A Machine Learning Approach for GPS Code Phase Estimation in Multipath Environments

Mohamad Orabi* , Joe Khalife† , Ali A. Abdallah‡ , Zaher M. Kassas†‡, and Samer S. Saab*
* Department of Electrical and Computer Engineering, Lebanese American University, Byblos, Lebanon
† Department of Mechanical and Aerospace Engineering, University of California, Irvine, U.S.A.
‡ Department of Electrical Engineering and Computer Science, University of California, Irvine, U.S.A.
Email: mohamad.orabi@lau.edu, khalifej@uci.edu, abdalla2@uci.edu, zkassas@ieee.org, ssaab@lau.edu

***Abstract*—A neural network (NN)-based delay-locked loop (DLL) for multipath mitigation in Global Positioning System (GPS) receivers is developed. The NN operates on equally-spaced samples of the autocorrelation function. The NN is trained using a statistical distribution model that takes into consideration multipath time delay and power attenuation. The performance of the proposed method is compared numerically and experimentally with three other conventional techniques: conventional early-minus-late DLL, narrow correlator, and high resolution correlator. It is demonstrated that the NN-based DLL produces smaller code phase root mean squared error compared to the three conventional techniques in high multipath environments.**

***Index Terms*—Navigation, localization, positioning, GPS receiver, machine learning, neural network, delay-locked loops (DLL).**

I. Introduction

Multipath interference is widely recognized to be among the major error sources in position determination via Global Positioning System (GPS), especially in deep urban canyons. Multipath signals distort the autocorrelation functions in the GPS receiver's delay-locked loops (DLLs) and phase-locked loops (PLLs), introducing biases in the code and carrier phase estimates, which in turn introduce errors in the navigation solution. Many techniques have been developed to mitigate the effect of mulitpath, most of which could be grouped into two main categories: (i) antenna techniques [1] and (ii) signal processing techniques, such as the narrow correlator [2], strobe edge correlator [3], and high resolution correlator (HRC) [4]. While the aforementioned approaches have been shown to outperform the standard early-minus-late (E-L) DLL, they are still susceptible to severe multipath and do not take advantage of the statistical models of multipath signals.

Ever since their development, neural networks (NNs) have found several applications in diverse domains: medical diagnosis, speech recognition, object detection, among others. This is due to their ability to learn relationships between inputs and outputs that are too complex for humans to model. Given a set of training data, a NN can learn a complex multi-dimensional relationship, and uses it to take decisions on new data that it has never been seen before. The significance of NNs lies in their ability to capture unmodeled aspects of the training data. In [5], a wavelet NN approach that predicts the satellite's clock bias based on its values at previous epochs was introduced. In [6], a multilayer NN was used to estimate a receiver's position outdoors from the received signal strength (RSS) of cellular signals, while in [7], a weighted k-nearest neighbor (WKNN) was used estimate a receiver's position indoors from the RSS of wireless local area network (WLAN) signals. In [8], a convolutional NN was proposed that uses the time series vector of the carrier-to-noise ratio (CNR) and pseudorange multipath to detect multipath in static and kinematic high precision GPS positioning. In [9], a deep NN was developed to spatially discriminate multipath signals in a synthetic aperture fashion and beamform towards the line-of-sight (LOS) component.

This paper develops a novel NN-based DLL (NNDLL) to mitigate positioning errors due to multipath. The proposed approach differs from previous machine learning-based multipath mitigation approaches by focusing on the autocorrelation function computed in the receiver, which conveys more information about multipath signals than the pseudorange measurement. While [10] also uses a NN that processes samples of the autocorrelation function, the NN's objective was to identify the type of multipath environment and receiver motion in order to adjust the receiver's tracking strategy. In contrast, the approach proposed in this paper uses samples of the autocorrelation function to produce an estimate of the code phase error, both in the presence and absence of multipath. While more samples of the autocorrelation function are needed compared to other conventional approaches, the proposed approach does not require any hardware upgrades. Simulation and experimental results are presented showing that the proposed NNDLL is robust against multipath and outperforms other conventional approaches in high multipath environments.

