Received: 10 October 2021 | Revised: 23 July 2022 | Accepted: 12 September 2022

DOI: 10.33012/navi.569

ORIGINAL ARTICLE

Optimized Position Estimation in Mobile Multipath Environments Using Machine Learning

Nesreen I. Ziedan

Computer and Systems Engineering Department, Faculty of Engineering, Zagazig University, Zagazig, Egypt

Correspondence
Nesreen I. Ziedan
Email: ziedan@ieee.org

Abstract

The positioning accuracy of global navigation satellite system receivers is frequently degraded in urban areas due to reflected signals. A moving receiver faces additional challenges because it needs to adjust to changes in the statuses of the signals received, including line-of-sight (LOS), multipath, non-LOS, or invisible. This paper proposes two new algorithms that can be used to enhance the accuracy of a moving receiver. The first algorithm is called Optimized Position Estimation (OPE). The OPE algorithm estimates the most likely paths and identifies the one with the optimal weight. The second algorithm is called Intelligent Signal Status Estimation (ISE). The ISE algorithm utilizes a self-organizing map machine-learning algorithm to estimate the probability of a change in signal status. The algorithms are tested using global positioning system C/A signals, which have over 50 changes in their statuses. The results obtained using these algorithms reveal that the accuracy is enhanced by as much as 96.3% (i.e., a 27-fold improvement) when compared to results using a conventional navigation algorithm.

Keywords

machine learning, multipath, self-organizing map, tracking, urban area

1 | INTRODUCTION

Multipath signals are a major source of positioning errors in urban environments. Global navigation satellite system (GNSS) signals often encounter obstacles on their path from a satellite to a receiver, including buildings, structures, and vegetation. These obstacles can obstruct, reflect, or diffract the signals to be received and may result in multiple versions of the same signal arriving at a receiver via different paths and with different code delays (McGraw et al., 2020). Therefore, multipath signals arriving with delays within 1.5 chips of the line-of-sight (LOS) signal correlate with the local replica signal and can lead to distortion in the cross-ambiguity function (CAF; Ziedan, 2014). A distorted CAF can lead to tracking and positioning errors. Positioning errors can also occur when the LOS signal is completely blocked and only non-LOS (NLOS) signals arrive at the receiver.

When processing GNSS signals, a conventional tracking module does not distinguish between LOS and multipath or NLOS signals. Therefore, a receiver continues to track a signal regardless of its status and generates pseudorange measurements for the navigation solution (Steingass et al., 2017; Ziedan, 2017). Erroneous pseudorange measurements that are contaminated by reflected signals can degrade positioning accuracy. One popular set of approaches used to overcome these errors focuses on the identification of multipath and NLOS signals and excludes these measurements from the positioning computation. These approaches range from simple integrity monitoring and consistency checking (Hwang & Brown, 2006; Parkinson & Axelrad, 1988; Yu, 1998) to those that involve more complex processing (Hsu et al., 2015; Jiang et al., 2011; Smolyakov et al., 2020; Wen et al., 2019; Ziedan, 2018). Some approaches use specifically designed receiver architecture to identify and exclude erroneous signals (Hsu et al., 2015; Smolyakov et al., 2020; Xu et al., 2020). While these approaches can be effective in increasing the positioning accuracy, discarding satellites can lead to poor geometry (Zhang & Zhang, 2009), an increase in the geometry dilution of precision (GDOP), and thus degraded positioning accuracy. Discarding data from contaminated satellites may also result in an insufficient number of satellites available for positioning computations.

Another set of approaches mitigates the multipath effect without discarding the contaminated satellites. These approaches usually work at the signal processing level. The earliest techniques that used this method were the Multipath Estimating Delay Lock Loop (MEDLL; van Nee, 1992) and the Double Delta approach (Garin & Rousseau, 1997; Garin et al., 1996). However, these approaches are not effective when dealing with multipath signal delays close to the LOS signal delay. Signals in urban multipath environments frequently include multiple reflections with a variety of path delays. More advanced methods include filter-based approaches (Closas et al., 2009; Giremus et al., 2007; Steingass et al., 2017; Ziedan, 2011) which can adjust to the presence of a small number of reflected signals before the processing overhead becomes a hindrance. Other approaches designed to address a relatively small number of reflected signals are those based on sparse estimation theory (Lesouple et al., 2019).

A recently developed category of approaches utilizes information from the surrounding environment for multipath mitigation. Three-dimensional (3D) building models, mapping, and ray tracing can be used to generate predictions regarding satellite status and the path lengths of the reflected signals. Examples of approaches that employ these predictions include shadow matching (Groves et al., 2020; Wang et al., 2013; Yozevitch & Ben-Moshe, 2015), ray tracing (Bradbury et al., 2007; Hsu et al., 2016; Lau & Cross, 2007; Miura et al., 2013; Ziedan, 2017), and neural networks (Li et al., 2019). While these approaches perform well in static multipath situations, they may not be fully adequate when used in dynamic mobile environments (Ziedan, 2022).

A receiver that is moving in an urban environment faces additional challenges for multipath mitigation. The status of the signals received is not constant and may change while the receiver is in transit. A signal can transition between four states, including LOS, multipath, NLOS, or invisible. For example, a LOS signal can become an NLOS signal when the surrounding structures block and reflect what was a direct signal. A tracking module can follow changes in these parameters and can continue to track an NLOS signal. Therefore, the objective of this paper is to develop an algorithm that utilizes both multipath and NLOS signals in position computations. The algorithms developed using this approach will be able to adapt to changes in the received signal status by identifying the transition from one status to another and adjusting the output accordingly.

Two algorithms are proposed. The first algorithm is called Optimized Position Estimation (OPE). OPE estimates the most likely sequences of positions on a map, where each sequence forms a path. OPE then identifies the path with the optimal weight. The second algorithm, which is called Intelligent Signal Status Estimation (ISE), contributes to the computation of the optimal weight. The ISE algorithm estimates the probability that a received signal will undergo a change in status and uses this information to compute the probability of transitions between these positions. Signal status changes based on the appearance or disappearance of LOS or reflected signals.

The ISE algorithm employs a supervised self-organizing map (SOM) neural network machine-learning algorithm (Kohonen, 1990) to compute the probability of a change in signal status. The features of the SOM network are extracted from a tracking module. The probabilities of status changes of the satellites above the horizon are used to compute the probabilities of a transition between positions. This information is then used to compute the weight function of the OPE algorithm. A conventional SOM network uses unsupervised learning and clusters inputs based on their features (Vesanto & Alhoniemi, 2000). By constrast, the proposed SOM network uses a supervised learning approach to estimate the probability of a change in signal status.

In a recent publication by Ziedan (2020), two position estimation algorithms that could be used in multipath environments were proposed. The first algorithm is called Map Matching with Tracking Feedback (MMTF). The MMTF algorithm finds the most likely candidate position on a map based on predictions from the received signal. The second algorithm is called Adaptive Position Estimation (APE). The APE algorithm uses output from the MMTF algorithm to compute code delay errors due to signal reflection and diffraction and removes them before proceeding to compute a position estimate. As observed by Ziedan (2020), the accuracies of the MMTF and APE algorithms differed from one another. In some cases, these differences were not small.

The proposed OPE algorithm integrates the MMTF and APE algorithms to achieve an optimized estimation of the signal position. The OPE algorithm extends the estimation to the most likely path instead of the most likely position. As described by Ziedan (2020), the status of a signal was estimated using a Gaussian-like function that generated a probability based on the estimated change in the carrier-to-noise ratio (C / N_0). In this paper, the machine learning ISE algorithm utilized several additional features to estimate the signal status and did not rely on the C / N_0 alone.

While the proposed OPE algorithm does not operate at the signal processing level, it does require information from the tracking module. Therefore, it can operate with any receiver as long as the tracking module can provide the necessary information (e.g., estimates of the C / N_0). Furthermore, because the OPE algorithm estimates the most likely path, it can be classified as working with multiple epochs.

The remaining sections of this paper are organized as follows. First, the design of the OPE algorithm is presented; this is followed by the introduction of the ISE algorithm. The selection of specific features is explained in detail, followed by an explanation of the design of the SOM algorithm. The experiment and results are then provided. Finally, the summary and conclusions are discussed.

