DTN routing protocols when SMOOTH is used. Lastly, we present our conclusions in Section VII.

II. Related Work

While synthetic mobility models are simple to use, they may not generate synthetic traces that mimic real human movement. Thus, the research community has recently focused on developing trace-based models (i.e., models that are constructed from real mobility traces). In trace-based models, real human movement traces are collected from various scenarios (e.g., campus, conferences, offices). The collected traces are then analyzed to identify the key characteristics of real human walks. Using this information, a model is constructed and is called a *trace-based* mobility model. The model is expected to generate synthetic traces that match well with the real human walk traces analyzed.

Different trace-based models developed are found in [9]–[12], [14]. In [15], the authors developed a simple mobility model, SWIM, that generates synthetic traces representing the social behavior among mobile nodes. The authors validated the synthetic traces generated by SWIM using the real GPS traces collected from three different experiments. In SWIM, every possible destination is assigned a calculated weight that increases with its popularity and decreases with its distance from the current GPS location of the mobile node. Two input parameters, K and α, are used for this purpose. The synthetic traces generated by SWIM seem to match the real GPS traces well. The GPS traces used for validation, however, are only from a campus and a conference scenario; thus, these traces do not capture the movement patterns from other scenarios, e.g., cities. In addition, the parameters K and α need to be set differently in order to match the real GPS traces. Also, the authors do not mention the flight distribution of mobile nodes, and this distribution appears to be a significant feature present in real human walk [2], [9].

In [2], the authors developed a mobility model, SLAW, that is based upon the real GPS traces collected from five different outdoor sites. The GPS traces used in SLAW are from diverse scenarios (two campuses, one state fair, one amusement park, and one metro city scenario); however, the synthetic traces generated by the SLAW mobility model heavily depend upon an input parameter called *hurst*. Our previous work in [18] and Lee et al. in [2] show that the *hurst* value has a significant effect on the performance metrics of a routing protocol. For example, in [18], we developed models that help researchers to generate rigorous simulation scenarios and these models were developed with the SLAW mobility model. During our analysis, we discovered SLAW's input parameters that most significantly affect the performance of a routing protocol. Specifically, we showed that the number of *nodes*, simulation *area* size, and the *hurst* value are the three input parameters that highly affect the performance of a routing protocol when used with SLAW. Unfortunately, as discussed in Section I, it is difficult to estimate the *hurst* value for a given data set. To illustrate, consider the following. The number of participants used in each outdoor site in [2] is small compared to the size of the measurement sites. Specifically, the measurement sites are of the order of several thousand meters squared and the number of participants used in each site ≤ 40. With such a difference in area size versus the number of participants, the *hurst* values extracted from the respective measurement sites may be biased. For example, we note that the site with the highest number of participants (i.e., Kaist with 34 participants) has the highest *hurst* value. In addition, due to the inherent complexity in techniques used for estimating *hurst* from a given data set and the possible biased trends that might be present in the measurements, one might find it complex to use SLAW the way one intends.

III. SMOOTH: Our Simple and Realistic Mobility Model

In the evaluation of real human walks, researchers have found several statistical features that exist [6]–[8], [13], [14], [25]. Specifically,

- **feature1:** The flights (i.e., a straight-line distance covered between two consecutive waypoints) distribution of mobile nodes follows a truncated power-law (TPL)[2].
- **feature2:** The ICTs (inter-contact times) (i.e., the amount of time between two successive contacts of the same pair of nodes) distribution of mobile nodes follows TPL.
- **feature3:** The pause-times (i.e., the amount of time a node pauses at a waypoint) distribution of mobile nodes follows TPL.
- **feature4:** Mobile nodes visit popular waypoints in the network.
- **feature5:** While moving, mobile nodes usually (but not always) visit the closest waypoint first.
- **feature6:** The distribution of mobile nodes is non-uniform in the network.
- **feature7:** Mobile nodes with common interests form communities and only tend to move around their communities of interests.

We capture all seven of these features in SMOOTH. Specifically, SMOOTH first creates communities with different levels of popularity, i.e., a few communities are more popular than the other communities. A mobile node in SMOOTH visits communities based upon their popularity level and, thus, meets a few mobile nodes more often than others. A mobile node in SMOOTH does not move around the simulation area randomly; instead, while moving, a mobile node is more likely to visit locations that are nearby to its current GPS location. We next describe how we created these communities in SMOOTH. We then show how SMOOTH meets **feature1** - **feature7** of real human walks.

