Dynamic Positioning Interval Based On Reciprocal Forecasting Error (DPI-RFE) Algorithm for Energy-Efficient Mobile IoT Indoor Positioning

Alper Saylam*, Nur Kelesoglu†, Rifat Orhan Cikmazel‡
Department of Electrical and Electronics Engineering
Yaşar University
Izmir, Turkey
*{saylamalper7}, †{nuurkelesoglu}, ‡{rcikmazel}@gmail.com

Mert Nakip
Institute of Theoretical and Applied Informatics
Polish Academy of Sciences (PAN)
Gliwice, Poland
mnakip@iitis.pl

Volkan Rodoplu
Department of Electrical and Electronics Engineering
Yaşar University
Izmir, Turkey
volkan.rodoplu@yasar.edu.tr

***Abstract*—We develop an algorithm called "Dynamic Positioning Interval based on Reciprocal Forecasting Error (DPI-RFE)" for energy-efficient mobile Internet of Things (IoT) Indoor Positioning (IP). In contrast with existing IP algorithms, DPI-RFE forecasts the future trajectory of a mobile IoT device by using machine learning and dynamically adjusts the positioning interval based on the reciprocal instantaneous forecasting error, thereby dynamically trading off transmit energy consumption against forecasting error. We compare the performance of DPI-RFE with respect to the total transmit energy consumption and the average forecasting error against Constant Positioning Interval (CPI) and Positioning Interval based on Displacement (PID) algorithms. Our results show that DPI-RFE significantly outperforms both of these benchmark algorithms with respect to transmit energy consumption while achieving a competitive average forecasting error performance. These results open the way to the design of machine learning based trajectory forecasting algorithms that can be utilized for energy-efficient positioning in next-generation wireless networks.**

***Index Terms*—Artificial Intelligence (AI), machine learning, Internet of Things (IoT), energy-efficient, mobility prediction, indoor positioning**

I. INTRODUCTION

Indoor positioning (IP) is expected to play a key role in next-generation wireless networks in order to enable a plethora of services such as navigation, proximity marketing, asset tracking as well as social distancing [1], [2]. While providing accurate IP is the main goal of these networks, it is equally important that such accuracy is provided without depleting the scarce battery resources of the mobile devices whose positions are being determined [3]. In particular, trading off positioning accuracy against energy consumption is crucial for mobile Internet of Things (IoT) devices, which constitute a rapidly growing segment of the Internet [4].

Artificial Intelligence (AI) is increasingly playing an important role in the design of wireless networks [5]. Recent work has also applied Machine Learning (ML) techniques to IP [6]–[10]. The main contribution of this paper is the development of a novel algorithm, which we call "Dynamic Positioning Interval based on Reciprocal Forecasting Error (DPI-RFE)" for energy-efficient IP. Our algorithm uses ML in order to forecast the future trajectory of a mobile IoT device and adapts the positioning interval of the device in a manner that is reciprocal to the instantaneous forecasting error.

Our DPI-RFE algorithm is distinct from existing algorithms that (1) forecast the future trajectory of a mobile device without any regard to energy efficiency [6]–[9], [11], [12], or (2) adjust the positioning interval of a mobile device based on motion detection by the device (i.e. motion-triggered energy-efficient IP) [13]–[17], or (3) forecast future trajectory of a mobile device in order to achieve energy-efficient IP [3], [18]. Our results show that with regard to total energy consumption of the mobile IoT device, DPI-RFE significantly outperforms both the Constant Positioning Interval (CPI) algorithm, which serves as a benchmark, and the Positioning Interval based on Displacement (PID) algorithm [18], which adapts the sleep duration in response to a significant forecast change in position. Furthermore, DPI-RFE achieves an average forecasting error that is close to that achieved by PID.

The rest of this paper is organized as follows: In Section II, we describe the relationship of this work to the state of the art. In Section III, we state our assumptions. In Section IV, we describe our DPI-RFE algorithm. In Section V, we discuss our results. In Section VI, we present our conclusions.