2 | OPTIMIZED POSITION ESTIMATION (OPE)

The OPE algorithm developed for this study keeps track of the number (N_g) of the most likely paths over N_n steps. Each path has an associated weight, or W^g. After N_n steps, the path with the optimal weight is selected as the estimated path and the positions that constitute the estimated path are identified as the estimated positions.

There are three functions that contribute to the determination of path weight (W^g). Various available data are utilized to determine path weight. 3D building models and the accelerated ray tracing algorithm (ART) developed as described in Ziedan (2017) are used to facilitate predictions of reflected signals and their path lengths. Analysis of information extracted from the tracking module is used to assess the probability of a receiver transitioning to a position based on the predictions of reflected signals. Therefore, the first function is computed from the transition probability, $U_{E^g_{n-1}\rightarrow E^g_n}$, between two consecutive positions, E^g_{n-1} and E^g_n, along a path, q^g. $U_{E^g_{n-1}\rightarrow E^g_n}$, using the developed machine-learning ISE algorithm.

The first function alone is not always sufficient because more than one position can yield similar predictions of reflected signals. The second function circumvents this issue by computing the probability that the E^g_n position is correct. This is done by accounting for the impact of the predicted code delays on pseudorange measurements and comparing the computed navigation solution to E^g_n. Therefore, the second function is computed from the difference between E^g_n and an estimated position calculated by the APE algorithm as described by Ziedan (2020).

The third function is computed from the distance between E^g_{n-1} and E^g_n based on the estimated velocity. The objective of the third function is to minimize jumps in the estimated positions that may occur if there are more than one E^g_n positions that yield similar transition probabilities, $U_{E^g_{n-1}\rightarrow E^g_n}$, even if these positions are not adjacent to one another. Figure 1 provides an outline of the functions that contribute to the computation of the path weight.

Each most likely path, q^g, consists of N_n most likely sequences of positions, E^g_n. The E^g_n positions determined at a step n are based on a set of candidate positions, C^j_n. In this case, E^g_n are a subset of C^j_n that have the highest transition probabilities from E^g_{n-1} to C^j_n, as computed by the ISE algorithm and the distance between E^g_{n-1} and C^j_n. Using the APE algorithm described by Ziedan (2020) for each position E^g_n, the estimated signal status, $\eta^{g_s}_n$, for a satellite s is used to estimate the code delay error, λ^s_n. The estimated λ^s_n is removed from the code delay estimate, $\Delta\tau^s_n$, and then a position B^g_n is computed. The difference between the most likely position E^g_n and the position B^g_n, computed as $||B^g_n - E^g_n||$, is used to compute a probability D^g_n. Following this computation, D^g_n and the transition probabilities are used to update the weight function, W^g_n. After N_n steps, the path with the optimal weight is identified as the estimated path.

There are two types of estimated positions. The first type is estimated from candidate positions that are located at the center of predefined discrete cells on a map. These are the E^g_n of these positions. The second type is computed from a navigation solution. These estimated positions have continuous values and are not restricted to predefined discrete cells. Therefore, the estimated path is based on two sets of estimated positions. The first set is composed of the E^g_n positions,

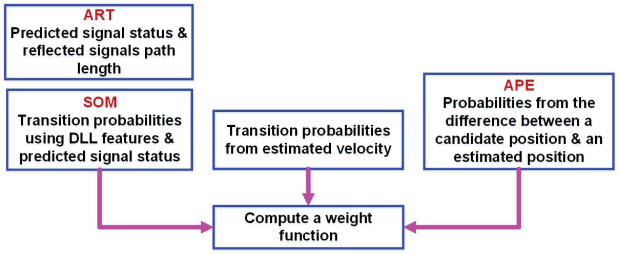

FIGURE 1 Outline of the functions that contribute to the computation of the path weight

which are taken from the discrete map cells C_n^j; this set of positions is called OPE map matching (OPE-MM). The second set includes the corresponding B_n^g positions, which are calculated from a navigation algorithm; this set is called OPE navigation (OPE-NAV).

Figure 2 provides an overview of the OPE algorithm. The algorithm can be summarized as follows:

1. An initial position estimate, P_0, is obtained. The most likely position E_0^g is set as P_0. The initial path consists of only E_0^g, with weight $W_0^g = 0$.
2. Candidate positions, C_n^j, located in the area around E_{n-1}^g are determined.
3. At each candidate position, C_n^j, the ART algorithm described by Ziedan (2017) is used to predict the signal status (i.e., LOS, multipath, NLOS, or invisible) of each satellite, s, above the horizon. The ART algorithm also provides predictions for reflected signal path lengths.
4. The transition probability, $U_{E_{n-1}^g \to C_n^j}$, between each E_{n-1}^g and C_n^j, is then computed using the ISE algorithm developed in this study in which $U_{E_{n-1}^g \to C_n^j}$ represents the probability that the predicted signal status identified above in Step 3 is correct.
5. Another transition probability, $V_{E_{n-1}^g \to C_n^j}$, is then computed based on the estimated velocity for the path q^g as follows:

5.1 A value less than 1 is defined as κ and an average distance is defined as d_{av}; both κ and d_{av} are tuning parameters. Using this method, one can assign a probability between κ and 1 to candidate positions that are separated from a predicted position, $\tilde{P}_n^g$, by a distance of less than d_{av}. As the distance increases beyond d_{av}, the probability will decrease more rapidly. This is addressed below in Steps 5.2 through 5.6.

5.2 The predicted position, $\tilde{P}_n^g$, at step n and path g, is computed as

$$\tilde{P}_n^g = E_{n-N_p}^g + \hat{v}_n^g \; T_{N_p} \tag{1}$$

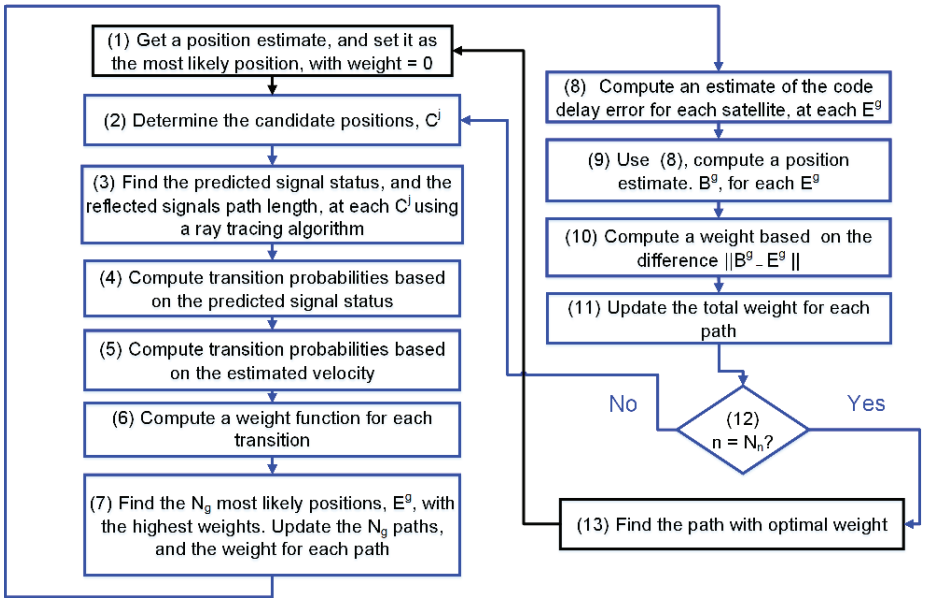

FIGURE 2 Overview of the developed OPE algorithm

where $\hat{v}_n^g$ is the estimated velocity for path g, as per the APE algorithm as described in Step 9. N_p is a number of steps and is also a tuning parameter. T_{N_p} is the time between step $n - N_p$ and n.

5.3 The distance between $\tilde{P}_n^g$ and each candidate position, C_n^j, is computed as $d_n^{g\,j}$. The minimum distance between $\tilde{P}_n^g$ and all the C_n^j positions is identified as $\tilde{d}_{min}^g$.

