Instinctive Assistive Indoor Navigation using Distributed Intelligence

MD MUZTOBA, ROHIT VOLETI, and FATIH KARABACAK, Arizona State University
JAEHYUN PARK, University of Ulsan
UMIT Y. OGRAS, Arizona State University

Cyber-physical systems (CPS) and the Internet of Things (IoT) offer a significant potential to improve the effectiveness of assistive technologies for those with physical disabilities. Practical assistive technologies should minimize the number of inputs from users to reduce their cognitive and physical effort. This article presents an energy-efficient framework and algorithm for assistive indoor navigation with multi-modal user input. The goal of the proposed framework is to simplify the navigation tasks and make them more instinctive for the user. Our framework automates indoor navigation using only a few user commands captured through a wearable device. The proposed methodology is evaluated using both a virtual smart building and a prototype. The evaluations for three different floorplans show one order of magnitude reduction in user effort and communication energy required for navigation, when compared to conventional navigation methodologies that require continuous user inputs.

CCS Concepts: • **Computer systems organization → Embedded and cyber-physical systems**; **Embedded systems**; **Embedded hardware**;

Additional Key Words and Phrases: IoT devices, assistive technologies, wearable computers, human-machine interface

ACM Reference format:
Md Muztoba, Rohit Voleti, Fatih Karabacak, Jaehyun Park, and Umit Y. Ogras. 2018. Instinctive Assistive Indoor Navigation using Distributed Intelligence. *ACM Trans. Des. Autom. Electron. Syst.* 23, 6, Article 80 (November 2018), 21 pages.
https://doi.org/10.1145/3212720

1 INTRODUCTION

The continuing expansion of the Internet of Things (IoT) will soon enable disruptive applications by interconnecting billions of smart devices with each other and their users [27]. One high-impact application area is in assistive technologies for the physically challenged population. The Annual World Report on Disability reveals that 15% of the world's population lives with a disability, and 110 to 190 million of these people have significant difficulties in functioning [1]. We can significantly

This work was supported partially by National Science Foundation (NSF) Grants No. CNS-1651624 and No. CNS-1526562, and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (Grant No. NRF-2018R1C1B50447150).
Authors' addresses: M. Muztoba, R. Voleti, F. Karabacak, and U. Y. Ogras, School of Electrical, Computer, and Energy Engineering, Arizona State University, Tempe, AZ 85281; emails: {mmuztoba, rnvoleti, fatihkarabacak, umit}@asu.edu; J. Park, School of Electrical Engineering, University of Ulsan, Ulsan, South Korea; email: jaehyun@ulsan.ac.kr.

1084-4309/2018/11-ART80 $15.00
https://doi.org/10.1145/3212720

improve the quality of life of this population by providing accessibility to smart cyber-physical systems (CPS) that process sensory inputs and assist with everyday tasks [13]. In this work, we focus specifically on an instinctive assistive indoor navigation system that minimizes the number of user inputs to reduce both physical and cognitive effort.

Traditional electronic systems rely heavily on user intervention for *all decision-making*. For example, electric wheelchairs require a continuous stream of direction commands, similar to driving a car. This puts all the decision making and navigation burden on the person with limited ability. The other extreme decision making is a fully autonomous navigation system, which has a list of all possible destinations. Setting the final destination and leaving the navigation to the wheelchair removes the burden of continuous control from the user. However, a fully autonomous system would be possible only if the user can specify all the possible destinations through a pre-defined interface. This alternative to user intervention lacks flexibility and has three drawbacks. First, use of a pre-defined framework itself requires a non-trivial interface, such as a graphical user interface, a touch screen, or speech recognition. A person with a disability may face difficulty in selecting the destination by interaction with this type of an interface. Second, all the potential destinations need to be known a priori and encoded in a menu. Third, the final destination is not always predetermined. In short, a fully autonomous solution pushes the complexity to receiving a "precise input." In contrast, this work aims to assist the user by receiving simple commands. To this end, we propose to distribute the intelligence between the person and the assistive device. In analogy to the nervous system, the person acts as the "brain" by indicating the intended action to the device through a seamless mechanism. The proposed navigation framework acts as the "spinal cord," which interprets the user's intent and manages the actions of the assistive device. For instance, the person may lean or turn towards a certain direction to indicate the intended movement in the indoor navigation example. Then, the proposed framework automates the immediate turn and also predicts the possible target(s) while moving the person towards them. Essentially, the combination of the proposed framework and the wheelchair serves as a natural extension of the body.

The goals of the proposed assistive cyber-physical system are to achieve natural human-machine communication and minimize the number of user inputs. To achieve these goals, we address two major challenges. The first challenge is communicating the user's intent with the device in a natural and simple way. The second is interpreting and utilizing this input to drive the physical system. To address the first challenge, we limit the user's inputs to a small set of simple commands and send them to the wearable device using multi-modal communication [34]. At the physical device side, we implement an algorithm to predict the intermediate destinations using the simple commands and floorplan information.

Human to machine communication: Machine to human communication can occur through a variety of mechanisms, ranging from display to audio. However, human to machine communication is more limited (and potentially more so for those with physical disabilities) when traditional mechanisms such as keyboards, touch screens, and mice, are not viable. This is often the case when dealing with a cyber-physical system. Speech recognition is a good candidate for a natural interaction with smart devices. However, we choose not to rely on any single modality for human-machine communication, as any given form of communication may have limitations in terms of its effectiveness (e.g. speech recognition in noisy environments). Moreover, users with certain disabilities may prefer some forms of interaction as opposed to others. For example, those with motor control disorders may have difficulty with hand gestures. As such, we propose a multi-modal communication architecture that allows more flexible interaction with smart devices. Multi-modal communication is defined as the use of multiple complementary mechanisms, such as speech, gestures, and facial expressions [34] as forms of communication with smart devices. In our experiments, we employ hand gesture recognition, speech recognition, and a brain-machine

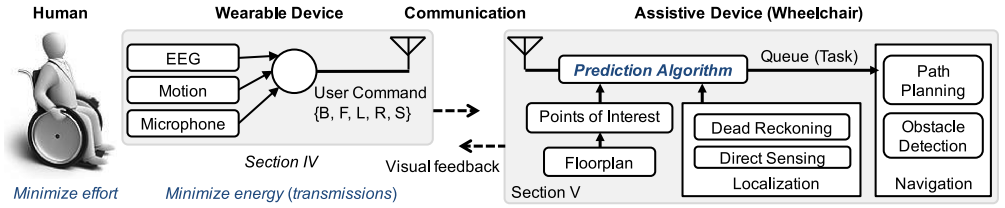

Fig. 1. The major components of the proposed system. Multi-modal communication architecture and assistive indoor navigation algorithms are described in Sections 4 and 5, respectively. *The wearable device is energy limited as it is physically separated from the wheelchair.*

interface (BMI) that records and analyzes electroencephalogram (EEG) signals. The proposed system allows users to potentially pick different mechanisms or a combination based on their ability to generate these inputs. As shown in Figure 1, the proposed system infers a minimal set of user inputs {Back (*B*), Forward (*F*), Left (*L*), Right (*R*), Stop (*S*)} using a combination of three modalities.

Processing the user input: To compensate for the reduction in user effort, the receiving device (i.e., the wearable device) must have the intelligence to interpret the user's intent. To be practical, the algorithms should run on the wearable device in *real time* with *minimum energy overhead*.[1] To address this need, we present an algorithm that can predict the possible set of destinations using only a few commands and floorplan information. To start, we encode each possible destination on the floorplan using a sequence of commands. Each new command is appended to a sequence of received commands to form a prefix. Then, this prefix is used to predict possible destinations, and automate navigation until the next decision point. We show that the proposed technique not only reduces the time to target, but more importantly, eliminates the need for a continuous user inputs, implying a reduction in communication energy consumption. As a result, we achieve 4.5× reduction in the number of user inputs and communication energy.

The major contributions of this article are:

- An algorithm to predict the user intent and automate navigation with minimum number of user inputs
- A system prototype that generates simple commands, and implements the proposed algorithm
- A co-simulation framework for modeling cyber-physical systems and extensive evaluations on a virtual smart building using three real floorplans.

The rest of this article is organized as follows. Related work appears in Section 2. We describe the proposed assistive system and the techniques for generating user commands in Sections 3 and 4, respectively. The proposed indoor navigation algorithm is presented in Section 5, while energy and performance evaluations appear in Section 6. Finally, conclusions and future directions are summarized in Section 7.