II. RELATIONSHIP TO THE STATE OF THE ART

In this section, we describe the relationship of our work to the state of the art. To this end, we have categorized the previous algorithms in this area as (1) those that forecast the future trajectory of a mobile device or a pedestrian without any

978-1-6654-4913-7/20/$31.00 ©2021 IEEE

regard to energy efficiency; (2) those that adjust the positioning interval of a mobile device based on the current motion that is detected by the mobile device (i.e. motion-triggered energy-efficient algorithms); and (3) those that forecast future trajectory of a mobile device in order to achieve energy-efficient positioning.

First, in order to determine the future trajectory of a mobile device or pedestrian, References [6]–[9] utilize ML methods. In [11], the trajectory is predicted based on Long Short Term Memory (LSTM) by using the data from distinct scenarios that include vehicles, bicycles, and pedestrians in order to predict real-time traffic. In [12], the proposed model is trained via recorded videos via bird's-eye tracks of multiple people. The model is targeted at walking speed and produces short-term predictions. In contrast, our work not only predicts the future trajectory of a mobile device but also determines the positioning interval at which the device wakes up in order to provide an effective trade-off of transmit energy consumption against forecasting error.

Second, we contrast our work against motion-triggered energy-efficient positioning algorithms that do not utilize any trajectory forecasting. Reference [13] collects mobility data at fixed intervals in order to achieve energy efficiency. In [14], a location tracking service called SensTrack is designed to reduce the Global Positioning Accuracy (GPS) usage and provides the user trajectory by using an accelerometer as well as orientation sensor readings on a smartphone. In [15], an energy-efficient system based on an accelerometer is used to detect motion. In [17], if the system detects no displacement of the mobile user, the application enters an energy-saving mode.

In [19], WLAN localization is performed by utilizing accelerometer data from a smartphone in order to localize the user only when the user moves, thereby reducing energy consumption. In [20], the EnTracked system is proposed in order to track the mobile devices in an indoor area in an energy-efficient and robust manner. The system utilizes the movement speed in programming the sleep cycles of GPS, and determines whether the target is moving or not by using the accelerometer. In contrast with all of these articles, in our work, our dynamic model does not operate in a manner that is directly triggered only in response to current displacement. Whether the user moves or not, in our work, we forecast the entire future trajectory and determine a positioning interval in a manner that adapts to the instantaneous forecasting error.

Third, we contrast our work against those that reduce energy consumption of the mobile device in a positioning system based on trajectory forecasting. In [18], the PID algorithm adjusts the sleep duration of a mobile device based on when the device is forecast to move to an adjacent cell. In [3], the Enloc localization framework is developed by using probabilistic prediction in order to increase the accuracy for a given energy budget. A prediction is formed only at uncertain points, which occur on the vertices of roads. In contrast with these works, our DPI-RFE algorithm first forecasts the future trajectory of a mobile device and then dynamically updates the positioning interval based on the instantaneous forecasting error.

III. ASSUMPTIONS AND BASIC SYSTEM DESIGN

Throughout this paper, we shall focus on a particular mobile IoT device, denoted by D, that roams an indoor deployment region, which we denote by $\mathcal{R}$. Furthermore, we observe this device D over an interval $\mathcal{T}$ in time. We assume that there is a set $\mathcal{M}$ of positioning anchors (or "anchors" for short) with which D is associated over $\mathcal{T}$.

We assume that device D has the ability to send a beacon signal that is heard successfully by each anchor in $\mathcal{M}$. We assume that the anchors in $\mathcal{M}$ are connected to a gateway G. Based on the beacon signals received by each of the anchors in $\mathcal{M}$, gateway G estimates the current position of device D.[1]

Device D wakes up and remains awake for a duration of T seconds, during which it sends L positioning beacons at regular intervals of T_b seconds. Thus, $T = LT_b$. Then, the device goes to sleep for T_s seconds. In our design, T is fixed, whereas T_s is potentially variable in each cycle. We define the "positioning interval", denoted by T_c, as the duration between two successive wake-ups of the device. Thus, $T_c = T + T_s$. Furthermore, in our design, T_s as an integer multiple of T.