5.4 The following difference is computed: $\Delta\tilde{d}^g = d_{av} - \tilde{d}_{min}^g$.

5.5 The ratio $\tilde{R}_d^g$ is computed as

$$\tilde{R}_d^g = \frac{1-\kappa}{\Delta\tilde{d}^g} \tag{2}$$

5.6 $V_{E_{n-1}^g \to C_n^j}$ is

$$V_{E_{n-1}^g \to C_n^j} = \begin{cases} 1-(d_n^{gj} - \tilde{d}_{min}^g)\tilde{R}_d^g, \text{ for } \tilde{d}_{min}^g \le d_n^{gj} \le d_{av} \\ \kappa-(d_n^{gj} - \tilde{d}_{min}^g)\tilde{R}_d^g, \text{ for } d_n^{gj} > d_{av} \end{cases} \tag{3}$$

6. The total transition probability is

$$\gamma_{E_{n-1}^g \to C_n^j} = U_{E_{n-1}^g \to C_n^j} \quad V_{E_{n-1}^g \to C_n^j} \tag{4}$$

where, $\gamma_{E_{n-1}^g}$ is used to update the weight function, W_n^j, for each candidate position as

$$W_n^j = W_{n-1}^g + w_t\, \gamma_{E_{n-1}^g \to C_n^j} \tag{5}$$

where, W_{n-1}^g is the total weight for path g at step $(n-1)$, and $w_t < 1$ is the contribution of a transition probability to a path weight.

7. The N_g most likely positions, E_n^g, are determined as the candidate positions, C_n^j, with the highest W_n^j. The weights, W_n^g, are set as W_n^j of the N_g most likely positions. The N_g paths are updated by appending E_n^g to each corresponding path, q_n^g.

8. Based on the signal status determined at each E_n^g, errors in the code delays are estimated. For example, an NLOS signal would generate an error in the code delay computed based on the extra path length of the NLOS signal compared to the path length of the LOS signal; the NLOS signal extra path length is obtained using the ART algorithm described in Step 3. More details on this calculation can be found in Ziedan (2020).

9. For each E_n^g, the APE algorithm described by Ziedan (2020) is applied to compute a position, B_n^g, and a velocity, $\hat{v}_n^g$.

10. A probability, D_n^g, is computed based on the difference $\left\| B_n^g - E_n^g \right\|$, where

$$D_n^g = \frac{1}{\sqrt{\pi\sigma_r}} \exp\left(-\left(\frac{\left\| B_n^g - E_n^g \right\|}{\sigma_r} \right)^2 \right) \tag{6}$$

with σ_r as a standard deviation. This is also a tuning parameter.

11. The weight function, W_n^g, is updated as

$$W_n^g = W_n^g + (1 - w_t) D_n^g \tag{7}$$

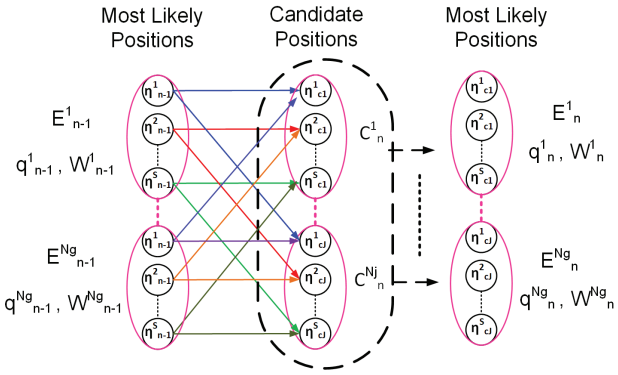

FIGURE 3 Illustration of the transition from the most likely positions, E_{n-1}^{g}, to the candidate positions, C_n^j, and then to the next E_n^g; each oval represents one position. The circles inside the ovals represent the statuses of the satellites above the horizon at that position.

12. If the current step number n is equal to N_n, then Step 13 is executed. Otherwise, the algorithm returns to Step 2.
13. The estimated path is identified as the path with the optimal weight. The estimated positions are set as the positions that constituted this estimated path. The position located at the end of the estimated path is used as an initial position, and the algorithm returns to Step 1.

Figure 3 illustrates the transition from the most likely positions, E_{n-1}^{g}, to the candidate positions, C_n^j, and then to the following E_n^g. Each oval represents one position. The circles inside each oval represent the statuses of the satellites above the horizon at that position.

3 | INTELLIGENT SIGNAL STATUS ESTIMATION (ISE)

The ISE algorithm uses several features to estimate the probability of transition between positions. This probability is computed from the probabilities of change in the signal status of the satellites above the horizon.

A probabilistic and supervised self-organizing map (SOM) neural network algorithm is introduced to estimate the probability of change in a signal status. SOM is a machine-learning approach used for data clustering (Kohonen, 1990; Vesanto & Alhoniemi, 2000). A conventional SOM is an unsupervised approach that can be used to classify input data into clusters with similar features. Several publications have described modified supervised and probabilistic approaches that use SOM functionalities to address problems in different fields (Barreto & Araujo, 2004; Papadimitriou et al., 2001; Polzlbauer et al., 2008; Song et al., 2007). A SOM consists of an input layer and an output layer. The input layer represents a number of N_{nf} features. The output layer transforms the input data into 1D or 2D maps with a total of N_{nu} neurons. A SOM involves both training and classification phases. Interested readers are referred to Kohonen (2001) for comprehensive details about SOM algorithms and their applications.

The features used as input for the SOM network include four features derived from the output of the delay-locked loop (DLL) and two features derived from the change in the C / N_0. Although the change in the C / N_0 was shown to be effective in estimating the transition probability in Ziedan (2020), machine learning was not used in that case.

3.1 | ISE Features

The ISE features are derived from the output of the tracking module. These features and their significance are discussed in this section.

The objective of using a SOM is to quantize the effect of a change in a signal status based on a probability by using various features. The probabilities of status changes of the satellites above the horizon are used to compute the probabilities of transition between positions. These values are then used to compute path weights.

A Doppler shift changes the received pseudorandom noise (PRN) code duration, as explained by Ziedan (2006). A negative Doppler shift increases the code duration, while a positive Doppler shift decreases it. An interval T, considering a Doppler shift f_d, was changed to T_{f_d} as follows (Ziedan, 2006):

$$T_{f_d} = T \frac{f_L}{f_L + f_d + \alpha_d T / 2} \tag{8}$$

where f_L is the carrier frequency and α_d is the Doppler rate. A tracking module generates an estimate of the start of the next integration interval as follows:

$$t_{s_i} = t_{s_{i-1}} + T_I \frac{f_L}{f_L + f_{d_i} + \alpha_i T_I / 2} + \tau_{e_i} \tag{9}$$

where T_I is the integration interval and t_{s_i} and $t_{s_{i-1}}$ are the estimated start of the i-th and (i-1)th integration intervals, respectively. f_{d_i} and α_i are the estimated Doppler shift and rate, respectively. τ_{e_i} is the estimated code delay error generated from the DLL. When a loop is in a steady state, then τ_{e_i} should result in stabilization (i.e., it should have a zero mean and a stable variance). The mean and standard deviation of τ_{e_i} are defined as μ_{τ_e} and σ_{τ_e}, respectively.

When a signal status changes due to the appearance or disappearance of a reflected signal, then τ_{e_i} will increase in magnitude, and μ_{τ_e} and σ_{τ_e} will change. Experimental testing using real and simulated data revealed that it is possible to use μ_{τ_e} and σ_{τ_e} to identify a probability of change in a signal status. One such experiment is shown in the text to follow. Therefore, μ_{τ_e} and σ_{τ_e} are used to generate features for the ISE algorithm.