2 RELATED WORK

IoT-enabled devices have a huge potential to assist people with disabilities [13, 14, 39], in addition to improving the quality of life in general. This has led to a large variety of application-oriented wearable devices that allow for natural forms of human-machine communication [25]. The user inputs that control these devices are typically generated by classifying a single mode of

[1]Note that wearable device, which interprets the user intent and communicates the commands, is energy-limited unlike the physical device (wheelchair).

communication, such as speech, hand gestures, facial expressions, brain-activity, and body, head, or eye movements [4, 21, 37].

Devices designed to communicate through a single input modality restrict the usage only to persons capable of producing that particular input [44]. For example, a person with an articulation disorder may face difficulty in using speech-recognition devices. Furthermore, devices that rely on a single modality can suffer from poor accuracy [34]. More than 30% of mobility aids are completely abandoned because of poor device performance or changes in user priorities, such as improvement or decline in their medical conditions [33]. Implementing multi-modal communication can alleviate these issues. By utilizing a variety of inputs, we can improve the flexibility of the device to support a wide group of users with differing abilities or changing priorities. For example, a user who suffers from a speech impediment after an accident may prefer gesture-based control of devices. Additionally, multi-modal communication has the potential to improve the recognition accuracy and robustness [15]. However, this also leads to a new challenge, because the final control decision has to be determined by fusing multiple inputs. A variety of decision-fusion algorithms have been developed to address this need [15, 24, 44]. We employ the majority voting algorithm to determine the final decision from our input devices [46].

Many assistive technologies rely heavily on tens of low-level commands directly from the user to actuators to complete a task. This requires high physical, cognitive, or linguistic effort [30]. It also increases the task completion time, since frequent responses from the user are required for task completion. In our work, we specifically focus on the technology surrounding assistive indoor navigation. Many researchers have explored ways in which automation can reduce the burden on the user for navigation tasks [5, 8, 43]. For example, vision-based robot navigation is presented in [6], where the trajectory is represented as a set of images. This type of an approach requires image processing, and it is limited to the path on the input images. Likewise, methodologies presented in References [5] and [43] require computationally intensive vision processing that involves extensive information processing. The work in Reference [17] addresses controlling wheeled and humanoid robots using a BMI. The authors present algorithms for learning command hierarchies using the history of the commands generated by the user. The researchers in Reference [40] performed a study on assistive indoor navigation with a semi-autonomous wheelchair with multi-modal inputs. Their work focuses mostly on the control framework around the wheelchair system, including a simulation environment to test the control system. In contrast, our study explores the navigation algorithm for an IoT-enabled wheelchair on an indoor floorplan that contains a grid of RFID sensors. The low-level control signals that drive the wheelchair are generated using a wearable device that interprets user inputs. We combine simple user inputs with the contextual information about the wheelchair environment (i.e., the floorplan, current state and location of the wheelchair). We also target reducing the user effort, communication energy cost, and computational requirements compared to more conventional methods for assistive indoor navigation.

3 CYBER-PHYSICAL SYSTEM OVERVIEW

The cyber-physical system we have designed is depicted in Figure 1. The first component of this system is the wearable device, which we model using a Raspberry Pi board [36]. This device is responsible for processing the user inputs from multiple sources by using the approach described in Section 4. It generates a final decision for wheelchair actuation and communicates with the wheelchair controller. We use the following three modalities to capture the user commands as inputs to the wearable device:

- **Gestures:** Hand, head, or body movements can be used to capture simple commands we employ. We utilize accelerometer and gyroscope sensors [42] to capture simple tilting and twisting movements on one hand that correspond to navigation commands. While

traditional controls (e.g., a joystick) require a large number of precise movements, a simple hand gesture can be less precise to accommodate users with motion-related disabilities. Gesture recognition can also be tailored for each user to improve the accuracy.

- **Speech:** While natural speech recognition is demanding, a handful of simple voice commands can be recognized with less processing power. We use the Pocketsphinx speech recognition system [23] to detect the user's voice commands for navigation.
- **Brain-machine Interface (BMI):** People who have difficulty in producing voice and gesture commands can employ wearable EEG-based BMIs to generate the inputs [29]. In this work, we use the Emotiv EPOC headset [16] to capture raw EEG inputs and facial expressions that represent navigation commands. BMI serves as a natural addition to our system, since users typically think about the gestures and speech commands they produce.

The next component of our system is the IoT-enabled wheelchair device. We use the Intel Edison [7] board to actuate all navigation-related tasks for the wheelchair. Traditional electric wheelchairs (without any navigation algorithm) require a continuous stream of manual inputs for directions to move. Every turn, stop, and movement command needs to be specified by the user. This requires the user to generate and transmit many commands, which may use up limited battery energy of the wearable device [3, 31]. Therefore, in our proposed solution, the Edison board runs a prediction algorithm that uses these inputs and the floorplan information to automate segments of the navigation. We show that this significantly reduces the number of user inputs and overall communication energy of the wearable device. The details of this implementation are discussed in Section 5.

4 MULTI-MODAL COMMUNICATION TECHNIQUE

4.1 Overview and Preliminaries

To enable intuitive human-machine communication, we employ a multi-modal communication approach. Each modality uses a classifier to interpret the user intent using its inputs. We make the following definitions to formulate the multi-modal communication problem:

- Let $H = \{H^i, 1 \leq i \leq L\}$ represent the set of classification modalities or devices, where L is the total number of classifier devices. In our experiments, we have $L = 3$ with the following:
  $-H^1$: BMI system
  $-H^2$: Speech recognition system
  $-H^3$: Gesture detection system
- Let $\Gamma = \{\gamma_j, 1 \leq j \leq M\}$ represent the set of possible user-generated commands, where M is the total number of commands. In our experiments, we have $M = 5$ with the following possible commands:
  $-\Gamma = \{B, F, L, R, S\}$, representing the commands of *Backwards*, *Forward*, *Left*, *Right*, and *Stop*, respectively.

Then, we denote the user's intended command and the classifier decision as follows:

- The *user's intended command* at time t is represented by $C_t \in \Gamma$
- The *classifier's decision* at time t from classification modality H^i is represented by $D_t^i \in \Gamma$

We state that classifier H^i successfully detects the user intent at time t, if $C_t = D_t^i$. Otherwise, the lack of any classifier output or $C_t \neq D_t^t$ imply an incorrect classification.

4.2 Fusion of Decisions

Our L-mode classification system generates L decisions (one for each modality) using the set of classifiers H^i. Therefore, we need a fusion algorithm to determine the final decision by using each individual classifier decision as an input. We propose the following two-step procedure:

(1) Use *optimal Bayesian classification* to determine the correct interpretation probability for each modality H^i. That is, compute $P(C_t = \gamma | D_t^i = \gamma)$ for each H^i and each possible command $\gamma \in \Gamma = \{B, F, L, R, S\}$. This techniques is detailed in Section 4.2.1.
(2) Employ a lightweight *majority voting* scheme to select the final output of the multi-modal system, denoted by D_t^f. This technique is explained in Section 4.2.2.

The following subsections detail these two steps.

4.2.1 Optimal Bayesian Classification. For a given classification modality H^i, the *conditional probability*$P(C_t = \gamma | D_t^i = \gamma)$ serves as a useful measure of the corresponding classifier's *credibility*. Since event distribution is unknown in practice, this conditional probability can be estimated by using an optimal Bayesian classifier, as follows:

$$P(C_t = \gamma | D_t^i = \gamma) = \frac{P(D_t^i = \gamma | C_t = \gamma) P(C_t = \gamma)}{P(D_t^i = \gamma | C_t = \gamma) P(C_t = \gamma) + P(D_t^i = \gamma | C_t \neq \gamma) P(C_t \neq \gamma)}, \tag{1}$$

where the denominator, $P(D_t^i = \gamma)$, is expanded using the total probability theorem.

To compute the credibility of each modality using Equation (1), we need the command generation ($P(C_t = \gamma)$, $P(C_t \neq \gamma)$) and conditional probability terms. The command generation probabilities can be estimated by using contextual information and history of commands, which are provided by our indoor navigation algorithm. Therefore, we describe their computation in Section 5.4 after presenting the necessary background. Next, we examine the conditional probability terms:

- $P(D_t^i = \gamma | C_t = \gamma)$ represents the *conditional probability* that modality H^i infers γ given the user command is γ.
- $P(D_t^i = \gamma | C_t \neq \gamma)$ represents the *conditional probability* that classification modality H^i infers γ, when the user actually sent a different command (false detection of command γ).

We conduct an offline analysis using user subject experiments to determine the accuracy of each classification modality. Once offline training is completed, the user begins to use the wheelchair and wearable device for indoor navigation task. We note that the decision accuracy of some classifiers may vary over time and use conditions. For example, the accuracy of speech recognition may decrease in noisy environments. Therefore, we also provide an online mechanism to continually adjust these conditional probabilities over time while the device is in regular use.

