GFG-Assisted Human Tracking using Smart Phones

Yingchi Mao† and Jie Wu*
†College of Computer and Information, Hohai Univesity, China
*Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122
Email: maoyingchi@gmail.com, jiewu@temple.edu

***Abstract*—Mobile phones are increasingly being equipped with hardware and software services that allow them to determine their locations; however, support for building location-based applications remains a challenging problem. The most widely used localization technology in smart phones is GPS, but it rarely works indoors and provides low energy efficiency. Cell tower-based localization is widely available, but can provide very poor accuracy without a fingerprint profile. WiFi localization, when available, provides reasonable accuracy, but is also much less effective in other areas. Constandache et al. proposed an Escort system to assist in localizing and tracking others in a public place without requiring either GPS, WiFi, war-driving, maps, or floor plans. However, the Escort system may route one person on a long path even though the person being tracked may be close by. In this paper, we will investigate the problem of finding better tracking paths in the Escort system. We propose a GFG routing assisted human tracking algorithm to reduce the length of the tracking path for every pair of users using smart phones in mobile social networks. Through adding one seeker in the Escort system, whose main function is to find better paths between any pair of two intersections by applying the GFG routing algorithm, the localization and tracking algorithm in the Escort system is more effective than the original algorithm. Finally, we conduct simulations of our proposed algorithm at the main campus of Temple University with different numbers of mobile users and different duration times. The simulation results show that the human tracking performance has been greatly enhanced.**

***Index Terms*—Escort systems, GFG-Routing, Human Tracking, Smart phone**

I. Introduction

With the proliferation of sensor-equipped smart phones, many location-based mobile services and mobile sensing applications have become a reality. Many location-based applications need to periodically probe the device's sensor to record the time-ordered position sequences, and then they process the sequences to locate other users for navigation services [1]. In these location-based applications, localizing and tracking nearby users is an important step towards reducing the time it takes for people to find others. As others have noted, the increasing pervasiveness of commodity smart phones that can provide localization estimates using a variety of sensors – GPS, WiFi, and/or cellular triangulation – opens up the attractive possibility of using position samples from users' phones at a fine spatiotemporal granularity [2]. While GPS provides highly accurate location estimates, it rarely works indoors. Furthermore, its accuracy degrades in urban canyons, and the energy consumed by GPS devices is a significant deterrent. When WiFi localization or war-driving is available, reasonable accuracy can be provided in an urban environment.

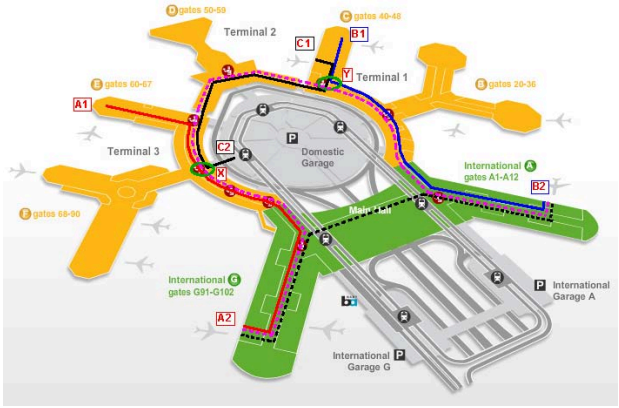


Fig. 1. Example of the localization and tracking problem in the Escort system.

However, it also is much less effective in other areas [3]. Cellular triangulation is widely available, but can provide very poor accuracy without a fingerprint profile or outside city centers.

On the other hand, people spend lots of time doing indoor activities. As a result, many of the most common social interactions occur in indoor environments [3]. For example, in [4], the authors consider localization and tracking with a hypothetical scenario. In a big conference hotel, one person A wants to locate, and can be navigated to, a specified person B without person B's precise location information. In that public environment, GPS, WiFi, and cell-tower triangulation localization can be ineffective. To solve the localization and tracking problem in that scenario, Constandache et al. developed a navigation system, called Escort, that can localize and route a person A to a specified person B in human populated public settings, such as airports, shopping malls, libraries, museums, and universities [4].

In the Escort system, each mobile phone, called an Escort client, is equipped with accelerometer and compass measurements. Mobile phones capture users' $movement\ traces$ by using those sensors. When one person A encounters another person B, each phone records these $encounters$ with a corresponding time stamp. A $< movement\ trace, encounter >$ is periodically uploaded to the Escort Server. The Escort server creates a trail graph, composed of users' positions and paths, using the $< movement\ trace, encounter >$ information. With the trail graph, person A can be routed to person B. The situation is depicted in Figure 2.

In the Escort system, tracking a person just depends on their movement trails and encounters, which may result in a long tracking path from person A to B, even if person B may be close to A. For example, suppose that there are three persons, A, B, and C, walking in the San Francisco International Airport. Person A walks from the boarding area E of Terminal 3 to the boarding area G of the International Terminal (solid red line from A_1 to A_2 in Fig. 1). Person B walks from the boarding area C of Terminal 1 to the boarding area A of the International Terminal (Solid blue line from B_1 to B_2). Person C walks from the Domestic Terminal 3 Station to the boarding area C of Terminal 1 (Solid black line from C_1 to C_2). As Fig. 1 shows, Person A and C encounter each other at the position **X** of Terminal 3, and Person B and C meet at the position **Y** of Terminal 1. When person A expresses an interest in navigating to B, the Escort server can create a route path based on their trails. Person A can be routed back to the position **X**, **Y** and the boarding area A of the International Terminal along the purple dash line with the original routing algorithm. In fact, there is one shorter path, which can be navigated from person A to person B – the dashed black line on the map. It is obvious that the tracking performance in the Escort system is of low efficiency because the Escort server is not aware of a possible trail along the shorter path.

In this paper, we will investigate the problem of having a better tracking path in the Escort system. We propose a GFG routing assisted human tracking algorithm to reduce the length of the tracking path between every pair of people in mobile social networks. In the original Escort system, we add one seeker, who stores the map or floor plan in his/her smart phone. The seeker is used to find better paths between any two intersections by applying the GFG routing algorithm in the map. When the seeker has found a new path after applying the GFG routing algorithm, it immediately reports the new path information to the Escort server. The server can compute the shorter routing path by merging the GFG routing path with the current trail graph. In addition, if there is no specific explanation given, $routing$ and $tracking$ express the same meaning in this paper.

The main contributions of this paper are listed as follows:

1) We propose a new human tracking algorithm by applying the GFG routing method in mobile social networks, which can provide the fundamental support for many location-based or location-aware applications.
2) We analyze the properties of the proposed tracking algorithm and give the complexity analysis.
3) A comprehensive comparison experiment is conducted. Simulation results show that the proposed GFG route assisted tracking algorithm can achieve a higher tracking performance. The average length of the tracking path after applying the GFG route, is much shorter than that of the original tracking path, and has a performance close to that of the shortest path.