We divide up the time axis into slots of duration T. We let k denote the discrete-time index of a time slot. For every k, we let $\mathbf{x}[k]$ denote the 2D vector, composed of the x and y coordinates of the position estimate of the mobile device in slot k. This estimate is based on the L beacons received by the anchors during slot k, if the device is awake in slot k. In particular, no such position estimate is formed at G during a slot if the device D sleeps in that slot.[2]

IV. DYNAMIC POSITIONING INTERVAL BASED ON FORECASTING ERROR (DPI-RFE) ALGORITHM

Fig. 1 shows the architecture that implements our DPI-RFE algorithm. We note that this architecture resides at gateway G. The architecture is comprised of four modules: Forecasting; Position Selection (PS); Forecasting Error Calculation (FEC); and Positioning Interval Update (PIU). Below, we describe each of these modules.

First, the Forecasting module in Fig. 1 takes as input the past position estimates formed when the device was awake as well as the past forecast positions for those slots at which the device was not awake. (The Accumulator in the figure accumulates the past forecasts. The Forecasting module uses these during those slots at each of which a past position estimate is not available since the device was not awake in that slot.) The Forecasting module takes this superposition of position estimates and position forecasts for a total of U slots into the past and forms position forecasts V slots into the future.[3] (Recall that each slot is of duration T seconds.)

[1]For example, if Angle of Arrival (AoA) is used as the underlying positioning technology, each anchor records the AoA measurements based on the beacon signal received from D. The gateway G combines these AoA measurements in order to compute the current position of D.

[2]Thus, forecasting is needed for each slot in which no position estimate is available at G.

[3]Note that past forecasts are used in all slots for which no position estimates are available in order to form forecasts of future positions.

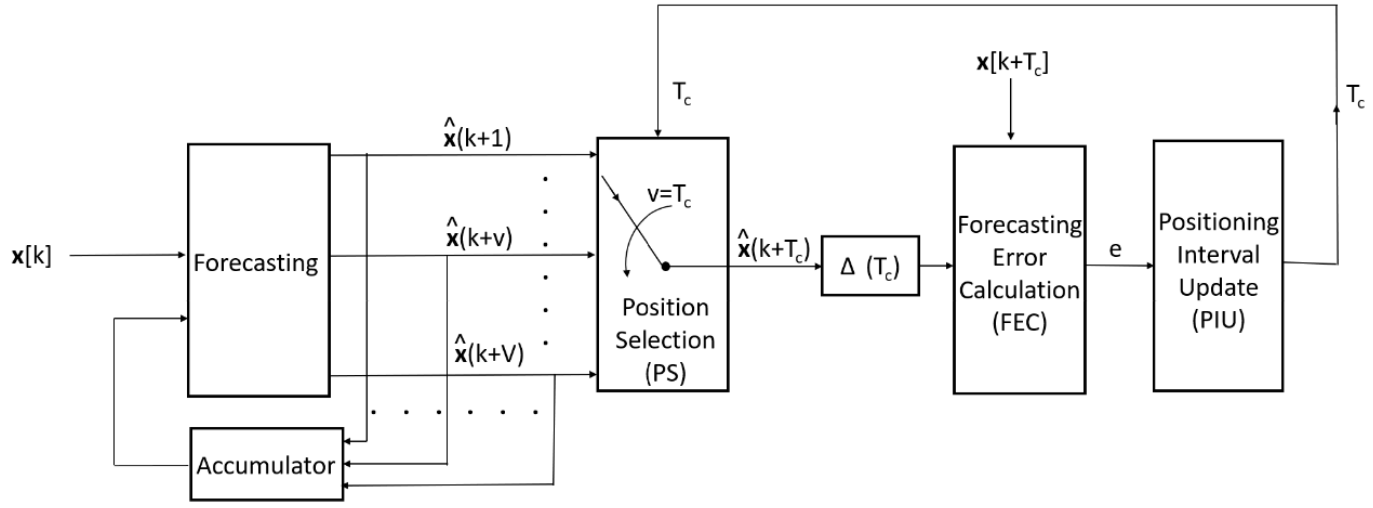

Fig. 1. Architectural description of the Dynamic Positioning Interval based on Reciprocal Forecasting Error (DPI-RFE) algorithm

The output of the Forecasting module is the vector of forecast positions, denoted by $\hat{\mathbf{x}}(k+v)_{v\in\{1,\ldots V\}}$.[4]

Second, the PS module selects a position among the forecast positions based on the value of the positioning interval T_c that is fed into this module. This selected position is $\hat{\mathbf{x}}(k+T_c)$.