A. Creating Communities

When evaluating real GPS traces (e.g., traces in [2]), it is evident that mobile nodes do not move randomly over the simulation area. Specifically, a mobile node visits a few

[2]The truncated power-law distribution follows power-law upto a certain time after which it is truncated by an exponential cut-off.

(i.e., popular) waypoints more frequently than waypoints that are less popular. Such movement patterns of mobile nodes represent the social behavior of humans and, thus, form communities. In SMOOTH, we represent communities by clusters, where popular communities are denoted by clusters of bigger sizes. (A *cluster-size* is the number of waypoints that belong to the cluster.) In SMOOTH, clusters are formed per the following technique.

We assume that the total number of waypoints, N, and the total number of clusters, C, available for a mobile node to visit are fixed and are specified as user inputs. N is divided into C clusters of unequal sizes. The cluster sizes are unequal to mimic realistic scenarios (i.e., popular communities). For example, students spend more time in an academic building (including classrooms) than they spend in a food court. Similarly, within a building, there are places that are visited more often than others (e.g., classrooms are visited more often than a conference room). Therefore, within a cluster, we plot groups (of random sizes). Specifically, to plot the waypoints of clusters and groups (within a cluster), we use the following approach:

1) for every cluster:
   a) for first group in this cluster:
      i) Plot the first waypoint uniformly over the simulation area with the following caveat; i.e., no two waypoints that belong to different clusters are within each other's transmission range.
      ii) Plot other waypoints that belong to this group within 0.1R (transmission range) of any of the waypoints that belong to this group.
   b) for every other group in this cluster:
      i) Plot the first waypoint such that it is within distance d from the last waypoint plotted for the previous group, $\frac{Y}{4} \leq d \leq \frac{Y}{3}$, where

$$Y = \frac{2Z}{C}, \quad (1)$$

Z is the side of the simulation area and C is the total number of clusters in the network. The area spanned by a cluster depends upon the simulation *area* size and the total number of clusters in the network. For example, if a user wants to create a single community, then SMOOTH plots it as one large cluster with various groups of random sizes. Thus, our choice of d (to separate random groups) depends upon the simulation *area* size (i.e., Z) and the total number of clusters (i.e., C).

B. SMOOTH: A Simple Algorithm

In this section and the next, we describe how SMOOTH satisfies all the seven statistical features (**feature1** - **feature7**) found in real human walks. Algorithm 1 describes an easy step-by-step working of SMOOTH. Further details on the algorithm follow.

In SMOOTH, a mobile node chooses a cluster (i.e., a community) with probability $\propto$ to its size (Step 3), i.e., a bigger cluster is chosen more often than a smaller cluster and, thus, more mobile nodes visit the bigger clusters making them more popular (**feature4** and **feature6**). From the selected cluster, a mobile node (uniformly) chooses a subset of waypoints to visit (Step 4). Since a bigger cluster has more waypoints, a mobile node chooses more waypoints to visit in a bigger cluster than a smaller cluster and, thus, spends more time in bigger clusters. Since every mobile node chooses a cluster with probability $\propto$ to its size, a few mobile nodes meet each other more often than others (**feature2** and **feature7**). While moving, a mobile node tends to visit the closest waypoint (i.e., the waypoint closest to its current GPS location); however, it may occasionally visit a farther location first (e.g., to attend a class that starts in a few minutes). To simulate this movement, the authors of [2] use a distance parameter *alpha* that determines the probability of choosing the next waypoint via the LATP (Least-Action Trip Planning) algorithm. The LATP algorithm uses a distance function (i.e., $\frac{1}{d^\alpha}$) to calculate the probability of selecting the next waypoint to visit; d is the distance between the current waypoint and the unvisited waypoint and α is the value of parameter *alpha* that controls the selection of the next waypoint. For $\alpha = 0$, the next waypoint is selected randomly; for $\alpha = \infty$, the next waypoint to visit is the closest waypoint. In SMOOTH, we follow the same approach; that is, we use the LATP algorithm to select the next waypoint to be visited by the mobile nodes (**feature1** and **feature5**). For details, please refer to Section III.C of [2]. Pause-times are distributed according to truncated power-law (Step 6), i.e., mobile nodes pause for a long time at a few waypoints and pause for a short time at the majority of waypoints visited (**feature3**). The input parameters to SMOOTH follow.

