IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received October 5, 2020, accepted October 21, 2020, date of publication October 27, 2020, date of current version November 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3034114

An Indoor Positioning and Navigation System Using Named Data Networking

SIFAT UT TAKI[1], AMITABHA CHAKRABARTY[1], MD. JALIL PIRAN[2], (Member, IEEE), QUOC-VIET PHAM[3], (Member, IEEE), AND DOUG YOUNG SUH[4], (Member, IEEE)

[1]Department of Computer Science and Engineering, Brac University, Dhaka 1212, Bangladesh
[2]Department of Computer Science and Engineering, Sejong University, Seoul 05006, South Korea
[3]Research Institute of Computer, Information and Communication, Pusan National University, Busan 46241, South Korea
[4]Department of Electronics Engineering, Kyung Hee University, Yongin 17104, South Korea

Corresponding authors: Doug Young Suh (suh@khu.ac.kr) and Md. Jalil Piran (piran@sejong.ac.kr)

This work was supported by the Korea Electric Power Corporation under Grant R18XA02.

ABSTRACT Named data networking (NDN) is a new and promising Internet architecture, which aims to replace the current transmission control protocol/Internet protocol (TCP/IP)-based Internet. The NDN internet architecture has introduced several benefits in numerous applications. An NDN-based indoor positioning and navigation system can further optimize existing localization technologies with reduced the server load and faster response time. In this study, an NDN-based approach for the existing Wi-Fi fingerprinting-based indoor positioning and navigation was investigated. Among the many features of NDN, the network-level caching can reduce the computational load and response time of the localization and navigation server. The theoretical analysis of the runtime complexity shows that the NDN's network-level caching performance is better than those of the conventional algorithms. In this paper, naming methods for different services and a server-side algorithm for handling the NDN requests are proposed. The real-world implantation and testing results show a better overall performance than that of TCP/IP. This network-level optimization for indoor positioning and navigation opens new opportunities because it can be combined with other application-level optimization techniques for more efficient indoor positioning and navigation in the future.

INDEX TERMS Indoor positioning, indoor navigation, localization, named data networking (NDN), RSS fingerprint.

I. INTRODUCTION

The global positioning system (GPS) in modern smartphones makes life simpler by helping users to locate their position and navigate around different places. One of the challenges of the GPS is that it does not perform effectively inside buildings [1]. Making matters worse, GPS receivers in smartphones are not powerful enough to accurately measure altitude in multi-storey buildings [2]. Nevertheless, users require localization and navigation in many modern complexes such as shopping malls, airports, and hospitals. Finding a specific shop in a shopping mall, locating an ATM booth, identifying a terminal in an airport or locating a ward in a large hospital is very common nowadays. To address this problem, researchers have developed various indoor positioning techniques based on Wi-Fi [3], Bluetooth [4], ultrasonic sound [5], radio frequency identification [6], light signals [7] and ultra-wideband [8]. Some of the indoor positioning and navigation technologies can run on a user device independently but most of them require an external server to estimate user location and to find the correct path for navigation inside a building.

There are two phases in fingerprinting-based technologies. The first is the offline phase during which a radio map of an indoor area is generated with the received signal strength (RSS) fingerprints. This radio map database is stored in a server responsible for the calculation of positioning or navigation requests. The second phase is the online phase during which the RSS intensity of the selected radio sources is observed by a user with a device and sent to the server. The server estimates the user's current location by comparing the received RSS values with the existing RSS values from the radio map database and sends the estimated coordinates back to the user. Indoor navigation is also achieved in

The associate editor coordinating the review of this manuscript and approving it for publication was Seung-Hyun Kong.

a similar fashion where the user sends destination information to the server and in response, the server sends back a plausible path (we explain the procedure in details in Section III).

Fingerprinting-based localization has several limitations regarding the accuracy and initial RSS data collection for the radio map. However, the computational load and server response time are also some of the concerns which are the primary focus of this research. The computational complexity of the positioning and navigation algorithms running on the server mainly depends on the size of the indoor area. In addition to acceptable accuracy, real-time responsiveness is also a key to functional indoor positioning and navigation systems. Researchers have adopted different approaches to mitigate this problem. In the current literature, all proposed approaches are based on the application layer. However, there is still room for improvement in the network layer, which is yet to be investigated. Network-level improvements can be combined with other efficient application-level implementations to further optimize the performance of indoor positioning and navigation systems. All the proposed application-level efficiencies rely on the transmission control protocol/Internet protocol (TCP/IP) for the communication between the server and client. While TCP/IP is still robust and fulfilling our needs, it has some problems; TCP/IP is not the most efficient way for transmitting data when designing the Internet of Things (IoT) systems. IoT requires a network with reduced traffic load, fast response time, better mobility and security. TCP/IP cannot meet all these requirements. Therefore, information-centric networking (ICN) is receiving much attention in the IoT research field. ICN shifts the paradigm of point-to-point networking to data-centric networking. In ICN, it does not matter where the data is coming from as long as they are authentic. It helps the networking devices to retrieve data from the nearest possible node (when the data are available) without communicating with the original server from where the data originated.

Named data networking (NDN), which falls under the umbrella of ICN, is a strong candidate for the future of Internet architecture [9]. The fundamental difference between TCP/IP and NDN is that instead of using an IP address to find the destination node, NDN uses a name to find specific data. Traditional TCP/IP uses an IP address, a MAC address, and a port number to establish an end to end connection with the server and the client. NDN shifts this paradigm to a data-centric network in which data are the key component. Instead of IP addresses, names are used. A name indicates the data someone is looking for and not the end point where the data are being produced. This allows intermediate routers to look at the name and identify which data are being requested. In addition, NDN routers carry caches of previously requested data such that the router can satisfy future requests independently. Moreover, when one request is in progress, other identical requests are stalled in the router, which reduces unnecessary network traffic (detailed explanation in Section III-A). NDN also offers better security by encrypting the data instead of the connection between the server and client. Features like adaptive forwarding, better security, and data caching make NDN a better communication protocol over TCP/IP, particularly in IoT applications.

To address the aforementioned problem with the inefficient transmission protocol, an NDN-based approach is proposed in this paper. The main contributions are as follows:

1) A novel NDN-based approach for indoor positioning and navigation systems is proposed to reduce the server load and response time. This approach optimizes the network layer and will be applicable to any server-based indoor positioning or navigation system in the future. It can complement other application layer optimizations and make a system more efficient.
2) A complete NDN-based indoor positioning and navigation system is designed with Wi-Fi RSS fingerprints and floor detection with barometer sensors. The approach is based on fundamental indoor positioning and navigation algorithms.
3) The proposed NDN-based approach is evaluated by comparing it with traditional TCP/IP-based systems. The real-world experimental results have shown reduced server load and response time compared to those of TCP/IP in floor detection, indoor positioning, and indoor navigation services.

The primary objective of this research study was to introduce a better communication protocol in indoor positioning and navigation systems to make the network layer more efficient. This approach can offer reduced the server load and faster response time. Accuracy of the indoor positioning system is outside the scope of this paper.

The remainder of this paper is organized as follows: Section II presents related study reports on different optimization techniques for indoor positioning and navigation. Section III presents the architecture and modeling of the proposed system. Moreover, Section IV presents the implementation of the proposed NDN-based system and tests done in a real-world scenario. The performance of the NDN-based system is evaluated with real results obtained from a mobile application in Section V. Finally, the key conclusions and future study directions are presented in Section VI.

II. RELATED WORK

NDN is getting is receiving much attention from researchers in the IoT field [10]–[12]. Being a data-centric network, NDN particularly favors low powered IoT devices. In addition, NDN provides certain key features for IoT such as caching, data aggregation, adaptive forwarding, and security. Researchers are exploring how NDN can improve wireless networks such as ad-hoc and wireless sensors [13], [14]. The popularity of NDN in IoT applications is growing rapidly as more and more researchers are investigating the improvements and use cases of this promising Internet architecture [15]. Outdoor location-based services have increased in demand in the last couple of decades. Indoor positioning systems are being widely studied recently, and providing robust indoor positioning and navigation solutions is a concern. Moreover, many researchers are studying the accuracy of indoor positioning systems [16]–[19] and how to reduce the server load with different algorithms and implementation techniques [20]–[22].

Wi-Fi fingerprinting-based indoor positioning is one of the most popular indoor positioning techniques currently available. Because most indoor areas already have a Wi-Fi infrastructure, Wi-Fi fingerprinting-based indoor positioning is a cost-effective method with acceptable accuracy. In this localization technique, some Wi-Fi routers are selected as access points (APs) and reference points (RPs) are chosen in the indoor area. The numbers of APs and RPs increase with the size of the indoor area, which directly impacts the performance of the positioning algorithms running in the server. To improve the efficiency of various indoor positioning techniques, many solutions have been proposed in recent years. Some researchers suggested using the Kalman filter to constrain the search space of WiFi fingerprints to improve the accuracy and computational efficiency [23]–[25]. One team in the *IPIN 2018* competition introduced a more efficient algorithm that considers other features of the Wi-Fi RSS such as the entropy parameter and their efficiency weight [26].

Because this research work exclusively focuses on reducing the server load and response time, some of the previously published related works are presented in this section. Dong *et al.* [27] proposed a new efficient fingerprint vector matching algorithm for a large dataset that employs three match making processes. Instead of directly mapping between the location and fingerprint, which takes a lot of computing time, matching the head node is done first in order to narrow down the search area. In the second step, the vectors are matched within the search area. Subsequently, the absolute RSS values are matched to optimize the database search in the third step. Zhang *et al.* [28] adopted a grid search algorithm that optimizes the parameters of the kernel support vector, which improves the computational efficiency of the positioning algorithm. Each RP stores data from all the available APs, which makes them high-dimensional. Processing data with high-dimensions increases the computational complexity of the positioning algorithm. By using an efficient principal component analysis model, the researchers reduced the dimension and complexity of the algorithm. Moreover, Luo *et al.* [29] used a machine learning approach and employed linear discriminant analysis based classification model for floor identification for three-dimensional indoor positioning using Wi-Fi RSS only. In addition, they used a modified algorithm based on K-nearest neighbors algorithm named LL_KNN for positioning on a particular floor. Their approach uses only two APs for floor detection, which significantly reduces the computational complexity. Furthermore, a convolutional neural network (CNN)-based indoor localization system was proposed by Song *et al.* [30], which uses the SAE network to extract one-dimensional CNN and key features of the dataset. Their optimized CNN replaces general matrix multiplication, which reduces computational complexity. Subedi and Pyun *et al.* [31] proposed an affinity propagation clustering (APC)-based fingerprinting localization system with Gaussian process regression to estimate RSS values in the offline phase and used APC to reduce the search space during the online phase; this method improved the accuracy and reduced the computational load.

However, all the proposed approaches are based on the application layer. A network-level optimization of an indoor positioning and navigation system has not been studied. Making the network layer more efficient can further optimize an indoor positioning and navigation system. To implement the fingerprinting-based indoor positioning system with NDN, a more traditional and widely accepted weighted K-nearest neighbors (WKNN) approach was applied. Because this study focuses more on the network layer optimization, the application layer was simplified with fundamental algorithms.

