspend the prediction.

Assuming the timing of location measurements follows a Poisson process, we can compute the probability of receiving a new measurement as a function of time. A Poisson process is characterized by λ, which is the average number of events occurring in some predefined interval ΔT. For us, these events are location measurements, and the predefined interval is set arbitrarily to one minute. The parameter λ is called the event rate or the rate parameter. The Poisson distribution gives the probability of receiving at least one measurement by time t as $1 - e^{-\lambda t}$. Our algorithm stops making predictions when the probability of having at least one measurement is sufficiently high, *i.e.* when $1 - e^{-\lambda t} > 1 - \epsilon$, for some small value of ϵ, $0 < \epsilon < 1$, called *maximum prediction threshold*. This occurs at time $t^* = -\ln(\epsilon)/\lambda$.

IV. Experimental Data

A. Data Source

The location data came from a commercial aggregator [10] that ingests, cleans, and sells location data gathered from mobile phones. The data comes from individuals using their phone normally, occasionally running applications that trigger location measurements, e.g. weather, web browsing, or navigation. This data simulates a geofencing application that is not actively taking location measurements, instead relying passively on measurements triggered by other applications.

For each of 1000 randomly selected users with at least one data point, we extracted all their data for the date of 1 June 2019 to understand representative statistics on the quantity and frequency of location data that is normally available from a user. As we show below, the data from these 1000 users varied in terms of the number of data points.

For our experiments, we used the same data source from the same day, but this time we extracted data from the 1000 users with the *most* data points for that day. We refer to these two data sets as the "random" and "high density" sets. The random data set is used to understand representative statistics of available data, while the high density data is used to simulate various data densities, by controlled subsampling, to understand how our algorithm performs.

B. Data Statistics

A trajectory $\mathcal{S}$ of a user is a sequence of location measurements $\{\boldsymbol{x}(t_1), \boldsymbol{x}(t_2), \ldots, \boldsymbol{x}(t_{N_\mathcal{S}})\}$ where $N_\mathcal{S}$ is the number of measurements and $\forall i < N_\mathcal{S} : t_i < t_{i+1}$. High density means that, on average, the time gap $|t_i - t_{i+1}|$ is small.

In Section III-C we show how to compute a cutoff time for predictions assuming that timing of new location measurements is governed by a Poisson process. The sole parameter of the Poisson distribution is λ, which is the mean number of events over some unit time. For each of the 1000 random trajectories, we computed the maximum likelihood value of λ. The mode of this distribution of λ's occurs at 0.04, which corresponds to one measurement every 25 minutes.

We subjected each random trajectory to a statistical test to determine if the timing of the measurements was Poisson distributed. Using the chi-square test [11] for Poisson distributions, we found that 88.3% of the random users passed at the $p = 0.05$ level. This indicates that the Poisson distribution is appropriate for modeling the arrival time of location measurements in our data. For the high density data, none of the users passed the statistical test, but these trajectories are outliers, chosen for experimental advantages.

C. Data Pre-Processing

We processed the high density data to simulate the Poisson processes of the randomly chosen users. Our goal was to find, in the high density data, long sequences of measurements with temporally uniform sampling, since uniform sampling is convenient for down-sampling to measurement times that are Poisson distributed. Due to space limitation, details can be found in our extended version [12]. This process results in 530 total trajectories from 165 unique users. For these trajectories, we used the first five minutes for training the parameters of the GP and used the remaining data for testing.

In the test period (*i.e.* after five minutes) for each of these trajectories, we computed the first time it entered the geofence, t_{in}, and the first time it exited the geofence, t_{out}. Normally this required interpolation, where we assumed constant, straight line speed between temporally adjacent points.

We converted the latitude/longitude coordinates in each trajectory to local Euclidean coordinates (x, y) in meters. To simulate a Poisson process from our uniformly sampled trajectories, we sample points from the uniform trajectories with a Bernoulli process [13], which is the discrete-time version of a Poisson process.

V. Experiments

A. Evaluation metric

We first describe our evaluation metric to evaluate the effectiveness of our technique for the geofence decision problem. We propose a “realized value” score that is analogous to the expected value in Equation 4. The realized value is the actual payoff, as per the payoff matrix, of using an algorithm to make the act or wait decisions for geofences. Starting first with a single point below, we show how we aggregate the realized value over multiple points in a trajectory $\mathcal{S}$, multiple geofences on the map $\mathcal{R}$, and multiple trajectories in a test set.

