This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2010 proceedings.

Energy Efficient Assisted GPS Measurement and Path Reconstruction for People Tracking

Robin Wentao Ouyang*, Albert Kai-Sun Wong*, Mung Chiang†, Kam Tim Woo*,
Victoria Ying Zhang*, Hongseok Kim† and Xiaoming Xiao*
*Department of Electronic and Computer Engineering
The Hong Kong University of Science and Technology, Kowloon, Hong Kong
Email: oywtece@ust.hk, eealbert@ust.hk, eetim@ece.ust.hk, victoria@ust.hk, xiao@ust.hk
†Department of Electrical Engineering
Princeton University, Princeton, NJ 08544, USA
Email: chiangm@princeton.edu, hongseok@ieee.org

***Abstract*—In the use of a wearable GPS and cellular tracker for applications such as elderly tracking, device power consumption is an important consideration. To save power, assisted GPS (AGPS) location fixes should not be performed frequently. On the other hand, we also do not want to lose important information about the user's mobility patterns and routines. To solve this dilemma, in this paper, we present the design of a system that intelligently schedules on-line AGPS location fixes only when necessary based on information extracted from user's historical mobility data, and then reconstruct the user path based on these sparsely taken on-line location fixes. Experimental results show that our on-line algorithm can significantly reduce the number of AGPS fixes needed and the reconstruction method works well without a priori knowledge of a map and streets information.**

***Index Terms*—Assisted Global Positioning System (AGPS), tracking, energy efficiency, intelligent location fix, path reconstruction.**

I. Introduction

Location has played an important role in ubiquitous computing environments. There exist many techniques to determine a user's location (e.g., [1]–[3]), and the Global Positioning System (GPS) [1] is the most popular one for its convenience, precision and global availability. Information about the user location can enable numerous innovative applications of wireless systems and many compelling location-based services [4]–[6]. For example, Ashbrook and Starner [4] use GPS data to learn significant locations and predict movement across multiple users. Krumm and Horvitz [5] design a method that uses a history of a driver's destinations based on GPS data, along with data about driving behaviors, to predict where a driver is going as a trip progresses. Liao *et al.* [6] use a hierarchical Markov model and the particle filter to learn and infer a user's daily movements and high level behavior through an urban community.

In existing works, little attention has been paid to the issue of power saving in a wearable tracker device, and it is often assumed that the GPS device would report the user's current location (may include other measurements to form a state) very frequently - for example, once every second.

In our work on elderly tracking using Assisted GPS (AGPS) [7], social service organizations have expressed the desire to have a tracker that can work continuously for 2 weeks or more, as the targeted carrier of the tracker are elders with diminished capability who may not be capable of recharging the battery daily themselves. Even with AGPS, which is far more energy efficient than GPS, our previous work indicated that each AGPS fix would on average reduce the battery standby time by at least 5 minutes. Hence, in order for the device to work longer, we want the device to perform and report AGPS location fixes as infrequently as possible. At the same time, we also do not want to lose important information about the user's mobility pattern (such as the paths, speed and range of travels that needed to be recorded and can be used for abnormality detection). We therefore face a tradeoff between energy efficiency and tracking accuracy, controlled in part by deciding when to make each location fix.

To obtain an efficient and flexible tradeoff, in this paper, we present the design of a system that analyzes a user's historical data, intelligently schedules on-line AGPS fixes in order to save device power and then reconstructs the user's travel path. The complete process is as follows:

1) Apply filtering, clustering and pruning to densely collected historical data to reduce the number of data points and obtain a set of more representative locations;
2) Identify significant locations and estimate the transition probabilities among them;
3) Apply the information extracted for on-line AGPS fix scheduling;
4) Reconstruct the user's travel path from the sparsely collected on-line AGPS fixes.

For path reconstruction, although mapping and restricting the data points on streets can bring performance enhancements, map information is not easy to obtain, model and integrate into the system. Therefore, in this paper we assume that the historical data are processed without a priori knowledge about the real world (e.g., a map and the streets information).

The remainder of this paper is organized as follows. Section II presents the off-line preprocessing of historical data. Section III describes the on-line intelligent location fix scheduling algorithm. Section IV details the path reconstruction method.

978-1-4244-5638-3/10/$26.00 ©2010 IEEE

Experimental results are then shown in Section V. Finally, Section VI draws the conclusions.