While GPS for outdoor navigation is fairly common and part of our daily lives, navigating indoors and fine-grained navigation have seen a recent increase in popularity [32]–[34]. Dijkstra's algorithm is being widely used for pathfinding problems because it is simple to use and provides robust results [35]. Algorithms like Floyd–Warshall and Bellman–Ford are also used in some indoor navigation systems to optimize the results [36]. One of the most common algorithms is A*. It is more efficient and used in outdoor environments [37] and for indoor navigation [38], [39]; A* is efficient because it calculates the path with heuristic values. Provided a graph, A* will find the shortest path from the given source node to the destination node. While traversing the graph, A* calculates the cost of each possible route with the cost function $F(n) = G(n) + H(n)$ at each node n, where H is the heuristic cost and G the covered distance. A* always chooses the route to the next node with the minimal cost. The algorithm is both optimal and complete. In this study, A* was used for finding the shortest path and navigating the user. All navigation calculations were handled by an external server. By improving the communication protocol with NDN, it was possible to further optimize the indoor navigation system and make it more efficient.

Many researchers study indoor localization and navigation with diverse technologies and techniques. However, an NDN-based indoor positioning and navigation system has not been explored. Because NDN is designed to optimize IoT applications, the use of NDN in indoor positioning and navigation systems can provide various benefits. The possibility and potential of an NDN-based indoor positioning and navigation system was the focus of this research study. Therefore, a Wi-Fi RSS fingerprinting-based indoor positioning and navigation system with barometer sensors for floor detection was implemented with NDN.

III. SYSTEM ARCHITECTURE

Most modern smartphones include micro-electromechanical system (MEMS) pressure sensors, which can measure the pressure with relatively high accuracy [40].

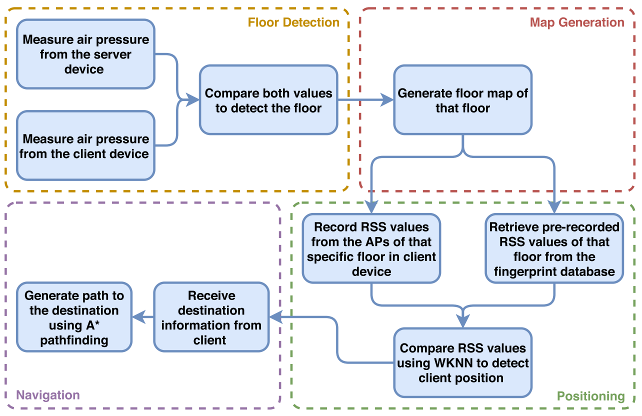

Floor Detection
Measure air pressure from the server device
Measure air pressure from the client device
Compare both values to detect the floor
Map Generation
Generate floor map of that floor
Record RSS values from the APs of that specific floor in client device
Retrieve pre-recorded RSS values of that floor from the fingerprint database
Compare RSS values using WKNN to detect client position
Positioning
Receive destination information from client
Generate path to the destination using A* pathfinding
Navigation

FIGURE 1. Application layer framework of the system.

Recent studies [41]–[43] have proved that barometer sensors are compatible with other sensors and systems. For floor detection, two barometer sensors are used: one is the reference barometer sensor connected to the server, and the other is the user device's barometer sensor. The air pressure values from these two barometer sensors are compared to identify the floor. The reference barometer sensor can be placed on any floor.

As previously mentioned, there are two phases in a Wi-Fi RSS fingerprinting-based positioning system. In the offline phase, some Wi-Fi routers are selected as APs, and some RPs in the indoor area are selected. The RPs are selected such that an almost uniform grid is created. The RSS values of each AP are recorded from all the selected RPs. Each RP contains RSS fingerprints and the corresponding two-dimensional coordinate. This way, a radio map is generated. In the online phase, the user device records the RSS values from the selected APs and sends them to the server where they are matched with the pre-recorded RSS values from the calibration phase with WKNN. Subsequently, the current position of the user is estimated. The grid of the RPs from the generated radio map is also used as the traversal tree for navigation with A* pathfinding. The navigation algorithm generates a path from the user's nearby RP to the nearest destination RP.

The setup in Figure 2 presents the server and client which are connected to an NDN router. The server has a local database for storing the fingerprint data. A barometer sensor is also connected to the server. First, the user's floor is detected, and information required to generate the map of that specific building floor is sent to the client's device. The location of the user is estimated by matching the observed RSS values of the selected APs from the client's device and the existing RSS values of the RPs from the radio map. Indoor navigation requires users to send destination information to the server. Once the server has the destination information, it can calculate the shortest possible route and send the result to the client's device. Figure 1 demonstrates the flow of the system.

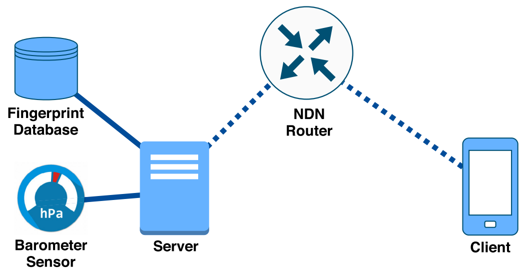


FIGURE 2. Network setup of the system.

A. NDN ARCHITECTURE

There are two types of packets in NDN: 1) the interest packet, which is sent by the consumer (client) and contains information about the requested data and 2) the data packet, which is produced by the producer (server) and contains the data requested by the consumer. NDN maintains three data structures: forwarding information base (FIB), pending interest table (PIT), and content store (CS). the FIB contains the information about next hops, the PIT stores all the identical interest packets that are currently being processed, and the CS is responsible for caching all the previously requested data packets. In NDN, the routers are stateful, i.e., the router knows which data the user is asking for.

When an interest packet arrives at a router, the router first checks if it has the requested data in its CS. If the requested data is available, the router can satisfy the request. If the data is unavailable in the CS, the router checks the PIT in which all the pending requests are stalled. If the requested data is already pending, the router puts the interest packet in the PIT while waiting for the response from its next hop. If the router receives multiple interest packets for identical data from the same consumer, the router only keeps one interest packet in the PIT. Moreover, if there is a new incoming interest packet (which is unavailable in the CS or PIT), the router searches its FIB table to forward the interest packet accordingly and create a new PIT entry. Once the data packet arrives, it is cached in a new CS entry for future requests. Figure 3 presents this process.

B. DIFFERENT SERVICES AND ALGORITHMS

Floor detection, map generation, indoor positioning, and indoor navigation use different services to process individual NDN requests from the user. In this section, the application layer implementations are discussed in this section. the network layer NDN implementation is discussed in Section IV.

1) FLOOR DETECTION SERVICE

The atmospheric pressure decreases with the altitude. By using a barometric sensor to measure the air pressure changes, it is possible to calculate the change of altitude corresponding to change in the pressure with the following

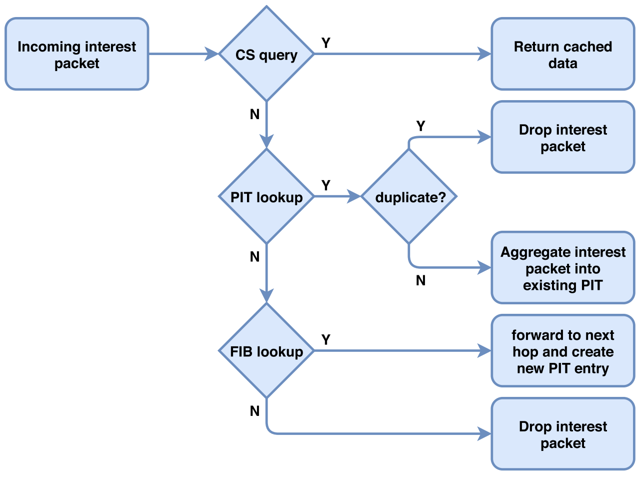


FIGURE 3. Processing of NDN interest packets in a router.

equation:

$$h = \left(1 - \sqrt[5.255]{\frac{p}{1013.25}}\right) \times \frac{288.15}{0.0065}, \tag{1}$$

where h is the altitude in meters, and p is the pressure in hectopascal [44]. Using (1), when the height is increased by approximately 8.37 m (27.46 ft), the pressure changes by approximately 1 hPa [45]. Therefore, (1) can be further simplified to following the equation:

$$p = \frac{h}{8.37}, \tag{2}$$

because the absolute altitude is not required for floor detection. The height of each floor of the building is stored in the database of the server. With (2), The user's floor is determined from the difference in height between the two sensors and the floors. In this study, the reference barometer sensor was installed on the ground floor. The equation for estimating the floor can be defined as follows:

$$floor \approx (alt_{ref} - alt_{user})/h_{floor}, \tag{3}$$

where alt_{user} and alt_{ref} are the estimated vertical distances of the user and reference sensor from the ground, respectively; h_{floor} is the height of each floor of the building. The air pressures P_{ref} from the reference sensor and P_{user} from the user are used to calculate alt_{ref} and alt_{user} using (2). With these values, Algorithm 1 uses (3) to detect the user's current floor.

2) POSITIONING SERVICE

The positioning service uses the WKNN algorithm to estimate the user's current location on a specific floor. This algorithm is one of the most widely used algorithms for indoor localization. The observed RSS values from all the pre-selected APs are sent from the user's device to the server through NDN. Subsequently, The server calculates the user's current location with the WKNN algorithm.

Algorithm 1 Floor Detection

```
Input : The reference pressure $P_{ref}$ (hPa), the user
        device pressure $P_{user}$ (hPa), height of each
        floor $h_{floor}$ (m) from the database
Output: floor number
begin
    $alt_{ref} = 8.37 * P_{ref}$;
    $alt_{user} = 8.37 * P_{user}$;
    floor = $Round((alt_{ref} - alt_{user})/h_{floor})$;
    return floor
end
```

If there are N_{AP} number of APs and N_{RP} number of RPs, the RSS vector of the i^{th} RP from the existing fingerprint database is $RSS_i = RSS_{i1}, RSS_{i2}, \ldots, RSS_{ij}$, where $i = 1, 2, \ldots, N_{RP}$ and $j = 1, 2, \ldots, N_{AP}$. The RSS vector from the user's device is $RSS_{user} = RSS_1, RSS_2, \ldots, RSS_j$. First, the Euclidean distance D_i between RSS_{user} and all the RSS_i is calculated with the following equation:

$$D_i = \sqrt{\sum_{j=1}^{N_{AP}} (RSS_{ij} - RSS_j)^2}. \tag{4}$$

Because the WKNN is used, the weight is assigned with the following equation:

$$w_i = \frac{1/D_i}{\sum_{j=1}^{k}(1/D_j)}, \quad i = 1, 2, \ldots, k, \tag{5}$$

where the value of k is 4. Finally, the user's coordinates (x_u, y_u) are estimated using the following equation:

$$(x_u, y_u) = \sum_{i=1}^{k} w_i(x_i, y_i). \tag{6}$$

3) NAVIGATION SERVICE

For the navigation service, the system uses the existing grid of RPs to create a graph, and A* algorithm is used to find the path. Each RP has a corresponding two-dimensional coordinate used by the server to identify the user's current location and destination. The A* algorithm works with heuristic values. Because a uniform grid is used, the heuristic value is calculated with the Euclidean distance. By using Algorithm 2, the service calculates all the required nodes to create the shortest path from the user's current nearest RP to the nearest RP of the destination. Thus the algorithm uses the current location node (x_1, y_1, z_1) and destination location node (x_2, y_2, z_2), where x and y indicates the two-dimensional coordinates an RP, and z is the floor number. When a user requests navigation to a different floor, the algorithm shows the path to the nearest exit point. *Closed* and *Open* keep track of all the visited nodes and next possible nodes, respectively. By using the heuristic cost H and covered distance G, $F = G + H$ can be calculated to determine the shortest path from (x_1, y_1) to (x_2, y_2).