The paper is organized as follows. Section II formally outlines the mathematical model of GPS signals in the presence of multipath. Section III introduces the multipath simulator used to train and test the proposed NNDLL. Section IV presents the NN architecture. Section V presents simulation results characterizing the performance of the proposed NNDLL. Section VI shows experimental results demonstrating the robustness of the proposed NNDLL against multipath. Concluding remarks are given in Section VII.

This work was performed under the financial assistance award 70NANB17H192 from U.S. Department of Commerce, National Institute of Standards and Technology (NIST).

II. Model Description

This section presents the mathematical models of the received GPS signals. Note that only one GPS satellite is considered in the model, since signals from the others GPS satellites can be modeled as white noise, as a result of the autocorrelation properties of pseudorandom noise (PRN) sequences. In the presence of M multipath signals, the received signal at the input of a GPS receiver can be modeled as

$$r(t) = \sum_{i=0}^{M} a_i p(t-\tau_i)\cos(\omega t + \theta_i) + n(t), \tag{1}$$

where the 0-th signal is the direct-path (DP) signal; $p(t)$ is the PRN code; a_i, τ_i, and θ_i are the amplitude, code phase, and carrier phase of the i-th multipath signal, respectively; and $n(t)$ is the measurement noise, which is modeled as a zero-mean white Gaussian noise process.

Assuming the receiver to be accurately tracking the carrier phase of the received signal, correlating the received signal after carrier wipe-off with the replicated PRN code yields

$$R(t) = \sum_{i=0}^{M} a_i R(t-\tau_i) + N(t), \tag{2}$$

where a_i is the amplitude of the i-th path signal, $R(t-\tau_i)$ is the autocorrelation function of the PRN code shifted by the code phase τ_i, and $N(t)$ is the low-pass filtered noise.

III. Multipath Simulator

This section discusses the multipath simulator, which is developed to assess the performance of the proposed NNDLL compared to conventional approaches. Subsection III-A elaborates on the statistical multipath model used for simulating the characteristics of the non LOS (NLOS) signals. Subsection III-B discusses the simulation of the DP and multipath signal parameters, mainly Doppler frequency and code phase. The latter also discusses the generation of the samples of the autocorrelation function that will be used to train the NN-DLL and later estimate the code phase.

Since the proposed approach utilizes a NN, it needs a large amount of training data composed of autocorrelation functions corrupted by mutlipath and their corresponding true DP code phases. For this reason, a multipath simulator was developed in MATLAB to generate the required training data, as well as test the performance of different approaches. Samples of the received signals given by (1) were simulated and then processed. One main challenge in simulating such samples is that the PRN code shifts are limited to integer multiples of the sampling period T_s. To correctly simulate delays that are a fraction of T_s, the simulator generates codes at a high sampling frequency of 500 Msps, introduces the desired delays, and then decimates the signal back to the sampling frequency used by the receiver.

A. Statistical Multipath Model

The simulator uses statistical models for multipath signals in static urban canyon environments developed by [11]. In the development of this model, more than 20,000 multipath signals were sampled and tracked using the coupled amplitude delay-locked loop (CADLL) algorithm in [12], with 127 correlators to extend the detection scope to 4 chips. The multipath time delay is modeled as a gamma-distributed random variable with shape and scale parameters γ and ς, respectively. The shape and scale parameters are in turn functions of the satellite's elevation angle, as shown in Table I.

TABLE I
GAMMA DISTRIBUTION PARAMETERS FOR DIFFERENT ELEVATION ANGLES

Elevation angle	γ	ς
(0, 15°)	2.62	129.83
(15°, 30°)	2.77	105.52
(30°, 45°)	2.81	80.93
(45°, 60°)	2.56	65.12
(60°, 75°)	2.47	53.22
(75°, 90°)	2.40	43.24

Multipath power attenuation can be modeled as a decreasing affine function of the time delay τ given by [11]

$$f(\tau; a, b) = a\tau + b, \tag{3}$$

where the 95% confidence intervals of a and b are $[-0.0039, -0.0025]$ and $[-12.7, -11.9]$, respectively. For a given τ, the parameters a and b are generated according to

$$a \sim \mathcal{N}\left(\bar{a}, \sigma_a^2\right), \qquad b \sim \mathcal{N}\left(\bar{b}, \sigma_b^2\right),$$

where the means $\bar{a}$ and $\bar{b}$ and the variances σ_a^2 and σ_b^2 are obtained from the confidence intervals.