Offline Training to Determine Conditional Probabilities: During the training phase, we ask the user to generate each command a total of T times ($T = 50$ in our experiments). We record the performance of each classifier H^i in a matrix, denoted by $\boldsymbol{O}^i$, as follows:

$$\boldsymbol{O^i} = [o_{t,j}^i] = \begin{array}{c} \begin{array}{ccccc} \textit{Intent} \rightarrow & B & F & L & R & S \\ \textit{Time} \downarrow & & & & & \end{array} \\ \begin{array}{c} t_1 \\ t_2 \\ \vdots \\ t_T \end{array} \begin{pmatrix} o_{1,B}^i & o_{1,F}^i & o_{1,L}^i & o_{1,R}^i & o_{1,S}^i \\ o_{2,B}^i & o_{2,F}^i & o_{2,L}^i & o_{2,R}^i & o_{2,S}^i \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ o_{T,B}^i & o_{T,F}^i & o_{T,L}^i & o_{T,R}^i & o_{T,S}^i \end{pmatrix} \end{array} \tag{2}$$

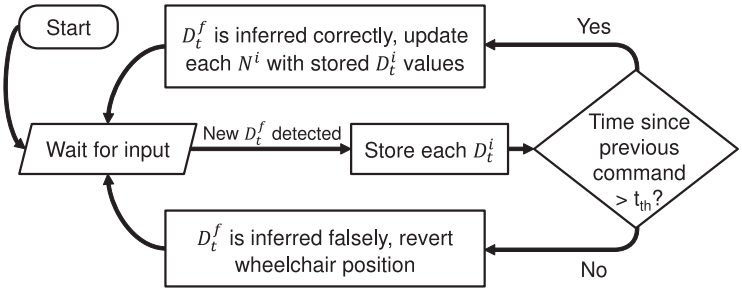

Fig. 2. Online update of $\boldsymbol{O}^i$.

The columns of $\boldsymbol{O}^i$ from $j = 1, \ldots, M$ represent the outcomes the ith modality $D_t^i \in \Gamma$ used in the experiment. Each rows of $\boldsymbol{O}^i$ from $t = 1, \ldots, T$ represent a different trials for each intended command. For example, $o^i_{1,F}$ represents the outcome of the ith modality in the first trial, while testing for the forward command.

Once this matrix is generated, computing the conditional probabilities $P(D_t^i = \gamma | C_t = \gamma)$ and $P(D_t^i = \gamma | C_t \neq \gamma)$ is a matter of counting the frequency of each outcome. For example, $P(D_t^i = F | C_t = F)$ is computed using the second column of $\boldsymbol{O}^i$. The number of rows for which $o^i_{j,F} = F$ gives the total number of trials that the ith modality generates the outcomes correctly as F. Hence, $P(D_t^i = F | C_t = F)$ is simply the number of of correct outcomes divided by the total number of trials T.

Online Training to Update O^i: Since various environmental factors can affect any particular classifier's accuracy over time, we also propose an online technique to update the elements of matrix $\boldsymbol{O}^i$ at runtime.

O^i is a $T \times M$ (i.e., 50×5) vector, as defined in Equation (2). Each column stores the last 50 decisions made by modality "i" for a given intent. For instance, the first column of O^i stores the decisions made by modality "i," when the intent was **B**ackwards (Column 1 in Equation (2)). At runtime, we employ a sliding window of length T = 50 (i.e., the last 50 decisions for each intent are stored). When a new decision is made, the oldest entry is dropped and replaced with the new one. In this way, each column of O^i reflects the latest 50 decisions for the corresponding intent. Then, these up-to-date columns are used to compute the conditional probability of generating each action, given an intent.

The main challenge in implementing the sliding window is the uncertainty in the true user intent. That is, there is no ground truth reference for intended user commands. Therefore, we must decide upon a method to determine if any particular final classifier decision, D_t^f, is the true intended user command. We observe that a user will immediately try to correct an incorrect classification decision. Similarly, we observe that the same user will not issue another immediate command if he is satisfied with the fused decision from all classification modalities, likely indicating a correct final decision. Therefore, we assume that if a new command does not arrive within a predefined time threshold, t_{th}, the final decision D_t^f matches the user's intended command C_t. In this case, we store these newest D_t^i value for each H^i in the column of each $\boldsymbol{O}^i$ that corresponds to this true intended command, while discarding the oldest values, as described in Figure 2. If, instead, the user does generate a new command within the same pre-defined time threshold, then we assume that the user was not satisfied with the last generated final decision D_t^f. This implies that the command was incorrectly classified by our multi-modal system. Unlike the other case, we cannot simply update the $\boldsymbol{O}^i$ matrices, since the correct intended decision is still unknown. Therefore,

when an incorrect event is interpreted, the system will attempt to re-interpret the user's intent until a correct decision can be determined. When a user is finally satisfied with the final fused decision, D_t^f, we conclude that all the previous incorrect attempts were made with the same intended command. Then, the corresponding columns in each $\boldsymbol{O}^i$ are updated with the previous D_t^i values generated by the multi-modal system, as shown in Figure 2.

We note that reinforcement learning (RL) algorithm [41] is a powerful alternative technique for making online updates. We opted for a simpler online update mechanism, because we target low-power systems with very limited computational power and memory. In contrast, RL is data intensive and requires more computations than our simple updates, which have linear complexities.

4.2.2 Majority Voting for Final Decision. From the above discussion, we know that the conditional probability $P(C_t = \gamma | D_t^i = \gamma)$ gives a measure for the *credibility* of a decision from classifier H^i. We utilize this credibility as a *weight*, denoted by $w_{j,t}^i$, for each decision in the multi-modal classification system and form the weight matrix $\boldsymbol{W}_t = [w_{j,t}^i]$. Here, j represents the user commands $C_t \in \Gamma$ used in the experiment. Hence, $\boldsymbol{W}_t$ is an $L \times M$ matrix that contains the weight for all supported commands related to each classifier, at time t. In our system, each classifier can output a decision from the same set of five commands, which are B, F, L, R, and S. For each classifier H^i at time t, we can then define a *decision matrix* (of size $L \times M$). An element of the decision matrix, $\boldsymbol{\Phi}_t = [\phi_{j,t}^i] \in \{1, 0\}$ is 1 if the jth command in Γ is detected by classifier H^i at time t. Otherwise, $\phi_{j,t}^i$ is set to 0. For a particular classifier H^i, i.e., along the ith row, the decision matrix will only have one nonzero elements corresponding to the generated command from classifier H^i. This is because each classifier only outputs one decision.

We calculate the score for each potential decision $\gamma \in \Gamma$ by fusing the weights and decisions from all classifiers $H^1 - H^L$, as follows:

$$score = diag\left(\boldsymbol{W}_t^T \times \boldsymbol{\Phi}_t\right). \tag{3}$$

The entries of this array provide a measure of the likelihood of each command. Therefore, we determine the final fused decision D_t^f by simply choosing the decision corresponding maximum element of *score* as

$$D_t^f = argmax\,(score). \tag{4}$$

We also experimented with alternative voting rules discussed in Reference [35] and observed that weighted majority voting provides the most robust solution. Indeed, the majority voting rule works well on most domains and is one of the most robust voting rules [11].

5 ASSISTIVE INDOOR NAVIGATION

5.1 Preliminaries

Each potential destination on the target floorplan, e.g., a room, is called an explicit point of interest (POI). For example, Figure 4 shows one of the floorplans used in our simulations and highlights the explicit POIs with ⋄ markers. When two explicit POIs cannot be connected by a straight path due to an obstacle, such as a wall, we introduce implicit POIs. The implicit POIs are the points on the floorplan that connect two or more explicit POIs using a straight path. They can also be considered the turning points that the wheelchair needs to pass through to go from one explicit POI to another. The implicit POIs in Figure 4 are shown with ∘ markers.

Definition 1 (Point of Interest Graph). The POIs in the target floorplan are represented by the directed graph $G_F(V, E)$,

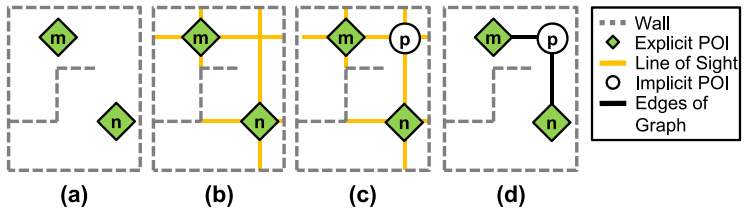

Fig. 3. Implicit POI generation. (a) A sample floorplan with two explicit POIs, m and n, (b) Lines of sight of explicit POIs, (c) Generation of implicit POI, p, from the intersection of lines of sight, and (d) POIs with connected edges.