Algorithm 2 Navigation Using A*

```
min_cost = function to find min cost of an RP node;
Input  : current location $(x_1, y_1, z_1)$, destination
         $(x_2, y_2, z_2)$
Output: route
begin
    if $z_1 \neq z_2$ then
        return route to the nearest exit point;
    Closed ⟵ empty set;
    Open ⟵ include $(x_1, y_1)$;
    while Open set $\neq \emptyset$ do
        Current ⟵ min_cost(Open);
        if Current == $(x_2, y_2)$ then
            route ⟵ Backtrack to $(x_1, y_1)$;
            return route;
        for each neighbor N of Current do
            if N is in Closed then
                Continue;
        end
        else
            calculate N's G and H;
            $F = G + H$;
            N's parent ⟵ Current;
            Open ⟵ N;
        end
    end
end
```

C. THEORETICAL ANALYSIS OF SYSTEM PERFORMANCE

A uniform grid of RPs and APs (with δ and Δ intervals, respectively) is considered as a model (Figure 4). This simplified model helps to correlate the numbers of RPs and APs with the area size to understand how different algorithms are impacted when the indoor area increases. In addition the model helps to demonstrate how NDN cache reliance

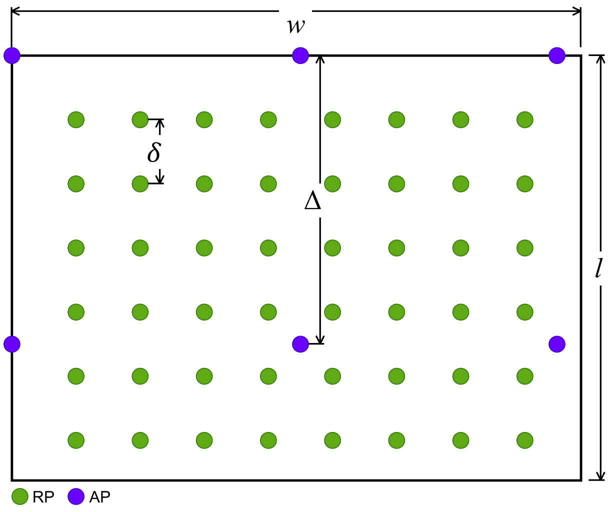


FIGURE 4. **AP and RP selection for an indoor area.**

can improve the performance and efficiency compared to the traditional TCP/IP approach.

The number of RPs varies with the area size and distance between them. For example, if the width and length of a rectangle are x units and y units, respectively, the number of squares in the grid with one unit width that would fit into the rectangle can be easily calculated. Because the model does not consider points along the edges, they must be subtracted and one unit must be added because of the intersection of two edges. Thus, the number of squares N in the grid is as follows:

$$N = (x-1)(y-1). \tag{7}$$

Hence, the model has a uniform grid of RPs with δ intervals. Dividing the edges by δ and determining the ceiling of the value will provide the exact number of squares. By modifying (7) for the model, the following equation is obtained:

$$N_{RP} = \left(\left\lceil \frac{w}{\delta} \right\rceil - 1\right)\left(\left\lceil \frac{l}{\delta} \right\rceil - 1\right), \quad \text{where } \delta < w, l, \tag{8}$$

where N_{RP} is the number of RPs required for the system, w the width, l the length of the area, and δ the interval between each RP. For square areas, (8) can be rewritten as follows:

$$N_{RP} = \left(\left\lceil \frac{l}{\delta} \right\rceil - 1\right)^2, \quad \text{where } \delta < l. \tag{9}$$

In addition, the AP positions for the model are selected in a grid formation (Figure 4). However, in a real-life scenario, it may not be possible to select APs in a grid formation as the model proposes. The main purpose of this model is to correlate the numbers of APs and RPs with the area size which is impossible without considering the uniformity of the APs. Similar to (8), the number of APs can be calculated. However, unlike the RPs, the APs are selected along the edge in this model. Thus, the value is floored, and one unit is added to each edge to determine the exact number. The required number of APs for an indoor area can be calculated as follows:

$$N_{AP} = \begin{cases} \left(\left\lfloor \frac{w}{\Delta} \right\rfloor + 1\right)\left(\left\lfloor \frac{l}{\Delta} \right\rfloor + 1\right), & \text{if } \Delta \le w, l \\ 4, & \text{otherwise.} \end{cases} \tag{10}$$

For square areas, (10) can be rewritten as follows:

$$N_{AP} = \begin{cases} \left(\left\lfloor \frac{l}{\Delta} \right\rfloor + 1\right)^2, & \text{if } \Delta \le l \\ 4, & \text{otherwise,} \end{cases} \tag{11}$$

where N_{AP} is the number of required APs, and Δ the distance between two APs. At least four APs are required for the system to work properly. The value of Δ should be sufficiently small such that it remains within the range the of Wi-Fi radio signal.

The performance heavily relies on the NDN router's data cache. WKNN is a very common algorithm used in indoor positioning systems based on Wi-Fi RSS fingerprinting. The A* algorithm is another common path finding algorithm. These algorithms have high computational complexity.

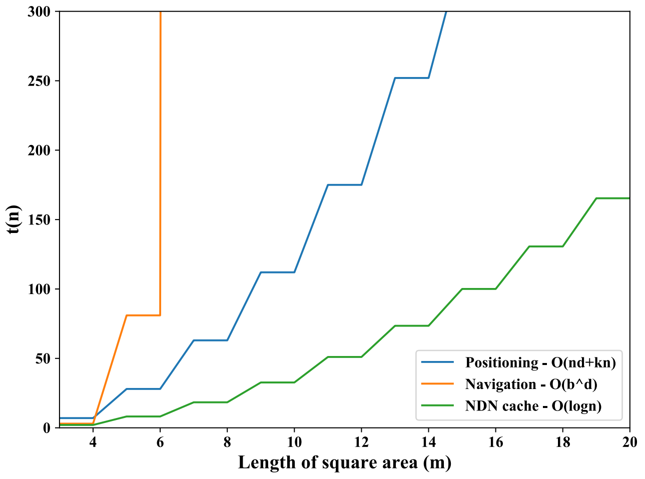


FIGURE 5. Computational complexity analysis of different algorithms with respect to area size.

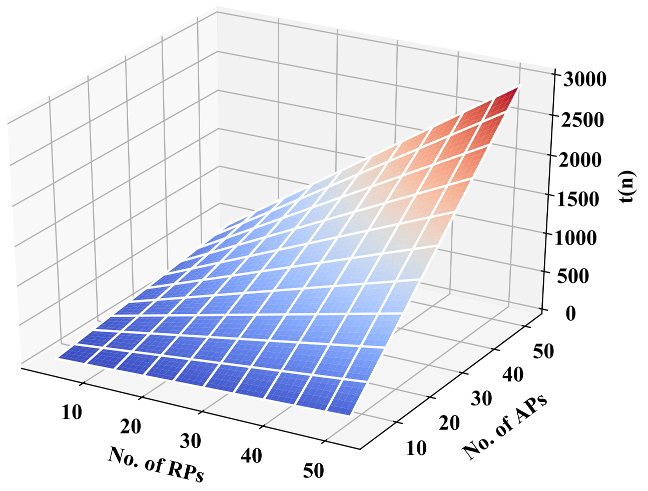


FIGURE 6. Computational complexity analysis of positioning algorithm with respect to numbers of APs and RPs.

When the indoor area increases, the complexity increases because the numbers of RPs and APs increase. By using the presented model, the numbers of RPs and APs can be calculated, and the performance characteristics of the algorithms for an increasing area size can be compared. For the positioning algorithm, the computational complexity for the distance computation for a single RP is $\mathcal{O}(d)$ where d is the size of the vector. The distance computation for n number of RPs require $\mathcal{O}(nd)$ runtime. Finding the nearest K-number of the neighbors require $\mathcal{O}(kn)$ time. Thus, the total computational complexity for the positioning algorithm is $\mathcal{O}(nd + kn)$. In this study, WKNN is used which requires additional $\mathcal{O}(k)$ time to assign weight. However, because the value of k is considered 4, it becomes a constant. The A* algorithm has a computational complexity of $\mathcal{O}(b^d)$, where b is the branching factor, and d is the depth. The tree is a uniform grid in this model. As a result, the branching factor b is 3 on average, and the depth d is the total number of RPs.

An NDN router stores cache in the CS, and the CS uses Set as its underlying data structure [46]. Set uses a red–black tree which has a computational complexity of $\mathcal{O}(\log n)$ for searching, inserting, and deleting data. Moreover, the CS stores data in packets. A data packet can carry a maximum of 8800 bytes of data. The default size of the CS is 65536 packets, which correspond to approximately 500 MB; however, the value can be increased if required.

The scenario considered for the theoretical model has a square indoor area in which δ is 2 m and Δ is 15 m. By using (11) and (3), it is possible to calculate the numbers of RPs and APs for that square area. When the length of the area is increased, the numbers of APs and RPs increase proportionally, thus increasing the complexity. Figure (5) presents the length of the square area on the x axis and the growth of algorithmic computational complexity $t(n)$ of the algorithms on the y axis. For each sample from the length on the x axis, numbers of RPs and APs are calculated using (11) and (3). The growth of algorithmic computational complexity is calculated using the numbers of RPs and APs and the aforementioned big O notation for each algorithm. For NDN, the cache size increases with the increasing length to accommodate every possible combination of RSS (up to the tenth decimal place) along with every navigation combinations. Although it may be unfeasible in real life to increase the cache infinitely, theoretically, the caching algorithm still outperforms other algorithms when comparing the growth of algorithmic computational complexity for large indoor areas; thus, NDN's caching algorithm is more efficient.