B. Training and Testing Data Generation

In an effort to make the simulator as realistic as possible, the simulator utilizes GPS ephemerides extracted from publicly available two-line element (TLE) files to calculate the positions and motion of GPS satellites via the SPG4 orbit determination algorithm. It is worth noting that the ephemeris errors in the TLE files will result in satellite position errors that are a negligible fraction of the true range between the satellite and the receiver. Hence, the training and testing data is not impacted by ephemeris errors. Then, given a receiver position, the time history of DP parameters, such as elevation angles, code and carrier phases, and Doppler frequency are computed. After obtaining the DP signal characteristics, multipath signal parameters, such as attenuation, relative delay, and Doppler shifts are then generated based on the model described in Subsection III-A. The simulator then uses the aforementioned parameters to generate the combined DP and multipath signals at the receiver level according to equation (1), at a sampling rate of 500 Msps in order to capture code phase delays that are a fraction of the receiver's sampling time T_s. The signal is then decimated to the lower sampling frequency used by the receiver, after which zero-mean white Gaussian noise is added. The variance of the noise is derived from the desired CNR [13]. Then, the simulated received signal is correlated with the corresponding PRN, resulting in the signal modeled in (2). Samples of this signal are then processed using several techniques to estimate the code phase of the DP signal.

IV. Neural Network Architecture

This section describes the architecture of the proposed NNDLL approach, along with the training strategy. Fig. 1 shows the architecture of a conventional E-L DLL [14], [15] and the proposed NNDLL.

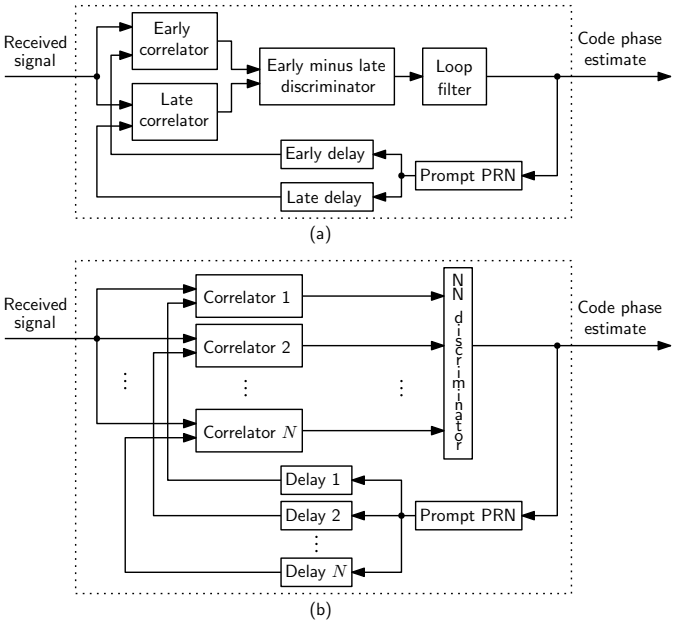

Fig. 1. (a) Conventional E-L DLL architecture and (b) proposed NNDLL architecture, where N is the number of autocorrelation function samples.

A machine learning algorithm is an algorithm that is capable of learning a model from datasets. This is achieved by employing NNs that consist of a combination of neurons and layers. The neuron is nothing but a mathematical function that computes a weighting average of incoming input(s), then applies a predefined nonlinear function (known as the activation function). A collection of neurons forms a layer. The number of layers in a NN defines its depth, i.e., the number of sequential operations needed to perform a specific task. The NN consists of an input layer, output layer, and a number of intermediate layers, known as hidden layers. A connection between neuron j in layer $l-1$ and neuron i in layer l is given by the weight w_{ij}^l, which represents the significance of the connection in a given NN. The learning process of a NN seeks to find the optimal weights that describes the learned model from the training data. For any given neuron, its output x_i^l could be computed according to

$$x_i^l = f(b_i + \sum_{j=0}^{D} w_{ij}^l x_j^{l-1}), \tag{4}$$

where b_i is the bias value for the given neuron, D is the number of neurons in the previous layer $l-1$, and f is the activation function for the given neuron. From (4), the outputs of a layer of size n with inputs z is given by