The remainder of the paper is organized as follows. In section II, we introduce the related work. Then, an overview of the Escort system, the original tracking scheme and the problem formulation are presented in Section III. Then, we propose the GFG routing assisted human tracking algorithm in detail and analyze its properties. In Section V, we conduct simulations of the proposed algorithm at the main campus of Temple University with a varying number of trails, a varying duration times, and a varying number of seekers. The conclusions are drawn in Section VI. In subsequent discussion, we use "$user$" for "$person$".

II. Related Work

A. Localization

In recent years, there has been a lot research done in localization technology, which can be grouped into three branches. The first branch focuses on the tradeoff between energy efficiency and location accuracy [5], [6], [7], [8]. Although GPS can provide high accuracy, the energy consumption is a significant challenge for mobile devices. Alternative WiFi or GSM-based schemes offer longer battery life, but at the expense of lower accuracy. The second one proposes collaborative methods, like combining GPS, WiFi, and/or GSM, so that multiple devices can determine the position of mobile devices [3], [9]. The third one is to propose methods that identify logical locations, as opposed to physical coordinates [10], [11].

Previous localization solutions considered deploying radios or specialized hardware (e.g., GPS, WiFi beacons, and cell tower triangulation) in the environment to assist localization. The user's location can be estimated based on the overhead signals and the collected data during the calibration phase. Cricket [12], VOR [13], and Pinpoint [14] rely on these techniques. Radar [15], Active Campus [16], and PlaceLab [17] rely on access points in the public environments to enable localization. These solutions require calibrating WiFi signal strengths at many physical locations. The calibration process is time-consuming and may not scale over large areas. In addition, some research applies floor plans and/or maps to assist in user localization. CompAcc [18] is an outdoor localization scheme that builds a user trail similar to Escort. Authors in [19] rely on a floor plan coupled with WiFi war-driving and inertial sensors to enable localization.

Unlike the above localization schemes, Escort requires minimal hardware support, compass and accelerometer sensors in off-the-shelf smart phones, and a beacon [4]. In addition, Escort only relies on trails and encounters to achieve localization and provides tracking directions between users.

B. Geographic Routing

Geographic routing is an attractive routing protocol in the location-based applications because it has lower route discovery overhead than the topology-based protocols. In geographic routing, it is assumed that each node is aware of its own position and the positions of its neighbors, and also that the source node is aware of the destination's position.

In the greedy routing algorithm [20], each node forwards a message to its neighbor that is closest to the destination node.

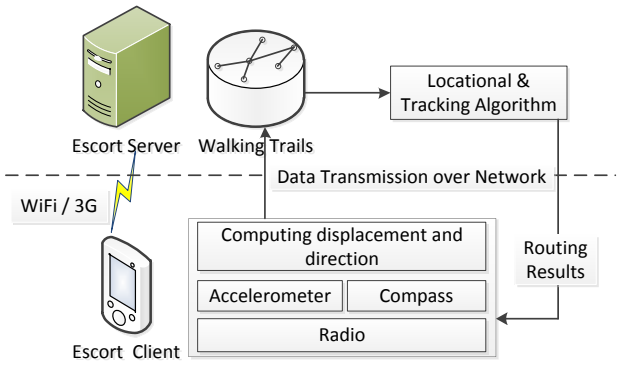


Fig. 2. The overview of the Escort system [4].

Its major drawback is the high failure rate. However, it has a close performance to that of the shortest path algorithm. Another geographic routing algorithm is called Face [21], which guarantees message delivery in the connected graphs. Most efficient geographic routing protocols are greedy-face combinations. In GFG [22], and its variant GPSR [23], a message starts with greedy routing until it reaches a local minimum. The algorithm then changes to face routing mode, which forwards the message along the perimeter of the face that is next to the local minimum in the direction of the destination. Face routing mode stops and reverts back to greedy routing when the message is forwarded to a node that is closer to the destination than the local minimum. GFG switches between greedy mode and face mode in order to make sure that the message is continuously getting closer to the destination.

To reduce transmissions, Datta et al. [24] proposed to run GFG in its face routing mode on the network of internal nodes, which is defined as a connected dominating set (CDS). GOAFR [25] and GOAFR+ [26] use a distance-bounded face traversal. This traversal iteratively traverses both sides of the face for a bounded distance with the right-hand rule and the left-hand rule, respectively. It also increases the bound if the condition to return to greedy forwarding mode is not satisfied after some iterations. The resulting paths of GOAFR and GOAFR+ are asymptotically optimal and are, on average, the shortest among the proposed geographic protocols.

III. Preliminaries

In this section, we briefly describe the overview of the Escort system, and we give our model and problem formulation.

A. The Main Idea of the Escort System

The proposed Escort system by Constandache et al. [4] is of client/server architecture, as shown in Fig. 2. We briefly introduce the client component and the server component.

The Escort client is a smart phone that is equipped with an accelerometer and compass. The accelerometer is used to record the number of steps and the speed. When multiplied by the user's step size, the smart phone can compute the user's displacement. The compass readings offer the direction of movement. The user's trail can be expressed as a sequence of the $< displacement, direction, time >$ tuples. The smart phone periodically uploads the trail information to the Escort server over a WiFi/3G wireless connection. In addition, the mobile phone can detect other users by using audio signals if one user encounters others. If one user detects another user, the two users can log this intersection and upload it to the Escort server. Thus, the Escort server obtains all of the users' trail information and creates a trail graph.

If one user A wants to locate the user B, the Escort server computes the tracking path, which routes A back to B in the trail graph. There are three steps: 1) creating a trail graph; 2) pruning the trail graph by using a pruning heuristic; 3) pruning the graph by running the Floyd-Warshall algorithm. In a trail graph, the edges are segments of user trails, while the vertices are either spatial intersections or the current user locations [4]. Fig. 3(a) shows a trail graph after tracking 8 users for 10 minutes, where the sample interval is 5 seconds.

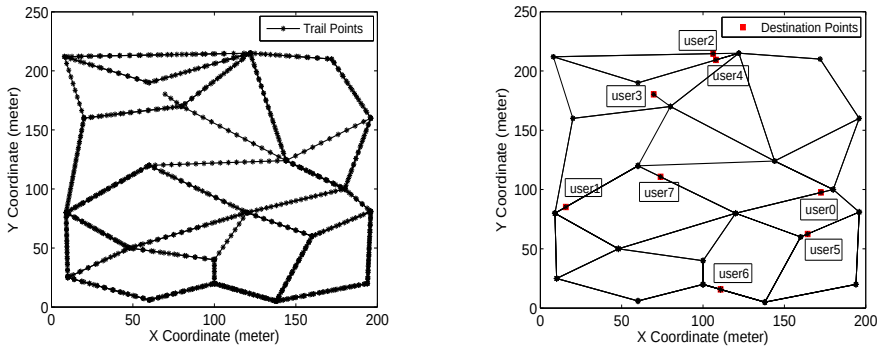
To maintain such a complex graph and compute the routes over it efficiently, the Escort server runs a graph pruning heuristic for every pair of user trails. It is assumed that two users' trails, $Trail_A$ and $Trail_B$, intersect with each other at several positions. The intersections are at different distances, with respect to each user. The pruning heuristic selects the closest intersections for both users A and B, respectively. The two intersections and the two paths joining them are retained in the trail graph, while other intersections are eliminated. Thus, the resulting graph is a fully connected graph, as shown in Fig. 3(b). Fig. 3(b) is the resulting graph of Fig. 3(a) after applying the pruning heuristic.