In addition, the algorithm complexity can be compared based on the numbers of RPs and APs instead of the area size. The growth of the algorithmic computational complexity $t(n)$ of the algorithms is directly proportional to the numbers of RPs and APs used in the indoor positioning and navigation system. Figure 6 presents the growth of the computational complexity of the positioning algorithm and Figure 7 shows

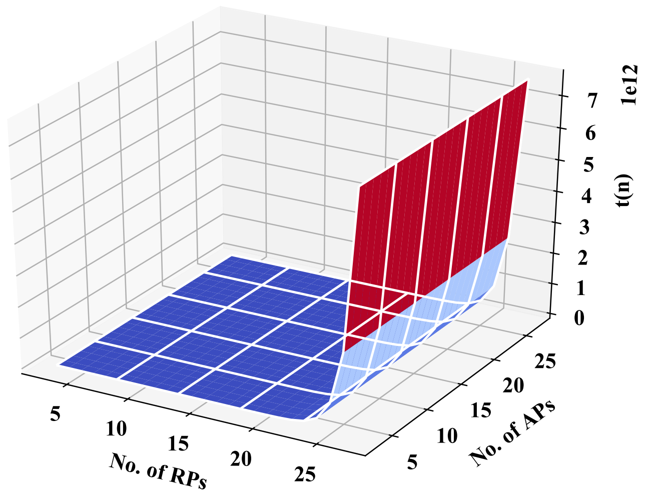


FIGURE 7. Computational complexity analysis of navigation algorithm with respect to numbers of APs and RPs.

the growth of the computational complexity of the navigation algorithm with respect to the numbers of RPs and APs. In the positioning algorithm, the growth depends on the numbers of RPs and APs used, whereas in the navigation algorithm, the growth only depends on the number of RPs because it uses a grid of RPs as the tree. Figure 8 presents the computational complexity of the NDN caching algorithm with respect to the numbers of APs and RPs. Because the RSS values vary more (from 0 to -110 dBm in this study), the APs create significantly more combinations than the RPs, which causes the complexity to grow significantly when the number of APs increase.

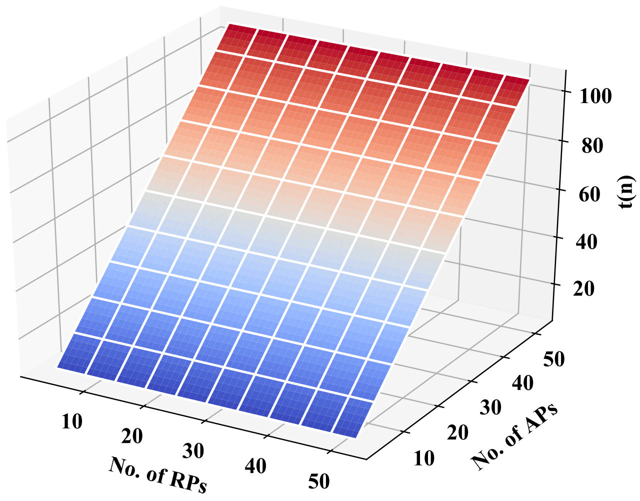


FIGURE 8. Computational complexity analysis of NDN caching algorithm with respect to numbers of APs and RPs.

D. PROBABILITY OF HITTING NDN CACHE

The probability of hitting cache in an NDN router can be calculated with respect to the CS size of that router. The same model as in Figure 4 is used. The data will be retrieved from the cache if users request data with identical information; this will generate the same name, thereby enabling the routers to understand if the user is asking for the same data. The more often one service is used, the more combinations of data will be generated and stored in the CS of a router. An increase in the CS entries for a service will increase the probability of hitting cache for that service. For a sufficient cache size, if every possible combination is generated and stored in the CS, the probability of hitting cache will be 1. All the notations used throughout the paper are available in Table 1.

The navigation service requires two values: the current position (nearest RP) of the user and the the destination position. Both values must be identical to produce the same name in NDN. The probabilities of having the same current location and destination can be calculated and combined. The destination can be any RP except for the current location. However, the current position's RP is limited to the router's wireless radio signal range. Because the RPs form a grid, the number of squares n that will fit into a circle can be calculated. If the radius of a circle is R, and the length of each square of the grid is L, the ratio of the area is $\pi R^2/L^2$. For the approximation of squares removed at the circle's circumference, $\sqrt{2}L$ is used as the average, which is the diagonal length of the square. By using this average, the number of cutoff squares can be calculated with $2\pi R/\sqrt{2}L$. Hence, the total number of squares can be defined as follows:

TABLE 1. Notation for system model.

Notation	Description
δ	Distance between RPs in the grid
Δ	Distance between APs in the grid
r	Router's theoretical range
N_{AP}	Total number of APs
N_{RP}	Total number of RPs
n_{AP}	Number of APs within user's range
n_{RP}	Number of RPs within router's range
$d_{\max}$	Range of an AP (maximal distance)
f	Frequency of radio signal of an AP
σ	Standard deviation of RSS
Δ_{alt}	Change in altitude for 0.1 hPa difference in air pressure
CS_{nav}	CS entries in an NDN router for navigation service
CS_{pos}	CS entries in an NDN router for positioning service
CS_{floor}	CS entries in an NDN router for floor detection service
$P(Hit)_{nav}$	Probability of hitting cache in navigation service
$P(Hit)_{pos}$	Probability of hitting cache in positioning service
$P(Hit)_{floor}$	Probability of hitting cache in floor detection service

$$n \approx \frac{\pi R^2}{L^2} - \frac{2\pi R}{\sqrt{2}L}. \tag{12}$$

Equation (12) can be used to estimate the number of RPs within a Wi-Fi router's range with:

$$n_{RP} \approx \frac{\pi(r+\delta/2)^2}{\delta^2} - \frac{2\pi(r+\delta/2)}{\sqrt{2}\delta}, \tag{13}$$

where n_{RP} is the number of RPs within the range of that router, and r is the theoretical range of the Wi-Fi router. Here, the diameter of the circle is increased by δ. This results in a better approximation of the maximal RPs if the AP is not perfectly aligned with the grid of RPs. The value of n_{RP} from (13) is used for the following equation:

$$P(Hit)_{nav} = \frac{1}{n_{RP}} \times \frac{1}{N_{RP}-1} \times CS_{nav}, \tag{14}$$

to determine the probability of hitting cache in the navigation service; N_{RP} is the total number of RPs, and CS_{nav} is the number of CS entries in the NDN router for navigation requests.

For the positioning service to hit cache, all the RSS values from the selected APs must be the same. This way, the name will be the same, and the router will be able tell whether the user is requesting identical data or not. Owing to multipath propagation, it is difficult for each RSS value to be identical, even when standing in the exact same location. However, for enough tries and a sufficient CS_{pos} size, it is possible to hit cache. Section III-C theoretically demonstrated that having more cache size does not affect the performance significantly. Here, the minimal signal strength for an AP is assuned to be -110 dBm to guarantee an adequate number of APs within the users range; This number can be changed if necessary.

If RSS ≤ -110 dBm, it is considered out of range. Thus, for the service to hit cache, only the APs within the acceptable range must match. The APs outside the range will always generate -110 dBm. The theoretical maximal distance from an AP before it is considered out of range is calculated with the free space path loss model for Wi-Fi signals:

$$\begin{aligned} 110 &= 20\log(d_{\max}) + 20\log(f) - 27.55 \\ \implies d_{\max} &= 10^{(27.55-20\log(f)\,+110)/20}, \end{aligned} \tag{15}$$

where $d_{\max}$ is the maximal distance in meter from the router for -110 dBm, and f is the frequency of the radio signal in mHz. With (15), the maximal number of acceptable APs within the user's range can be determined as follows:

$$n_{AP} \approx \frac{\pi(d_{\max} + \Delta/2)^2}{\Delta^2} - \frac{2\pi(d_{\max} + \Delta/2)}{\sqrt{2}\Delta}. \tag{16}$$

eqnarray (16) provides n_{AP}, which is the number of APs within the user's range. This is based on the assumption that the APs form a grid. The probability of generating the same request can be calculated if a user stands in the exact same location. If the standard deviation of the RSS is known, the probability of hitting the cache in one location can be calculated as follows:

$$P(Hit)_{pos} = \sum_{n=1}^{n_{AP}} \frac{1}{\sigma_n} \times CS_{pos}, \tag{17}$$

where σ is the standard deviation of the RSS for each AP, which depends on the fluctuation of the RSS; CS_{pos} is the number of CS entries for the positioning requests.

During floor detection, the system measures the air pressure in hectopascal and rounds the value off to one decimal place. If the change in altitude for 0.1 hPa difference in the air pressure is known, the maximal number of air pressure samples within the router's range can be calculated. The equation for calculating the probability of hitting cache in the floor detection service is as follows:

$$P(Hit)_{floor} = \frac{\Delta_{alt}}{2r} \times CS_{floor}, \tag{18}$$

where r is the theoretical range of the Wi-Fi router, Δ_{alt} the change in altitude for 0.1 hPa difference in the air pressure, CS_{floor} the number of CS entries for floor detection requests.

IV. NDN IMPLEMENTATION

To implement and test the proposed NDN-based indoor positioning and navigation system, a server was created with Java. MongoDB was used as the local database for storing the RSS fingerprints. The database was running on the server computer. MongoDB is a NoSQL document-based database. Consequently, the database can access large volumes of data at a very high speed and can be scaled-out easily with a minimal performance loss. Moreover, an Android mobile application was developed to test the performance of the system. The application had TCP/IP and NDN options to communicate with the server. This enabled the comparison of the performance difference between the TCP/IP and NDN. Another computer (Raspberry Pi) was installed between the server (producer) and client (consumer) as a router because no NDN router was available at the time of this research study. NDN forwarding daemon (NFD) was running on that router. Although NDN has the native capability of running over Ethernet, there is no global-scale native NDN network because NDN is still at a very early development stage. Instead, NDN can run as an overlay network on top of a traditional IP network using NFD. This router computer was connected to the server and the client with WLAN 802.11ac wireless connection; it was responsible for handling the TCP/IP and NDN connections.

The smartphone application was designed to show a simplified view of the map. It shows the user's estimated position inside the building and provides navigation to the given coordinate. In addition, the application shows the time interval between sending a request and receiving a response in milliseconds. The BCS Computer City, which is a shopping complex in Dhaka, Bangladesh, was chosen as the test site (Table 2). The RPs and APs with intervals of approximately 3 and 15 m were chosen, respectively. The test was done in a relatively small-scale environment to show the potential of the NDN in indoor positioning and navigation systems. The authors assumed that if NDN can deliver promising performance in a small scale environment in which the complexity is already low for the traditional TCP/IP-based systems, it will be suitable for large-scale deployments.

TABLE 2. Test site information.

Type	Shopping mall
Story	4
Height	17 m
No. of shops	322
Area per floor	3500 m^2 (approximately)
δ	3 m
Δ	15 m
No. of APs (per floor)	18
No. of RPs (per floor)	92

In NDN, naming the data is one of the crucial tasks. Each interest and data packet should generate non-identical names. Based on the requirements, NDN allows flexible naming conventions for different applications. The NDN names follow a hierarchical naming structure, and the components are separated with '/'. The names for all the services were designed such that they could take the advantage of NDN's advanced features. The naming structure of all the services has this pattern: "/ips/place_name/service_name/floor_number/service-specific_information".

By the time of the experiment, there was no dedicated NDN capable router available. Therefore, a Raspberry Pi 3 Model B that ran the NFD was used as a router. The server application ran on a MacBook Pro 2019, and Samsung Galaxy Note 9 was used as the client device (Table 3).

