Scenariot: Spatially Mapping Smart Things Within Augmented Reality Scenes

Ke Huo, Yuanzhi Cao, Sang Ho Yoon, Zhuangying Xu, Guiming Chen, Karthik Ramani *
School of Mechanical Engineering, Purdue University
West Lafayette, IN 47907, USA
{khuo, cao158, yoon87, xu970, chen1956, ramani}@purdue.edu

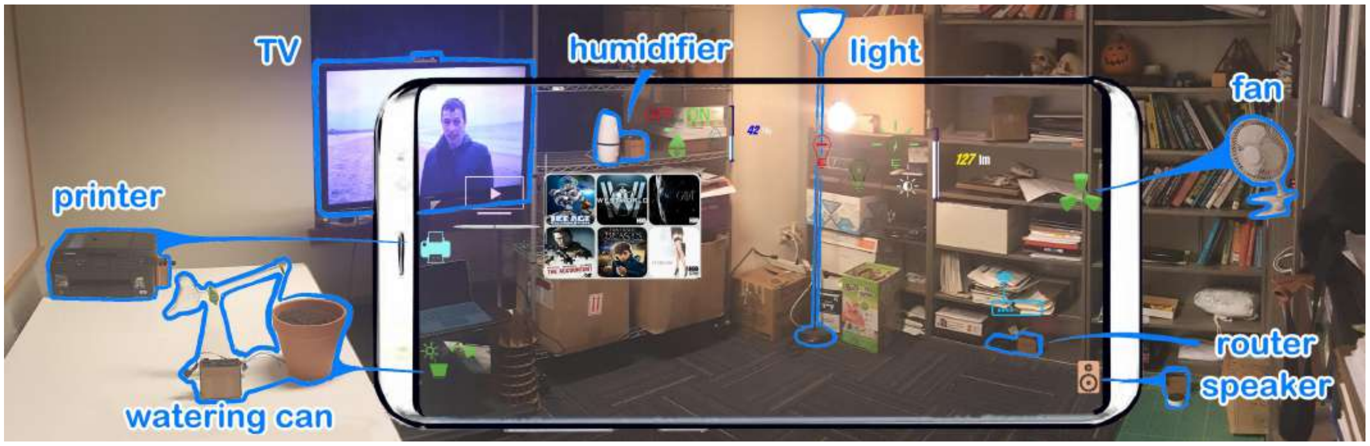

Figure 1. *Scenariot* is a method for discovering and localizing IoT devices with a SLAM-based AR device. We register the discovered devices spatially in the AR scene to enable new spatial aware interactions.

ABSTRACT
The emerging simultaneous localizing and mapping (SLAM) based tracking technique allows the mobile AR device spatial awareness of the physical world. Still, smart things are not fully supported with the spatial awareness in AR. Therefore, we present *Scenariot*, a method that enables instant discovery and localization of the surrounding smart things while also spatially registering them with a SLAM based mobile AR system. By exploiting the spatial relationships between mobile AR systems and smart things, *Scenariot* fosters in-situ interactions with connected devices. We embed Ultra-Wide Band (UWB) RF units into the AR device and the controllers of the smart things, which allows for measuring the distances between them. With a one-time initial calibration, users localize multiple IoT devices and map them within the AR scenes. Through a series of experiments and evaluations, we validate the localization accuracy as well as the performance of the enabled spatial aware interactions. Further, we demonstrate various use cases through *Scenariot*.

*School of Electrical and Computer Engineering (by courtesy)

CHI 2018, April 21–26, 2018, Montreal, QC, Canada
© 2018 ACM. ISBN 978-1-4503-5620-6/18/04…$15.00
DOI: https://doi.org/10.1145/3173574.3173793

ACM Classification Keywords
H.5.2 Information Interfaces and Presentation (e.g. HCI): User Interfaces, Interaction Styles; H.5.1 Multimedia Information Systems: Artificial, augmented, and virtual realities

Author Keywords
Augmented Reality; IoT; Smart Environment; Spatial Interactions; Context Awareness; Localization; UWB; SLAM

INTRODUCTION
The ecology of connected smart devices is being rapidly interwoven with people's daily lives and work environments. People envision that their surrounding physical world will largely be enhanced with ubiquitous computing [31]. However, accessing and interacting with the Internet of Things (IoT) remains challenging due to the increasing diversity and complexity of the connected devices [6]. Traditionally, the digital interfaces of the interactive devices have been realized with a self-equipped touch screen display which has a limited adaptability. But now, contemporary IoT devices allow users

to access full functionalities remotely by using an offloaded interface on a smartphone. Still, in order to discover and access the devices, users need to browse through a specific webpage on-line or search for the corresponding applications. To alleviate the cumbersome processes, we leverage the spatial information of the devices relative to the environment and propose a physical browsing approach with AR.

As a novel interface which bridges the real and the digital, Augmented Reality (AR) has become a promising surrogate for interacting with the proliferating smart things [22, 26, 28, 39]. By superimposing the graphical digital interfaces on the physical world, users are exposed to the functionalities of the devices together with their physical affordance. This way, users are able to directly and intuitively access the smart environment. Moreover, the emerging visual SLAM technique equips a mobile AR device with spatial awareness about the surrounding environment.Thus further spatial references based interaction metaphors can be realized in AR [17, 19, 25].

To this end, the key part of interacting with the smart environment in mobile AR is mapping of the smart objects globally, i.e., knowing where the smart things are located in the AR scene. Simple scene augmentation has been achieved by detecting the objects in the view of a camera. More recent works have shown progresses in multi-view object detection [32] and pose estimation [38] during consecutive movements of the camera. But, computer vision approaches largely rely on keeping the object of interest in the camera's view locally, which implies that users are aware of the identities and locations of the devices ahead.

In contrast, we primarily aim at enabling AR interactions with the surrounding smart environment as a whole ecology. This requires discovering and localizing the smart things globally without prior location information of the devices. Wireless techniques such as Bluetooth, Zigbee, and WiFi allow for automatic discovery of the connected devices in an area network. Yet, a received signal strength indication (RSSI) based localization with the above technology suffers from low accuracy (from only a few meters) [1]. An accurate alternative utilizing Ultra-wide Bandwidth (UWB) based RF technology has been advanced and made accessible recently. Therefore, we develop a distance based localization method which integrates UWB with SLAM to achieve quick mapping of smart devices spatially in the AR scene.

We present *Scenariot* (Figure 1), an AR system which provides fast estimation of the 3D locations of smart things and exploits the spatial relationships for location aware interactions. To achieve this, we equip the IoT controllers and the AR device with distance measurement units. The user carries the distant sensing capable AR device and surveys the surrounding environment while moving. We develop a distance based localization algorithm to estimate the positions of the IoT devices. By mapping the IoT devices into the coordinate system of the AR environment, *Scenariot* enables spatial context aware interactions instantly, including distant pointing, proximity based control, and visual navigation. In our current prototype, *Scenariot* supports a single user to map up to 10 IoT devices which are distributed in a room ($\sim 10 \times 10$m) with an accuracy of ~ 0.4m. Following is a list of the contributions:

- An approach to estimating the 3D locations of distributed smart things using a SLAM based AR device;
- Implementation and evaluation of hardware and software systems allowing users to rapidly map the smart things and interact with them in AR scenes; and
- A wide range of example applications demonstrating the usage of the proposed localization method and the enabled interaction metaphors.

RELATED WORK

Context Awareness of Ubiquitous Computing Devices

Moderate pervasive computing devices such as mobiles and wearables, are able to discover the smart things connected to the same network and retrieve the corresponding interfaces effortlessly. Enabling context awareness of the surrounding smart environments on mobile devices has been the focus of ubiquitous computing community [17]. Researchers attempted to identify and select smart devices through a *touching* or a close proximity interaction [36], which means that the user needs to either physically contact or be present in close proximity (within 1m) to the target. Early works incorporated short-range RFID readers [45] or near field communication (NFC) chip [36] in mobile devices to link with a smart device. Recently, through leveraging the electromagnetic (EM) emissions from the smart devices, researchers have investigated using machine learning to recognize the EM signatures [24, 47, 44] without instrumenting the devices. To achieve selecting the device at a distance, various technical approaches have been considered including ultra-high frequency (UHF) RFID [42], infrared targeting [10], visible light sensing [39], magnetic sensing [48], visual fiducial tags [22, 20], and visual natural features [12]. To this extent, previous efforts primarily focused on local interaction with a device with prior knowledge of where it is located in the environment. The spatial knowledge about the smart devices played an important role in context awareness [9, 17, 25, 35]. Here, we emphasize the discovery of the devices' absolute positions in a smart environment using a wireless localization method. We bring location awareness to the smart devices and enable mobile spatial interactions within an AR scene.