II. Preprocessing

This section presents the preprocessing phase. The preprocessing is done off-line and utilizes historical tracking data which are collected by setting the GPS device in the tracking mode. The device will report the estimated latitude (ϕ_a), longitude (ϕ_o), date & time (dt) and an accuracy measure - the horizontal dilution of precision (HDOP) [1] at an interval of once every 2 seconds. The (ϕ_a, ϕ_o) pairs are then translated to the (x, y) coordinates in a local east, north coordinate system. The tracking mode is turned on as the device is first being carried by a user, until sufficient historical data are collected. Afterwards, the tracking mode can be turned on intermittently to produce updated historical data.

Based on the recorded GPS data, we first filter and cluster them into groups to produce a more concise data set. Then, we detect frequently visited locations (FVLs) and crucial locations (CLs; for path reconstruction purpose) to form a set of significant locations (SLs). We assume the transitions among these SLs obey a first order Markov model, and then estimate the corresponding transition probabilities.

A. Filtering, Clustering and Pruning

The HDOP is related to the horizontal standard deviation of the location estimation. A large reported HDOP indicates that the corresponding location estimation is not good enough. Therefore, upon obtaining historical GPS data, we first filter out those bad estimates based on the associated HDOP values. In our experiments, we retain only those GPS data whose HDOP values are less than $thre_h = 20$ for training.

For those retained training data, due to GPS measurement errors, two GPS location fixes taken at different time epochs can still vary as much as 15 meters even when the user stands at the same location. To reduce the measurement error and obtain a more concise and representative data set, we then group those training data into clusters. We use a variant of the k-means clustering algorithm [4].

The idea is to take a location point $P = [x, y]^T$ and a radius R. The mean of all the location points within the circle centered at P with radius R is calculated as M. Then M is taken as the new center of the circle and the process is repeated. This process terminates when the mean M stops changing. M is then regarded as a new representative location and a unique identity I_i is assigned to it. Then all the location points within its radius are marked and removed from consideration for the remaining clustering. The clustering continues until no recorded location points remain and finally we obtain a set of new representative locations $\{I_i\}$ (cluster centers), the number of which is much smaller than that of the training location points. The starting point P for each cluster is chosen as the one with the smallest dt value in the remaining training data (not have been clustered yet).

After clustering, there may still exist some outliers. Therefore, we finally prune out these clusters (as well as the corresponding training data) which contain less than $thre_n$ training points. We denote the remaining cluster centers as L_i's.

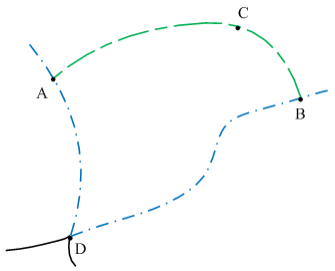


Fig. 1. An illustration of crucial locations.

B. FVL Detection

It is reasonable to consider a location as a frequently visited one by a user if the user spends long enough time there [4]. For example, a bus stop, a supermarket, his home, his office and his friends' homes.

For each path formed by consecutive location points in the pruned training set, we examine whether it satisfies either of the following two conditions.

1) Possible outdoor FVL: There exist consecutive location fixes inside a cluster and the time duration from the first location fix to the last one exceeds $thre_{t1}$. This indicates the user is waiting or hovering there.
2) Possible indoor FVL: The GPS signals are lost for at least time interval $thre_{t2}$ after the last point (belonging to a cluster) of a path. This indicates the user has entered into a building or a tunnel (if the user does not turn off the GPS device).

The FVLs are then detected on a cluster basis. That is, if either of the two conditions is satisfied, the corresponding counter C_i (initialized as zero) for the cluster center L_i involved will be increased by one. After examining all the paths in the training set, those L_i's whose C_i's exceed certain threshold will be detected as FVLs.

For an indoor FVL, since L_i is a cluster center, it will not center at the end points of the paths in the training data (very close to the corresponding building) but be a little far away from the location of the corresponding building.

C. CL Detection

We define a CL as a location where there exist more than one route before or after it. A CL informs us that we need to take at least one location fix before arriving at it or after passing though it in order to unambiguously determine the user path. For example, in Fig. 1, locations A and B are two crucial locations. From A to B, a user can either goes through $A - C - B$ or $A - D - B$. Only knowing that the user has gone from A to B cannot unambiguously determine the user path. Therefore, making a location fix after A and before B is very important to distinguish the route that the user has taken.