To further reduce the complexity of the resulted graph, the graph is then pruned again by applying the Floyd-Warshall algorithm and by keeping the graph of the shortest paths between users. The new, smaller graph is created.

In the real application, the practical challenge is that Escort may route a human on a long path, even though they may be close, because the trail graph grows large. Although the Escort system applies pruning heuristics to reduce the computation complexity, there are some limitations. In this paper, we will investigate the problem of obtaining a better routing path by merging the GFG-based routing path with the trail graph.

B. Analysis

The original tracking algorithm in the Escort server has some drawbacks. Firstly, the computational complexity of the Floyd-Warshall algorithm is $O(n^3)$, where n is the number of nodes in the trail graph. When the trail graph grows over time or the number of users increases, the computational overhead increases. Secondly, due to being unaware of the real map, the Escort server may miss a possible shorter path along the shortest path. For example, Fig. 4(a) is the three users' trails in the resulted graph after applying a pruning heuristic. If the original routing algorithm is being applied, the resulting routing path is $ABCDEHGLKRQPO$, as shown in Fig. 4(b). Obviously, this routing path is too long. If one seeker knows the map information, he can find the shorter path by



(a) The trail graph after tracking 8 users for 10 minutes.

(b) The resulted graph after applying the pruning method.

Fig. 3. Trail graph example.

applying the GFG-routing algorithm. If the new path is found and added to the trail graph, the server can compute the shorter routing path, $ALKRO$, as shown in Fig. 4(c).

C. Problem Formulation

There are N users walking in a public environment, e.g, a shopping mall, airport, convention center. The Escort system can provide tracking directions to user A for navigation to user B. There are two major phases during tracking: data collection and user routing. In our model, all users have a mobile phone equipped with sensors (e.g., an accelerometer and compass) and are walking in the random viewpoint model. All mobile phones register the Escort service and periodically report its sensors' readings to the server. In addition, one seeker, who has a mobile phone that is storing the map information of the public area, is added to the Escort system. The mobile phone is also equipped with the accelerometer and compass. The seeker is applied to find the new path based on the GFG-routing algorithm. The notations are listed in Table I.

In our model, there are several constraints:

1) There are one or multiple beacons in the Escort system that are applied to calibrate the position error and trail drift.
2) Due to our focus on efficient tracking, the position estimates the errors of users that are ignored. That is to say, it is assumed that the captured position information of the users is correct.
3) If one seeker finds a new path, it immediately uploads the path to the Escort server. The server stores all of the GFG routing paths.
4) It is assumed that one person should encounter at least one other person while tracking others.

With the scenario and assumptions, our objective is to find the shorter tracking path while navigating from A to B, compared to the path that was computed by the original routing algorithm in the Escort system.

TABLE I
LIST OF NOTATIONS

Notations	Meaning
G	the trail graph
V	set of nodes
E	set of edges
U	set of users' trails
pos_i	the current position of user u_i
$intersect_{ij}$	the intersection position of u_i and u_j
$P_{original}$	the tracking path by applying the original tracking algorithm
P_{GFG}	the routing path by applying GFG routing algorithm
$P_{GFG-new}$	the new found routing path by applying GFG routing algorithm in each interval
$P_{GFG-included}$	the tracking path including one of GFG routing paths
$P_{GFG-assisted}$	the tracking path by applying GFG-assisted tracking algorithm
$L_{original}$	the length of one $P_{original}$
$L_{GFG-assisted}$	the length of one $P_{GFG-assisted}$
$N_{GFG-included}$	the number of $P_{GFG-included}$
$N_{GFG-assisted}$	the number of $P_{GFG-assisted}$

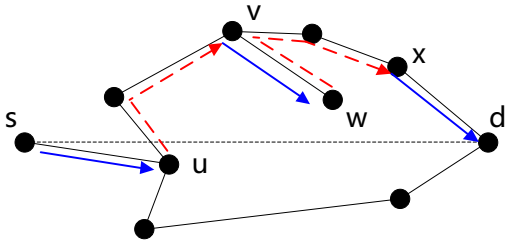
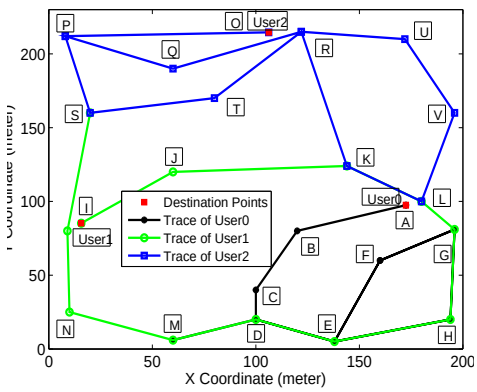


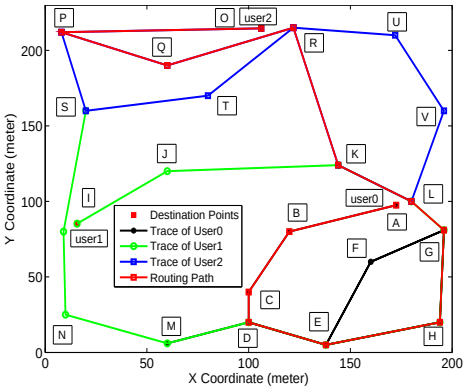
Fig. 5. Example of GFG routing algorithm.

IV. GFG-ASSISTED TRACKING SCHEME

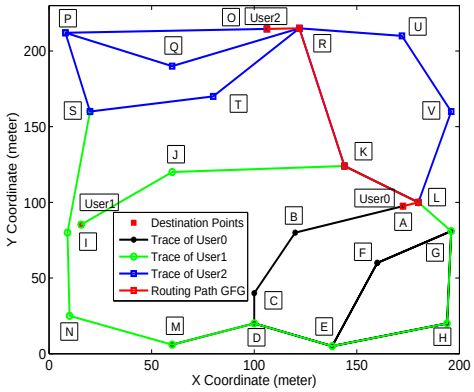
In order to avoid the drawbacks of the original routing algorithm, we propose a GFG-assisted human tracking algorithm. In our algorithm, one seeker is added to look for the new path by using the GFG routing algorithm in the specific area. The seeker owns a mobile phone that is equipped with the accelerometer and compass, and the area map has been downloaded to the phone. After the new path is found, it is uploaded to the server and is then merged with the trail graph. If one user A wants to track another user B, the server will run the GFG-assisted tracking algorithm and select the shorter path to route A to B.