Third, this forecast position $\hat{\mathbf{x}}(k+T_c)$ is input from the PS to the FEC. Recall that device D sends a sequence of L beacons in a slot of duration T when it is awake. The FEC module takes the position estimate of D denoted by $\mathbf{x}[k+T_c]$ and the selected forecast position $\hat{\mathbf{x}}(k+T_c)$ and computes the Euclidean distance between these two positions in order to obtain the instantaneous forecasting error denoted by e in Fig. 1.

Fourth, the PIU module takes the instantaneous forecasting error as an input and determines the next positioning interval, namely T_c, based on this forecasting error. After the positioning interval has been determined, PIU passes to the PS the value of T_c as shown in Fig. 1.

Our DPI-RFE algorithm performs a particular selection of T_c based on the forecasting error e, which we now describe. We shall denote the nth positioning interval by $T_c^{(n)}$ and the nth forecasting error by $e^{(n)}$. The DPI-RFE algorithm computes the next positioning interval denoted by $T_c^{(n+1)}$ based on the current positioning interval $T_c^{(n)}$ and the nth forecasting error $e^{(n)}$ as $T_c^{(n+1)} = T_c^{(n)}/e^{(n)}$ provided that this value does not fall below T and does not rise above VT.[5] (DPI-RFE sets $T_c^{(n+1)}$ to the lower or the upper bound, respectively, if $T_c^{(n)}/e^{(n)}$ hits any one of these limits.) That is, the value of the next positioning interval is chosen to be that of the current positioning interval times the reciprocal of the current forecasting error.[6]

V. RESULTS

A. Simulation Setup

In this section, first, we describe our positioning data set. Second, we explain our forecasting methodology for the DPI-RFE algorithm. Third, we describe the benchmark algorithms against which we compare the performance of our algorithm.

1) Description of the Positioning Data Set: The data set [21] used in this work consists of kinematically collected positions of a human moving in a rectangular deployment region that has a length of 70 m and a width of 35 m. We scaled the data set to a deployment region that has size 4×4 m. In the data set, there are two features, namely the x and y coordinates of the human, who carries the mobile device.[7]

2) Forecasting Methodology: The duration for which the device remains awake, namely T, is set to 1 s. We use a Multilayer Perceptron (MLP) forecasting scheme in the DPI-RFE algorithm. The number of layers and the number of neurons in each layer of the MLP neural network are optimized in order to minimize Mean Square Error (MSE).

The Forecasting module has a distinct MLP model in order to forecast the future positions on each of the x and y axes. The number of past samples, namely U, is set to 20, and the number of future forecasts, namely V, is set to 10. Each MLP model is comprised of a single hidden layer with 20 neurons and an output layer with 10 neurons. Rectified Linear Unit Function (ReLU) is utilized at both layers. In the training stage, 10-fold cross-validation is applied. For each fold, the number of epochs is 250 and the batch size is 20.

[4]The argument that appears in the parentheses is the time index *for* which the forecast is formed.

[5]The former case corresponds to zero sleep duration; the latter case is the maximum duration for which forecasts into the future are available.

[6]In our future work, we shall explore alternative choices in the functional form that characterizes the dependence of the positioning interval on the forecasting error.

[7]In the data set, there is a total of 5718 samples for training and 1907 samples for testing.

3) Benchmark Algorithms: We examine the performance of our DPI-RFE algorithm against two benchmark algorithms: Constant Positioning Interval (CPI) and Positioning Interval based on Displacement (PID) [18].

In the CPI algorithm, the duration between two successive wake-ups of the mobile device, namely T_c, is constant. This algorithm reports forecast positions during those slots at which the device sleeps. In this work, we use the same MLP-based forecasting scheme for CPI as we do for DPI-RFE.[8]

The PID algorithm adaptively chooses the positioning interval based on the forecast displacement of the mobile device D as follows: The deployment region $\mathcal{R}$ is divided into sub-regions called "cells". Based on the forecast trajectory of the device, the PID algorithm sets the positioning interval to be the duration until the first time that the device is forecast to cross over to an adjacent cell.

B. Performance Evaluation

We shall evaluate the performance of DPI-RFE against the CPI and PID algorithms with respect to both the total transmit energy consumption and the average forecasting error measured over a 300 s observation interval.