We first calculate the realized value, called $V_{\mathcal{R},\mathcal{S}}(t_i)$, for each measurement $\boldsymbol{x}(t_i) \in \mathcal{S} \setminus \boldsymbol{x}(t_{N_\mathcal{S}})$. The last measurement $\boldsymbol{x}(t_{N_\mathcal{S}})$ is used as a pivot and is not evaluated.

Given t_{in} and t_{out} as the first timestamps that the user enters and exits a geofence $\mathcal{R}$, t_i and t_{i+1} as the timestamps of i-th and $(i+1)$-th measurements, and $\hat{t}$ as the predicted timestamp that the geofence should ”act” when we receive $\boldsymbol{x}(t_i)$, the realized value of the i-th measurement $V_{\mathcal{R},\mathcal{S}}(t_i)$ is defined as:

$$V_{\mathcal{R},\mathcal{S}}(t_i) = \begin{cases} \alpha, & t_{i+1} < \hat{t} \wedge t_i \leq t_{out} \leq t_{i+1}, \\ \beta, & \hat{t} \leq t_{i+1} \wedge \hat{t} \in [t_{in}, t_{out}], \\ \delta, & \hat{t} \leq t_{i+1} \wedge \hat{t} \notin [t_{in}, t_{out}]. \end{cases} \tag{5}$$

Elaborating on the compact notation in Equation 5, the condition $\hat{t} \leq t_{i+1}$ indicates we ”act” before the next measurement arrives. Then, we receive reward β or penalty δ if we act when the actual user state is inside (*i.e.*, $\hat{t} \in [t_{in}, t_{out}]$) or outside (*i.e.*, $\hat{t} \notin [t_{in}, t_{out}]$), respectively. This explains the β and δ payoffs in Equation 5.

In the case of $t_{i+1} < \hat{t}$, which means we ”wait” before the next measurement arrives, we further investigate different cases of $[t_{in}, t_{out}]$ and $[t_i, t_{i+1}]$ to decide the reward/penalty. These cases are illustrated in Figure 3. In case (1) where $t_{out} < t_i$, we ”wait” while the user is outside, thus, the payoff is 0. In cases (2) and (3) where $t_i \leq t_{out} \leq t_{i+1}$, we ”wait” while the user is inside, thus the payoff is α. In cases (4), (5), and (6) where have $t_{i+1} < t_{out}$, although the user might be inside, we still have a chance to act when the next measurement arrives. Therefore, the evaluation is deferred to the next measurements and the payoff of the current measurement is 0. In short, we only receive payoff α when $t_i \leq t_{out} \leq t_{i+1}$. This explains the α payoff in Equation 5.

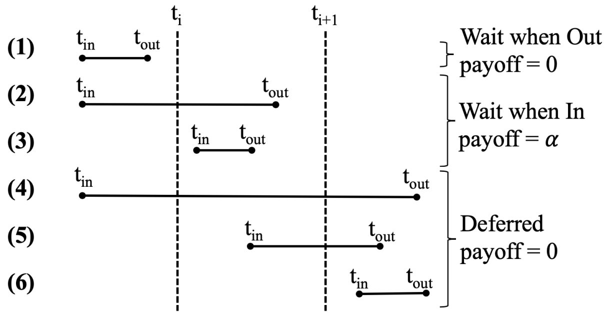


Fig. 3: Illustrations of cases when we ”wait” (*i.e.*, $t_{i+1} < \hat{t}$).

For a given trajectory, we should only act on a geofence a maximum of one time. Thus, for evaluation purposes, we stop calculating the realized value for this trajectory after the measurement $\boldsymbol{x}(t_k)$ for which we receive β (*i.e.* act when inside the geofence), and set $V_{\mathcal{R},\mathcal{S}}(t_i)$ of all following measurements to 0, *i.e.* $\forall i \in [k+1, N_{\mathcal{S}}-1] : V_{\mathcal{R},\mathcal{S}}(t_i) = 0$. This means we can receive the payoff β only once.