- The set of vertices, V, represents all possible POIs. Each vertex $v_i \in V$ has two attributes. The first one specifies the position (P_{xi}, P_{yi}), and the second one shows whether the corresponding POI is explicit or implicit.
- The edges E denote the set of paths that connect the POIs. Each edge $e_i \in E$ has an attribute d_i, which gives the distance between its end points.

Definition 2 (Wheelchair State). The state of the wheelchair at time t is given by the triple $S_t = (x_t, y_t, \theta_t)$. x_t and y_t denote the coordinates of the wheelchair, while θ_t is the orientation at time t.

Definition 3 (User Command). The user inputs needed for navigation are referred to as *user command*. In this work, we use $\Gamma = \{F, R, L, B, S\}$ commands to predict the user intent and automate navigation towards the intended target.

Problem Formulation: *Given* the current state of the wheelchair $S_t = (x_t, y_t, \theta_t)$ and the user command $C_t \in \{F, R, L, B, S\}$,

Determine the set of possible target vertices and *automate* the wheelchair navigation until the next decision point.

The proposed solution to this problem consists of three steps:

(1) generating the POI graph $G_F(V, E)$,
(2) predicting intended task using the fused command from the user, and
(3) planning and automating the navigation until the next decision point.

5.2 POI Graph Generation

We generate the POI graph $G_F(V, E)$ at the beginning of the navigation task using the image of the floorplan that marks only the explicit POIs. The implicit POIs are inserted automatically by dividing the path between two explicit POIs into multiple segments, when there is a wall or another obstacle on the path. This process is illustrated step by step in Figure 3. The point of interest generation algorithm considers the obstacles that are known at the time of planning. In extreme circumstances, moving objects or obstacles may block parts of the planned trajectory. As soon as such an obstacle is detected, the POI graph is constructed using the current position as the initial point. Then, a new trajectory is generated following the user input at that location.

Once the explicit and implicit POI locations are known, we mark the starting position as the root of $G_F(V, E)$, as illustrated in Figure 4(b). The rest of the POI graph $G_F(V, E)$ is constructed using a breadth-first search on the floorplan. Starting with the root, the breadth-first search checks if moving forward would bring the wheelchair to an explicit or implicit POI in the graph. If this is true, then it inserts an edge between the current vertex and the new vertex. Then, the new edge is labeled as F and its distance is recorded. After repeating this process for right and left turns,

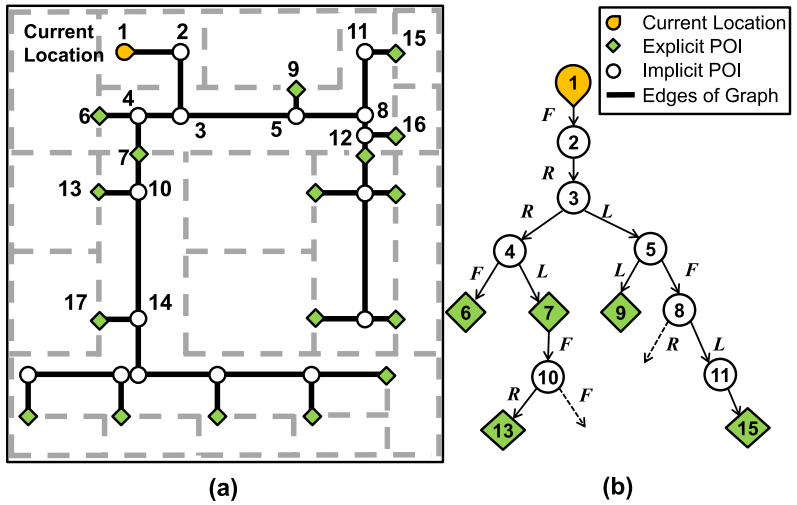

Fig. 4. (a) A sample floorplan used in this work, (b) An illustrative portion of floorplan graph automatically generated from the floorplan.

the algorithm moves to the next unvisited POI until all POIs are visited. The result is the final POI graph $G_F(V, E)$, where the edges are labeled with F, R, or L. For example, the POI graph in Figure 4(b) shows that moving forward from the starting position would bring the wheelchair to the implicit POI represented by vertex 2. Likewise, a right turn at vertex 2 would move the wheelchair to implicit POI 3.

In summary, the POI graph and the labels enable us to encode the path from the current position to all possible destinations using the labels. For example, the path from the starting position (vertex 1) to the explicit POI represented by vertex 6 can be encoded by $\{F, R, R, F\}$. This encoding is used by the prediction step of the proposed technique to automate the navigation, as described next.

5.3 Prediction Algorithm

The navigation algorithm starts with an empty command queue and the POI graph $G_F(V, E)$, as outlined in Figure 5. When a new user input is received, the algorithm traverses the outgoing edge labeled with the received command, and appends it to the queue. If an explicit POI is reached, then the algorithm starts automated navigation from the wheelchair's current position to this POI. Otherwise, the algorithm checks if the outdegree of the new vertex is equal to one. An outdegree of one means that the wheelchair can pass through the POI without waiting for any further user input. Therefore, the algorithm moves to the next POI, and adds the label of the outgoing edge to the queue. This process repeats until an explicit POI or a POI with outdegree greater than one is reached. By default, our algorithm returns the command queue constructed until that point, since an outdegree greater than one means that the corresponding vertex is a decision point. Hence, the wheelchair moves automatically until reaching the last POI encoded in the command queue. If the user provides the next input before this POI, then the algorithm runs again with the new input to construct the next command queue *without interrupting the operation*. Otherwise, it stops at the last POI and waits for the next user input.

We note that it is also possible to speculate the edge that will be taken upon reaching a vertex with outdegree greater than one. Speculation can be made with the help of most commonly visited explicit POIs, and can potentially automate the navigation all the way to the final target. However,

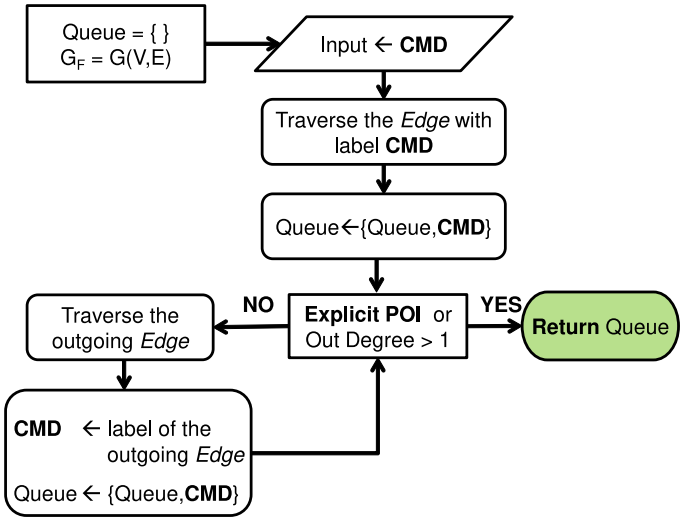

Fig. 5. The flow diagram of the prediction algorithm.

the downside of speculation is the probability of a wrong decision, *unless the user provides the next command before the speculation point.* We do not exercise speculation, since the default mode already operates uninterrupted if the next user input arrives before reaching the decision point. Furthermore, the cost of an incorrect decision is significantly larger than stopping, and we do not want the accuracy of the navigation algorithm to depend on the timing of the user inputs.

Computational Complexity and Scalability: The size of the POI graph affects two parameters. First, the POI has to be stored in memory. Therefore, it is important to fit POI to the on-chip memory for fast response. This is not a major limitation, since we can easily fit a POI of around 250 thousand nodes to 1MB on low-cost platforms like Intel Edison. The second and more important consideration is the computational complexity of the proposed technique. The worst-case complexity of the POI graph generation step grows quadratically with the number of explicit POIs in the floorplan. The worst-case happens only if no three explicit POIs are collinear. The complexity of the prediction step, which is the critical portion for real-time operation, grows linearly with the diameter of $G_F(V, E)$. Since only the prediction part needs to be performed repeatedly in real-time, the computation time does not present a major limitation, either. Experimental runtime and power overhead measurements on the Intel Edison board show that our algorithm can run in real-time with $\approx 0.7\%$ power consumption overhead.

5.4 Updating Command Probabilities using Contextual Awareness

As mentioned in Section 4, Equation (1) requires computing the command generation probabilities $P(C_t = \gamma)$ and $P(C_t \neq \gamma)$, which depend on the context [28]. For example, users are unlikely to go *Forward* while facing a wall. Since we compute these probabilities using the contextual information provided by the user's destination preferences and floorplan, this computation is presented in this section.