1) *nodes* is the total number of mobile nodes in the network.
2) *area* is the size of the simulation area.
3) *waypoints (N)* is the total number of waypoints in the network that mobile nodes can visit.
4) *range (R)* is the transmission range of a mobile node[3].
5) *clusters (C)* is the total number of clusters (to be plotted) in the network.
6) *percent_waypoint (y)* is the percentage of waypoints visited by a mobile node from the selected cluster.
7) *alpha* controls the selection of the next waypoint visited

Algorithm 1 SMOOTH: pseudocode

1. Divide the simulation area into several communities (see Section III-A for details).
2. **for** each mobile node **do**
3. Select a community with probability $\propto$ to its *size*.
4. Select a subset (y%) of waypoints to visit from the selected community.
5. Visit the selected waypoints via the LATP algorithm (see Section III-B for details).
6. At each waypoint, pause for a *pause-time* distributed by power-law.
7. **end for**

[3]Many mobility models do not require the transmission range. SMOOTH needs the transmission range to form communities.

by a mobile node in the LATP algorithm.

8) *beta* is the levy exponent used for the pause-time distribution (i.e., describes the asymptotic behavior of the distribution).
9) (*min_pause, max_pause*) is the minimum and the maximum allowed pause-time (in seconds) for a mobile node.

C. Power-law flights and ICTs

In this section, we validate that our simple technique (described in Section III-A) for creating communities generates synthetic traces that preserve the statistical features observed in real human movement. Specifically, we validate that SMOOTH generates synthetic traces that have flights and ICTs (inter-contact times) distributed according to truncated power-law (i.e., **feature1** and **feature2**). The distribution of flights is of interest, as it helps to predict the trajectories of human walks. Similarly, the ICTs distribution is important, as it represents how often the mobile nodes meet each other and, thus, predicts the probability of forwarding a packet during routing. Recent studies [8], [13] have shown that flights (of mobile nodes) and ICTs (among a pair of mobile nodes) follow truncated power-law.

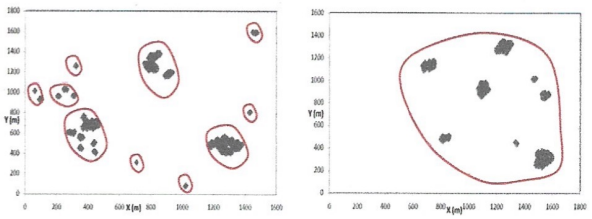
To validate that SMOOTH preserves the statistical features of real human movement, we simulated the two scenarios listed in Table I. (We choose only two scenarios due to space constraints; however, we note that the results are the same regardless of the scenarios chosen.) Figure 1 shows the synthetic maps of waypoints generated by SMOOTH for these two scenarios; the simulation area size and the total number of waypoints are kept constant to illustrate the clustering effect. Specifically, as shown, Scenario I represents 10 communities and, thus, is shown as various clusters distributed over the simulation area. Scenario II, on the other hand, represents a single community and is, thus, represented by a few groups over the simulation area. We extracted the distributions of flights and ICTs among nodes for both of these scenarios. Figure 2 shows the CCDFs (Complementary cumulative distribution functions) of flights and ICTs generated by SMOOTH for the Scenario I listed in Table I (satisfies **feature1** and **feature2**). Results for Scenario II are similar.

Parameter	Scenario1	Scenario2
Nodes	20	10
Area (m^2)	1600x1600	1600x1600
Clusters	10	1
Waypoints	1000	1000

TABLE I
EXAMPLE SCENARIOS SIMULATED ON SMOOTH.