The following is an example of an experiment that illustrates the effect of a status change on μ_{τ_e} and σ_{τ_e} that was run using simulated GPS C/A signals. GPS signals are simulated using the GNSS software receiver developed as described by Ziedan (2006) and Ziedan and Garrison (2008, 2009, 2011). A coherent integration of 20 ms is used, with $f_d = -1500$ Hz and $\alpha_d = 0.5$ Hz/sec. The sampling rate is $f_s = 6500$ kHz. The measurement noise is generated from a white Gaussian distribution, while the oscillator phase and frequency noises are generated from normal random walks as described by Ziedan (2006). The runtime is 40 seconds. During the first 10 s, only the LOS signal is present with $C / N_0 = 40$ dB-Hz. At $t = 10$ s, the LOS signal is blocked, and an NLOS signal appears with $C / N_0 = 35$ dB-Hz and a code delay of $\tau_m = 0.23$ chips. At $t = 20$ s, the LOS signal reappears. A new multipath signal appears with $C / N_0 = 30$ dB-Hz and a code delay of $\tau_m = 0.314$ chips. At $t = 30$ s, the multipath signal disappears. Therefore, this scenario presents three distinct status changes, which include LOS to NLOS, NLOS to multipath, and multipath to LOS, at $t = 10$, 20, and 30 s, respectively.

For illustration, μ_{τ_e} and σ_{τ_e} are computed over intervals with lengths that range from 0.02 s to 0.5 s. This means that the two parameters are computed over 0.02 s, then 0.04 s, and so on. Figure 4 shows the heat map for μ_{τ_e}, where the horizontal

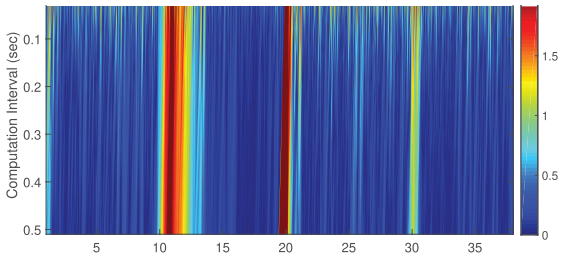

FIGURE 4 Illustration of the change in the mean, μ_{τ_e}, for the investigated scenario

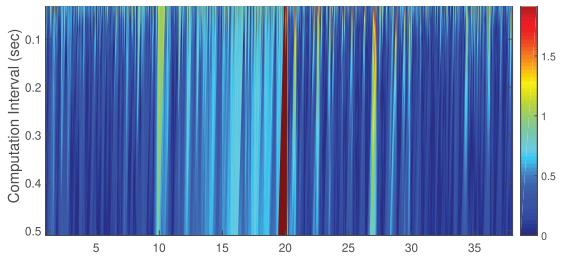

FIGURE 5 Illustration of the change in the standard deviation, σ_{τ_e}, for the investigated scenario

axis is the start time of a computation interval and the vertical axis is the computation interval. The μ_{τ_e} values are normalized. As shown, at $t = 10$ s, there is a large increase in μ_{τ_e} when the LOS signal disappears and an NLOS signal appears. As the DLL converges, μ_{τ_e} decreases gradually, until it returns to a level similar to that observed before the change in signal status. At $t = 20$ s, the time point at which the LOS signal and a new multipath signal appears and the aforementioned NLOS signal disappears, the increase in μ_{τ_e} is larger than in the previous case, but the DLL converges more rapidly. At $t = 30$ s, the time point at which the multipath signal disappears and only the LOS signal remains, the increase in μ_{τ_e} is the smallest, and the DLL converges rapidly. Similarly, the heat map shown in Figure 5 documents the change in σ_{τ_e} in which different status changes exhibited different patterns in the change of σ_{τ_e}.

Based on the analysis presented above, four features are computed from μ_{τ_e} and σ_{τ_e}. The feature computation is performed using a sliding window approach. This means that if a window has a size of $(2j + 1)$, then for a position at time t_i, the computation is performed using data from t_{i-j} to t_{i+j}. Meanwhile, for a position at time t_{i+1}, the computation is performed using data from t_{i+1-j} to t_{i+1+j}, i.e., the window slides forward by a factor of 1. t_i is defined as the time to check for a transition. Two time periods are defined; the first time period, T_{Pi-1}, begins and ends before t_i, while the second one, T_{Pi}, begins and ends after t_i. t_{si-1} and t_{ei-1} are defined as the start and end time of $T_{p_{i-1}}$, respectively. t_{s_i} and t_{e_i} are defined as the start and end time of T_{p_i}, respectively. T_i^* is defined as the time period between $t_{e_{i-1}}$ and t_{p_i}, where t_{p_i} is located inside T_{p_i}. Figure 6 illustrates the relationship between these different time periods. The four features include:

- The difference between the average μ_{τ_e} over $T_{p_{i-1}}$ and T_{p_i}, defined as $\Delta\mu_{\tau_{ei}}$
- The maximum change between the average μ_{τ_e} over $T_{p_{i-1}}$ and the individual μ_{τ_e} over T_i^*
- The difference between the average σ_{τ_e} over $T_{p_{i-1}}$ and T_{p_i}, defined as $\Delta\sigma_{\tau_{ei}}$

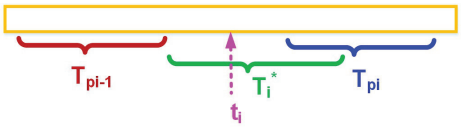

FIGURE 6 Illustration of the relationship between different time periods

- The maximum change between the average σ_{τ_e} over $T_{p_{i-1}}$ and the individual σ_{τ_e} over T_i^*

The two additional features are computed based on the estimated $\hat{C}/N_0$. This computational approach is similar to the one used for the first four features, which is as follows:

- The difference between the average $\hat{C}/N_0$ over $T_{p_{i-1}}$ and T_{p_i}, defined as $\Delta\hat{C}/N_{0_i}$
- The maximum change between the average $\hat{C}/N_0$ over $T_{p_{i-1}}$ and the individual $\hat{C}/N_0$ over T_i^*

3.2 | Supervised Self-Organizing Map (SOM) Neural Network

This paper proposes a novel supervised SOM neural network algorithm. Conventional SOM algorithms are unsupervised and used to classify input data into clusters with similar features.

Four satellites statuses are defined as described by Ziedan (2020). These include LOS (L), multipath (M), NLOS (N), and invisible (I). Therefore, as explained by Ziedan (2020), there are 16 possible transitions. Each satellite, s, can have only one status, η_n^s, at a single instance n. A transition from a source status, X, to a destination status, Y, can take place with X and Y as either L, M, N, or I. The 16 potential transitions are as follows:

- If $X = L$, the potential transitions are $L \rightarrow L$ (LL), $L \rightarrow M$ (LM), $L \rightarrow N$ (LN), and $L \rightarrow I$ (LI).
- If $X = M$, the potential transitions are $M \rightarrow L$ (ML), $M \rightarrow M$ (MM), $M \rightarrow N$ (MN), and $M \rightarrow I$ (MI).
- If $X = N$, the potential transitions are $N \rightarrow L$ (NL), $N \rightarrow M$ (NM), $N \rightarrow N$ (NN), and $N \rightarrow I$ (NI).
- If $X = I$, the potential transitions are $I \rightarrow L$ (IL), $I \rightarrow M$ (IM), $I \rightarrow N$ (IN), and $I \rightarrow I$ (II).

Four SOM networks are used (i.e., one for each source status). Each SOM network has a size of $N_{nu} = 36$ neurons. The inputs of each SOM network are the features extracted from a signal with a satellite source status that is similar to that of the SOM network. This means that the SOM network is designed to generate transition probabilities based on the assumption of a specific source status.

For the training phase, the inputs of each SOM network are used as training data for each of the four transitions with the same source status. The number of training samples for each SOM network are defined as N_{t_X}. Each training sample included

$N_{nf} = 6$ features. SOM network training is performed using the MATLAB neural network clustering tool (nctool).