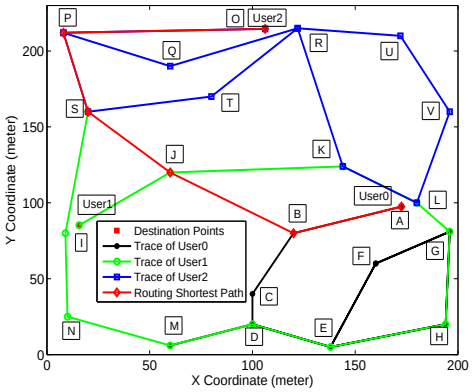
(a) The trails of 3 users for 10 minutes.



(b) Original routing path from user0 to user2.



(c) Routing path with GFG algorithm from user0 to user2.



(d) Routing path with the shortest path from user0 to user2.

Fig. 4. Routing path algorithm simulation example.

A. GFG Routing Algorithm

In the proposed algorithm, due to owning the area map, the seeker can know his own position and the positions of all of the passageway intersections in the specific area. Thus, the seeker can apply the GFG routing algorithm to search the shorter paths for every pair of intersections and report them to the server. Those GFG routing paths can assist the server in finding a shorter path for human tracking.

At the present time, the geographic routing algorithms include three categories: *Greedy*, *Face* and *Greedy-Face* combinations, in which greedy-face combinations are the most efficient geographic routing algorithms. In greedy routing, each node forwards a message to the node that is closest to the destination node. Only neighbors closer to the destination node are considered, otherwise forwarding fails. The greedy routing algorithm does not guarantee delivery since a message can be trapped in a local minimum. In the GFG routing algorithm, each node sends a message to its neighbor that is closest to the destination node. If greedy routing fails, face routing mode starts, which forwards the message along the perimeter of the face next to the local minimum in the direction of the destination, until the greedy routing mode can be resumed. When starting recovery, the distance of the source to destination d_r and the first edge e_r have to be stored in the routing packet header. If the first edge e_r is visited again for a second time, then the destination is not reachable, and the packet is dropped. The distance d_r is used to check whether or not the next hop on the face is closer to the destination than the node entering recovery mode. If such a node is found, greedy

Algorithm 1 GFG Routing Algorithm with sooner-back procedure [24]

if packet in greedy mode **then**
  select next hop node v according to the greedy rule
  if node such neighbor exists **then**
    select next hop node v, direction from (u, d)
    switch packet to face mode
    store current distance to the destination d_r and $e_r \leftarrow (u, v)$ in the packet header
else
  if there is a neighbor v with $\|v - d\| < d_r$ **then**
    switch packet to greedy mode
  else
    select next hop node v direction from (u, p)
    if (u,v)equals the first edge e_r in face mode **then**
      drop packet and return
forward packet to next hop node v

mode can be resumed instead of continuing the face traversal until crossing the $s-d-$line. GFG switches between greedy mode and face mode in order to make sure that the message is continuously getting closer to the destination. This is known as *sooner-back procedure* [24]. The GFG routing algorithm is presented in Algorithm 1.

An example is shown in Fig. 5. Source node s sends the message to the closest neighbor u. After reaching local minimum u in the greedy mode (solid line), a face traversal is started (dashed line) until a node v is found which is closer to the destination than u. Then, switching to greedy mode, the message is sent to the closest neighbor w. After reaching local minimum w, a face traversal is started again until node x is found. Finally, the message is forwarded to the destination d via the node x in greedy mode.

B. The GFG-assisted Tracking Algorithm

The GFG-assisted tracking algorithm includes four phases, respectively. The first and second phases include the trail graph construction and pruning the graph, which are the same as in the original tracking algorithm.

The third phase is called merging the GFG routing path. In this phase, the new found GFG routing paths are reported to the server. The server merges them with the resulting trail graph after the second phase. For the new trail graph, it creates the new vertices when the new path intersects with the users' trails, $V \leftarrow V \cup \{\text{new vertices}\} \cup \{\text{the intersections in the new GFG path}\}$, where V is the set of nodes in the trail graph. Meanwhile, the new segments are also merged into the trail graph, that is, $E \leftarrow E \cup \{\text{new segments}\} \cup \{\text{the edges in the new GFG path}\}$, where E is the set of edges.

After the merging GFG routing path phase, the GFG-assisted tracking algorithm enters into the forth phase. The server selects the shortest path to be routed from one user back to the other. If one user A wants to be routed to another B, the GFG-assisted tracking algorithm computes the tracking path by using two methods. Firstly, it checks whether the GFG routing paths can connect their trails. If the GFG routing paths can, A can be routed back to B along that GFG routing path. Secondly, it checks whether or not their trails have the intersection points. If they do, A can also be routed back to B. If they don't, based on our constraints, one user can encounter at least one other user C while tracking others during the data collection phase. A can still be routed back to B through the user C. This is because user C has the intersection points with A and B, respectively. Finally, the server calculates the length of two paths by applying the above two methods, respectively, and chooses the shortest path as the tracking path from A to B. It is obvious that the tracking length of our algorithm is no longer than that of the original algorithm.

Fig. 4(c) illustrates the procedure of the selection of the shortest path for navigating one user to another user. Fig. 4(a) shows that there is no intersection point between $user_0$'s and $user_2$'s trails. The server finds that $user_1$ can connect their trails; so the tracking path, after applying the original algorithm, is $ABCDEHGLKRQPO$, as shown in Fig. 4(b).

Algorithm 2 GFG-assisted Tracking Algorithm

```
/*Construction trail graph phase*/
for ∀u_i ∈ U do
    record its current position pos_i
    V ← pos_i
    compute the spatial intersections intersect_ij with u_j, u_j ∈ U
    V ← intersect_ij
    E ← uv, vu is the neighbor in the one segment
/*Pruning graph phase*/
for ∀pair(u_i, u_j) ∈ U do
    select the closest points v_i and v_j ∈ intersect_ij for both u_i and u_j
    retain v_i and v_j
    delete other intersections
/*Merging GFG path phase*/
for ∀newfoundP_GFG_i do
    compute the intersection points intersect_ij with u_j ∈ U
    V ← V ∪ intersect_ij
    E ← E ∪ P_GFG_i
/*Computing GFG-assisted path phase*/
for ∀pair(u_i, u_j) ∈ U do
    compute the length of path P_original
    for ∀P_GFG_k ∈ P_GFG do
        if P_GFG_k have the intersections both with u_i and u_j then
            P_GFG-included ← {pos_i → intersect_ik} ∪ P_GFG_k ∪ {intersect_kj → pos_j}
            compute the length of path L_GFG-included
            P_GFG-assisted ← MAX{L_original, L_GFG-included}
        else
            P_GFG-assisted ← P_original
```

Due to knowing the map information, the seeker finds several new paths from node L to R, R to P, and B to A, by applying the GFG routing algorithm. Then, those new paths are merged with the trail graph, and the new trail graph is created. The server computes the tracking path $P_{GFG-included}$, involving the GFG routing path P_{GFG}. The resulting routing path is $ALKRO$, in which LKR is one GFG routing path, as shown in Fig. 4(c). Finally, the server compares the length of two paths and selects the shorter one as the final tracking path. The GFG-assisted tracking algorithm is presented in Algorithm 2.