Let P_{tx} denote the transmit power consumption incurred by the mobile device when the L beacons are transmitted. We let T_{on} denote the duration during which the L beacons are transmitted.[9] Then, the total transmit energy consumption of device over one cycle of duration T_c is $P_{tx}T_{\text{on}}$. We take $P_{\text{tx}} = 16.25$ mW, and $T_{\text{on}} = 3$ ms in this work.[10]

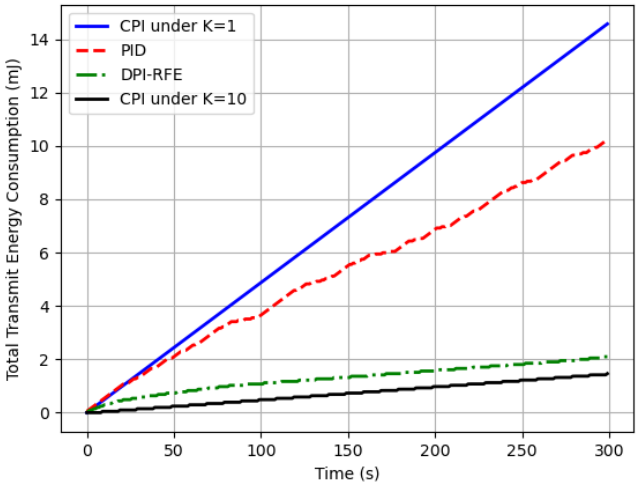

Fig. 2. Comparison of the total transmit energy consumption incurred by the mobile IoT device under our DPI-RFE algorithm versus those under the CPI and PID algorithms

[8]Hence, the key difference is that while T_c is constant in CPI, it is variable in DPI-RFE.

[9]Note that P_{tx} is incurred only when a beacon is transmitted. We do not quantify the idle power when no transmission occurs while the device is awake. Since we model only beacon transmission and no downlink reception, this model is reasonable in an IP setting.

[10]These are the specifications of the Texas Instruments CC2640R2F evaluation board for an AoA-based positioning system, which is taken as a representative platform.

Fig. 2 shows the total transmit energy consumption of each algorithm over the observation interval. We let $K \equiv T_c/T$. Note that this ratio is constant for CPI. In this figure, CPI under $K = 1$ (i.e. zero sleep duration) provides an upper bound to the total transmit energy consumption, while CPI under $K = 10$ serves as a lower bound (for $V = 10$ step ahead forecasting).[11] In the figure, first, we see that the total transmit energy consumption of DPI-RFE remains close to that of the CPI for $K = 10$. Second, we see that DPI-RFE significantly outperforms PID in total energy consumption across the entire observation interval.[12]

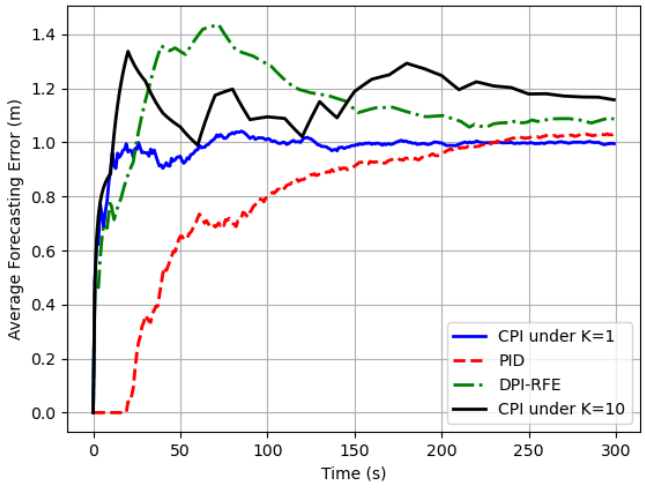

Fig. 3. Comparison of the forecasting error of our DPI-RFE algorithm against those of the CPI and PID algorithms

Fig. 3 displays the time-averaged forecasting error so far (in meters[13]) of each algorithm over the same observation interval. At $t = 300$ s, where approximate convergence has been attained, we see that the average forecasting error of DPI-RFE is slightly higher than that of the PID algorithm. Note that the average forecasting error of both DPI-RFE and PID are above the CPI that has $K = 1$.[14]

Fig. 4 displays the instantaneous positioning interval T_c as a function of time. Note that $T_c = 1$ and $T_c = 10$ s for CPI under $K = 1$ and $K = 10$, respectively. In this figure, we see

[11]The total energy consumption of CPI under $K = 1$ increases linearly as a function of time, as the device consumes constant transmit power and does not sleep in this case. The total energy consumption of CPI under $K = 10$ is a piecewise linear function that is constant whenever the device is asleep.