$$\boldsymbol{f}_l(\boldsymbol{z}) = f(\boldsymbol{b_l} + \boldsymbol{W_l z}), \tag{5}$$

where f is the aforementioned activation function, $\boldsymbol{b_l} = [b_1, b_2, \ldots, b_n]^\mathsf{T}$ is a vector containing the biases for each neuron in layer l, $\boldsymbol{z}$ is a vector of size m containing the input of layer l (output of layer $l-1$), and $\boldsymbol{W_l z}$ is an $n \times m$ matrix, where each row corresponds to the weights of a neuron in the layer. Finally, a NN can be modeled as the mapping

$$y = f_{NN}(x) = f_3(\boldsymbol{f}_2(\boldsymbol{f}_1(\boldsymbol{x}))), \tag{6}$$

where $\boldsymbol{f}_2$ and $\boldsymbol{f}_3$ are vector-valued functions of the form given in (5).

In this paper, the training algorithm adopted for the proposed NN is a supervised training, where the target output is predefined in the training dataset (i.e., the code phase estimate). The neural network design requires defining the terms that characterize its performance and shape, such as: (i) the number of layers, (ii) the number of neurons, (iii) the connections between layers and neurons, and (iv) the function that the network is capable of learning. A near-optimal design of the proposed NN was obtained by trial and error. Numerical analysis shows that a feed-forward multilayer perceptron (MLP) with two hidden layers achieves acceptably low training and selection errors. The number of neurons in each layer is equal to the number of inputs N. A diagram of the proposed NN is shown in Fig. 2.

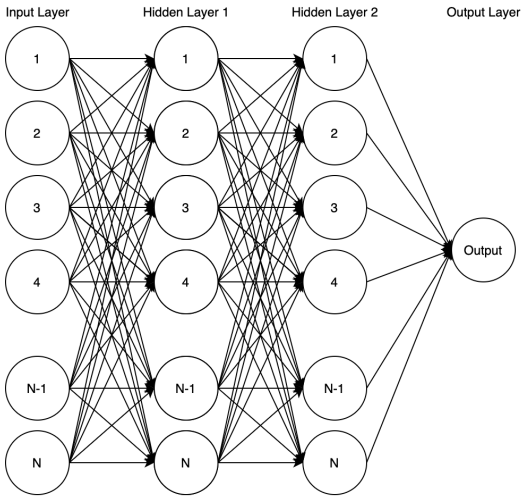

Fig. 2. Diagram of the proposed NN architecture.

The inputs of the NN is N equally-spaced samples of the autocorrelation function spanning 4 chips and centered around the estimated code shift. The number of inputs N is given by $4f_s/f_{chip}$, where f_s and f_{chip} are the sampling and chip rates, respectively. Several realizations of this input are generated to train the NN. The NN training seeks to find a combination of weights and biases that would map the input to a desired output. This could be cast as a minimization problem. The root mean squared propagation (RMSProp) was used as a training algorithm, which seeks to minimize the mean squared error (MSE) cost function, denoted by E and defined as

$$E = \frac{1}{N} \sum_{i=0}^{N} (y_i - \hat{y_i})^2 \tag{7}$$

where $\hat{y_i}$ is the code phase estimate at each learning epoch. The RMSProp optimizer has several advantages over the

traditional gradient descent optimizer. Mainly, it reduces oscillations while finding the minima. It also chooses different learning rates for each parameter, reducing the size of the optimization problem.

Fig. 3 plots the training and validation MSEs as a function of training epochs. Fig. 3 only shows the MSEs in the interval $[0.01, 0.09]$, ignoring the first few training epochs. It can be seen that the training and selection errors for the specified NN design are low, with the selection error being almost as low as the training error, showcasing the ability of the NN to function acceptably with new, non-training data. This is crucial for the application considered herein, since the testing data is from real GPS signals, while the training data is obtained from simulated GPS signals.

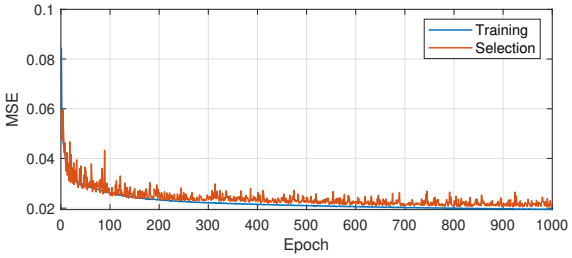

Fig. 3. Training and selection MSEs as a function of the training epoch.