C. Correctness and Complexity Analysis

Theorem 1. *The GFG-assisted tracking algorithm can ensure that any user can be routed back to the other user in the planar network.*

Proof: The seeker runs the GFG-routing algorithm with the map. The road intersections and all of the road segments can form a planar connected graph. It has been proven that the GFG routing algorithm can guarantee message delivery

TABLE II
SIMULATION SETTINGS

Parameters	Values
Monitored Region	Main Campus of Temple University
Mobile Model	Random Waypoint Model
Number of Mobile Users	[8, 20]
Mobile Velocity	[1.5, 2.0] meters/second
Sensing Sampling Frequency	5 seconds
Data Collection Time	5 minutes
Amount of Stored Data	10 minutes
Total Duration Time	60 minutes

in the planar connected graph [22]. Thus, the seeker can be guaranteed a routing path for every pair of intersections in the road map. After the routing path is found, it is merged with the trail graph by applying the pruning algorithm. Because the resulting trail graph after pruning is a fully-connected graph, the resulting trail graph after merging the GFG routing paths is still a fully-connected graph.

If one user A wants to be routed to another user B, the GFG-assisted tracking algorithm provides two methods for finding the routing path. One method determines the route path by applying the GFG routing path. The other computes the routing path by finding the encounter points with the other users. After that, the server calculates the length of two routing paths, respectively, and chooses the shortest path as the final routing path. Thus, the server can ensure user A will be routed back to user B.

■

Theorem 2. *The worst case complexity of the GFG-assisted tracking algorithm is $O(N^2)$.*

Proof: In the GFG routing algorithm, it is assumed that each node is aware of its own position and the positions of neighbors, and it is also assumed that the source is aware of the destination's position. A message is forwarded to its neighbor based just on the local information and the destination location. Thus, the complexity is $O(m)$, where m is the average degree of a node. During the routing path selection phase in the GFG-assisted tracking algorithm for two users, at the worst case, the server needs to search all of the intersection points to determine the final routing path; thus, the complexity is $O(N)$. Therefore, for all pairs of users, the worst case complexity is $O(N^2)$. ■

V. SIMULATION EVALUATION

In this section, we evaluate the tracking performance with the proposed tracking algorithm in this paper, and we compare it with the original tracking algorithm in the Escort system.

A. Simulation Settings & Methodology

We create an event-based simulator for the evaluation of our proposed tracking algorithm. The mobility area of all of the users is a $350m \times 350m$ public area in the main campus of Temple University. In the simulations, we consider the random waypoint movement on the networks [27]. In the random waypoint model, each mobile user moves independently of others, with a speed that changes only when entering a new road segment. When walking along one segment, the mobile user maintains a constant velocity. Meanwhile, the mobile user selects and traverses a new segment at random at each intersection; it then repeats. In the simulation, the mobile velocity varies from 1.5 to 2.0 m/s.

The methodology of the experiments is the same as that of the Escort system. In order to test the tracking performance across a range of users, movements, and encounter patterns, we split the simulation experiments into two stages: trail data collection and user routing. In the trail data collection stage, a group of N mobile users log into the Escort server and are monitored for some time T. The movement model is the random waypoint model. When the data collection phase ends at time T, the server stores the movement and position information of each user and can run the tracking algorithm for the users. In the simulation, the sensing sample frequency is set to 5 seconds, and the data collection period for each time is set to 5 minutes. Data is collected 12 times, and the total duration time of experiments is 60 minutes. Considering the workload of the server and the computational complexity of the trail graph, in the simulation, the server just stores the related data about the system running for 10 minutes.

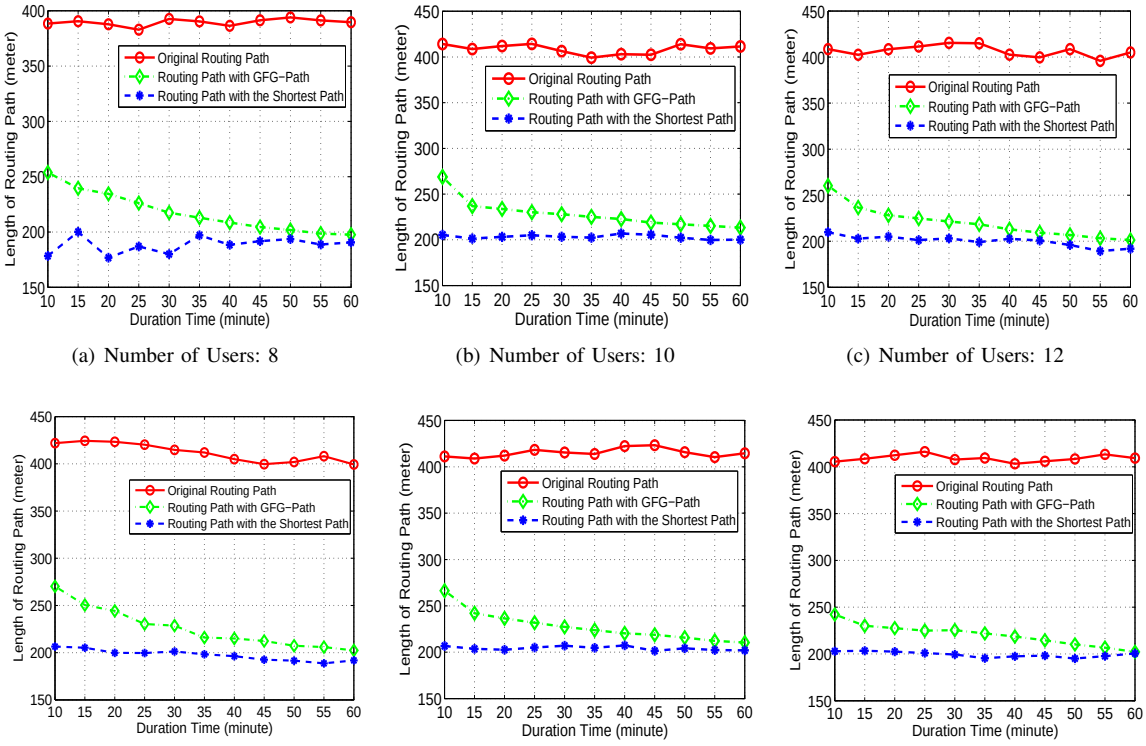
In our proposed algorithm, we add one seeker to find a better tracking path by applying the GFG-routing scheme. The seeker walks around in the main campus of Temple University to find a new path, with a constant velocity of 1.5 m/s. The seeker walks up to one intersection and then selects one road (next intersection in the map) based on the GFG-routing scheme. If a new GFG path in the main campus is found, it would be reported to the Escort server and merged with the trail graph. The seeker does not walk until the duration time has expired or no new GFG path can be found.