Interacting with Smart Environment in AR

Recent works have shown great interests in leveraging mobile AR technology to interact with the smart environment [22, 26, 28, 39, 40]. In these works, the spatial relationship between the AR device and the smart device remains local, which means the augmentation only applies on specific devices in the instant view. Yet, the awareness of the surrounding environment as a whole ecology in AR requires mapping the devices using their 3D locations. The SLAM based tracking technique, which brings the AR device awareness about the physical environment, has significantly matured in past years and has started to appear on commercialized product [4, 18, 29, 46]. However, the SLAM map itself has no semantic information. Recent researches have shown progress in object detection [32] and pose estimation [38] working with visual SLAM. But it is still challenging to discover the 3D locations of all smart devices scattered in a cluttered scene by only using computer vision. Therefore, we propose the use of a wireless localization technique together with SLAM, which requires no prior knowledge of the smart environment, to estimate the absolute positions of the smart devices instantly.

Wireless Localization and Mapping Technique

The wireless localization and mapping problem, especially in indoor environment, has been studied extensively [1]. We primarily consider the infrastructure-free localization techniques since we aim at instantly estimating the locations of the devices. We draw inspiration from the concept of wireless sensor network (WSN) localization which estimates nodes' positions as the smart devices naturally form a network. A common solution which deals with indoor environment is a distance-based localization method which derives the coordinates by measuring the distances across the nodes [2]. For examples, Multidimensional Scaling (MDS) and its variations are widely used distance based methods [7, 11]. Recent developed UWB technology which provides an accurate distance measurement($\sim$ 0.1m) leads to a highly accurate localization [15]. However, these approaches usually consider only stationary nodes. Hahnel et.al. proposed an idea using a mobile robot which was equipped with UHF RFID reader and two antennas to survey an environment and localize the RFID tags placed in the environment [21]. We merge this surveying idea into the WSN localization solutions. We leverage the mobility of our mobile AR device to survey the smart environment. With the SLAM capability from the AR device, we generate an arbitrary number of nodes whose positions are known and collect the distance measurements along the surveying path. We develop an adapted MDS algorithm to estimate the locations of the smart devices.

SCENARIOT

We embed UWB units on IoT controllers and mobile AR devices. The distributed smart devices in the surrounding environment together with the AR device form a UWB network as shown in Figure 2. Unlike the conventional localization in wireless sensor networks where all nodes are stationary, we incorporate a dynamically moving node (the mobile AR device), along with a group of stationary nodes (distributed smart things). We are interested in finding the positions of the *stationary* nodes relative to the *dynamic* one. Due to the visual SLAM built in the mobile AR device which creates and updates a global map of the surrounding environment, the dynamic node is capable of real-time *self-localizing* on the map. We leverage the mobility and treat the *dynamic* node as a mobile *surveying* platform. Along the moving path of the dynamic node, we collect the distance measurements between dynamic node and each of the stationary nodes together with the positions at every measuring instance. We then employ the MDS technique and derive the 3D coordinates of the nodes in the coordinate system of the built SLAM map. Note that, in order to achieve 3D localization, we require 3D movements instead of planar ones from the dynamic node.

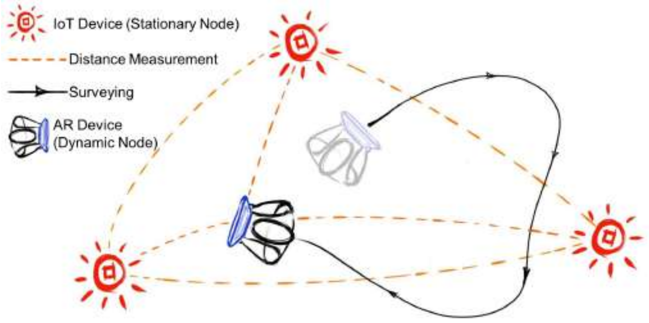

Figure 2. ***Scenariot*** **localization principle.**

To this end, we discover and map the smart devices spatially in the AR scenes. It is worth noting that, the surveying movement only needs to be conducted once for an unknown environment. We store the 3D locations of the devices as well as the created SLAM map of the scene so that when users revisit the same region, the spatial registration is retained as long as the smart devices remain at the same locations and the environment has not changed much. We can render an AR scene with the digital representation of the smart devices superimposed at the physical objects' locations instantly. By exploiting the spatial relationship between the user and the connected devices, e.g., distance, orientation, and movement, we further enable context aware in situ AR interactions.

Reviewing MDS Localization Principles

We first describe a traditional localization problem in a wireless network solved with MDS. MDS is a general technique which recovers the coordinates of a collection of nodes by minimizing the mismatch between the measured distances and the distances calculated from the estimated coordinates [16]. Consider that we have N nodes to be localized in a fully connected network, in which the the Euclidean distance matrix across all N nodes is complete. We denote the coordinates as $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^T \in \mathbb{R}^{N\times 3}$. The MDS algorithm estimates the relative coordinates of the nodes by minimizing the stress function $S(\mathbf{X})$:

$$\min_{\mathbf{X}} \mathrm{S}(\mathbf{X}) = \min_{\mathbf{X}} \sum_{i\leq j\leq N} \omega_{ij}(\hat{d}_{ij} - d_{ij}(\mathbf{X}))^2, \qquad (1)$$

where $\hat{d}_{ij}$ is the distance measurement, $d_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|$, and the weight ω_{ij} is defined based on the quality of the measurements. We denote a weight matrix $\mathbf{W}$ with the size of $N\times N$ which includes ω_{ij} as an element.

To solve this optimizing problem, an iterative method called "Scaling by MAjorizing a COmplicated function" (SMACOF) has been widely used with high guarantees and speeds of convergence [13]. We introduce a majorizing function as $\mathrm{T}(\mathbf{X},\mathbf{Z}) \geq \mathrm{S}(\mathbf{X})$ which bounds S from the above and touches the surface of S at $\mathbf{Z} \in \mathbb{R}^{N\times 3}$:

$$\mathrm{S}(\mathbf{X}) \leq \mathrm{T}(\mathbf{X},\mathbf{Z}) = C + \mathrm{tr}(\mathbf{X}^T\mathbf{V}\mathbf{X}) - 2\mathrm{tr}(\mathbf{X}^T\mathbf{B}(\mathbf{Z})\mathbf{Z}) \qquad (2)$$

where the matrix element of $\mathbf{V}$ and $\mathbf{B}(\mathbf{Z})$ are defined as follows:

$$v_{ij} = \begin{cases} \sum\limits_{k=1,k\neq j} & -\omega_{kj} \quad if i \neq j, \\ \sum\limits_{k=1,k\neq j} & v_{kj} \quad if i = j, \end{cases}$$

$$b_{ij} = \begin{cases} \sum\limits_{k=1,k\neq j} & \omega_{kj}\frac{\hat{d}_{ij}}{d_{ij}(\mathbf{Z})} \quad if i \neq j, \\ \sum\limits_{k=1,k\neq j} & -b_{kj} \quad if i = j, \end{cases}$$

$\mathrm{T}(\mathbf{X},\mathbf{Z})$ is a quadratic and thus convex function [13]. Further, we compute the minimum of the function as:

$$\mathbf{X} = \min_{\mathbf{X}} \mathrm{T}(\mathbf{X},\mathbf{Z}) = \mathbf{V}^{-1}\mathbf{B}(\mathbf{Z})\mathbf{Z} \tag{3}$$

The SMACOF as summarized in Algorithm 1 [16], iteratively minimizes the majorizing function $\mathrm{T}(\mathbf{X},\mathbf{Z})$. After solving the MDS localization using SMACOF, we obtain the relative coordinates of the nodes. However, the absolute positions of the nodes are lost when we rely only on the distance information. In order to recover the absolute positions fully, a set of at least 4 non-coplanar nodes (anchors) need to be localized a priori [16]. One common way to estimate the rigid body transformation, i.e., rotation-translation, between estimated coordinates of the anchors and the actual coordinates is by conducting a *Procrustes Analysis* [16].