After training, each SOM network generates a weight matrix with a size equal to $N_{nu} \times N_{nf}$. Each row in the weight matrix represents the weights of the N_{nf} features for one neuron. The weight matrices for the four networks are defined as W_x, X = *L*, *M*, *N*, or *I*. Figure 7 outlines the training steps. Of note, a conventional SOM network would classify a new sample by finding the closest neuron, i.e., the neuron located at a minimum distance from the sample to be classified. However, the SOM network developed here goes a step further and assigns each neuron with a probability of transition to each *Y*. This requires a conversion from unsupervised to supervised learning. Figure 8 outlines the steps involved in the conversion to supervised SOM, which is done as follows:

1. Each training sample is labeled by the transition that is used to generate it, *XY*.
2. The training is performed as is described above.
3. The Euclidean distance between each training sample and each neuron is computed. This produced a distance matrix, D_X, for each network, with a size of $N_{nu} \times N_{t_X}$. Each training sample is assigned to a neuron based on minimum distance. The minimum distance, $d_{min_{XY}}$, and the sample label are recorded.
4. Each neuron had labeled training samples assigned to it together with their $d_{min_{XY}}$. Different transitions with the same *X* could be assigned to the same neuron, albeit with different values of $d_{min_{XY}}$.
5. For each neuron, the probability of transition to each destination status is computed from $d_{min_{XY}}$. A probability matrix, P_{r_X}, of size $N_{nu} \times N_Y$ is computed for each network, where $N_Y = 4$ is the number of destination statuses with the same source. The probability matrix is computed using an approach similar to the one used to compute $V_{E_{n-1}^g \to C_n^j}$ in Section 2. The probability matrix is computed as follows:

5.1 For each neuron, the mean distance, $E_{u_{XY}}$, of $d_{min_{XY}}$ of the samples with the same label is computed, where the subscript *u* refers to a neuron and *XY* refers to a label. $E_{u_{XY}}$ is used as the main factor in the computation of the probability of transition assigned to a neuron.

FIGURE 7 Outline of the SOM training process

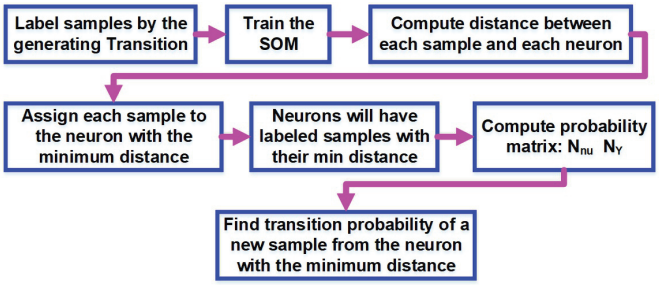

FIGURE 8 Outline of the proposed supervised SOM

5.2 For each destination Y, the mean of $E_{u_{XY}}$ calculated over all neurons is computed as E_{XY}. The minimum of $E_{u_{XY}}$ from all the neurons is identified and defined as $E_{min_{XY}}$. The difference between E_{XY} and $E_{min_{XY}}$ is computed and defined as $E_{diff_{XY}}$.

5.3 A ratio R_{XY} is computed as

$$R_{XY} = \frac{1-v}{E_{diff_{XY}}} \tag{10}$$

where v is a value less than 1.

5.4 The entry in the matrix P_{r_X} for a destination Y and a neuron u, which represents the probability of a transition $X \rightarrow Y$ at a neuron u, is

$$P_{r_{X_{Y_u}}} = \begin{cases} 1-\left(E_{u_{XY}} - E_{min_{XY}}\right)R_{XY}, & \text{for } E_{min_{XY}} \le E_{u_{XY}} \le E_{XY} \\ v-\left(E_{u_{XY}} - E_{min_{XY}}\right)R_{XY}, & \text{for } E_{u_{XY}} > E_{XY} \end{cases}$$

The goal is to assign a probability between v and 1 to transition with a mean distance less than the mean E_{XY}. The transition with the minimum mean distance $E_{min_{XY}}$ will have a probability of 1 at its assigned neuron. As the mean distance $E_{u_{XY}}$ increases beyond the mean E_{XY}, the probability will decrease more rapidly.

6. The probability of a transition $X \rightarrow Y$ for a new sample is determined by finding the neuron in the network X with the minimum distance to the new sample. The probability of transition would then be $P_{r_{X_{Y_u}}}$.

7. The probability of transition $U_{E^g_{n-1} \rightarrow C^j_n}$ between a position E^g_{n-1} and each candidate position, C^j_n, is determined based on the status X at E^g_{n-1} and the predicted status Y at C^j_n.

3.3 | Generating Training Samples

The training samples are extracted from the output of a tracking module that contained simulated GPS signals. GPS signals are simulated and tracked using the GNSS software receiver that was developed as described by Ziedan (2006) and Ziedan and Garrison (2008, 2009, 2011).

500 GPS signals are generated. Each signal included approximately 30 different transitions and lasted for approximately 150 seconds. This resulted in the generation of a minimum of 500 samples for each transition. Some transitions are present in more than 1,000 samples (e.g., multipath transitions) which covered a wide range of possibilities. Each signal parameter (e.g., C/N_0, Doppler shift, number of multipath signals, multipath delays, power, and phase) is generated based on random distributions. All transitions times are marked for use in generating the training samples.

The output, which is the tracking of each signal, is processed to extract the six features required for each training sample. Each training sample is marked by its transition and is used in the supervised version of the SOM network developed in this paper.

4 | EXPERIMENTS AND RESULTS

GPS C/A signals are used to evaluate the proposed OPE algorithm. The signals are generated using the Skydel software-defined GNSS simulator (Orolia, 2020). While the Skydel simulator can generate GNSS signals using a real route on a

map, it does not consider the effect of the surrounding environment on the signals received, for example, multipath reflections. However, the simulator can generate signals from multipath reflections when the multipath characteristics and timing have been set up appropriately.

In order to generate signals that are typically detected in an urban area, an actual urban area is used as the location of the simulated scenario. The area selected for this simulation is located inside the main campus of Zagazig University, Egypt. The scenario start time is April 4, 2021, at 8:00 AM GPS time. The route of the tested scenario (described further below) is analyzed using the ART algorithm described by Ziedan (2017) to obtain satellite statuses and the path lengths of the reflected signals at each point on the route. Furthermore, the timing and duration of each signal are recorded and used to code a Python script for the Skydel simulator. The simulator is then run to generate the signals that identified the correct satellite statuses.

Figure 9 shows a display from the user interface (UI) of the Skydel simulator. Figure 10 shows this area on Google Earth. Google Earth does not provide 3D images of the buildings in this area; thus, 3D building models are constructed as described by Ziedan (2017). The constructed models are superimposed over the view shown in Figure 10. This model is needed to run the ART algorithm inside the OPE algorithm; therefore, the model is reconstructed using MATLAB as described

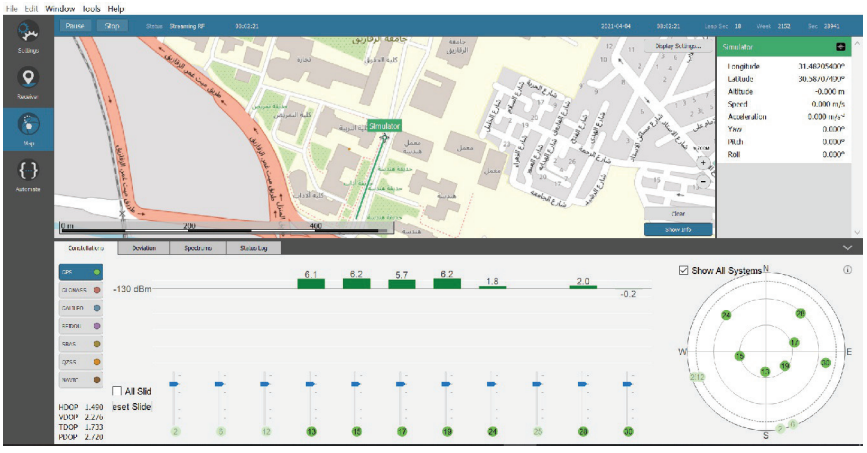

FIGURE 9 A view of the tested area from the UI of the Skydel simulator

FIGURE 10 A view of the test area from Google Earth with superimposed 3D building models

by Ziedan (2017). Figure 11 shows the reconstructed model. There are seven satellites above the horizon during the simulation with PRNs of 13, 15, 17, 19, 24, 28, and 30. Figure 12 depicts a sky plot of these satellites.