The realized value of a whole trajectory $\mathcal{S}$ is then the sum of realized values of all the measurements of $\mathcal{S}$:

$$V_{\mathcal{R},\mathcal{S}} = \sum_{i=1}^{N_{\mathcal{S}}-1} V_{\mathcal{R},\mathcal{S}}(t_i) \tag{6}$$

To avoid any bias towards any geofence positioning, we consider sum of $V_{\mathcal{R},\mathcal{S}}$ of a trajectory $\mathcal{S}$ over a set of multiple geofences as the score of $\mathcal{S}$:

$$V_{\mathcal{S}} = \sum_{\mathcal{R}} V_{\mathcal{R},\mathcal{S}} \tag{7}$$

Our set of experimental geofences is a grid described below.

The final realized value is the average of $V_{\mathcal{S}}$ over all N_T trajectories:

$$V = \frac{1}{N_T} \sum_{\mathcal{S}} V_{\mathcal{S}} \tag{8}$$

B. Baselines

Our baseline algorithm is called Passive Wait (PW), where the geofence passively waits to act until there is an actual location measurement inside. This appears to be the basis of the algorithm used by both iOS [14] and Android [15]. Formally, when receiving the most recent measurement $\boldsymbol{x}(t_0) = \big(x(t_0), y(t_0)\big)$ at time $t = t_0 = 0$ with the measurement noise defined by the standard deviation σ_m, then for each $t < t^*$, the PW method makes a prediction:

$$\boldsymbol{x}(t) \sim \mathcal{N}\big(\boldsymbol{\mu}(t), \boldsymbol{\Sigma}(t)\big) = \mathcal{N}\left(\begin{bmatrix} x(t_0) \\ y(t_0) \end{bmatrix}, \sigma_m^2 I\right) \tag{9}$$

where I is the 2×2 identity matrix. Although this is a simple approach, it appears to be the most sophisticated existing algorithm for geofences, both in practice and in research.

C. Experiment Setup

We present several experiments on 530 trajectories from 165 unique users, processed from high-density data described in Section IV. We experiment with a thorough range of experimental parameters to show our algorithm performs well in a variety of operating conditions. Due to space limitation, some experiments are only shown in the extended version [12].

For each trajectory, we created a grid of geofences centered at the last training point. The grid covered the entire bounding box of the testing data of the trajectory. In order to account for predictions beyond the bounds of the trajectory, we expanded the grid on all four sides by 18 kilometers. This expansion assumed a maximum driving speed of 30m/s (or 108km/h) and a duration of 600 seconds. The choice of 600(s) is explained in our discussion of ϵ in Section V-D2.

Each grid cell is considered as a geofence with size $L \times L$. We tested with sizes $L \in \{500, \mathbf{1000}, 1500, 2000, 2500\}$ meters for square cells. The bold value indicates the default value used in the experiments where this parameter is fixed. With up to 13,824 geofences, combined with the 530 trajectories, this gives a test size of over 7 million trajectory-geofence pairs.

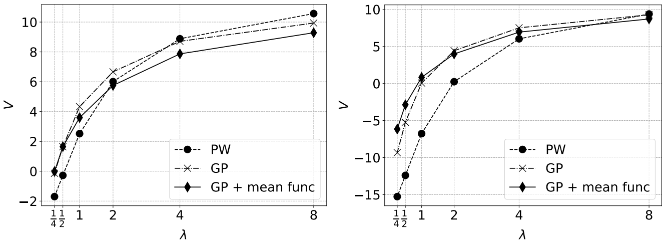
Each trajectory is sub-sampled with $\lambda \in \{0.25, \mathbf{0.5}, 1, 2, 4, 8\}$ with a Poisson time interval ΔT of one minute. Roughly speaking, these values of λ mean we expect to receive one measurement from every 4 minutes ($\lambda = \frac{1}{4}$) to every 7.5 seconds ($\lambda = 8$). The maximum prediction threshold ϵ is set to $\epsilon \in \{0.1, \mathbf{0.2}, 0.3, \ldots, 0.9\}$.