In the beginning, the probability that the user intends a specific command is distributed uniformly among all possible commands, i.e., $P(C_t = \gamma) = \frac{1}{M}$ and $P(C_t \neq \gamma) = \frac{M-1}{M}$. Since this is generally not true, we propose a method for updating these values when more information is gathered about the user and environment during *online training*.

Table 1. The Wheelchair Parameters Used in this Work

Wheelchair Specifications	
Maximum speed	2.24m/s
Distance between wheels	0.314m

First, we consider the history of a user's preferred destinations. To keep track of a user's destination preferences from each POI, we define the history matrix, Ψ^{V_k} for each point on the floorplan:

$$\mathbf{\Psi^{V_k}} = \left[\psi_1^{V_k} \ \psi_2^{V_k} \ \cdots \ \psi_s^{V_k} \right], \tag{5}$$

where s is the size of the history and $\{\psi_1^{V_k}, \psi_2^{V_k}, \ldots, \psi_s^{V_k}\} \in V$. By storing the destinations that a user visits on the floorplan, we can update the probability of the user visiting the destinations.

We calculate the probability of visiting the destinations based on context of the floorplan. In Figure 4(b), we see how the graph of POIs is generated as a tree, with branches connecting a given POI to possible POIs from that location. We also note the commands required to bring the user to each POI from that location. It is then apparent that not all possible commands can be used to reach a POI from each location on the graph (e.g., a user cannot issue a forward command if it makes the wheelchair run into a wall). We can further update the probabilities of certain commands by only considering the set of possible commands from each POI on the graph. By making these inferences, we can improve our estimates of the credibility for each classification modality.

The history matrix $\mathbf{\Psi^{V_K}}$ remembers the latest s destinations (in our case $s = 20$ locations). If the user starts abruptly visiting new destinations outside this set, then the list of the most recently visited destination will start changing accordingly. Due to the sliding window nature, the response of our algorithm to this change will not be instantaneous. That is, before the new set of destinations dominate the history, the POI will be affected by the older destinations.

5.5 Path Planning

We implement automated navigation using the kinematic model of two-wheeled robots:

$$\dot{x}_t = v_f cos(\theta_t), \ \dot{y}_t = v_f sin(\theta_t), \ \dot{\theta}_t = \omega, \tag{6}$$

where v_f and ω are the forward and angular velocities. We can express v_f and ω using the velocities of the right and left wheels (v_R, v_L), and the distance between the wheels (d) as

$$v_f = \frac{v_R + v_L}{2}, \ \omega = \frac{v_R - v_L}{d}. \tag{7}$$

We generate the trajectory using this kinematic model and the path following algorithm presented in Reference [10]. This algorithm computes the v_f and ω needed to follow a given trajectory from one POI to the next. Then, we use Equation (7) to calculate the reference velocities of the wheels. The wheelchair parameters used in this work are provided in Table 1.

5.6 Wheelchair State Estimation

The proposed algorithm requires the current state of the wheelchair to automate navigation. Therefore, there is a need for an indoor localization technique [18, 20]. We employ dead reckoning and absolute position sensing to accomplish this. Dead reckoning computes the speed of each wheel (v_R, v_L) with the help of optical encoders. Then, we estimate the wheelchair state S_t using Equations (6) and (7). However, the optical encoder is subject to localization uncertainties [12], which can lead to significant cumulative measurement error over time, as depicted in Figure 6.

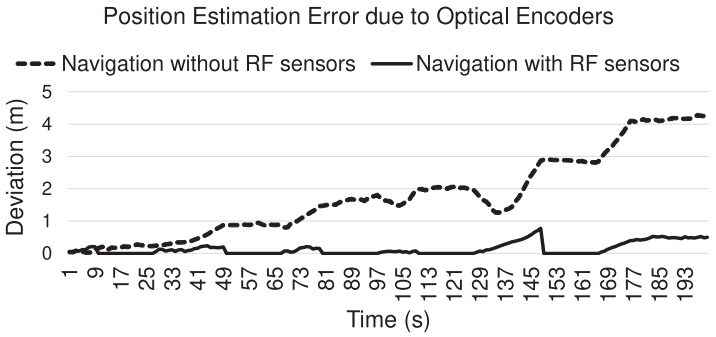

Fig. 6. Deviation from the actual path due to localization error.

Therefore, a direct sensing localization technique is essential to minimize the localization error. Direct sensing techniques include radio-frequency identification (RFID), infrared, ultrasound identification, Bluetooth beacons, or a barcode based localization [18]. Likewise, the orientation can be calculated using magnetic compasses, gyroscopes, or high-accuracy orientation sensors [26]. We chose the RFID based localization technique presented in Reference [32] due to the low cost and simple implementation. This method employs RFID tags, which are evenly spaced on a rectangular grid throughout the floorplan. It also assumes that the orientation (θ_t) is known at the beginning of the navigation task. When the wheelchair moves within the radius of detection of a new RFID tag (r), the coordinates of this RFID tag (x_{tag}, y_{tag}) and orientation are used to compute the absolute coordinates using

$$x_t = rcos(\theta_t) + x_{tag}, \tag{8}$$

$$y_t = rsin(\theta_t) + y_{tag}. \tag{9}$$

Consecutively, the last two known locations are used to update θ_t. We note that the position estimation from this method is associated with a measurement error, $\{\triangle d_{error} | \triangle d_{error} \in \setminus, -r < \triangle d_{error} < r\}$. However, the error from RF sensing is an order of magnitude smaller than the cumulative error from optical encoders and shown in Figure 6.

6 EXPERIMENTAL AND SIMULATION RESULTS

In this section, we first discuss experimental results obtained using our wearable system prototype. Then, we present thorough simulation studies on complex floorplans, performed with the help of experimental measurements.

6.1 Evaluation on an IoT Prototype

The IoT system shown in Figure 1 is prototyped using Raspberry Pi, the Intel Edison IoT board, and a Wi-Fi router. The Raspberry Pi models the *wearable device* used to capture the user intent and run the decision fusion algorithm described in Section 4. The inferred decision command D_t^f is transmitted to the Edison board, which controls the wheelchair. The Edison board generates the motor commands to navigate the wheelchair using the algorithm presented in Section 5.

The power consumption of the device physically connected to the wheelchair (in our system the Edison board) is not critical, since it can share the wheelchair's battery. However, the wearable device (in our system Raspberry Pi) relies on limited battery energy, so the energy consumption at the wearable device should be minimized. The wireless communication energy is the largest component of the total energy consumption. Hence, we employ communication energy saving

Table 2. Run-time and Power Consumption of the Proposed Algorithm Running on Raspberry Pi [36]

Proposed Algorithm		Communication Energy Overhead
Run-time	**Power Consumption Overhead**	
2.10ms	2.60mW	0.13mJ

Fig. 7. IoT-enabled wheelchair prototype and the Intel Edison board.

mechanisms while also limiting the added computational burden. The computation energy is limited using the light-weight decision fusion algorithm presented in Section 4, while the communication energy is saved by minimizing the number of commands transmitted to the wheelchair. To quantify this savings, we measure the power consumption of the Raspberry Pi using an external power meter, both with and without wireless transmission. By comparing the difference in total energy consumption, the energy consumed for transmitting each command is found as 0.13mJ, as summarized in Table 2. We note that using a low-power protocol like Bluetooth LE would reduce this overhead, but the relative savings reported under the simulation results would remain the same. On average, the proposed algorithm generates the queue of commands in 2.10ms and has a power consumption overhead of only 2.60mW, as shown in Table 2.

Our algorithm completes the navigation task with a small number of user inputs. Hence, the *main benefit* of the proposed algorithm is significantly reduced user effort. Furthermore, reduced effort translates directly into a smaller number of wireless transmissions and lower energy consumption.

Finally, the picture of our proof-of-concept wheelchair and Edison board is shown in Figure 7. Since systematic and repeatable study with different floorplans and user-input scenarios is not feasible on this prototype, we developed a simulation framework, as described next.