The simulated scenario began at the point marked *Start Point* and ended at the point marked End Point as shown in Figure 10. The total running time of the scenario is 180 seconds. A vehicle moving at a velocity of 4 km/hour is included in the simulation to represent heavy stop-and-go traffic. The scenario included four intervals. During the first and third intervals, the vehicle is moving. During the second and fourth intervals, the vehicle is static. The scenario began with the vehicle moving for approximately six seconds. The vehicle then stops for approximately 16 seconds, and then moves again for approximately 120 seconds until it reaches the *End Point*, after which it stops for approximately 40 seconds until the end of the scenario. The four intervals are defined as T_1, T_2, T_3, and T_4, respectively. The scenario includes more than 50 changes in signal status. Figure 13 shows the number of signals associated with each of the four statuses (L, M, N, and I) over the entire scenario. As shown, only five satellites are in LOS status between approximately 0 to 80 seconds, except for the short time during which a sixth satellite is in LOS. After 80 seconds, the number of LOS signals decreased, eventually reaching only two LOS signals during the final interval. Moreover, for most of the simulation, one or two of the satellites are invisible. The number of multipath and NLOS signals ranges from one to three each during the simulation. Figure 14 shows how the signal statuses of PRNs 24 and 30 change during

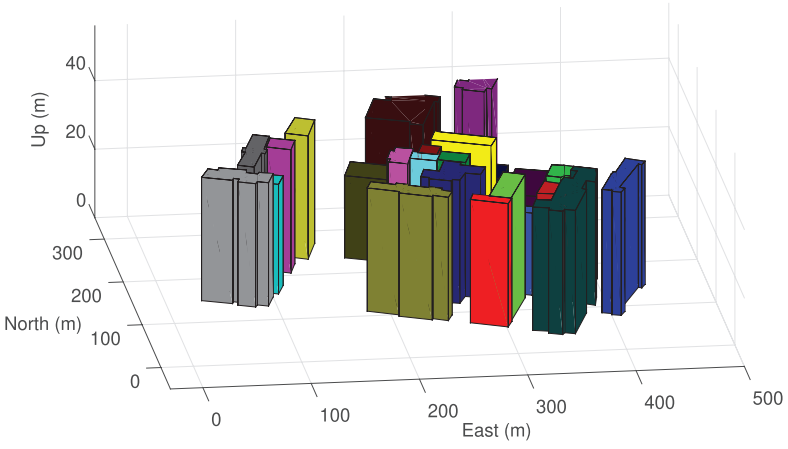

FIGURE 11 A reconstruction of the 3D building models

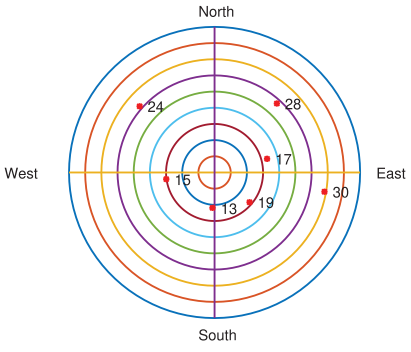

FIGURE 12 A sky plot of the satellites above the horizon

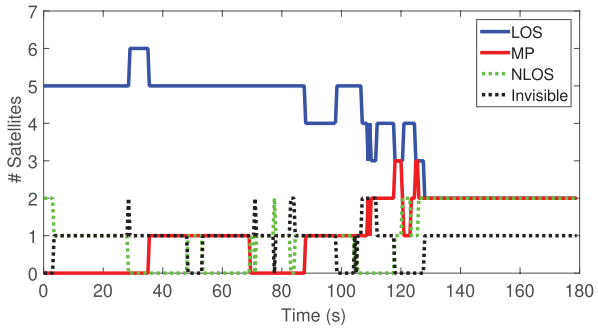

FIGURE 13 Illustration of the number of signals in each of the four statuses (L, M, N, and I) over the entire scenario

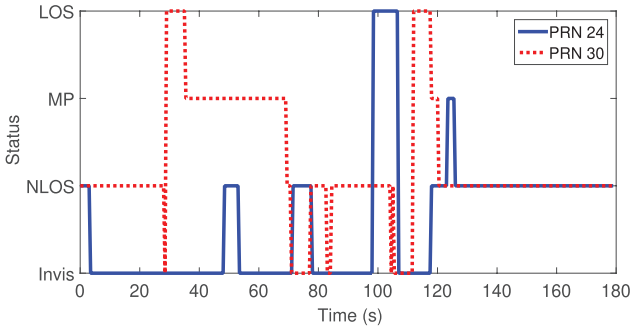

FIGURE 14 Illustration of the changes in the signal statuses of PRNs 24 and 30 during the entire scenario

the scenario. This scenario is thus suitable to test the OPE algorithm as it features a challenging environment with a continuous change in signal status.

Each satellite above the horizon is acquired and tracked using the software receiver developed as described by Ziedan (2006) and Ziedan and Garrison (2008, 2009, 2011). The navigation message is decoded and used to obtain the parameters needed to compute the navigation solution. This is done as described by Ziedan (2019, 2020). The Skydel simulator provides a RINEX navigation file that is also used in the computation of the navigation solution.

For the OPE algorithm, an area of approximately 250 m by 250 m (i.e., 62,500 m^2) is considered. The area is divided into cells with centers separated by 4 m. Cells located inside buildings are discarded. When testing, the ART algorithm is applied to each cell to obtain the status of each satellite and the path lengths of the reflected signals. The following values are used for the various tuning parameters: $d_{av} = 25$ m, $\kappa = 0.9$, $v = 0.9$, $N_p = 9$, and $w_t = 0.5$.

Positions are also estimated using a conventional navigation solution without applying the OPE algorithm for a comparison. Figure 15 shows the root-mean-square (RMS) of the ENU horizontal position error for the conventional approach as well as for the estimated OPE-NAV and OPE-MM over the duration of the scenario. The RMS error is computed every 200 ms. The jump in the positioning accuracy observed using the conventional approach is a direct response to signal transitions (i.e., becoming invisible or reappearing). The module used to track these signals can detect a loss of lock and can reacquire a signal once it becomes visible again. A complete description of the software receiver used here can be found in Ziedan (2006) and Ziedan and Garrison (2008, 2009, 2011).

Meanwhile, there is no jump in the positioning accuracy when using either OPE-NAV or OPE-MM to estimate positions. The OPE algorithm can detect changes in the signal statuses and adjust the estimated code delay accordingly to mitigate potential errors resulting from multipath or NLOS signals estimated by OPE-NAV. The adjustment of the code delay is done as described for the APE algorithm (Ziedan, 2020). The use of the weight function contributes to the enhancement of the estimated most-likely cell and improves the accuracy of the estimated OPE-MM position. This enhancement in OPE-MM accuracy also enhances the accuracy of OPE-NAV positioning.

There are only two LOS signals, two multipath signals, two NLOS signals, and one invisible signal at interval T_4. As shown in Figure 15, the RMS error of the conventional approach degrades rapidly during this interval. However, the estimation accuracy remains stable when using the OPE algorithm, which also provides high-level positioning accuracy.

Figure 16 documents the RMS error for each interval of the scenario using both the conventional approach and the OPE algorithm. The accuracy enhancement provided by the OPE-NAV algorithm over the conventional approach is 92%, 90%, 84%, and 89% at intervals T_1, T_2, T_3, and T_4, respectively. The accuracy enhancement of the APE algorithm described by Ziedan (2020) compared to a conventional approach ranged from 67% to 80%. The accuracy enhancement of the OPE-MM positioning over the conventional approach is 96.4%, 96%, 85%, and 94.7% at intervals T_1, T_2, T_3, and T_4, respectively. The accuracy enhancement of the MMTF algorithm described by Ziedan (2020) that estimates cells on a map compared to a conventional approach ranges from 42% to 96%.