In order to evaluate the tracking performance of our proposed algorithm, we use the shortest path between the two users as the benchmark. The metrics that we use to evaluate the tracking algorithms are the length of the tracking path, the ratio of $L_{GFG-assisted}$ to $L_{original}$, and the trade-off between the gain from the GFG routing path and the cost of one seeker under the different data collection times and the varying of users.

In addition, in order to accurately evaluate the tracking performance for the original tracking algorithm and the proposed GFG-based tracking algorithm, the border influence is ignored. The settings of parameters are shown in Table II. All of the statistics are averaged over 50 runs for high confidence.

B. Simulation Results

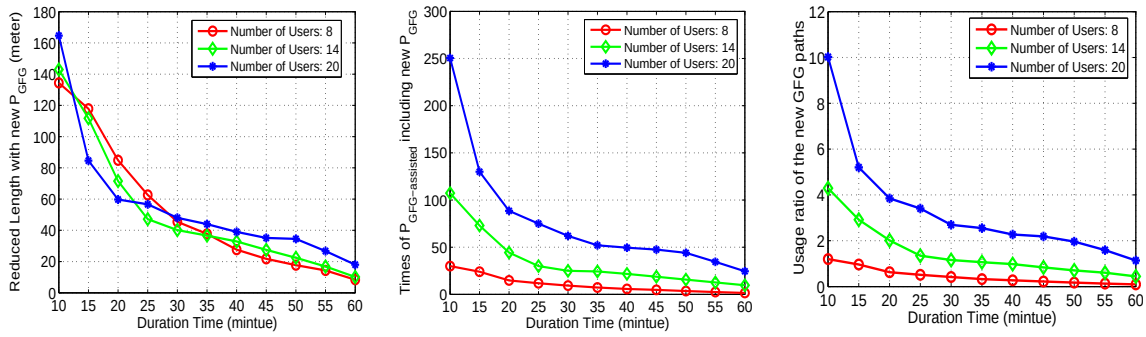
1) Visualized Simulation Example: We first give a visualized comparison result, applying the two tracking algorithms: the original tracking algorithm and the GFG-assisted tracking algorithm. In this example, the Escort server tracks 8 users for 10 minutes. Fig. 4(a) shows the three users' trails, $(user_0, user_1, user_2)$, in the resulting graph after applying the pruning heuristic. Fig. 4(b)-(d) illustrates the example of the routing path from $user_0$ to $user_2$ by applying the original



(a) Number of Users: 8 (b) Number of Users: 10 (c) Number of Users: 12

(d) Number of Users: 14 (e) Number of Users: 16 (f) Number of Users: 20

Fig. 6. Comparison results of routing paths using three schemes.

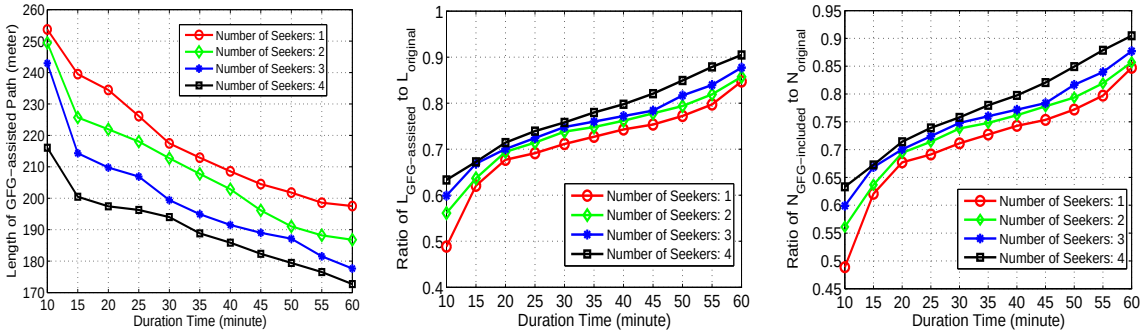


(a) Reduced routing length using the new P_{GFG} (b) Usage ratio of the new GFG paths (c) Usage times of $P_{GFG-assisted}$ including the new P_{GFG}

Fig. 7. Tradeoff between one seeker's cost and tracking path gain.

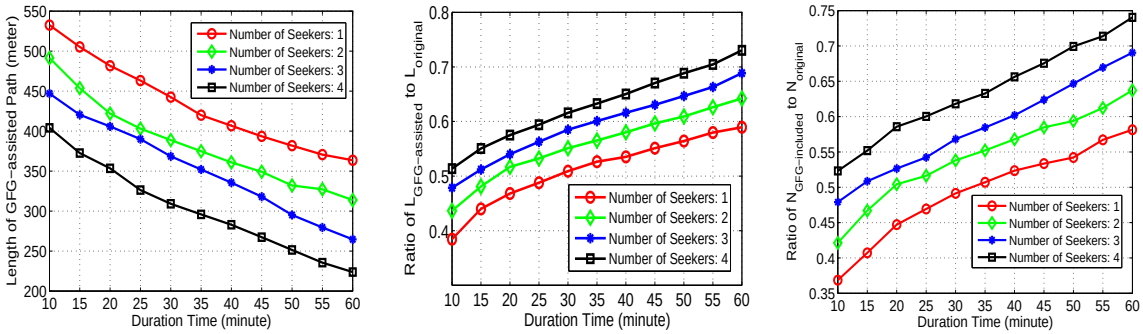
tracking algorithm, the GFG-assisted tracking algorithm, and the shortest routing path. It is obvious that the proposed GFG-assisted tracking algorithm can greatly reduce the length of the tracking path from $user_0$ to $user_2$ compared to the others. From Fig. 4(a), $user_0$'s trail cannot intersect $user_2$'s trail. If $user_0$ needs to route back to $user_2$ by applying the original tracking algorithm, the server should check whether or not they have encountered others. In this example, $user_0$ can be routed back to the point where she met $user_1$ then routed along the path that $user_1$ walked until she encountered $user_2$. Finally, $user_0$ can be navigated along $user_2$'s path to localize her current position. If the GFG routing algorithm is applied, one seeker can find a new path that can directly connect $user_0$ and $user_2$. Thus, $user_0$ can be routed back to vertex A and then routed along the GFG routing path P_{AB}; finally, $user_0$ can be routed back to $user_1$ from point B, as shown in Fig. 4(b). If the shortest path algorithm is applied in the trail graph, the server can compute a shorter tracking path compared to using the original one. However, due to there being no direct path from $user_0$ to $user_2$, the length of the final tracking path is longer than that of the GFG-assisted tracking path.