Two settings for the payoff matrix is considered to represent two real-world use cases: the *ads* matrix where $(\alpha, \beta, \delta) = (-\frac{1}{2}, 1, -\frac{1}{4})$; and the *alert-zone* matrix where $(\alpha, \beta, \delta) = (-2, 1, -\frac{1}{4})$. For the ads case, the geofence messenger wants to deliver an ads to users inside the geofence. The alert-zone case involves a messenger who wants to deliver an important message, e.g., a safety warning. While these values were set based on our own reasoning to mimic these scenarios, we assume the payoff matrix is given or can be learned.

D. Experimental Results

1) Varying Poisson λ: Focusing on the sporadic setting, we first evaluate the methods for different λ values. Figure 4 shows the realized values V when λ varies and other parameters are fixed, for both ads and alert-zone payoff matrices. Roughly speaking, each data point in a result graph shows the average dollar amount per trajectory that one model achieved over all geofences.

The general observation is that GP-based algorithms greatly outperform PW, especially with smaller λ, *i.e.*, more sporadic. This confirms our hypothesis that for sporadic measurements, using principled decision theory with a proper location prediction method brings significant improvement.



(a) Advertising

(b) Alert-zone

Fig. 4: The realized value with different values of λ

The improvement is even more prominent in the case of the alert-zone payoff matrix in Figure 4b, where the penalty is higher if the messenger waits until the user is inside the dangerous area (*i.e.*, larger $|\alpha|$). The reason is that we tend to act more readily when α is larger, while PW does not. We can gain more insight into this improvement by investigating our decision making process further. We decide to act when $\mathbb{E}\big[V(t) \mid \text{act}\big] > \mathbb{E}\big[V(t) \mid \text{wait}\big]$. After expanding and rearranging with the fact that typically $\delta + \alpha - \beta < 0$:

$$p_{\mathcal{R}}(t) > \frac{\delta}{\delta + \alpha - \beta} \tag{10}$$

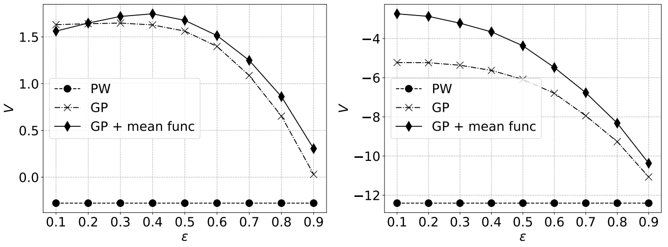
When $|\alpha|$ increases with fixed δ and β, the value of the right hand side of Equation 10 becomes smaller, which means that we might act with a smaller probability $p_{\mathcal{R}}(t)$. Therefore, an ads act needs a larger $p_{\mathcal{R}}(t)$ than an alert-zone to act, because missing an alert-zone act is is more costly.

When location measurements are frequent (*i.e.*, larger λ), prediction does not offer much benefit. This is understandable because with frequent measurements, there is a much higher chance a measurement arrives when the user is inside the geofence. Therefore, the prediction might be unnecessary.

The adaption with a linear mean function offers a higher realized value in the alert-zone setting compared to the standard GP. This improvement is also shown in other experiments.

2) Varying Prediction Threshold ϵ: Next, we consider the effect of the maximum prediction threshold ϵ. Recall that $t^* = -\ln(\epsilon)/\lambda$. For a fixed λ, a smaller ϵ leads to a larger t^*. In our most sporadic setting, where $\lambda = \frac{1}{4}$, and most conservative threshold, $\epsilon = 0.1$, we need to make predictions up to $600s$ ahead. That explains why $600s$ is used in our expansion of the geofence grid discussed in Section V-C.

Figure 5 shows the realized values for different values of ϵ when λ is fixed at $\frac{1}{2}$. The general trend is that smaller values of ϵ give larger realized values. With larger ϵ, the realized value becomes smaller and closer to the value obtained by the PW, which does not make predictions, thus, is not affected by the change of ϵ. In general, using a smaller ϵ is more conservative, because it leads to longer prediction times, lessening the chance that we will terminate the location prediction before the next measurement arrives.