Algorithm 1 SMACOF

```
1: procedure SMACOF($\mathbf{X}^{(0)}, \mathbf{W}$)
2:     calculate $\mathrm{S}(\mathbf{X}^0)$
3:     while $\delta \geq \varepsilon$ do
4:         $\mathbf{Z} = \mathbf{X}^{k-1}$
5:         $\mathbf{X}^k \leftarrow \min_{\mathbf{X}} \mathrm{T}(\mathbf{X},\mathbf{Z})$
6:         $\delta = \mathrm{S}(\mathbf{X}^{k-1}) - \mathrm{S}(\mathbf{X}^k)$
7:     return $\mathbf{X}^k$
```

SMACOF with Mobile "Anchors"

Our problem formulation differs from the above traditional approach with in three ways: (i) our network incorporates stationary nodes and a dynamic node, namely the smart devices and the mobile AR device; (ii) no prior location information on the stationary nodes is available, i.e., no physical anchors available from infrastructure; (iii) we are interested in recovering the absolute positions of the stationary nodes using location information of the AR device in the SLAM map. We have n stationary nodes in the network to be localized and m measurement instances which can be sampled during the surveying using the dynamic node. We tackle these problems as follows.

- Due to the self-localizing capability of the dynamic node, we remove the dynamic node, meanwhile insert a group of mobile "anchors" with known positions over a period of time to the network. We reinterpret this problem as a localization problem in a fully connected network with a total number of $N = n + m$ stationary nodes: Given the positions of the m nodes and the full Euclidean distance matrix, we localize the unknown positions of n nodes.
- Leveraging the mobility of the AR device, we can introduce an arbitrary number ($m \geq 4$) of "anchors" with diverse configuration into the network. Basically, we eliminate the requirement for the fixed and previously localized physical anchors by incorporating a self-localizing dynamic node.
- A straightforward way of estimating the absolute positions is performing a full SMACOF in N dimension followed by a *Procrustes Analysis* with the anchors. However, we observed two coupled drawbacks: (a) the distances across m anchors should not contribute to the stress function; (b) the search space in SMACOF increases from a dimension of $\mathbb{R}^{n\times 3}$ to $\mathbb{R}^{N\times 3}$ unnecessarily. We incorporate the idea of partitioning [14] to resolve these issues.

We now explain the specifics of the modified SMACOF. We separate the set of nodes into "unknown" ($\mathbf{X}_u$) and "anchors" ($\mathbf{X}_a$) partitions:

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_a \\ \mathbf{X}_u \end{bmatrix}, \quad \mathbf{Z} = \begin{bmatrix} \mathbf{Z}_a \\ \mathbf{Z}_u \end{bmatrix},$$

with,

$$\begin{aligned} \mathbf{X}_a &= [\mathbf{x}_1, \cdots, \mathbf{x}_n]^T \in \mathbb{R}^{n\times 3} \\ \mathbf{X}_u &= [\mathbf{x}_{n+1}, \cdots, \mathbf{x}_n + m]^T \in \mathbb{R}^{m\times 3} \\ \mathbf{Z}_a &= [\mathbf{Z}_1, \cdots, \mathbf{Z}_n]^T \in \mathbb{R}^{n\times 3} \\ \mathbf{Z}_u &= [\mathbf{Z}_{n+1}, \cdots, \mathbf{Z}_n + m]^T \in \mathbb{R}^{m\times 3} \end{aligned}$$

Similarly, we partition the weight matrix $\mathbf{W}$, as follows:

$$\mathbf{W} = \begin{bmatrix} \mathbf{W}_{11} & \mathbf{W}_{12} \\ \mathbf{W}_{21} & \mathbf{W}_{22}. \end{bmatrix},$$

where block matrices $\mathbf{W}_{11}$ is of size $n\times\ n$, $\mathbf{W}_{12} = \mathbf{W}_{21}^T$ is $n\times\ m$, $\mathbf{W}_{22}$ is $m\times\ m$. While $\mathbf{W}_{11}$ refers to weights of distances across $\mathbf{X}_u$ which are only measured once at first, $\mathbf{W}_{12}$ refers to the significance of the measurement instances between $\mathbf{X}_u$ and $\mathbf{X}_a$. Thus, we set elements in $\mathbf{W}_{11}$ as 1, and $\mathbf{W}_{12}$ as a larger number (e.g., 5). We then simplify $\mathrm{S}(\mathbf{X})$ by reducing $\mathbf{W}_{22}$ to $\mathbf{0}$ because distances among the anchors contribute nothing to the stress, followed by updating $\mathbf{V}$ and $\mathbf{B}(\mathbf{Z})$ accordingly. In the same way, we partition the auxiliary matrices $\mathbf{V}$ and $\mathbf{B}$ into block matrices. Further, we derive the partitioned $\mathrm{T}(\mathbf{X},\mathbf{Z})$, and differentiate it to solve the minimum of $\mathrm{T}(\mathbf{X},\mathbf{Z})$ [14]. Now we only account the nodes with unknown positions in the optimization procedure:

$$\mathbf{X}_u = \mathbf{V}_{22}^{-1}(\mathbf{B}_{22}\mathbf{Z}_u + \mathbf{B}_{12}^T\mathbf{Z}_a - \mathbf{V}_{12}^T\mathbf{X}_a). \tag{4}$$

We revise the Algorithm 1 with Eq. 4. In addition, we lower the computation complexity by splitting the matrices and reducing the dimensions. This is important for us, because (i) we need to deploy the algorithm on mobile devices; (ii) in our formulation, the number of the mobile anchors (m) can be arbitrarily large. Moreover, this way allows us to estimate the absolute positions in a single step manner by incorporating the anchors' absolute positions directly in the SMACOF procedure.

IMPLEMENTATION

Our prototype is composed of IoT controller modules, AR devices, firmware running on the microcontrollers (MCUs), and applications installed on the AR device. The AR device

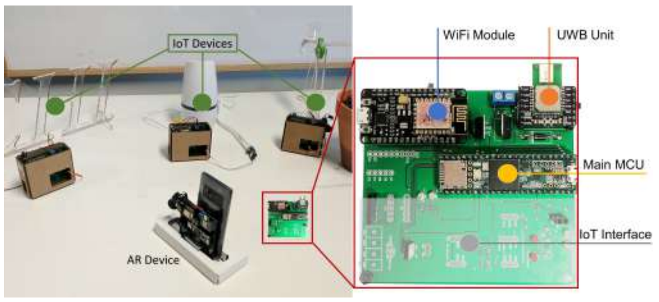

Figure 3. Overview of the *Scenariot* hardware. Deploy IoT controller boards (right) to IoT devices and AR device (left).

works as a host to handle the localization algorithm and interface with IoT devices. As shown in Figure 3, IoT controllers are deployed to smart things as well as to the AR device. All of the devices connect to a network through WiFi. Moreover, each IoT device is capable of measuring distances to the others. Note that we use off-the-shelf components and design the hardware as a development board for prototyping purposes. We believe the package size can be greatly improved after iteration. We developed the firmware and the mobile application with reliability as our primary goal at this stage. Thus, there is a lot of room for improvement in the efficiency.

Hardware

As shown in Figure 3, the overall size of the board is $100mm \times 100mm \times 20mm$ with the units installed in position. This board is designed to process distance measurements, deliver basic IoT functions, such as collecting sensor data and control appliances, and connecting with the smart environment network and the AR device over WiFi. The main MCU (Teensy 3.6) communicates with the DecaWave DWM1000 UWB module using SPI bus. Further it handles the WiFi communication by connecting a ESP8266 WiFi module (NodeMCU E12) via UART. The board also incorporates a set of general docking ports to interface with different IoT components such as sensors, and power relays. The board provides both 5V (1A max output) and 3.3V (1A max output) output from a rechargeable Li-ion battery (9V, 600mAh) using a dual regulator set. The battery lasts for ~ 1.5 hours with a continuous two-way WiFi communication and a UWB ranging. Our localization method works on mobile devices supporting a SLAM based AR environment. For our prototype, we adopted ZenFone AR (ZS571KL, Snapdrago$^{\text{TM}}$ 821 processor, Adreno$^{\text{TM}}$ 530 processor, 6GB RAM) which is embedded with Google Tango technology [4]. We attach one of the self-contained boards on the back of the phone. Together, they serve as the dynamic node in the wireless network.