2) Tracking Efficiency: We evaluate the tracking efficiency in terms of the length of the tracking path and the ratio of $L_{GFG-assisted}$ to $L_{original}$ under a varying number of users, ranging from 8 to 20. The system running time varies from 10 to 60 minutes. Fig. 6 illustrates the length of the



(a) Length of GFG-assisted routing path (b) Ratio of $L_{GFG-assisted}$ to $L_{original}$ (c) Ratio of $N_{GFG-included}$ to $N_{original}$

Fig. 10. Comparison results of multiple seekers' cost and tracking path gain in a $350m \times 350m$ area.



(a) Length of GFG-assisted routing path (b) Ratio of $L_{GFG-assisted}$ to $L_{original}$ (c) Ratio of $N_{GFG-included}$ to $N_{original}$

Fig. 11. Comparison results of multiple seekers' cost and tracking path gain in a $1000m \times 1000m$ area.

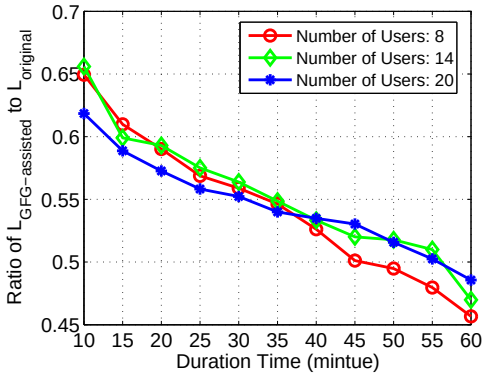


Fig. 8. Ratio of $L_{GFG-assisted}$ to $L_{original}$.

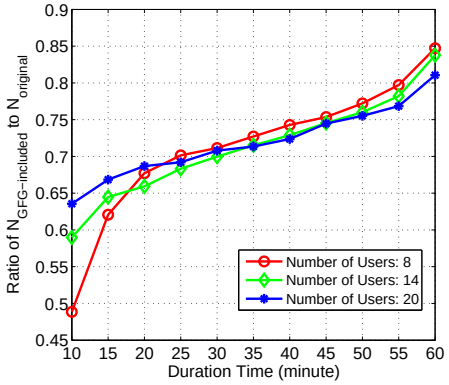


Fig. 9. Ratio of $N_{GFG-included}$ to $N_{original}$.

path by applying the original tracking algorithm, the GFG-assisted tracking algorithm, and the shortest path algorithm, respectively. As Fig. 6 shows: (i) the GFG-assisted tracking algorithm outperforms the original algorithm and has the closest performance to that of the shortest path algorithm. From the results in Fig. 6, the GFG-assisted tracking algorithm can reduce the length of the original tracking path by 35% to 55%, and the optimal results are increased 5% to 25%; (ii) with the increase in system running time, from 10 minutes to 60 minutes, the length of the GFG-assisted tracking path decreases. The reason is that the increase of the running time can increase the number of found GFG routing paths, which can provide a higher probability to reduce the tracking path via the GFG routing path; (iii) By varying the number of users from 8 to 20, the length of the GFG-assisted tracking path has no striking difference. Based on the simulation results, the number of users has no great impact on the tracking performance when applying the GFG-assisted tracking algorithm.

In addition, we evaluate two other metrics: the ratio of the reduced length $Ratio_{length}$ and the ratio of the GFG path

$Ratio_{GFGPath}$. $Ratio_{length}$ is defined as the ratio of the length of GFG-assisted tracking paths to the length of the tracking paths by applying the original algorithm:

$$Ratio_{length} = \frac{L_{GFG-assisted}}{L_{original}}$$

$Ratio_{GFGPath}$ is defined as the ratio of the number of tracking paths via GFG routing paths to the number of tracking paths by applying the original algorithm:

$$Ratio_{GFGPath} = \frac{N_{GFG-included}}{N_{original}}$$

Fig. 8 and Fig. 9 show the simulation results of the two metrics in the same scenario as the above. It is obvious that $Ratio_{length}$ decreases with the increase of the system running time, while $Ratio_{GFGPath}$ increases. With the increase of system running time, the new paths from applying the GFG routing algorithm grow over time. More and more new paths have been merged in the trail graph; thus, if one user wants to localize and track another user, there are more GFG routing paths to be used to find the shortest tracking path.

3) Tradeoff Gain and Cost: In our proposed tracking algorithm, one seeker is added to search the new path by applying the GFG routing algorithm to improve the tracking performance. If new GFG routing paths are found and merged with the trail graph, the server can compute the shorter tracking path with the found routing paths for each pair of two users. Thus, the mobile users can obtain benefit v. The benefit v is the reduced routing length using the new GFG path P_{GFG}. If the server computes the tracking paths for each pair of users without using the new found GFG path, the benefit is $v = 0$.

The cost in the GFG-assisted routing algorithm is attributed to the seekers. The cost c is the waking distance by the seeker for finding new GFG routing path. Due to the seeker walking with a constant velocity, the cost c for finding new GFG routing paths is as follows:

$$c = speed * T$$

where $speed$ is the average walking speed of one seek in one second, and T is the running time for finding a new GFG path by the seeker. Therefore, the tracking gain is $v - c$. If the server computes the tracking paths without using the new found GFG routing path, the tracking gain is $0 - c$.

In the simulations, we use three metrics to evaluate the tracking gain: the reduced length applying the new found GFG routing paths $P_{GFG-new}$, the times of using $P_{GFG-new}$, and the ratio of the number of GFG-assisted paths, including $P_{GFG-new}$, to the number of $P_{GFG-new}$, denoted as $L_{reduced}$, $UsageTimes_{P_{GFG}}$, and $UsageRatio_{P_{GFG}}$, respectively.

Fig. 7 illustrates the tradeoff between the tracking gain and the cost of one seeker with a varying number of users, ranging from 8 to 20. Fig. 10 and Fig. 11 show the tradeoff relationship between the gain and the cost of multiple seekers with 8 users walking within a $350m \times 350m$ public area and a $1000m \times 1000m$ area in the main campus of Temple University, respectively.

As Fig. 7, Fig. 10, and Fig. 11 show: (i) $L_{reduced}$, $UsageTimes_{P_{GFG}}$ and $UsageRatio_{P_{GFG}}$ show a significant drop when the running time grows. The reason is that the number of the new found GFG routing paths, $P_{GFG-new}$, in the beginning stage is much more than that in the subsequent stages. Hence, most of the obtained tracking paths, $P_{GFG-assisted}$, include $P_{GFG-new}$ at the beginning stage, which can greatly reduce the tracking length and enhance the tracking gains. However, when the system runs for a long time, less new GFG routing paths are found and merged with the trail graph, and the tracking gains gradually decrease; (ii) by varying the number of seekers from 1 to 4, it can be found that the more seekers there are, the better the tracking performance. The reason is that increasing the number of seekers can increase the number of found GFG routing paths during the same period of running time, which can provide a higher probability to improve the tracking performance; (iii) with the same experimental parameters, the enhancement of the tracking performance by adding more seekers in a larger area is more significant than that in a smaller area. More seekers can find more new and different GFG routing paths for the same running time in a larger public area. While in a smaller area, different seekers may find the same new GFG routing paths. Thus, the improvement of the tracking performance by adding more seekers is more significant in a larger public area. As Fig. 10 shows, when the simulation area is a $350m \times 350m$ area, the tracking gains show a small increase when the number of seekers increases from 1 to 4. Therefore, when the scenario can be presented as a 2-D model, if the Escort system is applied in the small-scale public area, such as the conference hall, adding more seekers does not help much improve the performance. Just one seeker can effectively improve the tracking performance. If the Escort system is applied in a large public area, such as an international airport, we need to add more seekers to find the new GFG paths and to improve the tracking performance. However, in the small-scale but complicated scenario, like a city hotel with multiple levels, it cannot be presented a 2-D model. GFG-assisted routing algorithm should be further modified to be applied in the small-scale but complicated scenario. In the future work, we will discuss the new routing algorithm and how to distribute the seekers to work together in such a city hotel scenario.