A. FLOOR DETECTION SERVICE NAMING METHOD

As previously mentioned, the floor detection service used in this research study requires at least two barometer sensors:

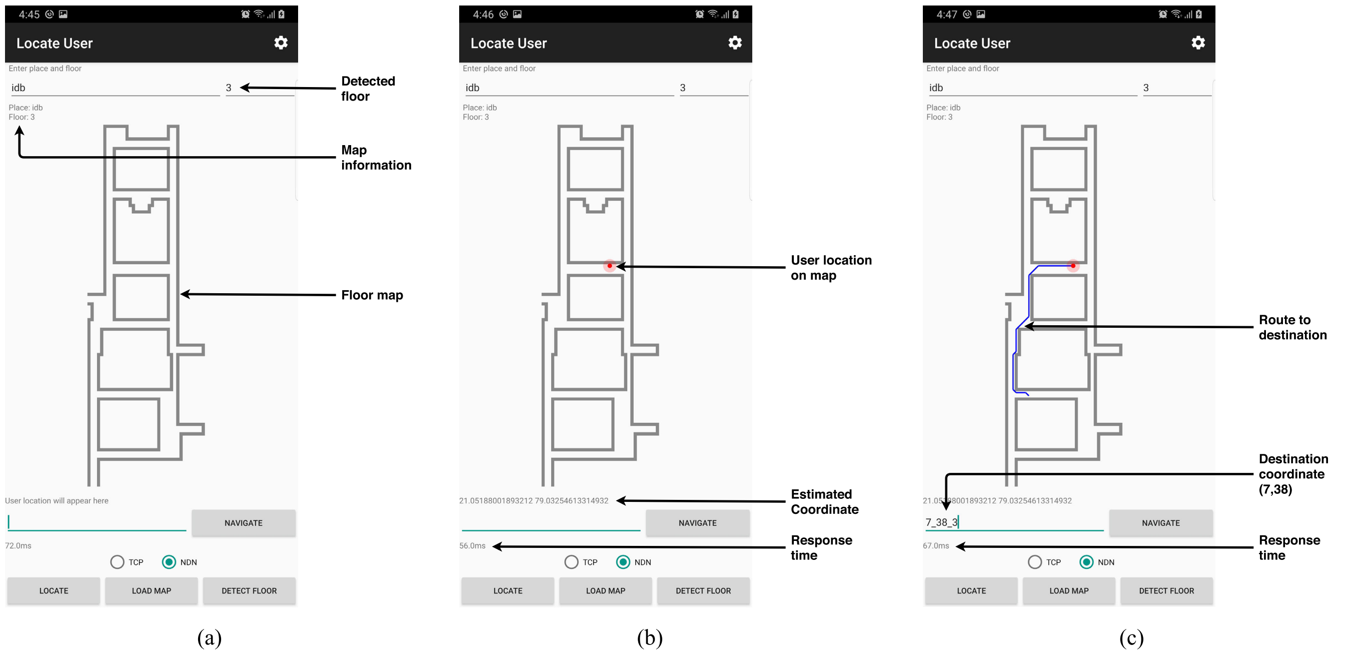


FIGURE 9. **Android client application: (a) Map view, (b) Current location, (c) Route to destination.**

TABLE 3. **Test device information.**

	Server	Router	Client
Name	MacBook Pro 2019	Raspberry Pi 3B	Samsung Galaxy Note 9
CPU	1.4 GHz i5	1.2GHz 64bit ARM	1.8 GHz Cortex-A55
Memory	16 GB	4 GB	6 GB

one on the server side and one on the client side. The client device records the atmospheric pressure in hectopascal and sends the request to the server. Subsequently, the server calculates the distance from its reference point with the value from its own reference air pressure sensor. The database contains information about the building, which: the server uses to detect the floor; "/ips/idb/floor_detection/1013.4" is an example for the naming method used in this service. Because names in NDN are hierarchical, each part of the name narrows down the information the client or consumer is looking for. In this case, "/ips" indicates that this component is for the indoor positioning system. The next part in the name, which is "/idb", contains the name of the building. The following part represents the service (in this case, it is the floor detection). The last part contains the value of the air pressure from the client's device (which is 1013.4 hPa in this example). When the server satisfies the request for the first time, the floor information for that specific air pressure is stored in the router's CS. Whenever there are other requests with the same air pressure value, the names of the NDN interest packets will be identical. Thus, the router can send the data packet from its cache.

B. MAP SERVICE NAMING METHOD

After successfully detecting the floor, the map must be generated in the client's device to compare the performance of NDN; "/ips/idb/load_map/3" is an example of how the name for loading the map is structured. The "/ips/idb/" part represents the floor name. The following part indicates the service used for loading the map of the floor. Finally, "3" is the floor number, for which the map is generated. The NDN data packet contains all necessary coordinate information for generating the map. In addition, it contains the information of the APs on that floor and the sequence, which is crucial for positioning. After acquiring the map information, the application uses the coordinates from the map information to draw the map on the user's device (Figure 9).

C. POSITIONING SERVICE NAMING METHOD

The client device first records the signal intensity from the selected Wi-Fi APs in decibel and sends the information to the server. The server accesses the database, which contains the information of all the RPs and corresponding RSS values of the APs. Based on the observed RSS from the client and existing RSS values from the database, the WKNN algorithm is used to estimate the user's location. The name is constructed as follows: "/ips/idb/position/3/-43_-32_-84_-57_-55…", where "/ips/idb/" part is identical to the floor detection name. The next part represents the service used to detect the user position. Moreover, '3/' is the floor number. Finally, the last part contains all the observed RSS values from the client device. The values are in a specific order such that the server can understand which RSS value corresponds to which AP. They are separated by "_".

Similarly to the floor detection, after supplying the client device with the position information for that specific name, the cache will be stored in the NDN router's CS. The future interest packets with identical RSS values will be satisfied by the NDN router. As previously mentioned, it is difficult to match all the RSS values for each AP, even if the user is standing in the exact same spot. However, after several requests, some interest packets with identical names might be generated. The approximate location of the user is shown on the application screen (Figure 9).

D. NAVIGATION SERVICE NAMING METHOD

Once the position of the user is determined, navigation to a desired location can be provided. The destination coordinates and destination floor are sent to the server from the client device. The server calculates the shortest possible route using Algorithm 2 and the route information is sent to the client device; "/ips/idb/navigation/21_79_3_7_38_3" is an example of the naming method for the navigation service. Again, "/ips/idb/" represents the same information as the floor detection, map, and positioning service. The following part indicates the service name (which is the navigation service in this case). In the last part, the first three values (21_79_3) represents the current position of the user. In this example, the current x coordinate is 21, y coordinate is 79, and 3 is the current floor. The last three values (7_38_3) indicates the destination information. the destination x coordinate is 7, Y coordinate is 38 and the destination floor is 3. By using the route information, the application can draw the navigation path on the map of the application (Figure 9).

All services follow similar processes when requesting information from the server (Figure 10). The server application (i.e., "the producer" in NDN) receives the interest packet from the client device (i.e., "the consumer" in NDN). Algorithm 3 is developed for the server to handle the naming methods of NDN used in different services. When an interest packet arrives at the producer, it can see the full *Name*. The components of the name are extracted and stored in the *Components* list in which the first item indicates the indoor positioning and navigation system, the second item represents the *Place*, and the third item indicates the *Service*. The other items in the *Components* list are service-specific information for other algorithms.

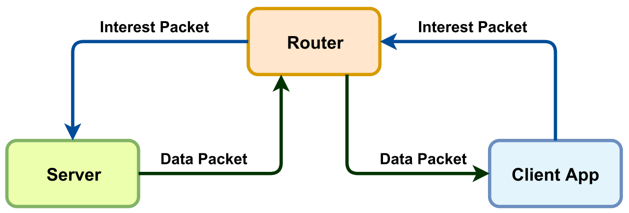
Algorithm 3 NDN Request Processing in Server Application

```
input  : Interest Packet
output: Data Packet
begin
    Name ⟵ get Name from the received Interst Packet;
    Components ⟵ List of components from Name separated by '/';
    Request ⟵ Components [0];
    if Request == ips then
        Place ⟵ Components [1];
        Service ⟵ Components [2];
        access database using Place information;
        if Service == floor_detection then
            $P_{user}$ ⟵ Components [3];
            run floor detection service (Algorithm 1) with $P_{user}$;
            return Data Packet containing floor information;
        else if Service == load_map then
            Floor ⟵ Components [3];
            get map information of the Floor from database;
            return Data Packet containing map information;
        else if Service == position then
            Floor ⟵ Components [3];
            $RSS_{user}$ ⟵ Components [4];
            run positioning service with $RSS_{user}$;
            return Data Packet containing location information;
        else if Service == navigation then
            $x_1, y_1, z_1, x_2, y_2, z_2$ ⟵ Components [3];
            run navigation service (Algorithm 2) with $(x_1, y_1, z_1), (x_2, y_2, z_2)$;
            return Data Packet containing route information;
        else
            Continue;
        end
    else
        Continue;
    end
end
```

V. PERFORMANCE EVALUATION

As the NDN-based indoor positioning and navigation system was implemented in the BCS Computer City, data for the proposed methodology could be gathered. Therefore, conducted multiple tests with different scenarios were conducted to compare NDN with TCP/IP. The following tests were conducted to compare NDN with TCP/IP.

- **Response Time:** the time it takes to receive data from the server for the client for each service was measured. The time was measured in nanoseconds and then converted to milliseconds. Because the client was an android application, *System.nanoTime()* method was used to measure the precise time required to execute one method. Each mode was measured 20 times with the NDN and TCP/IP protocols. For NDN, two different measurements were taken: one for the best-case scenario for NDN, in which same requests were generated to hit the cache. Instead of manually generating the same request with hard-coded data, a more real-world approach was considered for all of the services. The other scenario was the worst-case scenario in which the test subject tried to generate different requests each time;



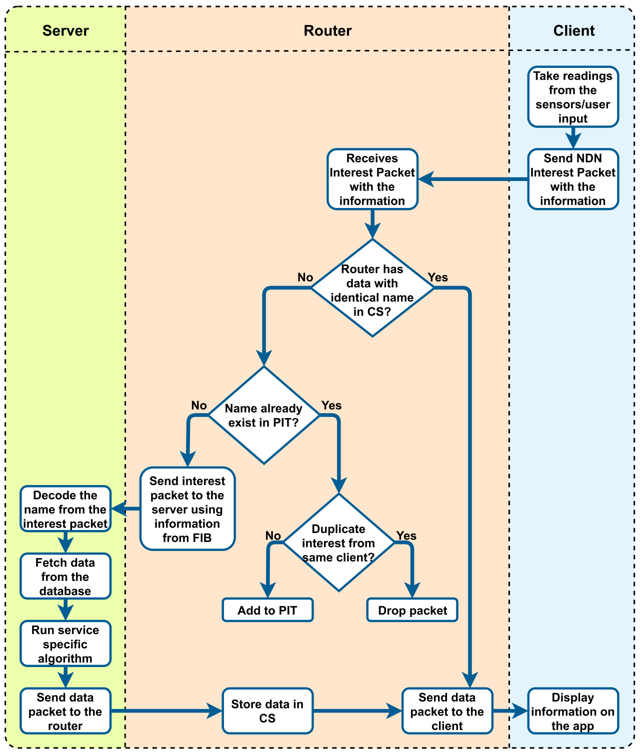

Server
Router
Client
Take readings from the sensors/user input
Send NDN Interest Packet with the information
Receives Interest Packet with the information
Router has data with identical name in CS?
No
Yes
Name already exist in PIT?
No
Yes
Send interest packet to the server using information from FIB
Duplicate interest from same client?
No
Yes
Add to PIT
Drop packet
Decode the name from the interest packet
Fetch data from the database
Run service specific algorithm
Send data packet to the router
Store data in CS
Send data packet to the client
Display information on the app

FIGURE 10. Packet handling process of the system.

thus, the requests did not hit the cache. This result should provide an idea about how NDN performs compared to TCP/IP when no cache is hit. There was only one test case for TCP/IP.