978-1-4244-5638-3/10/$26.00 ©2010 IEEE

To detect those CLs, a transition indicator matrix $\mathcal{B} = [b_{ij}]$ is created where $b_{ij} = 1$ if there exists transition from L_i to L_j, and $b_{ij} = 0$ otherwise. We define $b_{ii} = 0$. It should be noted that b_{ji} is not necessarily equal to b_{ij} since the route may be a uni-directional lane.

Here, we regard there exists transition from L_i to L_j if in the training set, there exists transition from a location belonging to the cluster centered at L_i to a location belonging to the cluster centered at L_j and $\|L_i - L_j\| \leq 2R$.

L_i is detected as a CL if either the summation of the ith row or the ith column of $\mathcal{B}$ is greater than or equal to 3. The reason of not setting this threshold to 2 is that every L_i has a chance to have two incoming or outgoing neighbors if the route is bi-directional.

We now term the collection of FVLs and CLs as SLs, and assign each of them a unique identity S_i. The number of S_i's will be much smaller than the number of L_i's.

D. Transition Probability Matrix

We consider these SLs S_i's as states and assume the transitions among them follow a first order Markov model. Then we can estimate the transition probability matrix $\mathcal{A} = [a_{ij}]$ (probability that S_j is followed by visiting S_i) for those states (we define $a_{ii} = 0$). We have

$$a_{ij} = \frac{\text{number of transitions from } S_i \text{ to } S_j}{\text{number of transitions from } S_i}.$$

We regard there exists transition from S_i to S_j if in the training set, a location belonging to the cluster centered at S_i can lead to a location belonging to the cluster centered at S_j without visiting other SLs.

The transition probability matrix $\mathcal{A}$ is used for movement prediction, e.g., if the user is now around an SL, what is the next SL he is likely to visit.

E. Possible Routes between SLs

For each $a_{ij} > 0$, we determine all the possible routes from S_i to S_j in terms of a sequence of L_i's from the paths in the training set. We denote each possible route from S_i to S_j as R_{ij}^k where k is from 1 to its possible maximum value. Since the transition probability matrix represents first order transitions, each R_{ij}^k indicates a possible route between neighboring SLs, which means besides the beginning and ending points of R_{ij}^k, it does not involve any SL intermediately. In addition, an ordinary L_i (not an SL) should appear on only one possible route. Therefore, some R_{ij}^k's should be merged into one possible route if they contain a common ordinary L_i. We denote the finally obtained possible routes from S_i to S_j as P_{ij}^k's.

III. On-line Scheduling

To save power, we set the device to make an ordinary on-line location fix once every $thre_t$ seconds, where $thre_t \gg 1$. Therefore, the tracking approaches based on the Kalman filter [8] and the particle filter [9] do not work in such a scenario. As a result, localization technique is used instead to determine the location of the user at each intended time epoch. In order to obtain important information about the user traveling, besides making an ordinary location fix every $thre_t$ seconds, the GPS device is also scheduled to make additional location fixes when necessary.

To guarantee the estimation accuracy, we allow the GPS device to make at most 5 consecutive location fixes at an interval of once every 2 seconds when it wakes up at a scheduled time epoch. Among the possible 5 consecutive location fixes, once the HDOP of a location fix is less than 20, that location fix is recorded (previous location fixes are discarded, if any) and the consecutive location fix process stops. Otherwise, the location fix with the smallest HDOP among the 5 consecutive fixes is recorded (others are discarded). That is to say, every recorded on-line location fix has relatively high accuracy. Hereafter, we refer to an on-line location fix as one that satisfies the above condition.

In the on-line phase, after turning on the GPS device, it makes a location fix and compares it with those SLs in the database. If the closest SL is within $thre_d$ meters, then this SL is determined as the starting point for this travel. Otherwise, this travel will be considered as a novel one which does not follow historical routes and the GPS device will work in the tracking mode (e.g., make a location fix once every 2 seconds) in order to record the user's novel behavior until an SL which satisfies the above condition is detected.