VI. Conclusion

To provide better tracking performance for many location-based applications, this paper studies the problem of obtaining better tracking paths in the Escort system. We propose a GFG routing assisted human tracking algorithm to reduce the length of the path for every pair of users using smart phones in mobile social networks. In the proposed algorithm, one seeker is added to find better paths for any pair of two intersections in the specific area by applying the GFG routing algorithm; then, the new path is merged with the trail graph. To evaluate the

tracking performance of the proposed algorithm, we analyze its properties and prove its correctness. Finally, extensive simulation experiments show that the proposed GFG assisted tracking algorithm outperforms the original one with the low cost when the system is applied in the scenario presented as a 2-D model.

ACKNOWLEDGMENTS

This research was supported in part by NSF grants ECCS 1128209, CNS 1065444, CCF 1028167, CNS 0948184, and CCF 0830289; Key Laboratory of Geo-informatics of State Bureau of Surveying and Mapping 201005.

REFERENCES

[1] A. Thiagarajan, L. Ravindranath, H. Balakrishnan, S. Madden, and L. Girod, "Accurate, low-energy trajectory mapping for mobile devices," in *Proc. of 8th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2011.

[2] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Toledo, and J. Eriksson, "Vtrack: Accurate, energy-aware road traffic delay estimation using mobile phones," in *Proc. of 7th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2009.

[3] N. Banerjee, S. Agarwal, and P. Bahl, "Virtual compass: Relative positioning to sense mobile social interactions," in *Proc. of 8th International Conference on Pervasive Computing (Pervasive)*, 2010.

[4] I. Constandache, X. Bao, M. Azizyan, and R. R. Choudhury, "Did you see bob?: Human localization using mobile phones," in *Proc. of the 16th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2010.

[5] Z. Zhuang, K. Kim, and J. P. Singh, "Improving energy efficiency of location sensing on smartphones," in *Proc. of 8th Annual International Conference on Mobile Systems, Applications and Services (MobiSys)*, 2010.

[6] M. Ra, J. Paek, A. B. Sharma, R. Govindan, M. H. Krieger, and M. J. Neely, "Energy-delay tradeoffs in smartphone applications," in *Proc. of 8th Annual International Conference on Mobile Systems, Applications and Services (MobiSys)*, 2010.

[7] K. Lin, A. Kansal, D. Lymberopoulos, and F. Zhao, "Energy-accuracy trade-off for continuous mobile device location," in *Proc. of 8th Annual International Conference on Mobile Systems, Applications and Services (MobiSys)*, 2010.

[8] M. B. Kjargaard, J. Langdal, T. Godsk, and T. Toftkjar, "Entracked: Energy-efficient robust position tracking for mobile devices," in *Proc. of 7th Annual International Conference on Mobile Systems, Applications and Services (MobiSys)*, 2009.

[9] A. Thiagarajan, J. Biagioni, T. Gerlich, and J. Eriksson, "Cooperative transit tracking using smart-phones," in *Proc. of 7th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2010.

[10] M. Azizyan, I. Constandache, and R. R. Choudhury, "Surroundsense: Localizing mobile phones via ambience fingerprinting," in *Proc. of the 15th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2009.

[11] N. Ravi, P. Shankar, A. Frankel, A. Elgammal, and L. Iftode, "Indoor localization using camera phones," in *Proc. of 7th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, 2006.

[12] N. B. Priyantha, A. Chakrakborty, and H. Balakrishnan, "The cricket location-support system," in *Proc. of the 6th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2000.

[13] D. Niculescu and B. Nath, "Vor based stations for indoor 802.11 positioning," in *Proc. of the 10th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2004.

[14] M. Youssef, A. Youssef, R. C, A. Shankar, and A. Agrawala, "Pinpoint: An asynchronous time-based location determination system," in *Proc. of the 4th International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2006.

[15] P. Bahl and V. N. Padmanabhan, "Radar: an in-building rf-based user localization and tracking system," in *Proc. of 19th IEEE Conference on Computer Communications (INFOCOM)*, 2000.

[16] W. G. Griswold, P. Shanahan, S. W. Brown, B. R, and M. Ratto, "Activecampus: Experiments in community-oriented ubiquitous computing," *IEEE Computers*, vol. 37(10), 2004.

[17] Y. Chen, Y. Chawathe, A. Lamarca, and J. Krumm, "Accuracy characterization for metropolitan-scale wi-fi localization," in *Proc. of the 3rd International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2005.

[18] I. Constandache, R. R. Choudhury, and I. Rhee, "Towards mobile phone localization without war-driving," in *Proc. of 29th IEEE Conference on Computer Communications (INFOCOM)*, 2010.

[19] F. Evennou and F. Marx, "Advanced integration of wi-fi and inertial navigation systems for indoor mobile positioning," *EURASIP Journal on Applied Signal Processing*, 2006.

[20] G. G. Finn, "Routing and addressing problems in large metropolitan-scale internetworks," in *ISI Research Report ISU/RR-87-180*, 1987.

[21] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia, "Routing with guaranteed delivery in ad hoc wireless networks," in *Proc. of the 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM)*, 1999.

[22] ——, "Routing with guaranteed delivery in ad hoc wireless networks," *Wireless Networks*, vol. 7(6), 2001.

[23] B. Karp and H. Kung, "Gpsr: greedy perimeter stateless routing for wireless networks," in *Proc. of 6th Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2000.

[24] S. Datta, I. Stojmenovic, and J. Wu, "Internal node and shortcut based routing with guaranteed delivery in wireless networks," *Cluster Computing*, 2002.

[25] F. Kuhn, R. Wattenhofer, and A. Zollinger, "Worst-case optimal and average-case efficient geometric ad-hoc routing," in *Proc. of 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2003.

[26] H. Hwang, I. Hur, and H. Choo, "Goafr plus-abc: geographic routing based on adaptive boundary circle in manets," in *Proc. of the 23rd International Conference on Information Networking (ICOIN)*, 2009.

[27] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*, 1996.