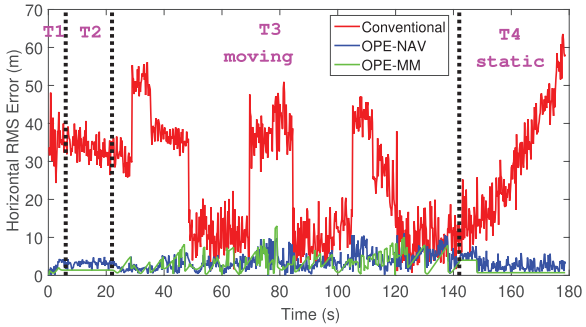

FIGURE 15 The RMS of the ENU horizontal position error for a conventional approach and the estimated OPE-NAV and OPE-MM positions over the entire scenario

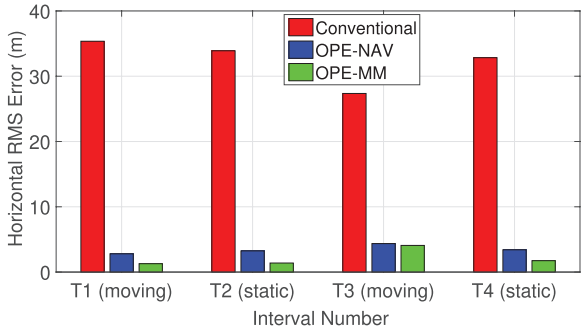

FIGURE 16 The RMS error at each interval of the scenario using a conventional approach and estimated OPE-NAV and OPE-MM positions

Collectively, these results suggest that the OPE algorithm can achieve superior accuracy enhancement when compared to the APE and MMTF algorithms. However, a direct comparison between the OPE, APE, and MMTF algorithms will be needed to assess actual differences in performance. This evaluation will be performed in a future study.

As shown by Ziedan (2020), no weight function was used, and the MMTF algorithm was used to estimate cells, not paths. In this situation, the APE and MMTF algorithms provided different accuracies when applied to different conditions. This result suggests that, in some cases, use of the APE algorithm results in better accuracy, while in other cases, the opposite is the case. Here, the OPE-MM estimated positions are more accurate than those provided by OPE-NAV by 54%, 57%, 6%, and 49% during intervals T_1, T_2, T_3, and T_4, respectively. The OPE-NAV positions are obtained using a navigation algorithm, while those generated by OPE-MM are based on cells on a map. This means that the computations used to generate OPE-NAV positions remain bounded by the signal characteristics (in this case, GPS C/A signals) as well as other error residuals (e.g., the ionospheric error residual when using the Klobuchar model). The OPE-MM estimated positions are limited by the position of the cell that is closest to the real position. Here, the cells have a 4-m^2 size. This means that, in the worst-case scenario, the closest cell will be separated from the real position by 2 m in the north and east directions. Therefore, because positions are determined more accurately by the OPE-MM than the OPE-NAV, one can conclude that the use of the weight function by the OPE-MM contributes positively to its accuracy.

It should be noted that urban environments can encompass countless different scenarios with varying positioning results. Some of the factors that affect the positioning results include building heights and density, the type and distribution of vegetation, and the number of surrounding vehicles. Some settings will require specialized algorithms that address their specific needs. For example, the application of 3D building models and ray tracing algorithms may not be suitable for highly dynamic applications because of the need for increased processing to analyze more cells in a shorter period. Furthermore, positioning algorithms that utilize 3D building models might be affected by modeling errors.

5 | SUMMARY AND CONCLUSION

Two new algorithms are introduced in this paper. The first algorithm, OPE, estimates the most likely path on a map, which is the path with the optimal weight. Three functions are used to compute the weight, including the probability of a transition between positions based on predictions of the signal status, the estimated velocity, and the difference between a position on a map and a corresponding position computed from a navigation algorithm. The second algorithm, ISE, is a probabilistic supervised SOM machine-learning algorithm that estimates the probability of transitions between positions.

The OPE algorithm generates two sets of estimated positions. The first of these is the set of OPE-MM positions, which correspond to cells on a map. The second set includes the OPE-NAV positions, which correspond to the OPE-MM positions, and are computed using a navigation algorithm after adjusting errors in the code delays based on reflected signals.

The estimations provided by the OPE-MM and OPE-NAV algorithms are not independent of one another. The OPE-MM positions are used in a function inside the navigation algorithm that provides estimates designed to minimize the difference between OPE-MM and OPE-NAV estimated positions. This difference is

then used to compute the weight function of the most-likely paths. This means that there is some degree of feedback between the estimates of the OPE-MM and OPE-NAV positions that aims at achieving superior positioning accuracy.

The results presented in this paper indicate that the OPE-MM estimated positions have enhanced estimation accuracies ranging from 85% to 96% compared to a conventional navigation approach. At the same time, the accuracy enhancement provided by the OPE-NAV estimated positions compared to a conventional navigation approach ranges from 84% to 92%. The OPE-MM provides more accurate position estimates compared to the OPE-NAV, where the accuracy enhancement ranges from 6% to 57%.

The accuracy of the OPE-NAV algorithm is limited by the GNSS signal and constellation used (in this case, the GPS C/A signal). The OPE-MM algorithm is limited by the size of the cells used. The feedback between the elements that generate estimates of the OPE-MM and OPE-NAV positions aims at enhancing the estimated positions. The results indicate that this feedback has a positive effect on positional accuracy.

Modern GNSS signals and multi-constellation receivers typically provide better positioning accuracy than the legacy GPS C/A signal; use of these signals may have a positive effect on the performance of the OPE-NAV algorithm. However, this hypothesis needs to be investigated in order to assess the effect of different signals and constellations on overall performance.

REFERENCES

Barreto, G. A., & Araujo, A. F. R. (2004). Identification and control of dynamical systems using the self-organizing map. *IEEE Transactions on Neural Networks*, *15*(5), 1244–1259. https://doi.org/10.1109/TNN.2004.832825

Bradbury, J., Ziebart, M., Cross, P., Boulton, P., & Read, A. (2007). Code multipath modelling in the urban environment using large virtual reality city models: determining the local environment. *The Journal of Navigation, 60*(1), 95–105. https://doi.org/10.1017/S0373463307004079

Closas, P., Fernandez-Prades, C., & Fernandez-Rubio, J. A. (2009). A Bayesian approach to multipath mitigation in GNSS receivers. *IEEE Journal of Selected Topics in Signal Processing*, *3*(4), 695–706. https://doi.org/10.1109/JSTSP.2009.2023831

Garin, L., & Rousseau, J. -M. (1997). Enhanced strobe correlator multipath rejection for code & carrier. *Proc. of the 10th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GPS 1997),* Kansas City, MO, 559–568. https://www.ion.org/publications/abstract.cfm?articleID=2800

Garin, L., van Diggelen, F., & Rousseau, J. -M. (1996). Strobe and edge correlator multipath mitigation for code. *Proc. of the 9th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GPS 1996),* Kansas City, MO, 657–664. https://www.ion.org/publications/abstract.cfm?articleID=2597

Giremus, A., Tourneret, J. -Y., & Calmettes, V. (2007). A particle filtering approach for joint detection/estimation of multipath effects on GPS measurements. *IEEE Transactions on Signal Processing*, *55*(4), 1275–1285. https://doi.org/10.1109/TSP.2006.888895

Groves, P. D., Zhong, Q., Faragher, R., & Esteves, P. (2020). Combining inertially-aided extended coherent integration (supercorrelation) with 3D-mapping-aided GNSS. *Proc. of the 33rd International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2020),* 2327–2346. https://doi.org/10.33012/2020.17767

Hsu, L. -T., Gu, Y., & Kamijo, S. (2016). 3D building model-based pedestrian positioning method using GPS/GLONASS/QZSS and its reliability calculation. *GPS Solutions*, *20*, 413–428. https://doi.org/10.1007/s10291-015-0451-7

Hsu, L. -T., Jan, S. -S., Groves, P. D., & Kubo, N. (2015). Multipath mitigation and NLOS detection using vector tracking in urban environments. *GPS Solutions*, *19*, 249–262. https://doi.org/10.1007/s10291-014-0384-6

Hwang, P. Y., & Brown, R. G. (2006). RAIM-FDE revisited: a new breakthrough in availability performance with NIORAIM (novel integrity-optimized RAIM). *NAVIGATION*, *53*(1), 41–51. https://doi.org/10.1002/j.2161-4296.2006.tb00370.x

Jiang, Z., Groves, P. D., Ochieng, W. Y., Feng, S., Milner, C. D., & Mattos, P. G. (2011). Multi-constellation GNSS multipath mitigation using consistency checking. *Proc. of the 24th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS 2011),* Portland, OR, 3889–3902. https://www.ion.org/publications/abstract.cfm?articleID=9944

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, *78*(9), 1464–1480. https://doi.org/10.1109/5.58325

Kohonen, T. (2001). *Self-organizing maps* (3rd ed.). Springer-Verlag.