Firmware

The firmware for the MCU is developed with the Teensyduino library and runs on an ARM Cortex-M4 chip (CPU speed $180M$Hz) that comes with the Teensy 3.6 board. The firmware mainly accomplishes the following tasks: (i) ranging to all available modules; (ii) connecting to a local area network through the WiFi module; (iii) communicating with the host AR device regarding localization and IoT function related messaging. Each MCU runs asynchronously with a tick function called from its own main loop and updates its state machine locally according to the tasks. We run a simple parsing and forwarding code on the NodeMCU chip after shaking hands with the main MCU. We support transmitting the distance data using User Datagram Protocol (UDP) via WiFi for high speed. We also support Transmission Control Protocol (TCP) if any IoT functionality requires large a file transmission.

Distance Measurements

We employ an asymmetrical double-sided two-way ranging scheme for time-of-flight ranging measurements between the IoT controller modules. This scheme is well known for correcting clock drift by exchanging two round-trip messages[23]. Although this approach is simple to implement, it works best for a small number of devices because it involves time-division multiplexing to range with multiple devices. We tune the tick timer in a conservative manner, which leads to an approximate upper bound of update rate $1000/(80+21n)$Hz for performing a *one to n ranging*. For our current prototype, we reach a ranging rate of ~ 3.7Hz when localizing a total number of 8 IoT devices at the same time. In this extreme condition, this update rate still allows users to move at a normal pace (~ 1m/s) without introducing many ranging errors.

Localization within AR

Our proposed localization method requires two groups of distance measurements: constant distances across all n stationary nodes, and continuous measurements from the dynamic node to the stationary nodes. We first poll a total number of $n(n-1)/2$ distance measurements across the IoT devices alternatively. Specifically, we perform $n-1$ times *one to i ranging*, where $i \in \{1, \cdots, n-1\}$. Then, during the surveying movements, the IoT module attached on the AR device collects the measurement instances and updates to the AR device using UDP. On Zenfone, the acquisition of the device position in the SLAM map is provide by the Google Tango API. We collect the position of the device when receiving a valid measurement instance. When the surveying ends, we launch the adapted SMACOF algorithm in a separate thread. Then after the algorithm terminates, we store the 3D location information of each connected device. For run time applications, we implement the proposed method on Zenfone. We developed the application within Unity3D [41] using C#. We employ an open source C# library Math.NET Numerics [33] to perform matrix calculations. To balance the computation resources and the localization accuracy, we empirically choose the number of samples from the surveying to be 100, the maximum iteration limit in SMACOF to be 500, and $\varepsilon = 1e-12$. This way, users spend less than 30s on the surveying. And running 500 iterations with 100 samples takes ≤ 10s to finish.

TECHNICAL EVALUATION

To analyze the performance of our localization method in terms of accuracy, we chose to evaluate our method under several possible surveying conditions. We illustrate the setup in Figure 4. We divided the surveying conditions into two levels. The primary conditions including surveying distance (r),

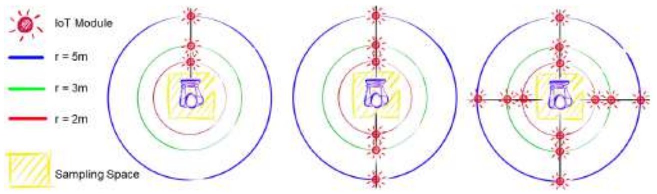

Figure 4. Technical evaluation setup.

i.e., the distance from the center of the surveying space to the devices to be located, and the number of devices (n) to be located, The secondary conditions included the surveying space, number of samples (m) collected in surveying, etc. We defined the surveying space using the axes (x,y,z) aligned bounding box ($l \times w \times h$) of the sample points, which is centered at the origin. Note, the origin of the SLAM map coordinate system was located at the point where the application launched. We launched the application with the phone being placed at a fixated location with a height of 1.5m (comparable to height of human body) above the floor.

In order to collect the data in a systematic manner, we decided to vary the primary conditions and fixate the secondary conditions when collecting the data. As shown in Figure 4, we conducted 9 surveyings to collect the surveying data with $r \in \{2,3,5\}$m and $n \in \{1,2,4\}$. For each surveying, we covered a sufficiently large survey space ($3 \times 3 \times 2$m) and collected 3000 samples. To achieve a uniform sampling as much as possible, we held the device at different heights and walked within the surveying region with different directions. Since the AR device is equipped with a depth camera, we manually tagged the center of the IoT module as ground truth locations. We recorded the position of the IoT modules relative to the instant AR device location and transformed it to the SLAM map coordinate system.

We first studied the effect of the secondary surveying conditions on the accuracy, by fixating the primary conditions. Based on the findings of the secondary conditions, we then evaluated the primary conditions and, designed studies with the suggested secondary conditions. For each studying test, we subsampled the dataset based on different conditions and fed the drawn samples to the localization algorithm. For evaluation purposes, we implemented the same algorithm with MATLAB and ran the algorithm on a desktop with a configuration of $\varepsilon = 1e-12$ and 500 maximum iterations. for all the experiments in this section. We used Root Mean Square Error (RMSE) between the localization results and the ground truth positions to indicate the accuracy.

Sampling Space

In order to gauge out the effect of the sampling space over the localization accuracy, we first assumed $l = w = h$, i.e., the surveying happening in a cube. We indicated the worst primary conditions as $r = 5$m, and $n = 4$, and the secondary condition as ($m = 100$). We varied $l = w = h = 1, 1.2, 1.4, 1.6, 1.8, 2$m to study the effect of the surveying space size on the accuracy of the localization. Then we randomly subsampled m points within the surveying space ($l \times w \times h$) from the overall dataset.

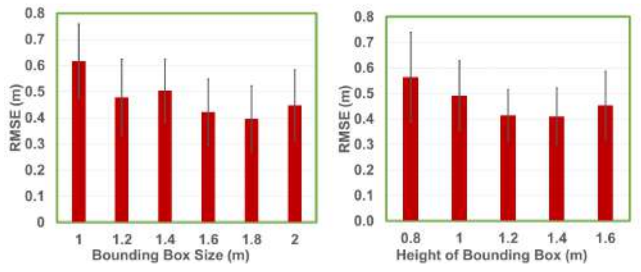

Figure 5. Effect of sampling space on the localization accuracy: (left)assume a cubic volume, (right) varying h and set $l = w = 1.6$(m).

We repeated the subsampling and localization 100 times for each of the variations. Then we took the average error of all 4 devices for the analysis. Last, we conducted a one-way univariate ANOVA and post hoc pairwise comparisons with Bonferroni correction. Overall, we found a significant difference across different survey space sizes ($p < 0.05$). Yet, within the set of $\{1.6, 1.8, 2\}$m, no significant difference was found ($p > 0.05$). As shown in Figure 6 (left), the mean error for $\{1.6, 1.8, 2\}$ was less than 0.5m which was less than 10% of the sampling distance $r = 5$m.

Second, reaching up to a large height limit involves an awkward motion. Taking into account a practical range of the arm motion without extra effort, we studied the effect of h on the accuracy with fixed l and w. Here, we varied $h = 0.8, 1.0, 1.2, 1.4, 1.6$m while fixating $l = w = 1.6$m. Other conditions remained the same as the first part. With the ANOVA test result, we found that there were significant differences across different height ranges, yet there were no significant differences across each other within the set of $\{1.2, 1.4, 1.6\}$m. From Figure 5 (right), we observed a mean error of 0.4m (SD = 0.1m) with a height range of 1.2m. To reach the range limit, an adult needs to fully stretch his/her arm up and down. On the other hand, even if there existed a degradation when $h \leq 1$m, we still observed a mean error ≤ 0.6m given the 5m sampling distance.

Sampling Number

We design the experiments in a similar way to the sampling space. We chose the primary condition as $r = 5$m, and $n = 4$, and the secondary condition as $l = w = h = 1.6$m, and vary the sampling numbers ($m = 20, 50, 100, 200, 300$) on the accuracy. From a one-way univariate ANOVA and post hoc pairwise comparisons, we concluded among $m = 100, 200, 300$ that there was no significant difference ($p > 0.05$), yet $m = 20, 50$ both showed significant differences with $m = 100, 200, 300$. From a computation efficiency point of view, we suggested a surveying with 100 sampling points.