IV. SLAW: A MODEL BASED ON REAL GPS TRACES

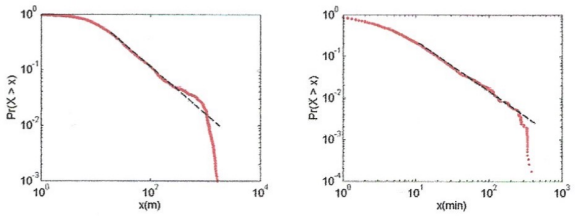
The SLAW mobility model was developed using the real GPS traces collected from five outdoor sites [2]. An analysis of SLAW concludes that SLAW meets **feature1** - **feature7**. SLAW also models the following statistical feature of real human movement:



Synthetic map of waypoints for Scenario I

Synthetic map of waypoints for Scenario II

Fig. 1. Synthetic map of waypoints generated by SMOOTH for the scenarios listed in Table I. The circled area illustrates the communities in each scenario.



CCDF of flights

CCDF of ICTs

Fig. 2. CCDFs of flights and ICTs extracted from the synthetic trace generated by SMOOTH for Scenario I listed in Table I.

- **feature8:** The distribution of waypoints over the simulation area is *self-similar*.

The synthetic mobility traces generated by SLAW are expected to be similar to the real GPS traces used in its development; however, using the SLAW mobility model can be extremely complex. For example, the waypoint placement technique used in SLAW to meet **feature8** requires a complete understanding of the *hurst* parameter. The real GPS traces used in [2] were analyzed to find the *hurst* values for each of the five outdoor sites. The *hurst* value is used to control the degree of self-similarity; a greater *hurst* value represents a higher degree of self-similarity in the network.

Algorithm 2 presents SLAW; x, y, and N are user inputs to the SLAW mobility model. The waypoints are distributed over the simulation area (Step 1) in a self-similar manner (**feature8**) using a technique related to Brownian Motion [16]. The *hurst* parameter can take values within a pre-defined range $0.5 < hurst < 1$; for a *hurst* value close to 1, the process is highly self-similar. In the context of waypoints' placement, a higher *hurst* value means that the network looks more self-similar at different scales than with a lower *hurst* value.

As shown in our previous work [18], a network remains mostly partitioned in SLAW. The high amount of network partitioning is due to two reasons: *inter-cluster partitioning* and *intra-cluster partitioning*. Specifically, in SLAW, clusters are formed (Step 2) via *transitive closure*, i.e., the waypoints

as $m \to \infty$, then $p(x) \sim x^{-\alpha-1}$, $0 < \alpha < 1$ as $x \to \infty$. Furthermore, $\alpha + \beta = 1$.

Proof: Due to the space limit, we provide a proof sketch below. The proof is adopted from [4], [18].

From [12], we can have the following relation between $v(m)$ and the autocorrelation function $\rho(x)$ of $Y(x)$.

$$v(m) = v \cdot \Big(\frac{1}{m} + \frac{2}{m^2}\sum_{n=1}^{m}(m-n)\cdot\rho(n)\Big) \quad (1)$$

where v is the variance of $Y(x)$.

From Eq. 1, $\rho(x) \sim x^{-\beta}$ since $v(m) \sim m^{-\beta}$.

A correlation function $c(x)$ of $Y(x)$ is asymptotically the same as its auto-correlation function $\rho(x)$ and can also be represented as a conditional probability to have a point at x, given that a point occurs at a $x = 0$. $c(x = n\Delta)$ can be represented as follows [36].

$$c(\Delta) = p(\Delta) \quad (2)$$
$$c(2\Delta) = c(\Delta)p(\Delta) + p(2\Delta) \quad (3)$$
$$\cdots \quad (4)$$
$$c(n\Delta) = \sum_{k=0}^{n-1} c[(n-k)\Delta]p(k\Delta) + \delta_{n,0} \quad (5)$$

The Fourier transform $\mathcal{F}$ of the last equation gives a power spectrum of $Y(x)$. Thus, $S(f) = \mathcal{F}(c(x)) = \frac{1}{1-\mathcal{F}(p(x))}$. Let $\hat{P}(f)$ be an asymptotic function of $\mathcal{F}(p(x))$.

$$\hat{P}(f) = 1 - \frac{1}{S(f)} \quad (6)$$

Since $c(x) \sim x^{-\beta}$, $S(f) = \mathcal{F}(c(x)) \sim f^{\beta-1}$ as $f \to 0$. Since $0 < \beta < 1$, the power spectrum of $Y(x)$ has $1/f$ noise.