V. Numerical Results

This section aims to asses the performance of the proposed NNDLL approach and compare it with existing conventional approaches. In Subsection V-A, the aforementioned simulator is used to obtain multipath error envelopes for the NNDLL, narrow correlator, and HRC. Then, the code phase tracking performance of these approaches is assessed in a multipath environments simulated according to (i) simulator discussed in Section III and (ii) Land Mobile Multipath Channel Model developed by the German Aerospace Center (DLR) [16].

A. Multipath Error Envelope

While results may differ in a real multipath environment, multipath error envelopes give useful insights of how a correlator will perform. The multipath error envelopes were obtained using the simulator proposed in Section III. To this end, a received signal is simulated consisting of the DP signal and a single multipath signal with relative complex attenuation factor α and a varying relative delay τ [13]. Throughout this subsection, the magnitude of α is set to 0.5 and its phase is uniformly randomized in the interval $[-\pi, \pi]$. The multipath error envelope was computed for (i) NNDLL, (ii) conventional E-L DLL with $1T_c$ correlator spacing, (iii) narrow correlator with $0.1T_c$ correlator spacing, and (iv) HRC. A plot of the multipath error as a function of the relative delay for all correlators is shown in Fig. 4. The sampling frequency was set to $f_s = 10$ Msps and the CNR to 47 dB-Hz.

Fig. 4 shows that the NNDLL performs poorly for very short delay multipath, i.e., $\tau \leq 15$ m, compared to the other correlators. However, for $\tau > 15$ m, the NNDLL performs

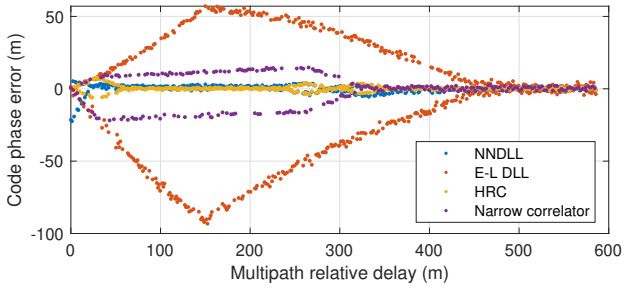

Fig. 4. Multipath error envelopes comparison between (i) NNDLL, (ii) E-L DLL with $1T_c$ spacing, (iii) narrow correlator with $0.1T_c$ spacing, and (iv) HRC. The sampling frequency was set to $f_s = 10$ Msps and CNR = 47 dB-Hz.

consistently better than the conventional DLL and the narrow correlator, for both constructive and destructive multipath. The NNDLL and HRC perform similarly for most delays greater than 55 m; however, the NNDLL outperforms the HRC for short delays between 15 and 55 m.

B. Code Phase Tracking Performance in Multipath Environments

To compare the performance of the NNDLL with conventional correlators, GPS signals from PRN 9 are simulated using the simulator discussed Section III for three multipath environments containing: (i) zero, (ii) one, (iii) and seven multipath signals. The receiver was assumed to be in Irvine, California, U.S.A. The output of the code phase tracking loops was used to asses the performance of (i) NNDLL, (ii) E-L DLL with $1T_c$ correlator spacing, (iii) narrow correlator with $0.1T_c$ correlator spacing, and (iv) HRC. The code phase tracking errors are shown in Fig. 5 and the resulting code phase root mean square errors (RMSEs) are summarized in Table II.

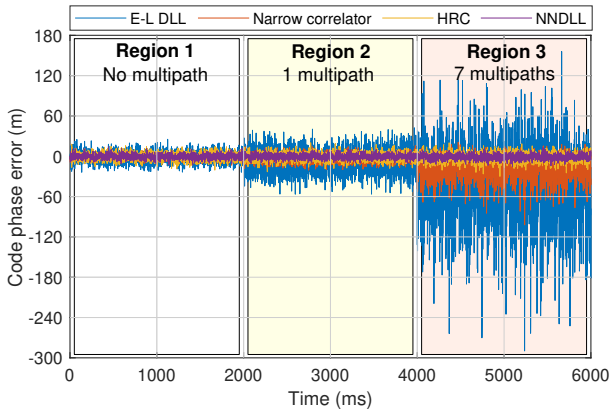

Fig. 5. Code phase tracking errors for (i) NNDLL, (ii) E-L DLL with $1T_c$ spacing, (iii) narrow correlator with $0.1T_c$ spacing, and (iv) HRC. The GPS signals were simulated according to Section III with a sampling frequency $f_s = 10$ Msps and CNR = 47 dB-Hz.