- **Server Load:** because server was created with Java, JProfiler was used to measure and monitor the server resource utilization for the previously mentioned test runs.

Because the implementation of the proposed system was a test, it was done with a single device that generated one request at a time for each attempt. This test should be sufficient for providing an idea about the advantages NDN will provide for a large-scale implementation with real users. The large-scale implementation and testing will be considered in the future.

A. ANALYSIS OF FLOOR DETECTION SERVICE PERFORMANCE

For the NDN best-case scenario test, the test subject stood on the same floor, and the height of the device from the floor remained constant. Thus, the barometer sensor data did not change, and the requests were identical. Every request was expected to be identical and hit the cache after the first request. To collect the worst-case results, the test was conducted on different floors, and the device height was changed to generate different requests. This way, the requests were not identical and did not hit the NDN router's cache stored in the CS. In addition, the test subject requested floor detection service with TCP/IP to compare the results with those of NDN.

1) FLOOR DETECTION RESPONSE TIME

According to the results in Figure 11, when the user and the device remained in the same location (which was the best-case scenario), the response time was much shorter after the first request. This is because after the first request, the information was retrieved from the NDN router. There is one spike in the graph owing to a slight change in the air pressure during the test. All the remaining results were between approximately 50 and 75 ms. In the worst-case scenario in which each requests reached the server for a response, the results were similar to those of TCP/IP (mainly between 100 and 250 ms).

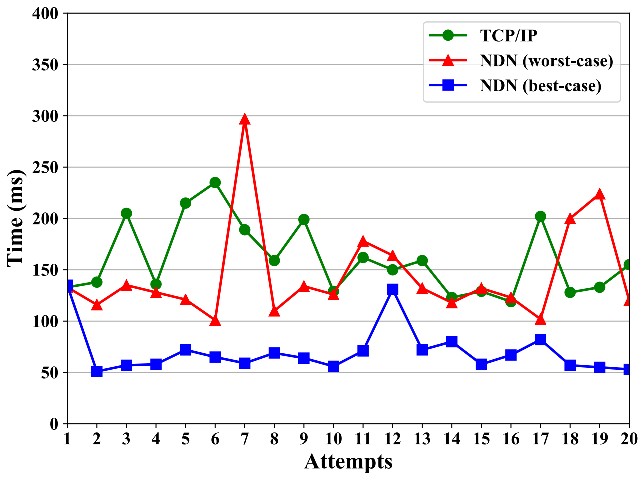


FIGURE 11. Floor detection response time with TCP/IP, NDN in the worst-case, and NDN in the best-case.

2) FLOOR DETECTION SERVER LOAD

During the test of the response time of the floor detection service, the server load was measured with JProfiler (Figure 12). With TCP/IP, every single request was handled by the server. The spikes indicate that when ever a request was made, the server responded by running the service-specific algorithm. (For the worst-case) scenario with NDN in which every request was different, showed similar results because every request was unique and handled by the server. However, for similar requests in the NDN best-case scenario, the server load was significantly reduced. This was because after the first request, most requests hit the router's CS cache and were

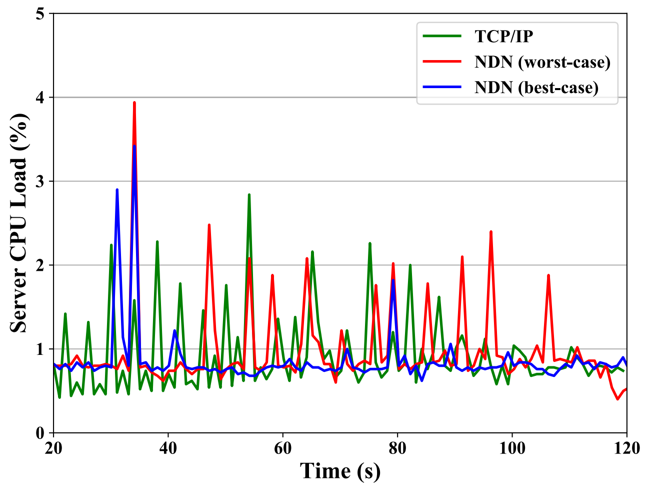


FIGURE 12. Floor detection server load with TCP/IP, NDN in the worst-case, and NDN in the best-case.

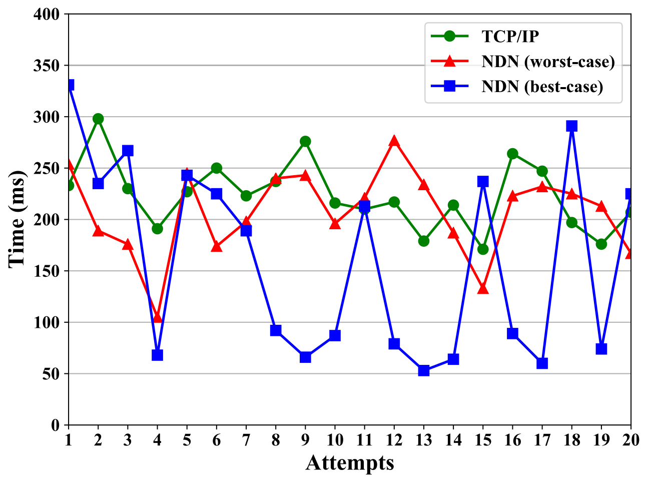


FIGURE 13. Positioning response time with TCP/IP, NDN in the worst-case, and NDN in the best-case.

processed by the router. This prevented the need to handle identical requests.

B. ANALYSIS OF POSITIONING SERVICE PERFORMANCE

When testing the NDN's positioning service performance in the worst-case scenario, the user moved on the floor. Thus, each request was different because of unique RSS values. In the best-case, testing the NDN caching performance was challenging because obtaining the same RSS values from the APs within the user's range.

1) POSITIONING RESPONSE TIME

According to Figure 13, in TCP/IP, all the positioning requests took similar times (approximately 200–300 ms). In the NDN best-case test, only 10 out of 20 requests hit the cache although the test subject remained in the same position. Those that did not hit the cache were processed by the server instead of the router. However, when the requests hit the cache, a response time of approximately 50–80 ms of response time was recorded. This should provide an idea about the NDN cache performance in the positioning service. In the worst-case scenario, TCP/IP and NDN both produced similar results (response time of approximately 200–300 ms).

2) POSITIONING SERVER LOAD

The server loads for the positioning service in different test cases were obtained by measuring the response time (Figure 14). The spikes in the graph indicate the requests, which were handled by the server. Such as for the floor detection service, when the requests were identical and handled by the router, the server load was significantly reduced compared to that of TCP/IP.

C. ANALYSIS OF NAVIGATION SERVICE PERFORMANCE

Two different measurements were taken for NDN: one for the best-case scenario in which the test user remained in the same location and requested the same destination for the navigation route. This way, all requests were expected to have identical names. Thus, the data could be retrieved from the router's CS. In the worst-case scenario, the user requested different destinations each time during the test. The requests were expected to be different and generate different names.

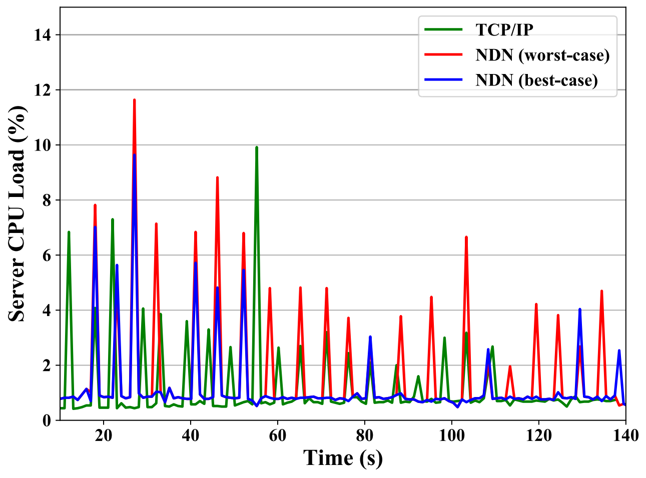


FIGURE 14. Positioning server load with TCP/IP, NDN in the worst-case, and NDN in the best-case.

The NDN worst-case test was conducted by sending different destinations from the same position of the device; thus, every request was different. In this test, coordinate (1,1) was the current position, which was in the farthest south-west corner. The chosen destination coordinates were (27,119), (27,118), (27,117), (27,116), (27,115), (27,114), (27,113), (27,112), (27,111), (27,110), (28,119), (28,118), (28,117), (28,116), (28,115), (28,114), (28,113), (28,112), (28,111), and (28,110); they were in the farthest north-east corner of the test site. For the worst-case and TCP/IP test, the staring point was (1,1), and the destination was (27,119).

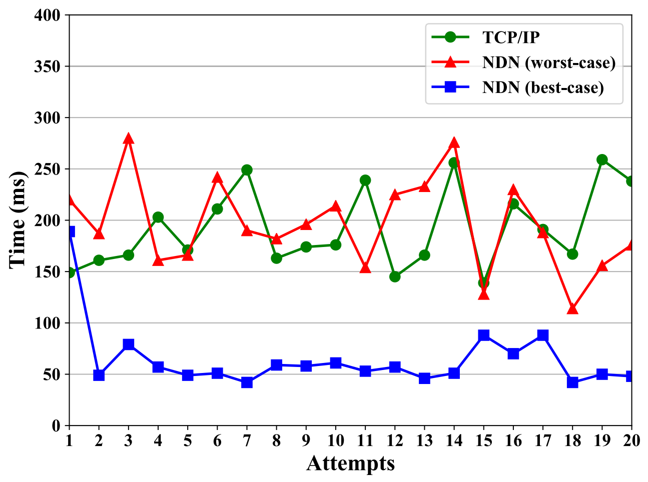


FIGURE 15. Navigation response time with TCP/IP, NDN in the worst-case, and NDN in the best-case.