Sampling Distance and Number of Devices

Based on studies on the secondary conditions, we set $l = w = h = 1.6$m and $m = 100$ to study the effect of sampling distance r and number of devices n. We designed this study with variations of $r = 2, 3, 5$m and $n = 1, 2, 4$. We calculated the average errors for the conditions with $n > 1$ and used them for the tests. We conducted a two-way univariate ANOVA

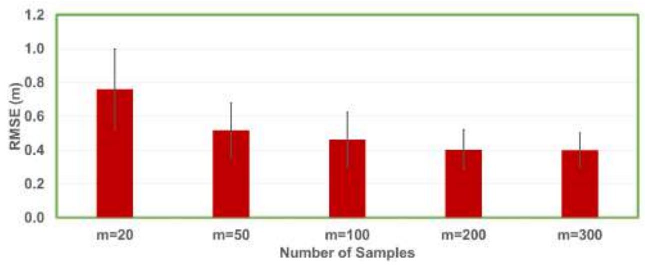

Figure 6. Effect of sampling Number (m) on the localization accuracy.

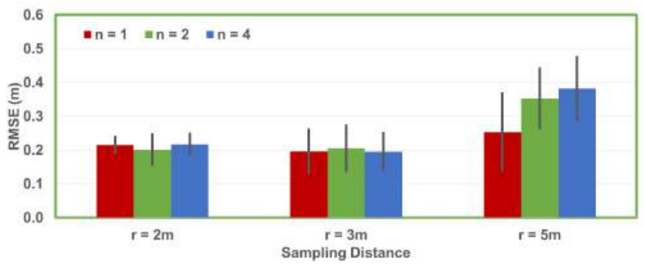

Figure 7. Effect of sampling distances (r) and number of devices (n) on the localization accuracy.

followed by post hoc pairwise comparisons. Overall, the ANOVA results indicated that both r and n were statistically significant ($p < 0.05$) over the accuracy. By examining the pairwise comparisons, we found out $n = 2$ and $n = 4$ showed no significant difference ($p > 0.05$) and that both of them were significantly different from $n = 1$. As shown in Figure 7, with the condition of $r = 5$m, $n = 2$ and $n = 4$ presented a larger error (> 0.3m). We observed that there was no significant difference between $r = 2$m and $r = 3$m, yet $r = 5$m yielded a significant difference from the others. From the figure, we confirmed that the mean errors of localizations at $r = 5$m increased but still remained < 0.4m.

Guidelines

From the study results, we summarized the following preliminary guidelines on utilizing the localization: (i) the surveying space should be sufficiently large ($l \geq 1.6$m, $w \geq 1.6$m, $h \geq 1.2$m); (ii) enough data should be sampled during surveying ($m \geq 100$); (iii) localization of multiple devices is feasible but likely to introduce more errors; (iv) the localization error increases as the IoT devices are located further from the survey region; (v)within a room of normal size($< 10 \times 10$m), surveying at the center of the room should localize the scattered devices with an average error at the level of $0.4m$ or less. With these guidelines, we further designed task evaluations and demonstration applications to verify our proposed method. Note that this technical evaluation was conducted with limited resources and so maybe less conclusive. This is why we suggest these guidelines conservatively.

Further, with the relative spatial relationships between the user and the IoT device, we extracted three basic spatial elements: the *orientation* of users with respect to the IoT devices, the direct *distance* measurement between a user and an IoT device, the *approaching direction* in which users walks. Based on these three relationships, we design and implement two location aware interactions, namely, distant pointing and proximity based control [25, 36]. In Task Evaluation section, we study the performance of these two widely accepted spatial interactions with users using our localization method.

TASK EVALUATION

Through the task evaluation, we expected to: (i) verify the localization performance with users in a realistic scene; (ii) examine whether the localization performance meets the requirements of the spatial interactions in AR. We deployed 8

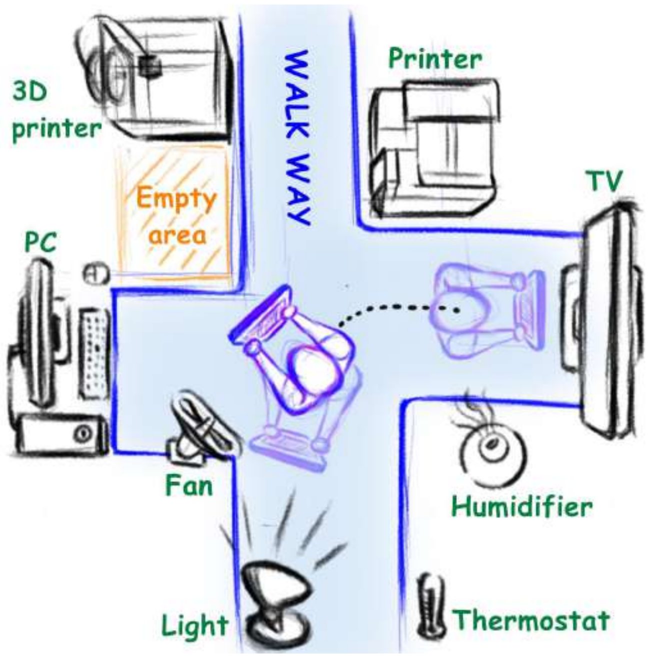

Figure 8. Task evaluation setup with 8 IoT devices in an office.

IoT controller modules onto 8 physical appliances in a cluttered office environment as illustrated in Figure 8. They were distributed within a region with a footprint of $\sim 10 \times 8$m at various heights. We also kept the existing common furnitures such as desks, shelves, and chairs in the testing area. Within this setup environment, we tested the localization accuracy by asking users to perform the surveying. After the surveying and the localization, we asked users to conduct these interactions. We then evaluated the performance in terms of targeting accuracy and completion time.

We recruited 11 participants with an average age of 25 for our study. Each user was asked to conduct a two-session study regarding the distant pointing and proximity based control respectively. Each session included 3 subtasks, where users first performed surveying movements then acted the designated interactions. Prior to the trial tests, we offered users a practice session to familiarize them with the system. We gave users a 5 minutes break between each session.

Localization Accuracy

For all 6 subtasks, users were asked to first perform surveying movements around the center area of the setup environment. We collected 6 sets of surveying movement trajectories and runtime localization results from each user, which resulted in 66 trials in total. After the surveying, the author manually tagged the ground truth positions of each IoT module as in the experiments in *Technical Evaluation* section. We displayed 4 progress bars on the screen to indicate the sampling number collected in the survey, as well as the expansion of the surveying space on 3 axes (x, y, z). We asked users to reach a minimum expansion of $l = w = 1.6$m, and $h = 1.2$m which was suggested by the technical evaluations. We did not ask the users to follow any specific trajectory as we tried to find out possible performance degradations in the realistic scenarios.

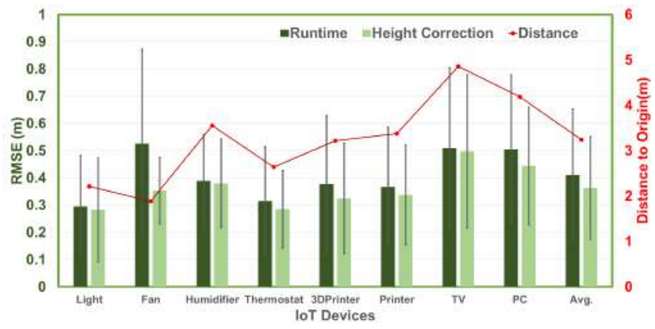

Figure 9. Localization accuracy study with users. Runtime: runtime localization result. Height Correction: results with height correction. Distance: the distances of IoT devices to the center of the surveying space.

Result As shown in Figure 9, the average of the localization error over all 8 devices yielded 0.41m (SD=0.24m). We expected this result based on the technical evaluation results. We ran a one-way ANOVA to find out if the localization accuracy was similar across different devices, and the result indicated the existence of significant differences. Further a post hoc pairwise test showed that the accuracies over Fan, TV, and PC were significantly different from those of the others. The TV and PC were placed at an extreme distance from the setup environment resulting in their being $\sim$ 5m and 4m away from the surveying region. As the distance increases, the localization performance may go down. Although the Fan was placed near the center ($\sim$ 2m), we deliberately left it on the floor under a desk. We suspect that the possible occlusion caused by the placement affected the localization performance.

We recalled that in the technical evaluations, the 100 samples were uniformly subsampled from the dataset. However, in real trials, we observed that users tended not to move much on the z axis. Sometimes if most of the sampled points lay approximately on a plane (horizontal), the flip ambiguity became more severe [5]. We inspected the collected data and found that flipping about an approximately horizontal plane happened occasionally which introduced an error on the z axis mainly. We observed that the average localization error on z (0.32m) was larger than on the other two axes (0.13, 0.15m).