6.2 Simulation Methodology

Systematic evaluation of the proposed assistive indoor navigation algorithm is essential to draw meaningful conclusions about its performance in realistic scenarios. To achieve this, we employ a Virtual Robot Experimentation Platform (V-REP) to construct a co-simulation framework for the floorplans given in Figures 8 and 9 [19]. The co-simulation framework consists of three major components, as illustrated in Figure 10. The physical layer models the physical world using V-REP,

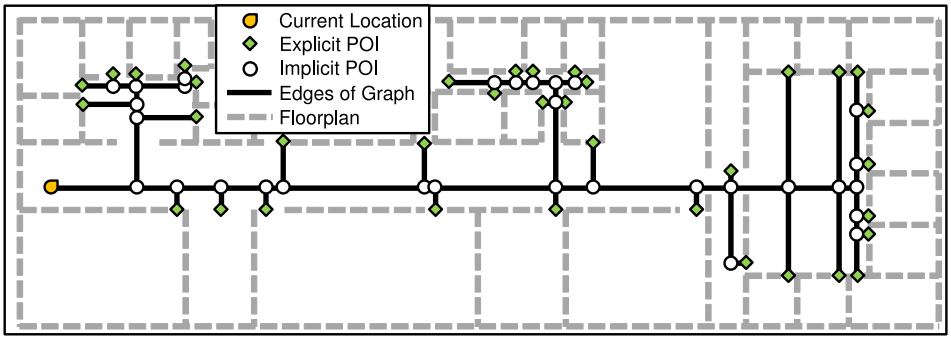

Fig. 8. Floorplan-B and the POIs used in our simulations.

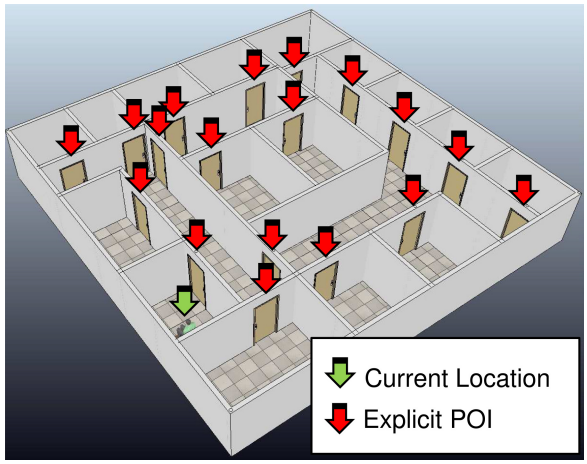

Fig. 9. 3D view of Floorplan-A. The POI graph of this floorplan is given in Figure 4.

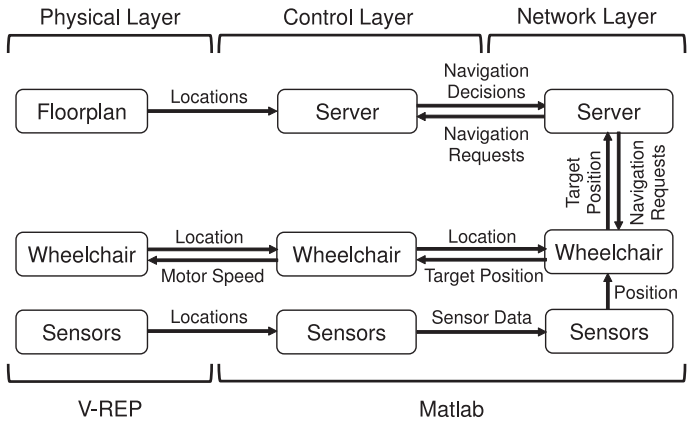

Fig. 10. The architecture of the co-simulation framework.

which accounts for the dynamics of the objects including the wheelchair. This model includes the walls, doors, sensors, human actors, the wheelchair, other objects and obstacles. The behaviors of the objects in the physical layer are modeled by the control layer implemented in Matlab. Finally, the network layer, also implemented in Matlab, models the communication network.

We opt for detailed simulation-based evaluation in order not to limit our experimental space with regards to floorplan generation, obstacles, wheelchair model and type, among several other variables. Our co-simulation framework enables evaluation of the complex scenarios by considering the tight interactions among the different layers, which are developed simultaneously. Modeling the physical environment using V-REP enables us to focus on developing and evaluating the target algorithm. Hence, this provides a valuable and versatile test bench for evaluating the effectiveness of our algorithm under different experimental environments.

Three sets of experiments are performed with each floorplan:

***Manual* navigation with ideal inputs:** The user generates the navigation commands to manually control the wheelchair using ideal inputs. *Ideal* means that the intended commands are generated with 100% accuracy.

***Manual* navigation with *realistic* inputs:** In these experiments, the wheelchair is controlled manually using the simple commands using the three modalities introduced in Section 3, i.e., gesture, speech recognition and BMI inputs. These commands *cannot* be interpreted with 100% accuracy due to imperfect sensing and processing. In our work, the interpretation accuracies of realistic inputs using BMI, speech recognition, and gesture recognition range between [54%, 68%], [70%, 92%], and [88%, 94%], respectively. On average, the interpretation accuracy of uni-modal classifiers ranges from 60% to 90%. During these experiments, the continuous command polling rate is set to 2Hz for manual navigation with both ideal and realistic inputs.

***Automated* navigations with realistic inputs:** We use the realistic inputs as in the second experiment. However, we employ the proposed indoor navigation algorithm described in Section 5, instead of manually driving the wheelchair. Fusion of the decisions improved the interpretation accuracy range and average to [94%, 98%] and 96%, respectively.

We sequentially navigate the wheelchair from the initial position to 18 different POIs, and repeat each simulation 50 times to obtain averages. During the experiments, we consider the essential elements of automated indoor navigation, such as localization, path planning, and obstacle detection. When not in range of a new RFID, the wheelchair estimates its location and orientation from the optical encoders installed on both wheels. Then, the wheelchair corrects its position according to the RF tag, when it comes inside RF transmission range, as discussed in Section 5. Obstacle avoidance is another issue considered in automated navigation [22]. In our virtual simulation, we employed proximity sensor-based object detection to avoid collision with people or other objects.

6.3 Simulation Results

Task completion times: Figures 11(a), 11(b), and 11(c) show the average completion times for Floorplan-A, Floorplan-B, and Floorplan-C, respectively. The destination POIs are sorted in decreasing order with respect to their distance to the starting point. For Floorplan-A, the navigation times with realistic inputs range from 32 to 183s, and have an average of 125s. With regards to navigation time, the proposed algorithm outperforms manual control of the wheelchair with continuous user inputs (in both the ideal input and realistic input scenarios). More precisely, it reduces the navigation time range to 21–121s, and the average to 86s. We observe that the proposed algorithm with realistic inputs achieves a lower completion time than manual wheelchair control using ideal inputs, even though the wheelchair speed remains the same. The reason for this improvement is the automation of the navigation without relying on continuous user inputs, leading

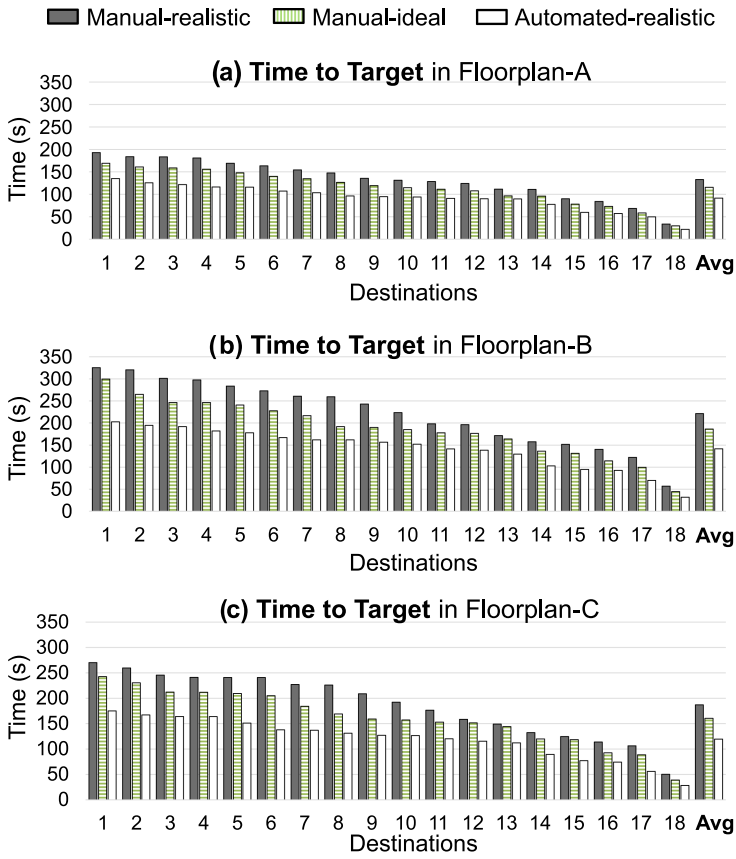

Fig. 11. *Time to target* for navigating from the initial position to 18 different POI targets.

to a smoother trajectory with fewer stops and turns. Moreover, the standard deviation reduces from 40 to 26s. This shows that the proposed algorithm not only decreases the navigation time but also makes the results more predictable.