[12]The plots for the total energy consumption of both DPI-RFE and PID are constant over the intervals on which the device sleeps. Recall that $T = 1$ s throughout our simulations.

[13]Since AoA-based positioning was used in the data set, the average forecasting error is relatively large compared with alternative technologies such as Ultrawideband (UWB). We emphasize that our DPI-RFE algorithm is not specific to AoA and can be used on top of any underlying IP technology.

[14]The forecasting error so far for CPI under $K = 1$ is calculated based on the past vector of position estimates (rather than forecasts) since the device does not sleep at all in this case. Even though the instantaneous forecasting error is not used at all by the CPI algorithm itself under $K = 1$, the average forecasting error attained after convergence in this case serves as a lower bound for other algorithms that are required to utilize past forecasts for those slots in which the device sleeps.

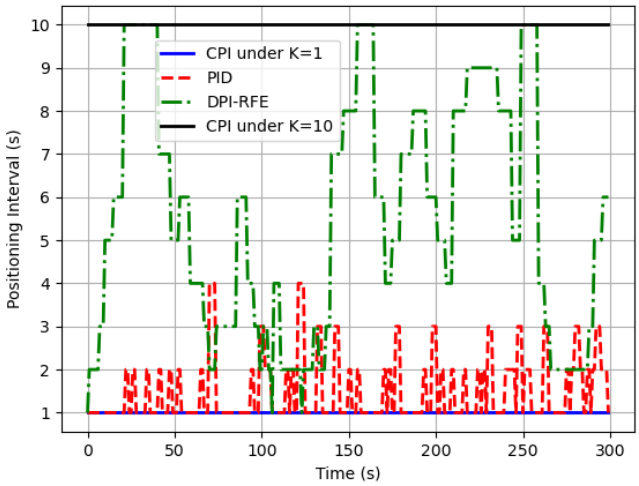

Fig. 4. Instantaneous positioning interval T_c of our DPI-RFE algorithm versus those of the CPI and PID algorithms

that the positioning interval of DPI-RFE varies between this lower and upper bound, while that of PID fluctuates between 1 and 4 s. The average positioning intervals of PID and DPI-RFE are 1.51 and 5.52 s, respectively. Hence, the average positioning interval of DPI-RFE is significantly higher than that of PID.

In summary, although the average forecasting error of DPI-RFE is slightly higher than that of PID as shown in Fig. 3, the total transmit energy consumption of DPI-RFE is significantly lower than that of PID as shown in Fig. 2. The reason is that DPI-RFE acts fast in response to the changes in the instantaneous forecasting error, as shown in Fig. 4, in order to decrease the energy expenditure.

VI. Conclusion

We have developed a novel algorithm called "Dynamic Positioning Interval based on Reciprocal Forecasting Error (DPI-RFE)" that dynamically trades off forecasting accuracy in trajectory prediction and total transmit energy consumption for mobile Internet of Things (IoT) devices by using Artificial Intelligence (AI). Our algorithm forecasts the future trajectory of a mobile device and updates the positioning interval based on the instantaneous reciprocal forecasting error. We have demonstrated our DPI-RFE algorithm outperforms Constant Positioning Interval (CPI) and Positioning Interval based on Displacement (PID) algorithms with respect to total transmit energy consumption, while achieving an average forecasting error that is close to that of PID.

In our future work, we plan to investigate alternative functional forms to the reciprocal of the forecasting error in order to improve the performance further with respect to total transmit energy consumption. In addition, we plan to compare the performance of alternative forecasting models besides the MLP forecaster utilized in this work in order to fully investigate the trade-off between forecasting error and transmit energy consumption.

References

[1] M. Alrashidi, "Social distancing in indoor spaces: an intelligent guide based on the Internet of Things: COVID-19 as a case study," *Computers*, vol. 9, no. 4, p. 91, 2020.