After determining the starting point (say, S_i), the GPS device looks up the transition probability matrix and find those a_{ij}'s which are greater than or equal to a threshold $thre_p$. The corresponding S_j's are then listed. The minimum time t_{ij}'s (there may exist more than one route from S_i to S_j) that will be used to travel from S_i to each of those S_j's are also estimated (by estimating the average time used from training data or by estimating the average velocity of the user from the previous location fixes). If $\min(t_{ij}) < thre_t$, the GPS device should make the next location fix after $\min(t_{ij})$. Otherwise, the GPS device should make the next location fix after $thre_t$ seconds.

After making a location fix (say, O_m) on the way from S_i to other SLs, the distances d_{mk}'s from O_m to the cluster centers L_k's that are around O_m are calculated. Assume the minimum distance obtained is d_{mj}. O_m is determined to be associated with L_j if $d_{mj} \leq thre_d$. The device then searches from the possible routes which starts from S_i and passes through L_j (detected off-line). The end point for the corresponding route is determined as the next possible SL (only one; say S_n). Though there may exist multiple routes from S_i to S_n, one that passes through L_j is unique since L_j is not an SL. If $d_{mj} > thre_d$, the tracking mode is turned on in order to record the user's novel behavior. It is turned off when an SL which satisfies the aforementioned condition is detected.

After determining S_n, the corresponding time variable t_{in} is updated, which represents the estimated time left to travel from current location to S_n. Similarly, if $t_{in} < thre_t$, the GPS device should make the next location fix after t_{in}. Otherwise, the GPS device should make the next location fix after $thre_t$ seconds. As long as $t_{in} > 0$, after each following on-line

location fix, the device may check again whether the next possible SL is still S_n.

When the distance between an on-line location fix and the target S_n is within $thre_d$ meters, the user is considered to be around S_n. Then S_n serves as the new starting point. The above process repeats until the GPS signals are lost for $thre_{t2}$ seconds.

To further save device power, the GPS device may send its location fixes to a remote server and the scheduling is done there. The remote server then sends signals to trigger the GPS device to make a location fix when necessary.

IV. Path Reconstruction

In the on-line phase, the obtained location fixes are taken sparsely, both in terms of time and space. Therefore, we need to interpolate some location points between consecutive on-line location fixes to obtain a more detailed path.

Using cluster centers L_i for interpolation is a good choice. Since each L_i is calculated as the average of all the training locations that fall into the corresponding cluster, its variance is lowered. As long as the cluster radius is small enough that the real path it covers can be considered as a line segment, for large enough training data, L_i will finally converge to the geometric center of the cluster, i.e., the center of the circle as well as the center of the real path, which means L_i is a good representative location. However, for large cluster radius, L_i may converge to a location that the user never visited.

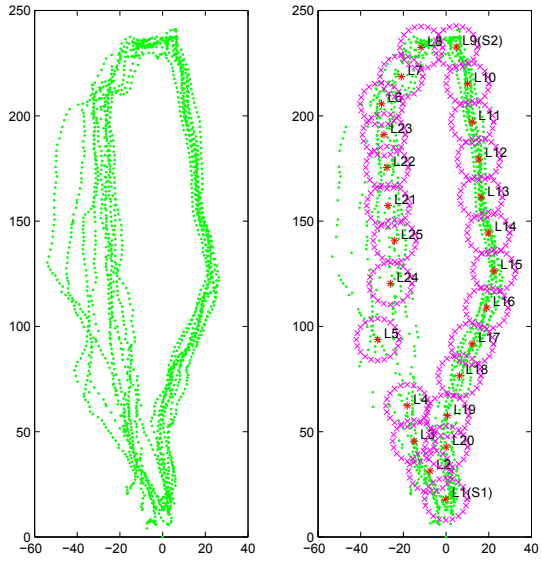
In the on-line phase, after obtaining a location fix O_m on the way from an SL to another, we have used O_m to detect the corresponding route and the target SL. Then the concatenation of those routes between consecutive SLs (in terms of L_i's) will form a more detailed reconstructed user path. Due to GPS measurement errors, even when the user is on the same route, associating different on-line location fixes (made at different time epochs) to a nearby cluster center may indicate different routes. In order to determine a unique route, a majority vote is then taken.