It can be seen from Fig. 5 and Table II that the NNDLL outperforms all other correlators in all three multipath regions. More importantly, the code phase tracking error of the NNDLL remains consistently low in all three regions, as can be seen from Table II.

TABLE II
CODE PHASE RMSES FOR THE E-L DLL, NARROW CORRELATOR, HRC, AND NNDLL IN THREE MUTLIPATH REGIONS

Correlator	Region 1	Region 2	Region 3
E-L DLL	8.158 m	19.624 m	70.739 m
Narrow correlator	3.846 m	6.762 m	25.634 m
HRC	5.424 m	5.685 m	6.762 m
NNDLL	3.560 m	3.492 m	3.823 m

C. DLR Simulation Results

This section uses a more realistic simulator developed by DLR [16] to analyze the performance of the proposed approach. It simulates a car moving from a multipath-free to a multipath environment, and produces the channel impulse response which is then used to generate the GPS signals. Fig. 6 shows the tracking performance of the (i) NNDLL, (ii) E-L DLL with $1T_c$ correlator spacing, (iii) narrow correlator with $0.1T_c$ correlator spacing, and (iv) HRC. The resulting code phase RMSEs are summarized in Table II.

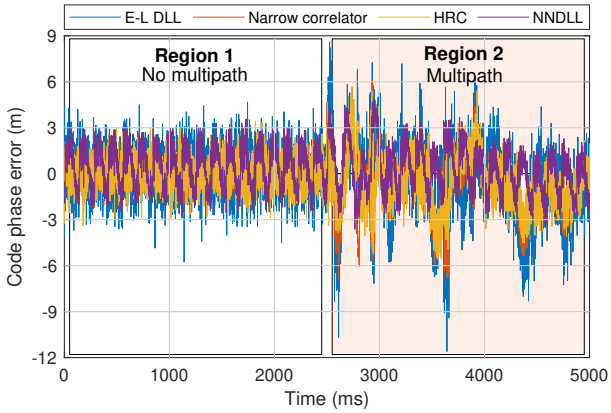

Fig. 6. Code phase tracking errors for (i) NNDLL, (ii) E-L DLL with $1T_c$ spacing, (iii) narrow correlator with $0.1T_c$ spacing, and (iv) HRC. The GPS signals were generated using the simulator developed in [16] with a sampling frequency $f_s = 10$ Msps and CNR=47 dB-Hz.

TABLE III
CODE PHASE RMSES FOR THE E-L DLL, NARROW CORRELATOR, HRC, AND NNDLL IN THE PRESENCE AND ABSENCE OF MULTIPATH USING DLR SIMULATOR

Correlator	Region 1	Region 2
E-L DLL	1.5159 m	2.8086 m
Narrow Correlator	0.8150 m	2.1737 m
HRC	1.0195 m	1.9145 m
NNDLL	1.3292 m	1.5405 m

It can be seen from Fig. 6 and Table III that, in a multipath-free environment, the NNDLL performs similarly to the conventional E-L DLL, while the narrow correlator and HRC perform better. However, the NNDLL outperforms all the other correlators in the presence of multipath regions. More importantly, the code phase tracking error of the NNDLL remains consistently low in both regions, as can be seen from Table III.

VI. EXPERIMENTAL RESULTS

This section validates the proposed NNDLL experimentally with real GPS signals. To this end, GPS L1 signals were collected at 2.5 Msps using an Ettus E312 USRP. The E312 is equipped with a GPS receiver and an inertial measurement unit. The ground truth was sampled from the E312's on-board GPS receiver. The sampled GPS L1 signals were then processed using the NNDLL, E-L DLL with $1T_c$ correlator spacing, and narrow correlator with $0.1T_c$ correlator spacing. Since the actual code phase value was not available, the comparison between the NNDLL, E-L DLL, and the narrow correlator was done based on the behavior of the code phase estimates. Both the DLL and narrow correlators had a loop filter bandwidth of 0.003 Hz. The E-L DLL and NNDLL code phase estimates are shown in Fig. 7, and the narrow correlator and NNDLL code phase estimates are shown in Fig. 8.