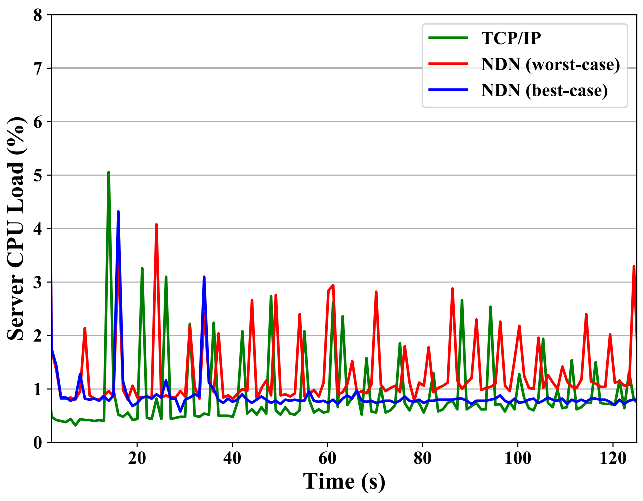


FIGURE 16. Navigation server load with TCP/IP, NDN in the worst-case, and NDN in the best-case.

1) NAVIGATION RESPONSE TIME

According to Figure 15, the TCP/IP response time was approximately 150–250 ms because all requests were handled by the server. The worst-case NDN results were similar to those of the TCP/IP. However, the navigation performance benefited from the NDN best-case scenario. After the first attempt, which took approximately 180 ms, all following attempts took approximately 45–80 ms because those responses were from the router's CS cache instead of the server.

2) NAVIGATION SERVER LOAD

When testing different cases with TCP/IP and NDN, the server load was recorded (Figure 16)). The results were as expected.In both TCP/IP and NDN worst-case scenario, there were greater numbers of spikes than in the best-case NDN scenario. Thus, every request was handled by the server. In the NDN best-case scenario, after the first request, the server load was drastically reduced because the router's cache was responsible for satisfying all the identical requests.

D. SERVER LOAD ANALYSIS FOR MULTIPLE REQUESTS

The previously presented test cases were conducted with a single device that sent one request at a time. To simulate the server load with multiple requests, the test device application was modified to send multiple requests at a time. The request number was increased from 1 to 100 at intervals of 20 requests. All of the requests were identical. Thus, NDN could use its caching feature. According to Figure 17, increasing the number of identical requests affects the server load in TCP/IP but does not affect in NDN.

E. OVERALL COMPARISON OF NDN WITH TCP/IP

According to the results, the NDN-based system performance is as good as that of TCP/IP when no cache is hit. However, NDN outperforms TCP/IP when hitting the cache. On average, the floor detection took 159.9 ms response time with

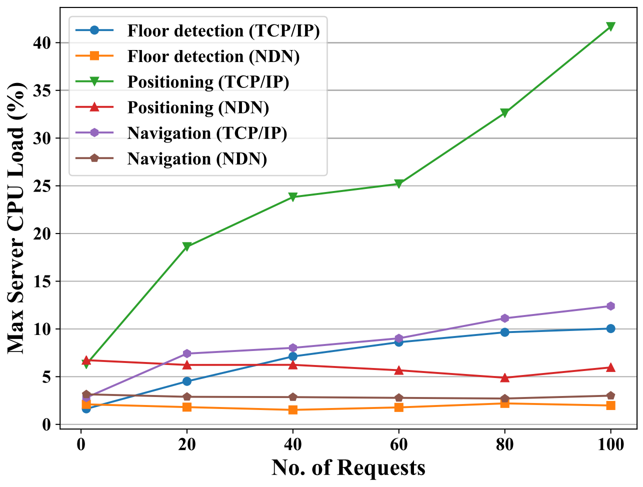


FIGURE 17. Server load with multiple requests of Floor Detection, Positioning and Navigation service.

TCP/IP, 144.7 ms with NDN in the worst-case scenario, and 70.6 ms with NDN in the best-case scenario. For positioning, an average response time of 223.2 ms was measured with TCP/IP, 159.4 ms with the NDN in the best-case scenario, and 206.6 ms with NDN in the worst-case scenario. For navigation, an average response time of 191.9 ms was measured with TCP/IP, 195.9 ms with NDN in the worst-case scenario, and 64.4 ms with NDN in the best-case scenario, as shown in Figure 18.

By taking the average time for each mode, it can be seen that the proposed system has a 77.5% better response time for floor detection, 33.4% better response time for positioning, and 99.5% better response time for navigation than TCP/IP in the best-case scenario. Moreover, NDN can drastically reduce the server load for all services for identical requests. The improved performance and server load may seem insignificant as there was only one test user. Nevertheless, this improvement will increase when more users start using the

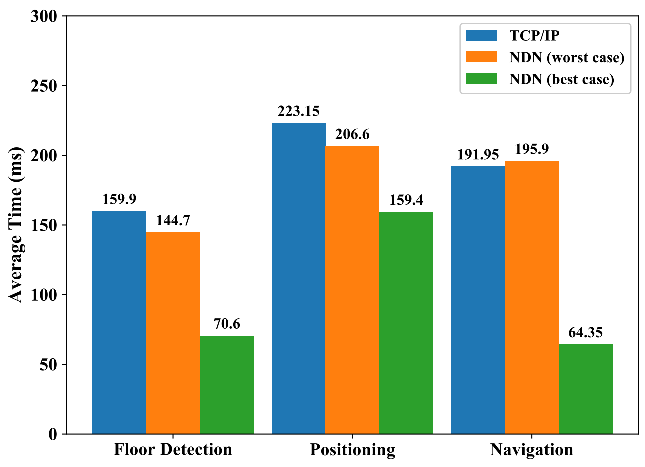


FIGURE 18. Average response time of NDN in compression to TCP/IP in different services.

services and more NDN routers are employed, which will help to offload more data from the server. This type of caching can be implemented in the server application with TCP/IP. However, it will not be as efficient as NDN because processing the cache in the server will cause extra server load. In NDN, the cache is handled by the routers.

VI. CONCLUSION

This paper presents the design, implementation and testing of a new NDN-based indoor positioning and navigation system with the existing localization and navigation algorithms. The overall performance and efficiency of the proposed NDN-based system is better than those of the traditional TCP/IP. The implementation of NDN in the network layer reduces the server load and response time. Because the traditional TCP/IP-based solutions are not very efficient, the proposed NDN-based approach can improve the existing solutions through its data caching and diverse packet forwarding features. When NDN is used as the communication protocol between the server and client, the routers can reduce the server load through data caching, which prevents the server from processing duplicate requests. In addition, the router's cache lookup performance is better than that of the algorithms running in the server, which reduces the response time. Thus, the proposed approach significantly improves the response time and server load of the existing indoor positioning and navigation technologies. The improvements of floor detection, localization, and navigation are approximately 77%, 33%, and 99%, respectively. However, in its current state, the improvements in indoor positioning and navigation can only be achieved in certain environments. Setting up this kind of environment is difficult, and the benefits can only be observed as long as the user is connected to an NDN network. NDN is at its early stage of development and requires further research. Although NDN does not require specialized hardware, it needs special application software and router firmware to work in an NDN network. Large-scale implementation and testing with more optimized positioning algorithms with better accuracy should be conducted in the future to assess the true potential of NDN-based indoor positioning and navigation systems. In addition, other indoor positioning technologies that require a server may be investigated with NDN communication protocol as additional future research direction.

ACKNOWLEDGMENT

This research was supported by Korea Electric Power Corporation (Grant number: R18XA02).

REFERENCES

[1] M. J. Piran, G. R. Murthy, G. P. Babu, and E. Ahvar, "Total GPS-free localization protocol for vehicular ad hoc and sensor networks (VASNET)," in *Proc. 3rd Int. Conf. Comput. Intell., Modeling Simulation*, Langkawi, Malaysia, Sep. 2011, pp. 388–393.

[2] C. Bauer, "On the (In-)Accuracy of GPS measures of smartphones: A study of running tracking applications," in *Proc. 11th Int. Conf. Adv. Mobile Comput. Multimedia (MoMM)*. Vienna, Austria: ACM, 2013, pp. 335–340.

[3] C. Feng, W. S. A. Au, S. Valaee, and Z. Tan, "Received-signal-strength-based indoor positioning using compressive sensing," *IEEE Trans. Mobile Comput.*, vol. 11, no. 12, pp. 1983–1993, Dec. 2012.

[4] R. K. Yadav, B. Bhattarai, H.-S. Gang, and J.-Y. Pyun, "Trusted K nearest Bayesian estimation for indoor positioning system," *IEEE Access*, vol. 7, pp. 51484–51498, 2019.

[5] R. Carotenuto, M. Merenda, D. Iero, and F. G. Della Corte, "An indoor ultrasonic system for autonomous 3-D positioning," *IEEE Trans. Instrum. Meas.*, vol. 68, no. 7, pp. 2507–2518, Jul. 2019.

[6] C.-Y. Yao and W.-C. Hsia, "An indoor positioning system based on the dual-channel passive RFID technology," *IEEE Sensors J.*, vol. 18, no. 11, pp. 4654–4663, Jun. 2018.

[7] M. Maheepala, A. Z. Kouzani, and M. A. Joordens, "Light-based indoor positioning systems: A review," *IEEE Sensors J.*, vol. 20, no. 8, pp. 3971–3995, Apr. 2020.

[8] Y. Xu, Y. S. Shmaliy, Y. Li, and X. Chen, "UWB-based indoor human localization with time-delayed data using EFIR filtering," *IEEE Access*, vol. 5, pp. 16676–16683, 2017.

[9] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos, "A survey of information-centric networking research," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 2, pp. 1024–1049, 2nd Quart., 2014.

[10] R. Ullah, M. A. U. Rehman, and B. S. Kim, "Hierarchical name-based mechanism for push-data broadcast control in information-centric multi-hop wireless networks," *Sensors*, vol. 19, no. 14, p. 3034, Jul. 2019.

[11] O. Akinwande, "Interest forwarding in named data networking using reinforcement learning," *Sensors*, vol. 18, no. 10, p. 3354, Oct. 2018.

[12] M. A. U. Rehman, R. Ullah, and B.-S. Kim, "NINQ: Name-integrated query framework for named-data networking of things," *Sensors*, vol. 19, no. 13, p. 2906, Jun. 2019.

[13] S. Muralidharan, A. Roy, and N. Saxena, "MDP-based model for interest scheduling in IoT-NDN environment," *IEEE Commun. Lett.*, vol. 22, no. 2, pp. 232–235, Feb. 2018.

[14] Z. Ren, M. A. Hail, and H. Hellbruck, "CCN-WSN—A lightweight, flexible content-centric networking protocol for wireless sensor networks," in *Proc. IEEE 8th Int. Conf. Intell. Sensors, Sensor Netw. Inf. Process.*, Melbourne, VIC, Australia, Apr. 2013, pp. 123–128.

[15] D. Saxena and V. Raychoudhury, "Design and verification of an NDN-based safety-critical application: A case study with smart healthcare," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 5, pp. 991–1005, May 2019.

[16] M. U. Ali, S. Hur, S. Park, and Y. Park, "Harvesting indoor positioning accuracy by exploring multiple features from received signal strength vector," *IEEE Access*, vol. 7, pp. 52110–52121, 2019.

[17] C.-H. Cheng, T.-P. Wang, and Y.-F. Huang, "Indoor positioning system using artificial neural network with swarm intelligence," *IEEE Access*, vol. 8, pp. 84248–84257, 2020.