V. Experimental Results

To assess the performance of our proposed approaches, we collected GPS data from one person wearing a GPS device in 5 days. He went to a restaurant from his dorm and then went back after dining twice every day. Finally, we collected 10 repetitions of each route he took. We used them as historical data to extract useful information. After processing historical data, we implemented our on-line location fix algorithm on the GPS device. At the same time, another GPS device which performed regular tracking (once every 2 seconds) was also taken by the user as a reference. We collected such GPS data for 2 days. We take some representative results to show and explain. The figures below are all in a local 2-D east, north coordinate system and the unit used is meter.

A. Preprocessing

As shown in Fig. 2, we denote the historical location data as green dots. It can be seen from Fig. 2 (a) that on the left



(a) (b)

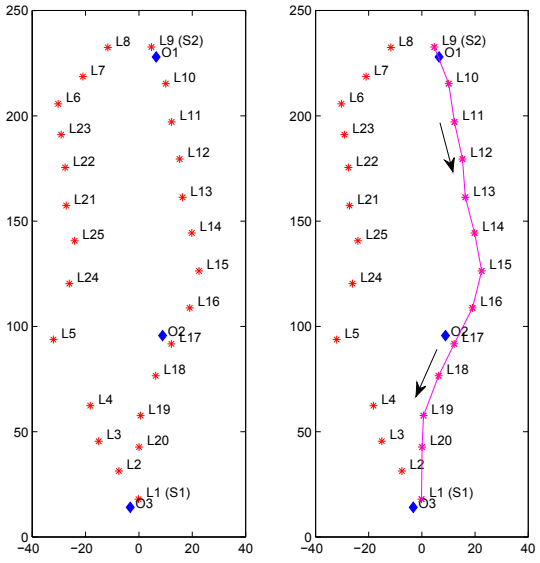
Fig. 2. (a) Historical data before preprocessing. (b) Historical data after filtering and cluster centers obtained after pruning.

half of the figure, there seems to exist a lot of different routes. However, there is only one route there. Multiple routes shown in historical data are caused by GPS measurement errors. We then perform filtering, clustering and pruning as stated in Section II (we set $R = 10$ meters and $thre_n = 15$), and finally we obtain 25 cluster centers as shown in Fig. 2 (b) where they are denoted as red stars and the boundaries of clusters are denoted as magenta crosses. Since we filter out some data points based on the HDOP values, while choose the starting point for a cluster center according to the dt values, we can observe that the cluster number assigned may not be continuous between neighboring clusters.

In Fig. 2 (b), L_1 and L_9 are detected as indoor FVLs. Actually, L_1 is close to the user's dorm and L_9 is close to the restaurant. These two locations are possible starting and ending points of a route. Due to the setting of R, two routes where the shortest distance between them is less than $2R$ can not be distinguished in that part. For example, L_2 covers two different routes since they are too close. In consequence, L_2 is detected as a CL, because from L_2, a user can go to L_1, L_3 and L_{20}. Since L_2 is adjacent to L_1 and the transition probability from L_1 to L_2 is 1, to simplify the on-line scheduling, we do not consider L_2 as an SL, but treat it as an ordinary cluster center which resides on the left route. Therefore, altogether we have two SLs : $S_1 = L_1$ and $S_2 = L_9$. From S_2 to S_1, there exist two possible routes: one is $P_{21}^1 = [S_2, L_8, \ldots, L_{22}, \ldots, L_5, \ldots, L_2, S_1]$, and the other is $P_{21}^2 = [S_2, L_{10}, \ldots, L_{15}, \ldots, L_{20}, S_1]$. Similarly, from S_1 to S_2, we also have two possible routes.

B. Intelligent On-line Location Fix

In the on-line phase, we set $thre_t = 120$ seconds and $thre_d = 15$ meters. The user started from S_2, and the



(a) (b)

Fig. 3. (a) On-line location fixes. (b) Path reconstruction.

destination is S_1. After the travel started, as shown in Fig. 3 (a), the first on-line location fix was O_1. It was then determined to be associated with S_2, and thus S_2 was the starting point. From the transition probability matrix, the next possible SL was S_1, and the minimum time t_{21} estimated (from historical data) to travel from S_2 to S_1 was 206 seconds, which was greater than $thre_t$. Therefore, the GPS device was scheduled to make the next location fix after $thre_t$ seconds, resulting in O_2. O_2 was then determined to be associated with L_{17}. We could then find the possible route the user took by checking P_{21}^k's which starts from S_2 and passes through L_{17}. We found that the end point for this route was S_1. t_{21} was then updated. It was smaller than $thre_t$ then, and the next location fix was scheduled to make after t_{21}, resulting in O_3. Since L_1 was a little far away from the location of the corresponding building, when taking O_3, the device could still detect GPS signals. O_3 was then determined to be associated with S_1. The next possible SL was then S_2. However, the GPS signals were then lost for $thre_{t2}$ seconds, which meant the travel terminated.