If we set $\alpha = -\beta+1$, then from Eq. 6, $\hat{P}(f) \sim 1-f^{\alpha}$. Note that the inverse Fourier transform of $1-Bf^{\theta}$, $0 < \theta < 1$ where B is a constant, is asymptotically $x^{-\theta-1}$. Therefore, $p(x) \sim x^{-\alpha-1}$ as $x \to \infty$. Since $0 < \alpha < 1$, $p(x)$ is asymptotically power-law and it proves $\alpha + \beta = 1$. ■

Although we can easily extend Theorem 3.1 over to multiple dimensions analytically, the definition of gaps over a multi-dimensional space is not intuitively clear. Gaite [15] uses Delaunay triangulation to measure the gaps over multi-dimensional spaces. We apply Delaunay triangulation on the waypoints we extracted from real traces [35]. The gap distributions over waypoints extracted from real traces can be seen in Fig. 3. Their power-law fitting is omitted and given in [34].

We measure the power-law slope (β) of the aggregated variances (β) of waypoints extracted from the GPS traces and the power-law slopes (α) of the CCDF of their corresponding gap distributions. We find that the values of $\alpha + \beta$ from real traces are close to 1.2. The margin of errors may arise from truncations caused by confined measurement areas as the truncations may significantly distort the power-law slope visible at the body of distribution. We also perform Delaunay triangulation on top of synthetically generated fractal waypoints using a simplified fBm technique [34] over a 2-D area and measure α and β. For each Hurst parameter value (H), we generate 10 synthetic waypoint maps. Figure 1 shows that $\alpha + \beta \approx 1$ with a similar margin of errors.

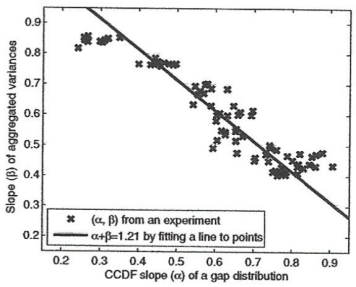


Fig. 1. α and β measured over synthetic 2-D waypoint maps.

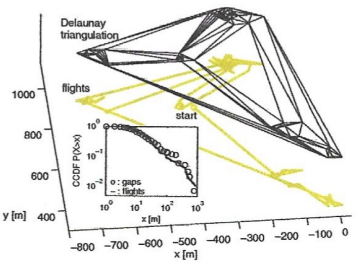


Fig. 2. Delaunay triangulation of waypoints extracted from one daily trace of KAIST. In the inset, the CCDFs of the line lengths from the triangles and the flights from the same trace are plotted. The CCDF of flights and gaps are closely matching.

C. Least-Action Trip Planning

In Section III-B, we showed that fractal points induce a power-law gap distribution. To find out how the power-law gaps are related to the actual human flights, we compare the flight distributions obtained from real traces and the gap distributions induced by the waypoints from the same traces. We find a strong similarity between these two distributions, especially in terms of their slopes and shapes. Figure 2 show that the Delaunay triangles on top of a daily trace of one participant, its line length CCDF forming the triangles and the CCDF of flights extracted from the same trace. We also perform the Delaunay triangulation on each individual trace and aggregate all the resulting triangle lines from all the traces. Figure 3 plots the resulting CCDFs. Their similarity with the corresponding flight distributions is strikingly impressive.

The similarity in the power-law slopes of gaps and flight distributions suggests interesting aspects about the order which a walker visits waypoints for a given set of waypoints. Obviously people are not conscious about gaps when they travel. However, as we can see from Figure 2, Delaunay triangles are formed among "neighboring" waypoints as Delaunay triangulation produces a planar graph where edges intersect only at their endpoints. From this, we conjecture that people

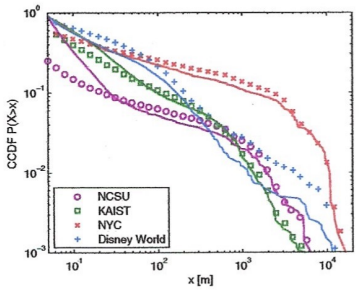


Fig. 3. Delaunay triangulation is performed on all individual daily traces. The line segment lengths in Delaunay triangles are aggregated and their CCDFs are plotted for different walkabout sites. The CCDF of flights obtained from the corresponding traces are also plotted for comparison.

might be minimizing the traveling distance. This makes sense intuitively when we are given a priori multiple destinations at different distances, we often strive to minimize the total distance of travel by first visiting nearby locations before visiting farther locations. In fact, this "greedy" way of trip planning is similar to a heuristic to the traveling salesman problem whose objective is to minimize the total distance of travel and aligns well with the least action principle of Maupertuis [13].