(a) Advertising

(b) Alert-zone

Fig. 5: The realized values when varying ϵ

t^* (s)	194	277	600	3600	36,000
Time (s)	0.38	0.53	1.08	6.12	60.52

TABLE II: Time to calculate $p_{\mathcal{R}}(t)$ for all $0 < t < t^*$

Another important effect of ϵ is that the suspension of the prediction greatly reduces computation time. For all $0 < t < t^*$, we need to compute the integrals on the predicted location distributions to compute $p_{\mathcal{R}}(t)$ as in Equation 1. Table II shows the time to compute $p_{\mathcal{R}}(t)$ for all $0 < t < t^*$, for different values of t^* with $\lambda = \frac{1}{2}$, for a single geofence, and running with a single thread on a personal computer. While all these values of t^* result in similar realized values, with $\epsilon = 0.1$ and 0.2, we can suspend the prediction at $t^* = 194$s and 277s, and the computation time is just 0.38s and 0.53s, respectively. Any computation after these values of t^* can be considered as wasted computation. Without this principled approach to identify t^*, one could arbitrarily choose some values for t^*, which would result in an almost linear increase in the computation time.

VI. Conclusions and Future Work

This paper proposes a novel framework for the problem of using geofences with sparsely sampled, sporadic data, which may increase the chances of missing a geofence. The framework uses short-term probabilistic location prediction and decision theory to decide when to activate a geofence. Our framework also uses a Poisson distribution to model arrival time of location measurements as an approach to stop the prediction process. While our algorithm consistently outperforms the baseline over a variety of settings, there are still opportunities for further work in this area. Example directions include investigating moving geofences; or better method to decide when to trigger, such as such as when entry into a geofence reaches a certain confidence level.

References

[1] J. Liu, B. Priyantha, T. Hart, H. S. Ramos, A. A. Loureiro, and Q. Wang, "Energy efficient gps sensing with cloud offloading," in *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*, 2012, pp. 85–98.

[2] E. G. Zimbelman, R. F. Keefe, E. K. Strand, C. A. Kolden, and A. M. Wempe, "Hazards in motion: Development of mobile geofences for use in logging safety," *Sensors*, vol. 17, no. 4, p. 822, 2017.

[3] M. B. Fattepur, G. Sharvani, and J. B. Huttanagoudar, "A solution to improve the performance of geofence enabled gnss chipset," in *2016 International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS)*. IEEE, 2016, pp. 112–116.

[4] T. Nakagawa, W. Yamada, C. Doi, H. Inamura, K. Ohta, M. Suzuki, and H. Morikawa, "Variable interval positioning method for smartphone-based power-saving geofencing," in *2013 IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. IEEE, 2013, pp. 3482–3486.

[5] C. Cheng, R. Jain, and E. van den Berg, "Location prediction algorithms for mobile wireless systems," in *Wireless internet handbook: technologies, standards, and application*, 2003, pp. 245–263.

[6] R. E. Kalman *et al.*, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[7] A. Doucet, N. De Freitas, and N. Gordon, "An introduction to sequential monte carlo methods," in *Sequential Monte Carlo methods in practice*. Springer, 2001, pp. 3–14.

[8] E. A. Wan and R. Van Der Merwe, "The unscented kalman filter for nonlinear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*. Ieee, 2000, pp. 153–158.

[9] C. K. Williams and C. E. Rasmussen, "Gaussian processes for regression," in *Proceedings of the 8th International Conference on Neural Information Processing Systems*, 1995, pp. 514–520.

[10] SafeGraph. Safegraph. [Online]. Available: https://www.safegraph.com/

[11] J. H. Zar, *Biostatistical Analysis*, 4th ed. Upper Saddle River, NJ: Prentice Hall, 1999, pages 575–578.

[12] K. Nguyen and J. Krumm. (2022) Reliable geofence activation with sparse and sporadic location measurements: Extended version. [Online]. Available: https://arxiv.org/abs/2204.00714

[13] M. Bonakdarpour. (2016) Poisson process: The limiting case of the bernoulli process. [Online]. Available: https://stephens999.github.io/fiveMinuteStats/bernoulli_poisson_process.html

[14] iOS. Monitoring the user's proximity to geographic regions. [Online]. Available: https://developer.apple.com/documentation/corelocation/monitoring_the_user_s_proximity_to_geographic_regions

[15] Android. Create and monitor geofences. [Online]. Available: https://developer.android.com/training/location/geofencing