[18] N. Bai, Y. Tian, Y. Liu, Z. Yuan, Z. Xiao, and J. Zhou, "A high-precision and low-cost IMU-based indoor pedestrian positioning technique," *IEEE Sensors J.*, vol. 20, no. 12, pp. 6716–6726, Jun. 2020.

[19] S. Yiu and K. Yang, "Gaussian process assisted fingerprinting localization," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 683–690, Oct. 2016.

[20] A. Achroufene, Y. Amirat, and A. Chibani, "RSS-based indoor localization using belief function theory," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 3, pp. 1163–1180, Jul. 2019.

[21] M. Lipka, E. Sippel, and M. Vossiek, "An extended Kalman filter for direct, real-time, phase-based high precision indoor localization," *IEEE Access*, vol. 7, pp. 25288–25297, 2019.

[22] M. Mizmizi and L. Reggiani, "Binary fingerprinting-based indoor positioning systems," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Sapporo, Japan, Sep. 2017, pp. 1–6.

[23] Y. Zhuang, Y. Li, L. Qi, H. Lan, J. Yang, and N. El-Sheimy, "A two-filter integration of MEMS sensors and WiFi fingerprinting for indoor positioning," *IEEE Sensors J.*, vol. 16, no. 13, pp. 5125–5126, Jul. 2016.

[24] C. Zhou, J. Yuan, H. Liu, and J. Qiu, "Bluetooth indoor positioning based on RSSI and Kalman filter," *Wireless Pers. Commun.*, vol. 96, no. 3, pp. 4115–4130, Jul. 2017.

[25] A. S. Paul and E. A. Wan, "RSSI-based indoor localization and tracking using sigma-point Kalman smoothers," *IEEE J. Sel. Topics Signal Process.*, vol. 3, no. 5, pp. 860–873, Oct. 2009.

[26] V. Renaudin, M. Ortiz, J. Perul, J. Torres-Sospedra, A. R. Jiménez, A. Pérez-Navarro, G. M. Mendoza-Silva, F. Seco, Y. Landau, R. Marbel, and B. Ben-Moshe, "Evaluating indoor positioning systems in a shopping mall: The lessons learned from the IPIN 2018 competition," *IEEE Access*, vol. 7, pp. 148594–148628, 2019.

[27] G. Dong, K. Lin, K. Li, H. Luo, and X. Zhang, "FMA-RRSS: Fingerprint matching algorithm based on relative received signal strength in indoor Wi-Fi positioning," in *Proc. IEEE 17th Int. Conf. Comput. Sci. Eng.*, Chengdu, China, Dec. 2014, pp. 1071–1077.

[28] L. Zhang, Y. Li, Y. Gu, and W. Yang, "An efficient machine learning approach for indoor localization," *China Commun.*, vol. 14, no. 11, pp. 141–150, Nov. 2017.

[29] J. Luo, Z. Zhang, C. Wang, C. Liu, and D. Xiao, "Indoor multifloor localization method based on WiFi fingerprints and LDA," *IEEE Trans. Ind. Informat.*, vol. 15, no. 9, pp. 5225–5234, Sep. 2019.

[30] X. Song, X. Fan, C. Xiang, Q. Ye, L. Liu, Z. Wang, X. He, N. Yang, and G. Fang, "A novel convolutional neural network based indoor localization framework with WiFi fingerprinting," *IEEE Access*, vol. 7, pp. 110698–110709, 2019.

[31] S. Subedi and J.-Y. Pyun, "Lightweight workload fingerprinting localization using affinity propagation clustering and Gaussian process regression," *Sensors*, vol. 18, no. 12, p. 4267, Dec. 2018.

[32] Y. Sun, L. Guan, Z. Chang, C. Li, and Y. Gao, "Design of a low-cost indoor navigation system for food delivery robot based on multi-sensor information fusion," *Sensors*, vol. 19, no. 22, p. 4980, Nov. 2019.

[33] V.-C. Ta, T.-K. Dao, D. Vaufreydaz, and E. Castelli, "Collaborative smartphone-based user positioning in a multiple-user context using wireless technologies," *Sensors*, vol. 20, no. 2, p. 405, Jan. 2020.

[34] C. Wang, L. Xing, and X. Tu, "A novel position and orientation sensor for indoor navigation based on linear CCDs," *Sensors*, vol. 20, no. 3, p. 748, Jan. 2020.

[35] A. Satan, "Bluetooth-based indoor navigation mobile system," in *Proc. 19th Int. Carpathian Control Conf. (ICCC)*, Szilvasvarad, Hungary, May 2018, pp. 332–337.

[36] J. C. Dela Cruz, G. V. Magwili, J. P. E. Mundo, G. P. B. Gregorio, M. L. L. Lamoca, and J. A. Villasenor, "Items-mapping and route optimization in a grocery store using Dijkstra's, Bellman-Ford and Floyd-Warshall algorithms," in *Proc. IEEE Region Conf. (TENCON)*, Singapore, Nov. 2016, pp. 243–246.

[37] Z. Boroujeni, D. Goehring, F. Ulbrich, D. Neumann, and R. Rojas, "Flexible unit A-star trajectory planning for autonomous vehicles on structured road maps," in *Proc. IEEE Int. Conf. Veh. Electron. Saf. (ICVES)*, Vienna, Austria, Jun. 2017, pp. 7–12.

[38] T. A. Dionti, K. M. Adhinugraha, and S. M. Alamri, "Inter-building routing approach for indoor environment," in *Proc. Int. Conf. Comput. Sci. Appl.* Trieste, Italy: Springer, Jul. 2017, pp. 247–260.

[39] H. Wu, A. Marshall, and W. Yu, "Path planning and following algorithms in an indoor navigation model for visually impaired," in *Proc. 2nd Int. Conf. Internet Monitor. Protection (ICIMP)*, San Jose, CA, USA, Jul. 2007, p. 38.

[40] F. Massé, A. K. Bourke, J. Chardonnens, A. Paraschiv-Ionescu, and K. Aminian, "Suitability of commercial barometric pressure sensors to distinguish sitting and standing activities for wearable monitoring," *Med. Eng. Phys.*, vol. 36, no. 6, pp. 739–744, Jun. 2014.

[41] J. Luo, C. Zhang, and C. Wang, "Indoor multi-floor 3D target tracking based on the multi-sensor fusion," *IEEE Access*, vol. 8, pp. 36836–36846, 2020.

[42] F. Haque, V. Dehghanian, A. O. Fapojuwo, and J. Nielsen, "A sensor fusion-based framework for floor localization," *IEEE Sensors J.*, vol. 19, no. 2, pp. 623–631, Jan. 2019.

[43] Y. Li, Z. Gao, Z. He, P. Zhang, R. Chen, and N. El-Sheimy, "Multi-sensor multi-floor 3D localization with robust floor detection," *IEEE Access*, vol. 6, pp. 76689–76699, 2018.

[44] T. Willemsen, F. Keller, and H. Sternberg, "Concept for building a MEMS based indoor localization system," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Busan, South Korea, Oct. 2014, pp. 1–10.

[45] S. S. Kim, J. W. Kim, and D. S. Han, "Floor detection using a barometer sensor in a smartphone," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Sapporo, Japan, Oct. 2017, pp. 1–9.

[46] A. Afanasyev, J. Shi, B. Zhang, L. Zhang, I. Moiseenko, Y. Yu, W. Shang, Y. Huang, J. P. Abraham, S. DiBenedetto, and C. Fan, "NFD developer's guide," Dept. Comput. Sci., Univ. California Los Angeles, Los Angeles, CA, USA, Tech. Rep. NDN-0021, 2014.

SIFAT UT TAKI received the B.Sc. degree in computer science and engineering from the School of Engineering and Computer Science, BRAC University, Dhaka, Bangladesh. His research interests include the IoT, networking, named data networking, indoor positioning systems, and embedded systems.

AMITABHA CHAKRABARTY received the M.Sc. degree from the Department of Computer Science and Engineering, University of Rajshahi, in 2004, the M.Sc. degree from Independent University, Bangladesh, majoring in telecommunication engineering, and the Ph.D. degree from the Faculty of Engineering and Computing, Dublin City University, Dublin, Ireland, in 2012. He is currently working as an Associate Professor with the Department of Computer Science and Engineering, Brac University, Dhaka, Bangladesh. He has published research papers in various national and international conferences, journals, as well as book chapters. He leads the IoT and Embedded System Research Group, BRAC University. He is involved in active research having a number of graduate and undergraduate research groups in different research projects. He is serving as a TCP member in various international journals and conferences. He is also serving as a senior judge in various national IT competition. His research interests include the Internet of Things (IoT), machine learning, deep learning, embedded systems, and switching theory.

MD. JALIL PIRAN (Member, IEEE) received the Ph.D. degree in electronics engineering from Kyung Hee University, South Korea, in 2016. He was a Postdoctoral Research Fellow in resource management and quality of experience in 5G cellular networks and the Internet of Things (IoT) with the Networking Laboratory, Kyung Hee University. He is currently an Assistant Professor with the Department of Computer Science and Engineering, Sejong University, Seoul, South Korea. He has published a substantial number of technical articles in well-known international journals and conferences in research fields of wireless communications, 5G/6G, the Internet of Things (IoT) Multimedia communication, streaming, adaptation, and QoE, applied machine learning, security, and smart grid. In the worldwide communities, he has been an Active Delegate from South Korea in the Moving Picture Experts Group, since 2013, and an Active Member of the International Association of Advanced Materials, since 2017. He received the IAAM Scientist Medal of the year 2017 for notable and outstanding research in new age technology and innovation, Stockholm, Sweden. He has been recognized as the Outstanding Emerging Researcher by the Iranian Ministry of Science, Technology, and Research, in 2017. His Ph.D. dissertation has been selected as the Dissertation of the Year 2016 by the Iranian Academic Center for Education, Culture, and Research in the Engineering Group.

QUOC-VIET PHAM (Member, IEEE) received the B.S. degree in electronics and telecommunications engineering from the Hanoi University of Science and Technology, Vietnam, in 2013, and the Ph.D. degree in telecommunications engineering from Inje University, South Korea, in 2017. He is currently a Research Professor with the Research Institute of Computer, Information and Communication, Pusan National University, South Korea. From September 2017 to December 2019, he was with Kyung Hee University, Changwon National University, and Inje University, in various academic positions. He received the Best Ph.D. Dissertation Award in Engineering from Inje University, in 2017. His research interests include convex optimization, game theory, and machine learning to analyze and optimize edge/cloud computing, and 5G and beyond networks.

DOUG YOUNG SUH (Member, IEEE) received the B.S. degree in nuclear engineering from Seoul National University, Seoul, South Korea, in 1980, and the Ph.D. degree in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 1990. In 1990, he joined the Korea Academy of Industry and Technology, and conducted research on HDTV until 1992. Since 1992, he has been a Professor with the College of Electronics and Information, Kyung Hee University, Seoul, South Korea. He has been a Korean Delegate for the ISO/IEC MPEG Forum since 1996. His research interests include networked video and computer games.