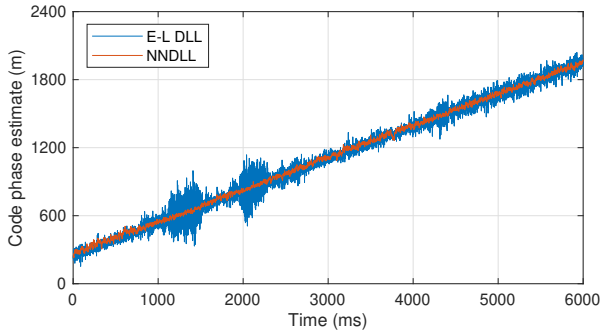

Fig. 7. Tracking performance comparison between the NNDLL and the E-L DLL with $1T_c$ correlator spacing with real GPS signals.

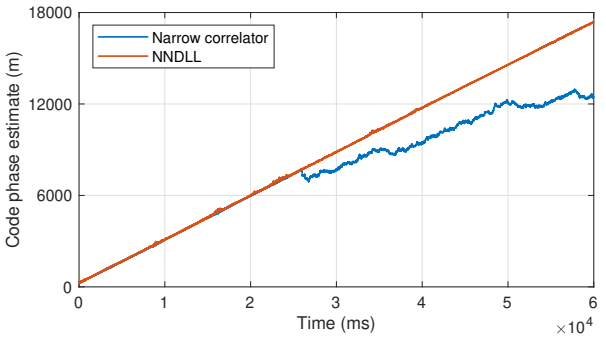

Fig. 8. Tracking performance comparison between NNDLL and narrow correlator with $0.1T_c$ with real GPS signals.

From Fig. 7, it is clear that both the NNDLL and E-L DLL are tracking the code phase. However, the NNDLL estimate seems more precise than the E-L DLL estimate. In addition, at around 1 and 2 seconds, the E-L DLL's code phase estimate seems to degrade, while the NNDLL estimate does not show any significant variations.

From Fig. 8, it can be seen that the narrow correlator initially seems to track the code phase more closely than the E-L DLL, performing similarly to the NNDLL. However, at around 26 seconds, the narrow correlator loses track of the code phase while the NNDLL manages to keep track of the incoming signal's code phase.

Fig. 9 shows the trajectory (white) traversed by the receiver on the campus of the University of California, Irvine, California, U.S.A, as well as the navigation solutions obtained with the NNDLL (green) and the E-L DLL with $1T_c$ correlator spacing (red). The NNDLL and E-L DLL solutions were obtained using 7 GPS satellites with no ionospheric or troposhperic corrections. The NNDLL remains closer to the ground truth than the E-L DLL, especially at the turns. Table IV shows the RMSEs and maximum absolute errors (MAEs) for both the NNDLL and conventional E-L DLL. While the errors for both solutions are large, the NNDLL solution is closer to the ground truth than the DLL solution, having and RMSE of 29 meters versus 57 meters. Note that smaller NNDLL and E-L DLL RMSEs may be obtained by improving the receiver design: increasing the subaccumulation period, which set to 1 ms, and applying ionospheric and tropospheric corrections. Nevertheless, the purpose of this paper is to conduct a preliminary study on the NNDLL and characterize the performance gain in a very basic receiver setting.

Fig. 9. Receiver trajectory (white) and navigation solutions obtained using the NNDLL (green) and the E-L DLL (red).