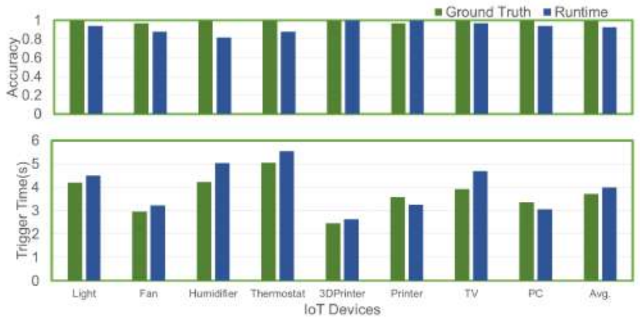

Figure 10. Distant pointing accuracy and completion time.

Here, we implemented a heuristic leveraging some meta information about the devices to compensate the error caused by flipping. We dissected 3 height levels with respect to the floor, namely, lower ($0 \leq z \leq 1$m), middle ($1 < z \leq 2$m), and upper ($z > 2$m). We designated the IoT devices in this way: lower (Fan), middle (Humidifer, 3D Printer, Printer, and PC), and upper (Light, Thermostat, and TV). Compared with the runtime results in Figure 9, the overall average error decreased to 0.36m (SD = 0.19m). The T-Test showed that there was a significant difference between the runtime result and the one with the heuristic ($p < 0.05$) and thus indicated a decreasing trend on the errors over all 8 IoT devices.

Distant Pointing

Distant pointing leverages the orientation of the AR camera and detects if the object of interest is located in the center of the view window. We placed a virtual spherical collider at the location of the IoT module. Next, we dissected the spherical colliders by diameters (d) into three groups: small ($d = 0.5$m), medium ($d = 1$m), and large ($d = 1.5$m). Then we categorize the corresponding 8 physical devices based on their physical sizes: the PC, and TV as large, the 3D Printer, and Printer as medium, and the rest as small. We implemented a pointing scheme which performs AABB collision tests with a viewfinder frustum (8 degrees [3]) over the colliders.

Within each subtask, we generated a randomized sequence, where each IoT device appeared twice in the sequence. We randomly assigned the ground truth position or the runtime localization result to the colliders. For each trial, user oriented the device towards the objects sequentially one by one. We suggested the users that they perform the distant pointing around the center of the setup environment though we do not limit their movements. We asked the user to place the whole physical device at the center of the view as we applied an offset to compensate for the deployment displacement. We counted a device as being triggered if user pointed to the correct device within 10s and dwelled for over 1s. We counted a negative trigger if during the dwelling time, any other device mis-triggered as well. After each trial, we asked the user to fully disengage with all of the objects and point to some empty space as shown in Figure 8. In total, we collected $8 \times 2 \times 3$ distant pointing trials from each user, which resulted in 528 trials across all users.

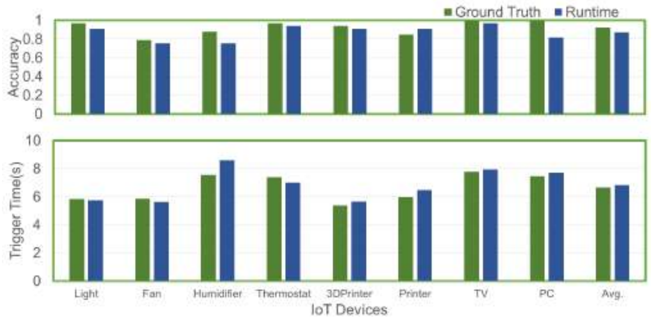

Figure 11. Proximity based control accuracy and completion time.

Result As shown in Figure 10, we observed an average of 0.99 pointing accuracy with ground truth. We achieved an average of 0.93 accuracy with runtime result, within which the Fan, Humidifier, and Thermostat had an accuracy less than 0.9. Compared to the localization accuracy shown in Figure 9, we suspect that the accuracy degradation was not just caused by the localization accuracy. We conjectured some potential reasons without verification: awkward installment positions, extra cognitive load from the cluttered scene, and selection ambiguities. For examples, the Fan was placed on the floor and the Thermostat was hanging around the ceiling, and the white Humidifier was hidden in a cluttered scene. In terms of completion time, we only counted the successful trials, and a T-Test showed there was no significant difference ($p > 0.05$).

Proximity based Control

Based on the proximic control framework proposed in a previous paper [25], we used three spatial elements for a proximity based interaction: *orientation*, *distance*, and *approaching direction*. The triggering conditions included facing towards, approaching towards and reaching into the proximity region of the hinted IoT device. The trial procedure was similar to that of Distant Pointing. We set a timeout limit of 15s, and we asked the users to return to approximately the same position to disengage from all of the objects.

Result The analysis showed an overall triggering accuracy of 0.92 with ground truth while 0.87 with runtime result. The ground truth accuracy suggested that we need to improve the interaction scheme. A paired T-Test on the accuracy between these two conditions indicated no significant difference($p > 0.05$). The accuracy with the results on both the Fan and the Humidifier were worse than others (< 0.8). We observed that users had unnatural motions, including bending towards the Fan and detouring before approaching the Humidifier. Therefore, we need to adjust the interaction design according to the possible obstacles in the way of the target and the height of the object located. For the completion time, the T-Test showed no significant difference between the ground truth and the runtime conditions ($p > 0.05$).

EXAMPLE USE CASES

Based on the localization result, we register the IoT devices spatially in the AR scene which empowers the IoT devices to have the spatial awareness of the physical world. We foresee

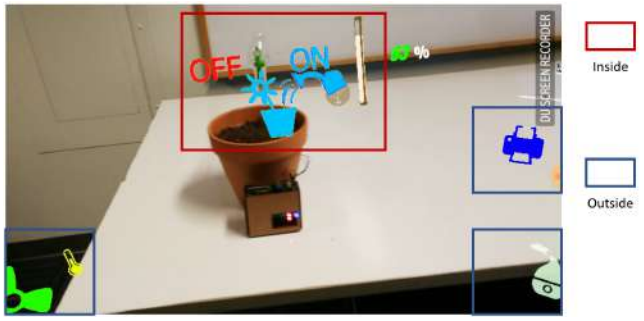

Figure 12. Discoverable World. The digital representations of the discovered IoT devices are visualized within the AR scene with spatial PiPs.

good potential of flexibility and applicability using *Scenariot*. Here we selectively deployed *Scenariot* in 4 use cases.

Discoverable World

When a user enters a new environment, the AR device broadcasts a discovery message to the network then all connected devices send an acknowledgement and register with their identities. After the user localizes the IoT devices, the digital interfaces will be relocated to the discovered 3D positions. Users can simply browse the digitally enhanced world within the augmented scene. Inspired by previous works [19, 27], we further deliver a spatial aware picture-in-picture (PiP) effect. As shown in Figure 1 and Figure 12, we not only visualize the digital interfaces when the corresponding physical object is located inside the view, but also the ones outside. To achieve this effect, we parameterize the outside view space using spherical coordinates and shift the outside locations to the peripheral region of the view frustum. This way, we preserve the spatial information of the outside view devices.

Proximity based Control

This interaction scheme has been studied in the Task Evaluation section also. We here demonstrate *Scenariot* being used for fabrication machine inspections as shown in Figure 13. Users approach the machine to examine the status or operate it through the AR interface. As users move closer to the target, the digital interface adjusts according to the distances for different levels of engagement [25].

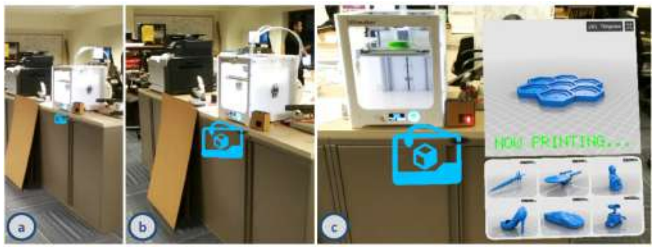

Figure 13. Proximity based Control. While users move closer to the machine (a, b, c), the level of engagement is adjusted accordingly.