Algorithm 1 *Least action trip planning* (LATP) algorithm with a distance weight function $d^{-\alpha}$

V: set of all vertices (waypoints)
V': set of all visited vertices
$s \in V$: starting vertex
$c \in V$: current vertex
$c \Leftarrow s$
$V' \Leftarrow \{c\}$
while $V' \neq V$ **do**
  Calculate distances to all unvisited vertices, $d(c,v) = \|c - v\|_2$ for all $v \in V - V'$
  Calculate probability to move to all unvisited vertices,

$$P(c,v) = \frac{\left\{\frac{1}{d(c,v)}\right\}^{\alpha}}{\sum_v \left\{\frac{1}{d(c,v)}\right\}^{\alpha}} \text{ for all } v \in V - V'$$

  Choose a next vertex $v' \in V - V'$ according to the probabilities $P(c,v)$
  $c \Leftarrow v'$
  $V' \Leftarrow V' \cup \{c\}$
end while

To measure how much real human traces follow the least action principle, we measure the degree that people chooses their next destination based on the distance. This can be estimated as follows. We first measure the *flight-to-nearest-waypoint* ratio. For a given flight from x to y, suppose k is the nearest unvisited waypoint from x. The flight-to-nearest-waypoint ratio is the ratio of $||x-y||$ over $||x-k||$. We then define the *least-action criterion*: for a give flight, it tests if its flight-to-nearest-waypoint ratio is less than some threshold. Figure 4 plots the percentage of flights meeting the least-action criterion in real traces for all participants when the threshold ratio (r) is less than 2. On average, 53% of flights meet the criterion. However, if we consider that humans are less sensitive to the distance when next destinations are all nearby. So if we exclude those flights whose length is less than a short distance (say 30 meters), we get more than 87%

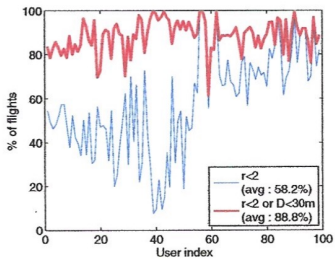


Fig. 4. For the GPS traces of 100 participants, we measure the percentage of flights meeting the least-action criterion. We also plot the case when we exclude those flights whose length is less than 30 meters.

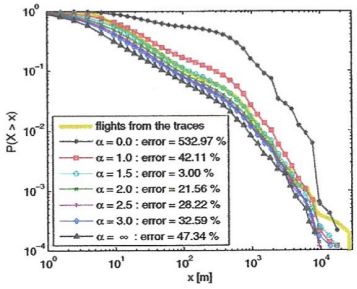


Fig. 5. Flight distributions obtained from LATP for different a values performed on top of waypoints extracted from the KAIST traces and the percentage difference between the sum of flights generated from LATP and that of flights from real traces

of flights meeting the criterion on average. We also tried other distances such as 20 meters and 10 meters and also varied r to a smaller ratio (1.5). All the measurements produce around 80% flights meeting the criterion. This indicates that most people in our traces use distance as an important metric for deciding the next waypoint, substantiating our conjecture.

We construct a new trip planning algorithm called *Least Action Trip Planning (LATP)* that given a set of waypoints to visit, decides the order in which a person visits. Algorithm 1 gives a pseudo-code of LATP. The algorithm selects a next waypoint to visit based on a weight function of $1/d^a$ where d is the distance from the current waypoint to an unvisited waypoint and a is a constant deciding the distance weight. If a is infinite, then the algorithm always chooses the nearest unvisited waypoint and if it is zero, then it randomly chooses the next waypoint. Figure 5 shows the flight distributions obtained from LATP for different a values performed on top of waypoints extracted from the KAIST traces and the percentage difference between the sum of flights generated from LATP and that of real flights from traces. In all cases, their difference is within a margin of around 10% when a is between 1 and 3. The figure shows that LATP generates traces very similar to the original traces.

D. Individual Walker Model

For a given input area S, our fractal waypoint generation generates a set W of the waypoints. We model an individual walker model that selects a subset of W and specifies the order