After traveling from S_2 to S_1, our reference GPS device (performing regular tracking) reported altogether 105 location fixes. In contrast, by our intelligent scheduling algorithm, only 3 location fixes were performed. Generally, the fewer location fixes we make, the more energy we save. To further save device energy, the scheduling can be made on a remote server.

C. Path Reconstruction

Finally, we reconstruct the user path by the method in Section IV based on our on-line recorded data points. The on-line location fixes indicated from S_2 to S_1, the user took the route which passed through L_{17}. We can then find the corresponding route from our off-line extracted information, which is P_{21}^2, and we connect the cluster centers which form it by magenta line segments as shown in Fig. 3 (b). This is then the reconstructed user path between two neighboring SLs. As has been mentioned in Section IV, if needed, a majority vote process can be done here to determine a unique route. For more complicated scenarios, the concatenation of all the routes between neighboring SLs then forms a more detailed reconstructed user path.

For reconstruction accuracy calculation, we use only those reference GPS location points whose HDOP values are less than or equal to 20. Assume altogether we collected N such points (denoted as D_i's) from the reference GPS device which performed regular tracking, then the accuracy of the reconstructed path is defined as

$$Accu = \frac{\sum_{i=1}^{N} I(\min_j \|D_i - L_j\|)}{N},$$

where the norm is the Euclidean norm, L_j's are the cluster centers which form the reconstructed path, $I(x) = 1$ if $x \leq 5$ and $I(x) = 0$ otherwise. $Accu$ measures the proportion of reference GPS location points whose distance to its closest reconstructed location point is less than or equal to 5 meters. Obviously, $0 \leq Accu \leq 1$, and the larger the better. For the reconstructed path shown in Fig. 3 (b), its $Accu = 0.92$.

VI. Conclusion

In order to save GPS device power while do not lose important information in tracking applications, this paper proposes to intelligently make on-line GPS location fixes only when necessary and then reconstruct a more detailed path later. The intelligence is achieved from extracting important information from a user's historical data. The proposed intelligent location fix algorithm greatly reduces the number of GPS points needed to take on line and the reconstruction method works well without a priori knowledge of the real word.

References

[1] E. Kaplan and C. Hegarty, *Understanding GPS: Principles and Applications Second Edition*, 2006.

[2] V. Zhang and A.-S. Wong, "Combined AOA and TOA NLOS localization with nonlinear programming in severe multipath environments," *Wireless Communications and Networking Conference, 2009. WCNC 2009. IEEE*, pp. 1–6, 2009.

[3] R. W. Ouyang, A. K.-S. Wong, C.-T. Lea, and V. Y. Zhang, "Received signal strength-based wireless localization via semidefinite programming," *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*, pp. 1–6, 2009.

[4] D. Ashbrook and T. Starner, "Using GPS to learn significant locations and predict movement across multiple users," *Personal and Ubiquitous Computing*, vol. 7, no. 5, pp. 275–286, 2003.

[5] J. Krumm and E. Horvitz, "Predestination: Inferring destinations from partial trajectories," *Lecture Notes in Computer Science*, vol. 4206, pp. 243–260, 2006.

[6] L. Liao, D. Patterson, D. Fox, and H. Kautz, "Learning and inferring transportation routines," *Artificial Intelligence*, vol. 171, no. 5-6, pp. 311–331, 2007.

[7] A.-S. Wong, T. K. Woo, A.-L. Lee, X. Xiao, V.-H. Luk, and K. W. Cheng, "An AGPS-based elderly tracking system," *Ubiquitous and Future Networks, 2009. ICUFN 2009. First International Conference on*, pp. 100–105, June 2009.

[8] S. Kay, *Fundamentals of statistical signal processing: estimation theory*, 1993.

[9] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on signal processing*, vol. 50, no. 2, pp. 174–188, 2002.