The improvements achieved by the proposed approach are even larger in Floorplan-B and Floorplan-C, as shown in Figures 11(b) and 11(c). More specifically, the reduction in navigation time ranges from 25% to 34% in Floorplan-B, and 24% to 30% in Floorplan-C. The reason for this larger improvement is the increased complexity of these floorplans, i.e., a larger number of turns and larger distances in these floorplans.

The number of user inputs and impact on communication energy: The main purpose and major benefit of using the proposed indoor navigation technique is the reduction in the number of commands the user has to generate. Our automated technique decreases the number of user inputs on average 4.0× in Floorplan-A, 5.2× in Floorplan-B, and 5.0× in Floorplan-C. The huge reduction in user effort translates into considerable savings in communication energy consumption, as shown in Figures 12(a), 12(b), and 12(c). In particular, the proposed technique cuts down the total communication energy on average from 9.9 to 2.4mJ in Floorplan-A, from 10.8 to 2.1mJ in Floorplan-B, and from 10.9 to 2.1mJ in Floorplan-C. Moreover, Figure 12 shows that the total number of inputs is also significantly lower than that required for manual driving of the wheelchair with ideal inputs. In summary, the proposed automated navigation algorithm results in 24.0%

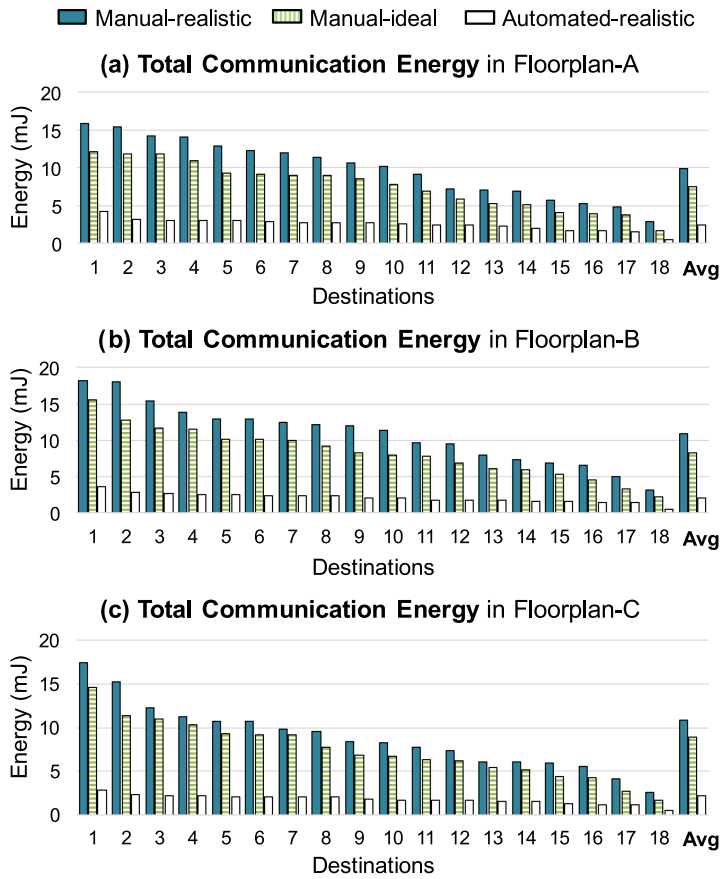

Fig. 12. *Communication energy cost* for navigating from the initial position to 18 different POI targets.

Table 3. Summary of the Improvements Obtained with the Proposed Algorithm

Improvement Over	Time		Number of Inputs	
	Average	Worst Case	Average	Worst Case
Realistic Inputs	34.1%	30.5%	78.5%	74.8%
Ideal Inputs	24.0%	20.8%	72.6%	67.2%

improvement in average navigation time and 72.6% reduction in the number of user inputs, as shown in Table 3.

Floorplan Complexity Analysis and Summary: We also analyze the average navigation time and the number of required user inputs as a function of the floorplan complexity. As a measure of complexity, we use the number of turns in the floorplan and distances between the POIs. Since the results are similar, we show only the plots for the number of turns. Figure 13 shows the variation in total communication energy as a function of number of turns in the floorplan. A larger number of turns implies a larger number of user inputs and, consequently, higher communication energy consumption. This plot also clearly reveals the effectiveness of the proposed indoor navigation algorithm in reducing the user effort and communication energy. More importantly,

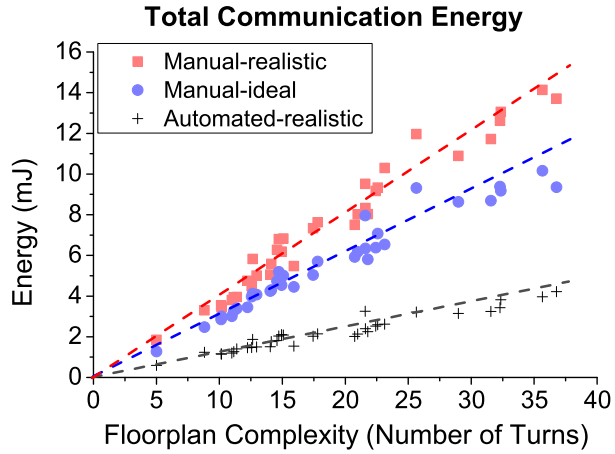

Fig. 13. The *total communication energy* as a function of number of turns in the floorplan.

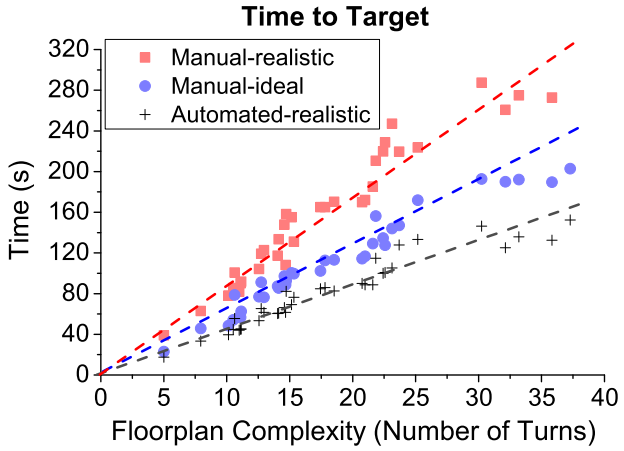

Fig. 14. The *navigation time* as a function of number of turns in the floorplan.

the savings grow with the floorplan complexity. We observed the same trends also for navigation time, as shown in Figure 14. The improvements are smaller for the navigation time, since the traveled distance and wheelchair speed do not change. Nevertheless, we still observe considerable improvement, which grows further with floorplan complexity.

6.4 Cost-Benefit Analysis

The prediction algorithm is implemented using Intel Edison and Raspberry Pi. These boards are available commercially and at a low-cost. RFIDs are gaining extensive attention due to their small size and low cost [2]. Each tag costs approximately $0.05–$0.10 [45]. Sensors and devices used for modality classification vary depending on the need of the user. As a result, the cost of such devices may vary widely. For example, classification modalities such as gesture and speech recognition can be implemented using low-cost development boards for less than $50 [38]. Portable EEG headsets are commercially available in the range of $200–$500 [9].

Hence, with a negligible added cost, a regular electric wheelchair can be used to implement a robust assistive indoor navigation system, which offers the advantages discussed in the experimental results.

7 CONCLUSIONS AND FUTURE WORK

IoT technology offers a huge potential to assist people with physical disabilities. Designing effective and practical solutions requires addressing two challenges. The first one is providing a natural and simple mechanism to capture user commands, while the second challenge is utilizing these commands to minimize user intervention. This article presented a low-cost and practical IoT system that addresses both of these challenges. Our assistive indoor navigation algorithm delivers one order of magnitude reduction in number of user inputs to complete complex tasks.

As future work, we plan to repeat the evaluations using a real wheelchair in addition to the V-REP simulation software. This is an important step in determining the true viability of this algorithm as an application for assisting those with disabilities. The proposed approach can also be extended for outdoor navigation. This extension requires accurate localization, e.g., using global positioning system instead of RFID markers, and the floorplan of a bounded region.