TABLE IV
RMSE AND MAE FOR THE NAVIGATION SOLUTIONS PRODUCED BY THE E-L DLL AND NNDLL

Correlator	RMSE	MAE
E-L DLL	57.71 m	77.79 m
NNDLL	29.96 m	40.52 m

VII. Conclusion

This paper proposed a machine learning approach to improve the GPS code phase estimate in multipath environments. The proposed NNDLL differs from other machine learning approaches in the literature in that it considers samples of the autocorrelation function replaces the discriminator of a conventional DLL with a NN. Simulation results were presented comparing the performance of the proposed NNDLL with conventional code phase tracking approaches: E-L DLL, narrow correlator, and HRC. It was demonstrated that the NNDLL was able to match the performance of conventional code phase tracking approaches in the absence of multipath, while the NN outperformed conventional approaches in multipath environments. Experimental results were presented, showing that while the NNDLL was trained using simulated GPS signals, it was still able to achieve nearly half the RMSE of an E-L DLL with real GPS signals.

VIII. Acknowledgments

The authors would like to thank Yanhao Yang for his help in data collection.

References

[1] D. Aloi and M. Sharawi, "High fidelity antenna model validation results of a GNSS multipath limiting antenna," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, no. 1, pp. 3–14, January 2011.

[2] A. van Dierendonck, P. Fenton, and T. Ford, "Theory and performance of narrow correlator spacing in a GPS receiver," *NAVIGATION, Journal of the Institute of Navigation*, vol. 39, no. 3, pp. 265–283, September 1992.

[3] L. Garin, F. van Diggelen, and J. Rousseau, "Strobe and edge correlator multipath mitigation for code," in *Proceedings of ION International Technical Meeting*, January 1996, pp. 657–664.

[4] G. McGraw and M. Braasch, "GNSS multipath mitigation using gated and high resolution correlator concepts," in *Proceedings of ION International Technical Meeting*, January 1999, pp. 333–342.

[5] Y. Wang, Z. Lu, Y. Qu, L. Li, and N. Wang, "Improving prediction performance of GPS satellite clock bias based on wavelet neural network," *GPS solutions*, vol. 21, no. 2, pp. 523–534, April 2017.

[6] A. Abdallah, S. Saab, and Z. Kassas, "A machine learning approach for localization in cellular environments," in *Proceedings of IEEE/ION Position, Location, and Navigation Symposium*, April 2018, pp. 1223–1227.

[7] R. Jaafar and S. Saab, "A neural network approach for indoor fingerprinting-based localization," in *Proceedings of IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference*, November 2018, pp. 537–542.

[8] Y. Quan, L. Lau, G. Roberts, X. Meng, and C. Zhang, "Convolutional neural network based multipath detection method for static and kinematic GPS high precision positioning," *Remote Sensing*, vol. 10, no. 12, p. 2052, December 2018.

[9] A. Abdallah and Z. Kassas, "Deep learning-aided spatial discrimination for multipath mitigation," in *Proceedings of IEEE/ION Position, Location, and Navigation Symposium*, April 2020, accepted.

[10] N. Sokhandan, N. Ziedan, A. Broumandan, and G. Lachapelle, "Context-aware adaptive multipath compensation based on channel pattern recognition for gnss receivers," *NAVIGATION, Journal of the Institute of Navigation*, vol. 70, no. 5, pp. 944–962, September 2017.

[11] Y. Wang, X. Chen, and P. Liu, "Statistical multipath model based on experimental GNSS data in static urban canyon environment," *Sensors*, vol. 18, no. 4, p. 1149, April 2018.

[12] X. Chen, F. Dovis, and M. Pini, "An innovative multipath mitigation method using coupled amplitude delay lock loops in GNSS receivers," in *Proceedings of IEEE/ION Position, Location and Navigation Symposium*, May 2010, pp. 1118–1126.

[13] P. Misra and P. Enge, *Global Positioning System: Signals, Measurements, and Performance*, 2nd ed. Ganga-Jamuna Press, 2010.

[14] Z. Kassas, J. Bhatti, and T. Humphreys, "A graphical approach to GPS software-defined receiver implementation," in *Proceedings of IEEE Global Conference on Signal and Information Processing*, December 2013, pp. 1226–1229.

[15] M. Braasch and A. Dempster, "Tutorial: GPS receiver architectures, front-end and baseband signal processing," *IEEE Aerospace and Electronic Systems Magazine*, vol. 34, no. 2, pp. 20–37, February 2019.

[16] A. Steingass and A. Lehner, "Software model for satellite to land mobile multipath propagation," 2019. [Online]. Available: hhttps://www.KN-S.dlr.de/COM-LMS