Monitoring Assets and Navigation

By attaching our IoT module to assets, we store the 3D locations of the assets together with the map created by the AR

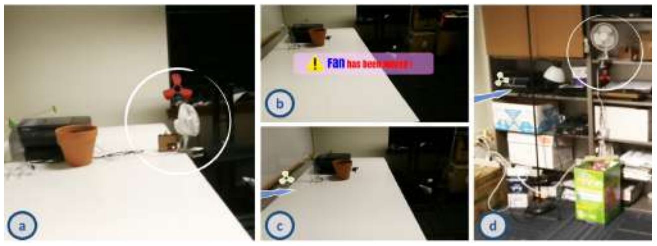

Figure 14. Monitoring the IoT assets (a, b) and navigating the user towards the assets by visualizing the direction on the screen(c, d).

device for later revisit usages as in Figure 14 (a). When the user reenter one of the discovered scenes, we check the distance measurements across the IoT devices and/or between the IoT device and the AR device. If they do not match with the calculations based on the last location records, we consider the IoT devices as having shifted from their original locations. We provide suggestions for the user to conduct a new surveying(Figure 14 (b)). After the new locations are discovered, we navigate users towards the new position by showing them an direction indicator on the AR device (Figure 14 (c, d)).

Miniature World

We consider another spatially aware interaction scheme for remote interaction with IoT devices, namely a miniature world [8]. With the depth camera equipped with Zenfone, we allow users to scan and reconstruct the mesh model of the surrounding environment. We can combine the surveying stage with scanning movements. With the discovered 3D location of the IoT devices, we superimpose the digital interfaces onto the virtual models. To this end, we develop an IoT-device-enhanced miniature world. Users can further interact with the miniature world to control the physical world.

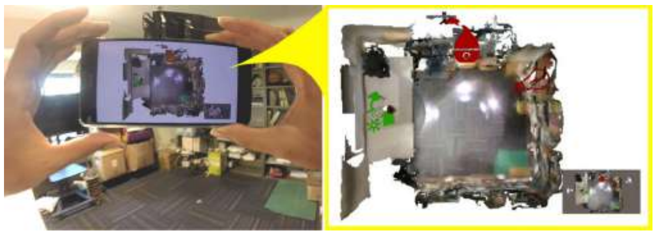

Figure 15. User creates a miniature world of the physical environment enhanced by the digital interfaces of IoT devices.

LIMITATION

Number of IoT Devices One essential bottleneck is the sampling rate of the distance sensing. With the current conservative parameters for 2-way ranging, we estimate the upper bound can be around 10 ($\sim$ 3.6Hz) given the assumption that the user moves at a speed of $\sim$ 1m/s. We consider to employ synchronized manner which only needs 1 message for distance measurement to increase the sampling rate [1]. Without strict proof, we expect to double the ranging frequency approximately as it requires only 1 round trip message instead of 2, thus doubling the bound (e.g., 20). For an even larger network, we need to localize the nodes patches by patches.

Multi-User The low update rate of UWB ranging also limits our prototype to working with single user only. Directly using the current implementation parallelly for a multi-user system will reduce the update rate by a fraction of the number of users (e.g. 1.85Hz for 2 users with 8 IoT). Instead, we plan to solve the dynamic registration between two users so that we can treat one user as host node, and other users as slave nodes.

Instrumenting To optimize the package in the future, we plan to remove the breakout boards, redesign the PCB boards, and replace the universal IoT interfaces with the desired interface only. Moreover, it will be interesting to generalize our concept of combining SLAM and UWB and replace the UWB with matured RF based technology.

DISCUSSION AND FUTURE WORK

Improving Localization Algorithm Currently, we assume a fully connected network, which means that the distances across all devices are available. However, in larger scale, this assumption may be invalid. In the future, we need to further evaluate the effect of missing distance measurements on the localization accuracy. For example, we can build a ranging quality metric mechanism to adjust the parameters (e.g., weights) during runtime. Further, heavy non-line-of-sight (NLOS) situations such as crossing walls needs to be identified and properly compensated [43] for a better accuracy. Moreover, we developed a heuristic for the flip disambiguation, yet we need a more general solution to resolve this problem [5].

Accuracy As the localization accuracies along x and y axes were 0.13m and 0.15m, we can still identify two adjacent devices if there are sufficient amount of differences along x and y axes. Further, we can introduce a mechanism with the users' input to distinguish two closely placed objects.

Power Consumption Given a full working condition in the localization phase with continuous transceiving, the whole board (including MCU) peak current reaches 350mA which is calculated based on the datasheet. It agrees with our reported results: a 600mAh battery lasts for $\sim$ 1.5 hours which means we can perform surveying ($\sim$ 30s) about 180 times. After localization, we keep DWM1000 in sleep mode (550nA) so that the battery can last substantially.

Scalability The current centralized localization approach may suffer high computation costs for a larger scale deployment (e.g.,factories). We are also considering implementing our method in a distributed way for large scale adoption. Technically, visual SLAM and UWB should both work in outdoor environment, yet for this paper, we only test in indoor environment. We would like to expand the study to outdoor setup in the future. Moreover, in real use scenarios, we need to address the heterogeneous interfaces with different IoT devices.

More Spatial Interaction Metaphors We envision advanced inter-devices interactions can be realized with the given spatial information. Currently, we support a single AR device to localize multiple stationary devices. In the future, we will consider to including multiple AR devices and dynamic IoT devices such as a service robot. Moreover, with the discovered locations, we can form an infrastructure based tracking by opportunistically referring the IoT devices (more than three) as anchors [34]. Further, we plan to incorporate multiple modalities for interacting with the smart environment. For examples, we can leverage the spatial relationships to provide both visual and auditory augmentation [37], and context-awareness with voice command [30].

Form factor of AR device For prototyping purposes, we use Google Tango devices which are specially designed and embedded with SLAM. Yet our localization method can be deployed to any moderate smartphones/tablets which are compatible with third party SLAM based AR SDK (e.g., Wikitude [46], ARCore [18], etc). Further, integrating *Scenariot* with the emerging head mounted display based AR devices, e.g., Hololens [29] is another alternative.

CONCLUSION

Our paper builds towards the broad goal of empowering users with the ability to quickly discover and intuitively interact with the connected smart things within the surrounding environment. We propose *Scenariot* to discover and localize the surrounding smart things as well as spatially register them with a SLAM based mobile AR device. By leveraging the spatial registration, in-situ AR interaction with the IoT devices is enabled. Through our experiments and user studies, we verified our method is capable of providing object level localization accuracy ~ 0.4m with multiple devices distributed in a cluttered scene ($\sim 10 \times 10$m). Therefore, we believe this work can bring spatial awareness to the IoT devices within an AR scene and further inspire advanced interaction designs.

ACKNOWLEDGEMENT

We thank the reviewers for their valuable feedback and comments. This work was supported partially by the NSF Awards No. 1538868 (CMMI) and 1632154 (IIP). We also acknowledge the Feddersen chair funding for early work. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

1. Abdulrahman Alarifi, AbdulMalik Al-Salman, Mansour Alsaleh, Ahmad Alnafessah, Suheer Al-Hadhrami, Mai A Al-Ammar, and Hend S Al-Khalifa. 2016. Ultra wideband indoor positioning technologies: Analysis and recent advances. *Sensors* 16, 5 (2016), 707.
2. Isaac Amundson and Xenofon Koutsoukos. 2009. A survey on localization for mobile wireless sensor networks. *Mobile entity localization and tracking in GPS-less environnments* (2009), 235–254.
3. Ferran Argelaguet and Carlos Andujar. 2009. Visual feedback techniques for virtual pointing on stereoscopic displays. In *Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology*. ACM, 163–170.
4. ASUS. 2017. ZenFone-AR. (2017). Retrieved September 1, 2017 from = https://www.asus.com/us/Phone/ZenFone-AR-ZS571KL/.
5. Shuanghua Bai and Houduo Qi. 2016. Tackling the flip ambiguity in wireless sensor network localization and beyond. *Digital Signal Processing* 55 (2016), 85–97.
6. Till Ballendat, Nicolai Marquardt, and Saul Greenberg. 2010. Proxemic interaction: designing for a proximity and orientation-aware environment. In *ACM International Conference on Interactive Tabletops and Surfaces*. ACM, 121–130.
7. Ingwer Borg and Patrick JF Groenen. 2005. *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media.
8. Sebastian Boring, Dominikus Baur, Andreas Butz, Sean Gustafson, and Patrick Baudisch. 2010. Touch projector: mobile interaction through video. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2287–2296.
9. Barry Brumitt, John Krumm, Brian Meyers, and Steven Shafer. 2000. Ubiquitous computing and the role of geometry. *IEEE Personal Communications* 7, 5 (2000), 41–43.
10. Yu-Hsiang Chen, Ben Zhang, Claire Tuna, Yang Li, Edward A Lee, and Bjorn Hartmann. 2013. *A context menu for the real world: Controlling physical appliances through head-worn infrared targeting*. Technical Report. CALIFORNIA UNIV BERKELEY DEPT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCES.
11. Jose A Costa, Neal Patwari, and Alfred O Hero III. 2006. Distributed weighted-multidimensional scaling for node localization in sensor networks. *ACM Transactions on Sensor Networks (TOSN)* 2, 1 (2006), 39–64.
12. Adrian A de Freitas, Michael Nebeling, Xiang'Anthony' Chen, Junrui Yang, Akshaye Shreenithi Kirupa Karthikeyan Ranithangam, and Anind K Dey. 2016. Snap-to-it: A user-inspired platform for opportunistic device interactions. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 5909–5920.
13. Jan De Leeuw and Patrick Mair. 2011. Multidimensional scaling using majorization: SMACOF in R. *Department of Statistics, UCLA* (2011).
14. Carmelo Di Franco, Enrico Bini, Mauro Marinoni, and Giorgio C Buttazzo. 2017a. Multidimensional scaling localization with anchors. In *Autonomous Robot Systems and Competitions (ICARSC), 2017 IEEE International Conference on*. IEEE, 49–54.
15. Carmelo Di Franco, Amanda Prorok, Nikolay Atanasov, Benjamin P Kempke, Prabal Dutta, Vijay Kumar, and George J Pappas. 2017b. Calibration-free network localization using non-line-of-sight ultra-wideband measurements.. In *IPSN*. 235–246.