REFERENCES

[1] World Report on Disability. 2017. Retrieved from http://www.who.int/disabilities/world_report/2011/en/.
[2] Luigi Atzori, Antonio Iera, and Giacomo Morabito. 2010. The internet of things: A survey. *Comput. Netw.* 54, 15 (2010), 2787–2805.
[3] Ganapati Bhat, Jaehyun Park, and Umit Y. Ogras. 2017. Near optimal energy allocation for self-powered wearable systems. In *Proceedings of the International Conference on Computer-Aided Design*.
[4] Luzheng Bi, Xin-An Fan, and Yili Liu. 2013. EEG-based brain-controlled mobile robots: A survey. *Hum.-Mach. Syst.* 43, 2 (2013), 161–176.
[5] Joydeep Biswas and Manuela Veloso. 2012. Depth camera based indoor mobile robot localization and navigation. In *Proceedings of the Conference on Robotics and Automation*. 1697–1702.
[6] Guillaume Blanc, Youcef Mezouar, and Philippe Martinet. 2005. Indoor navigation of a wheeled mobile robot along visual routes. In *Proceedings of the Conference on Robotics and Automation*. 3354–3359.
[7] Intel Edison Board. 2015. *Intel Edison Breakout Board Hardware Guide*.
[8] Francisco Bonin-Font, Alberto Ortiz, and Gabriel Oliver. 2008. Visual navigation for mobile robots: A survey. *J. Intell. Robot. Syst.* 53, 3 (2008), 263–296.
[9] Andrew Campbell, Tanzeem Choudhury, Shaohan Hu, Hong Lu, Matthew K. Mukerjee, Mashfiqui Rabbi, and Rajeev D. S. Raizada. 2010. NeuroPhone: Brain-mobile phone interface using a wireless EEG headset. In *Proceedings of the ACM SIGCOMM Workshop on Networking, Systems, and Applications on Mobile Handhelds*. 3–8.
[10] Oguz H. Dagci, Umit Y. Ogras, and U. Ozguner. 2003. Path following controller design using sliding mode control theory. In *Proceedings of the American Control Conference*, Vol. 1. 903–908.
[11] Partha Dasgupta and Eric Maskin. 2008. On the robustness of majority rule. *J Eur. Econ. Assoc.* 6, 5 (2008), 949–973.
[12] Berkan Dincay. 2010. GPS/optical encoder based navigation methods for dsPIC microcontroled mobile vehicle.
[13] Mari Carmen Domingo. 2012. An overview of the internet of things for people with disabilities. *J. Net. Comput. Appl.* 35-2 (2012), 584–596.
[14] Charalampos Doukas and Ilias Maglogiannis. 2012. Bringing IoT and cloud computing towards pervasive healthcare. In *Proceedings of the Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. 922–926.
[15] Bruno Dumas, Denis Lalanne, and Sharon Oviatt. 2009. Multimodal interfaces: A survey of principles, models and frameworks. *Hum. Mach. Interact.* 5440 (2009), 3–26.
[16] Emotivn. d.. EEG Neuroheadset. Retrieved from http://emotiv.com/store/hardware/epoc-bci-eeg/developer-neuroheadset/.
[17] M. Bryan et al. 2012. Automatic extraction of command hierarchies for adaptive brain-robot interfacing. In *Proceedings of the Conference on Robotics and Automation*. 3691–3697.
[18] Navid Fallah, Ilias Apostolopoulos, Kostas Bekris, and Eelke Folmer. 2013. Indoor human navigation systems: A survey. *Interact Comput.* 25, 1 (2013), 10.
[19] Marc Freese, Surya Singh, Fumio Ozaki, and Nobuto Matsuhira. 2010. Virtual robot experimentation platform V-REP: A versatile 3D robot simulator. In *Simulation, Modeling, and Programming for Autonomous Robots*. Springer, 51–62.

[20] Yanying Gu, Anthony Lo, and Ignas Niemegeers. 2009. A survey of indoor positioning systems for wireless personal networks. *Commun. Surveys Tutor.* 11, 1 (2009), 13–32.
[21] Haitham Hasan and Sameem Abdul-Kareem. 2014. Human–Computer interaction using vision-based hand gesture recognition sys.: A survey. *Neural Comput. Appl.* 25, 2 (2014), 251–261.
[22] Michael Hoy, Alexey S. Matveev, and Andrey V. Savkin. 2015. Algorithms for collision-free navigation of mobile robots in complex cluttered environments: A survey. *Robotica* 33, 03 (2015), 463–497.
[23] David Huggins-Daines, Mohit Kumar, Arthur Chan, Alan W. Black, Mosur Ravishankar, and Alexander I. Rudnicky. 2006. Pocketsphinx: A free, real-time continuous speech recognition system for hand-held devices. In *Proceedings of the Acoustics, Speech and Signal Conference*, Vol. 1. I–I.
[24] Alejandro Jaimes and Nicu Sebe. 2007. Multimodal human–Computer interaction: A survey. *Comp. Vision Image Understand.* 108, 1 (2007), 116–134.
[25] Matthias Kranz, Paul Holleis, and Albrecht Schmidt. 2010. Embedded interaction: Interacting with the internet of things. *IEEE Internet Comput.* 14, 2 (2010), 46–53.
[26] Wei-chen Lee and Cong-wei Cai. 2013. An orientation sensor for mobile robots using differentials. *Int. J. Adv. Robot. Syst.* 10 (2013).
[27] Daniele Miorandi, Sabrina Sicari, Francesco De Pellegrini, and Imrich Chlamtac. 2012. Internet of things: Vision, applications and research challenges. *Ad Hoc Netw.* 10, 7 (2012), 1497–1516.
[28] Md Muztoba et al. 2015. Context-aware control of smart objects via human-machine communication. In *Proceedings of the Biomedical Circuits and Systems Conference (BioCAS'15)*. 1–4.
[29] Md Muztoba, Ujjwal Gupta, Tanvir Mustofa, and Umit Y Ogras. 2015. Robust communication with IoT devices using wearable brain machine interfaces. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*. 200–207.
[30] Claudia Oppenauer. 2009. Usability of assistive technologies in ageing and disabilities. *Intl. J. Integr. Care* 9, 5 (2009).
[31] Jaehyun Park et al. 2017. Flexible PV-cell modeling for energy harvesting in wearable IoT applications. *ACM Trans. Embed. Comput. Syst.* 16, 5s (2017), 156.
[32] Sunhong Park and Shuji Hashimoto. 2009. Autonomous mobile robot navigation using passive RFID in indoor environment. *IEEE Trans. Industr. Electron.* 56, 7 (2009), 2366–2373.
[33] Betsy Phillips and Hongxin Zhao. 1993. Predictors of assistive technology abandonment. *Assist. Tech.* 5, 1 (1993), 36–45.
[34] Isabella Poggi. 2007. *Mind, Hands, Face And Body. A Goal And Belief View Of Multimodal Communication*. Weidler.
[35] Robi Polikar. 2006. Ensemble based systems in decision making. *Circ. Syst. Mag.* 6, 3 (2006), 21–45.
[36] Raspberry Pin. d.. Raspberry Pi Foundation. Retrieved from https://www.raspberrypi.org/.
[37] Siddharth S. Rautaray and Anupam Agrawal. 2015. Vision based hand gesture recognition for human computer interaction: A survey. *Artific. Intell. Rev.* 43, 1 (2015), 1–54.
[38] Ali Raza, Ataul Aziz Ikram, Asfand Amin, and Ahmad Jamal Ikram. 2016. A review of low cost and power efficient development boards for IoT applications. In *Proceedings of the Future Technical Conference*. 786–790.
[39] Vandana Milind Rohokale, Neeli Rashmi Prasad, and Ramjee Prasad. 2011. A cooperative internet of things (IoT) for rural healthcare monitoring and control. In *Proceedings of the International Conference on Wireless Communications, Vehicular Technology, Information Theory and Aerospace Electronic Systems (VITAE'11)*. 1–6.
[40] Dmitry Sinyukov, Ross Desmond, Matthew Dickerman, James Fleming, Jerome Schaufeld, and Taskin Padir. 2014. Multi-modal control framework for a semi-autonomous wheelchair using modular sensor designs. *Intell. Service Robot.* 7, 3 (Apr. 2014), 145–155. Retrieved from https://doi.org/10.1007/s11370-014-0149-7
[41] Richard S. Sutton and Andrew G. Barto. 1998. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA.
[42] Texas Instruments. 2015. Multi-Standard SensorTag. Retrieved from http://www.ti.com/tool/TIDC-CC2650STK-SENSORTAG.
[43] Sebastian Thrun. 1998. Learning metric-topological maps for indoor mobile robot navigation. *Artific. Intell.* 99, 1 (1998), 21–71.
[44] Matthew Turk. 2014. Multimodal interaction: A review. *Pattern Recogn. Lett.* 36 (2014), 189–195.
[45] Ju Wang, Jie Xiong, Hongbo Jiang, Xiaojiang Chen, and Dingyi Fang. 2017. D-watch: Embracing bad multipaths for device-free localization with COTS RFID devices. *IEEE/ACM Trans. Network.* 25, 6 (2017), 3559–3572.
[46] Yu A. Zuev and S. K. Ivanov. 1999. The voting as a way to increase the decision reliability. *J. Franklin Inst.* 336, 2 (1999), 361–378.

Received October 2017; revised March 2018; accepted April 2018