Lau, L., & Cross, P. (2007). Development and testing of a new ray-tracing approach to GNSS carrier-phase multipath modelling. *Journal of Geodesy*, *81*, 713–732. https://doi.org/10.1007/s00190-007-0139-z

Lesouple, J., Robert, T., Sahmoudi, M., Tourneret, J. -Y., & Vigneau, W. (2019). Multipath mitigation for GNSS positioning in an urban environment using sparse estimation. *IEEE Transactions on Intelligent Transportation Systems*, *20*(4), 1316–1328. https://doi.org/10.1109/TITS.2018.2848461

Li, D., Zhang, P., Zhao, J., Cheng, J., & Zhao, H. (2019). MP mitigation in GNSS positioning by GRU NNs and adaptive wavelet filtering. *IET Communications*, *13*(17), 2756–2766. https://doi.org/10.1049/iet-com.2018.5792

McGraw, G. A., Groves, P. D., & Ashman, B. W. (2020). Robust positioning in the presence of multipath and NLOS GNSS signals. In Y. T. J. Morton, F. van Diggelen, J. J. Spilker Jr., B. W. Parkinson, S. Lo, & G. Gao (Eds.), *Position, navigation, and timing technologies in the 21st century: integrated satellite navigation, sensor systems, and civil applications* (Vol. 1, pp. 551–589). John Wiley & Sons, Ltd. https://doi.org/10.1002/9781119458449.ch22

Miura, S., Hisaka, S., & Kamijo, S. (2013). GPS multipath detection and rectification using 3D maps. *16th International IEEE Conference on Intelligent Transportation Systems,* The Hague, Netherlands. https://doi.org/10.1109/ITSC.2013.6728447

Orolia. (2020). *Skydel software-defined GNSS simulator user manual.* http://manuals.spectracom.com/skydel

Papadimitriou, S., Mavroudi, S., Vladutu, L., & Bezerianos, A. (2001). Ischemia detection with a self-organizing map supplemented by supervised learning. *IEEE Transactions on Neural Networks*, *12*(3), 503–515. https://doi.org/10.1109/72.925554

Parkinson, B. W., & Axelrad, P. (1988). Autonomous GPS integrity monitoring using the pseudorange residual. *NAVIGATION*, *35*(2), 255–274. https://doi.org/10.1002/j.2161-4296.1988.tb00955.x

Polzlbauer, G., Lidy, T., & Rauber, A. (2008). Decision manifolds––a supervised learning algorithm based on self-organization. *IEEE Transactions on Neural Networks*, *19*(9), 1518–1530. https://doi.org/10.1109/TNN.2008.2000449

Smolyakov, I., Rezaee, M., & Langley, R. B. (2020). Resilient multipath prediction and detection architecture for low-cost navigation in challenging urban areas. *NAVIGATION*, *67*(2), 397–409. https://doi.org/10.1002/navi.362

Song, T., Jamshidi, M. M., Lee, R. R., & Huang, M. (2007). A modified probabilistic neural network for partial volume segmentation in brain MR image. *IEEE Transactions on Neural Networks*, *18*(5), 1424–1432. https://doi.org/10.1109/TNN.2007.891635

Steingass, A., Krach, B., & Crisci, M. (2017). Robustness versus accuracy: multipath effects on land mobile satellite navigation. *IET Radar, Sonar & Navigation*, *11*(3), 520–527. https://doi.org/10.1049/iet-rsn.2016.0361

van Nee, R. D. J. (1992). The multipath estimating delay lock loop. *IEEE Second International Symposium on Spread Spectrum Techniques and Applications (ISSTA),* Yokohama, Japan. https://doi.org/10.1109/ISSSTA.1992.665623

Vesanto, J., & Alhoniemi, E. (2000). Clustering of the self-organizing map. *IEEE Transactions on Neural Networks*, *11*(3), 586–600. https://doi.org/10.1109/72.846731

Wang, L., Groves, P. D., & Ziebart, M. K. (2013). GNSS shadow matching: improving urban positioning accuracy using a 3D city model with optimized visibility scoring scheme. *NAVIGATION*, *60*(3), 195–207. https://doi.org/10.1002/navi.38

Wen, W., Zhang, G., & Hsu, L. -T. (2019). Correcting NLOS by 3D lidar and building height to improve GNSS single point positioning. *NAVIGATION*, *66*(4), 705–718. https://doi.org/10.1002/navi.335

Xu, B., Jia, Q., & Hsu, L. -T. (2020). Vector tracking loop-based GNSS NLOS detection and correction: algorithm design and performance analysis. *IEEE Transactions on Instrumentation and Measurement*, *69*(7), 4604–4619. https://doi.org/10.1109/TIM.2019.2950578

Yozevitch, R., & ben-Moshe, B. (2015). A robust shadow matching algorithm for GNSS positioning. *NAVIGATION, 62*(2), 95–109. https://doi.org/10.1002/navi.85

Yu, J. Y. (1998). *Fault detection and exclusion used in a Global Positioning System GPS Receiver* (U.S. Patent No. 5841399A).

Zhang, M., & Zhang, J. (2009). A fast satellite selection algorithm: beyond four satellites. *IEEE Journal of Selected Topics in Signal Processing, 3*(5), 740–747. https://doi.org/10.1109/JSTSP.2009.2028381

Ziedan, N. I. (2006). *GNSS receivers for weak signals*. Artech House.

Ziedan, N. I. (2011). Multi-frequency combined processing for direct and multipath signals tracking based on particle filtering. *Proc. of the 24th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS 2011),* Portland, OR, 1090–1101. https://www.ion.org/publications/abstract.cfm?articleID=9666

Ziedan, N. I. (2014). Modeling and analysis of composite multipath characteristics for a Bayesian-based dense multipath mitigation algorithm. *Proc. of the 27th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2014),* Tampa, FL, 2754–2766. https://www.ion.org/publications/abstract.cfm?articleID=12528

Ziedan, N. I. (2017). Urban positioning accuracy enhancement utilizing 3D buildings model and accelerated ray tracing algorithm. *Proc. of the 30th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2017),* Portland, OR, 3253–3268. https://doi.org/10.33012/2017.15366

Ziedan, N. I. (2018). Improved multipath and NLOS signals identification in urban environments. *NAVIGATION*, *65*(3), 449–462. https://doi.org/10.1002/navi.257

Ziedan, N. I. (2019). Enhancing GNSS mobile positioning in urban environments through utilization of multipath prediction and consistency analysis. *Proc. of the 32nd International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2019),* Miami, FL, 3469–3483. https://doi.org/10.33012/2019.16929

Ziedan, N. I. (2020). Integrating GNSS signal tracking and map-matching for reflected and diffracted signals mitigation in urban environments. *Proc. of the 33rd International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2020),* 3012–3026. https://doi.org/10.33012/2020.17666

Ziedan, N. I. (2022). A novel model for multipath delay estimation and Its application to a modified optimized position estimation algorithm. *Proc. of the 35th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2022),* Denver, CO, 2138–2152. https://doi.org/10.33012/2022.18334

Ziedan, N. I., & Garrison, J. L. (2008). *System and method for acquiring weak signals in a global positioning satellite system* (US Patent No. 7,358,895). https://patents.google.com/patent/US20070046534A1/en

Ziedan, N. I., & Garrison, J. L. (2009). *Method and apparatus for detecting and processing Global Positioning System (GPS) signals* [US Patent No. 7,545,894]. https://patents.google.com/patent/US7545894B2/en?inventor=Nesreen+Ziedan

Ziedan, N. I., & Garrison, J. L. (2011). *System and method for high dynamic acquisition and tracking of signals from the Global Positioning System* [US Patent No. 8,031,112]. https://patents.google.com/patent/US20060115022A1/en

How to cite: Ziedan, N. I. (2023). Optimized position estimation in mobile multipath environments using machine learning. *NAVIGATION, 70*(2). https://doi.org/10.33012/navi.569