[2] J. Duque Domingo, C. Cerrada, E. Valero, and J. A. Cerrada, "An improved indoor positioning system using RGB-D cameras and wireless networks for use in complex environments," *Sensors*, vol. 17, no. 10, p. 2391, 2017.

[3] I. Constandache, S. Gaonkar, M. Sayler, R. R. Choudhury, and L. Cox, "Enloc: Energy-efficient localization for mobile phones," in *IEEE INFOCOM 2009*. IEEE, 2009, pp. 2716–2720.

[4] Ericsson, *Ericsson Mobility Report*, Jun. 2021, https://www.ericsson.com/en/mobility-report.

[5] D. C. Nguyen, P. Cheng, M. Ding, D. Lopez-Perez, P. N. Pathirana, J. Li, A. Seneviratne, Y. Li, and H. V. Poor, "Enabling AI in future wireless networks: a data life cycle perspective," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 553–595, 2020.

[6] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 961–971.

[7] C. Wang, L. Ma, R. Li, T. S. Durrani, and H. Zhang, "Exploring trajectory prediction through machine learning methods," *IEEE Access*, vol. 7, pp. 101 441–101 452, 2019.

[8] Y. Huang, H. Bi, Z. Li, T. Mao, and Z. Wang, "Stgat: Modeling spatial-temporal interactions for human trajectory prediction," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 6272–6281.

[9] H. Xue, D. Q. Huynh, and M. Reynolds, "SS-LSTM: A hierarchical LSTM model for pedestrian trajectory prediction," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018, pp. 1186–1194.

[10] E. Cakan, A. Sahin, M. Nakip, and V. Rodoplu, "Multi-layer perceptron decomposition architecture for mobile IoT indoor positioning," in *IEEE 7th World Forum on Internet of Things*, 2021, pp. 1–5.

[11] Y. Ma, X. Zhu, S. Zhang, R. Yang, W. Wang, and D. Manocha, "Trafficpredict: Trajectory prediction for heterogeneous traffic-agents," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 6120–6127.

[12] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *2009 IEEE 12th International Conference on Computer Vision*. IEEE, 2009, pp. 261–268.

[13] Y. Chon, E. Talipov, H. Shin, and H. Cha, "Mobility prediction-based smartphone energy optimization for everyday location monitoring," in *Proceedings of the 9th ACM conference on embedded networked sensor systems*, 2011, pp. 82–95.

[14] L. Zhang, J. Liu, and H. Jiang, "Energy-efficient location tracking with smartphones for IoT," in *SENSORS, 2012 IEEE*. IEEE, 2012, pp. 1–4.

[15] X. Liu, Y. Zhan, and J. Cen, "An energy-efficient crowd-sourcing-based indoor automatic localization system," *IEEE Sensors Journal*, vol. 18, no. 14, pp. 6009–6022, 2018.

[16] Z. Yin, C. Wu, Z. Yang, and Y. Liu, "Peer-to-peer indoor navigation using smartphones," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 5, pp. 1141–1153, 2017.

[17] J. L. S. González, L. M. S. Morillo, J. A. Álvarez-García, F. E. D. S. Ros, and A. R. J. Ruiz, "Energy-efficient indoor localization WiFi-Fingerprint system: An experimental study," *IEEE Access*, vol. 7, pp. 162 664–162 682, 2019.

[18] A. Saylam, R. O. Cikmazel, N. Kelesoglu, M. Nakip, and V. Rodoplu, "Energy-efficient indoor positioning for mobile Internet of Things based on Artificial Intelligence," in *Innovations in Intelligent Systems and Applications Conference*, 2021, pp. 1–6.

[19] I. Shafer and M. L. Chang, "Movement detection for power-efficient smartphone WLAN localization," in *Proceedings of the 13th ACM international conference on Modeling, analysis, and simulation of wireless and mobile systems*, 2010, pp. 81–90.

[20] M. B. Kjærgaard, J. Langdal, T. Godsk, and T. Toftkjær, "Entracked: energy-efficient robust position tracking for mobile devices," in *Proceedings of the 7th international conference on Mobile systems, applications, and services*, 2009, pp. 221–234.

[21] "Kinematically Collected Indoor Positioning Dataset," May 2019, https://zenodo.org/record/2647508#.YPBYO-gzbIU.