16. Ivan Dokmanic, Reza Parhizkar, Juri Ranieri, and Martin Vetterli. 2015. Euclidean distance matrices: Essential theory, algorithms, and applications. *IEEE Signal Processing Magazine* 32, 6 (2015), 12–30.

17. Hans Gellersen, Carl Fischer, Dominique Guinard, Roswitha Gostner, Gerd Kortuem, Christian Kray, Enrico Rukzio, and Sara Streng. 2009. Supporting device discovery and spontaneous interaction with spatial references. *Personal and Ubiquitous Computing* 13, 4 (2009), 255–264.

18. Google. 2017. ARCore. (2017). Retrieved September 1, 2017 from = https://developers.google.com/ar/.

19. Uwe Gruenefeld, Abdallah El Ali, Wilko Heuten, and Susanne Boll. 2017. Visualizing out-of-view objects in head-mounted augmented reality. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services*. ACM, 81.

20. Anhong Guo, Jeeeun Kim, Xiang'Anthony' Chen, Tom Yeh, Scott E Hudson, Jennifer Mankoff, and Jeffrey P Bigham. 2017. Facade: Auto-generating tactile interfaces to appliances. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 5826–5838.

21. Dirk Hahnel, Wolfram Burgard, Dieter Fox, Ken Fishkin, and Matthai Philipose. 2004. Mapping and localization with RFID technology. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, Vol. 1. IEEE, 1015–1020.

22. Valentin Heun, James Hobin, and Pattie Maes. 2013. Reality editor: programming smarter objects. In *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication*. ACM, 307–310.

23. Yi Jiang and Victor CM Leung. 2007. An asymmetric double sided two-way ranging for crystal offset. In *Signals, Systems and Electronics, 2007. ISSSE'07. International Symposium on*. IEEE, 525–528.

24. Gierad Laput, Chouchang Yang, Robert Xiao, Alanson Sample, and Chris Harrison. 2015. Em-sense: Touch recognition of uninstrumented, electrical and electromechanical objects. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. ACM, 157–166.

25. David Ledo, Saul Greenberg, Nicolai Marquardt, and Sebastian Boring. 2015. Proxemic-aware controls: Designing remote controls for ubiquitous computing ecologies. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*. ACM, 187–198.

26. Sikun Lin, Hao Fei Cheng, Weikai Li, Zhanpeng Huang, Pan Hui, and Christoph Peylo. 2017a. Ubii: Physical World Interaction Through Augmented Reality. *IEEE Transactions on Mobile Computing* 16, 3 (2017), 872–885.

27. Yung-Ta Lin, Yi-Chi Liao, Shan-Yuan Teng, Yi-Ju Chung, Liwei Chan, and Bing-Yu Chen. 2017b. Outside-In: Visualizing Out-of-Sight Regions-of-Interest in a 360 Video Using Spatial Picture-in-Picture Previews. In *Proceedings of the 30th Annual Symposium on User Interface Software and Technology*. ACM, –.

28. Simon Mayer, Markus Schalch, Marian George, and Gábor Sörös. 2013. Device recognition for intuitive interaction with the web of things. In *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication*. ACM, 239–242.

29. Microsoft. 2017. Hololens. (2017). Retrieved September 1, 2017 from = https://www.microsoft.com/en-us/hololens.

30. Gang Pan, Jiahui Wu, Daqing Zhang, Zhaohui Wu, Yingchun Yang, and Shijian Li. 2010. GeeAir: a universal multimodal remote control device for home appliances. *Personal and Ubiquitous Computing* 14, 8 (2010), 723–735.

31. Charith Perera, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. 2014. Context aware computing for the internet of things: A survey. *IEEE Communications Surveys & Tutorials* 16, 1 (2014), 414–454.

32. Sudeep Pillai and John Leonard. 2015. Monocular slam supported object recognition. *arXiv preprint arXiv:1506.01732* (2015).

33. Math.Net Project. 2017. Math.Net Numerics. (2017). Retrieved September 1, 2017 from = https://numerics.mathdotnet.com/.

34. Kun Qian, Chenshu Wu, Zimu Zhou, Yue Zheng, Zheng Yang, and Yunhao Liu. 2017. Inferring motion direction using commodity wi-fi for interactive exergames. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 1961–1972.

35. Jun Rekimoto, Yuji Ayatsuka, Michimune Kohno, and Haruo Oba. 2003. Proximal Interactions: A Direct Manipulation Technique for Wireless Networking.. In *Interact*, Vol. 3. 511–518.

36. Enrico Rukzio, Karin Leichtenstern, Vic Callaghan, Paul Holleis, Albrecht Schmidt, and Jeannette Chin. 2006. An experimental comparison of physical mobile interaction techniques: Touching, pointing and scanning. *UbiComp 2006: Ubiquitous Computing* (2006), 87–104.

37. Spencer Russell, Gershon Dublon, and Joseph A Paradiso. 2016. Hearthere: Networked sensory prosthetics through auditory augmented reality. In *Proceedings of the 7th Augmented Human International Conference 2016*. ACM, 20.

38. Renato F Salas-Moreno, Richard A Newcombe, Hauke Strasdat, Paul HJ Kelly, and Andrew J Davison. 2013. Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1352–1359.

39. Dominik Schmidt, David Molyneaux, and Xiang Cao. 2012. PICOntrol: using a handheld projector for direct control of physical devices through visible light. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*. ACM, 379–388.

40. Eldon Schoop, Michelle Nguyen, Daniel Lim, Valkyrie Savage, Sean Follmer, and Björn Hartmann. 2016. Drill Sergeant: Supporting physical construction projects through an ecosystem of augmented tools. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. ACM, 1607–1614.

41. Unity3D. 2017. Unity3D. (2017). Retrieved September 1, 2017 from = https://unity3d.com/.

42. Pasi Välkkynen and Timo Tuomisto. 2005. Physical Browsing Research. *PERMID* 2005 (2005), 35–38.

43. S Venkatesh and RM Buehrer. 2007. Non-line-of-sight identification in ultra-wideband systems based on received signal statistics. *IET Microwaves, Antennas & Propagation* 1, 6 (2007), 1120–1130.

44. Edward J Wang, Tien-Jui Lee, Alex Mariakakis, Mayank Goel, Sidhant Gupta, and Shwetak N Patel. 2015. Magnifisense: Inferring device interaction using wrist-worn passive magneto-inductive sensors. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 15–26.

45. Roy Want, Kenneth P Fishkin, Anuj Gujar, and Beverly L Harrison. 1999. Bridging physical and virtual worlds with electronic tags. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. ACM, 370–377.

46. wikitude. 2017. wikitude. (2017). Retrieved September 1, 2017 from = https://www.wikitude.com/.

47. Robert Xiao, Gierad Laput, Yang Zhang, and Chris Harrison. 2017. Deus EM Machina: On-Touch Contextual Functionality for Smart IoT Appliances. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 4000–4008.

48. Sang Ho Yoon, Yunbo Zhang, Ke Huo, and Karthik Ramani. 2016. TRing: Instant and Customizable Interactions with Objects Using an Embedded Magnet and a Finger-Worn Device. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. ACM, 169–181.