V.Ra: An In-Situ Visual Authoring System for Robot-IoT Task Planning with Augmented Reality

Yuanzhi Cao, Zhuangying Xu, †Fan Li, Wentao Zhong, Ke Huo, Karthik Ramani*
School of Mechanical Engineering, Purdue University, West Lafayette, IN 47907 USA
[cao158, xu970, zhong133, khuo, ramani]@purdue.edu
†School of Mechanical Engineering, Tsinghua University, Beijing, China
li-f15@mails.tsinghua.edu.cn

ABSTRACT
We present V.Ra, a visual and spatial programming system for robot-IoT task authoring. In V.Ra, programmable mobile robots serve as binding agents to link the stationary IoTs and perform collaborative tasks. We establish an ecosystem that coherently connects the three key elements of robot task planning , the human, robot and IoT, with one single mobile AR device. Users can perform task authoring with the Augmented Reality (AR) handheld interface, then placing the AR device onto the mobile robot directly transfers the task plan in a what-you-do-is-what-robot-does (WYDWRD) manner. The mobile device mediates the interactions between the user, robot, and the IoT oriented tasks, and guides the path planning execution with the embedded simultaneous localization and mapping (SLAM) capability. We demonstrate that V.Ra enables instant, robust and intuitive room-scale navigatory and interactive task authoring through various use cases and preliminary studies.

Author Keywords
Robotic task authoring, Human-robot interaction, Augmented Reality, SLAM, Internet-of-Robotic-thing, Robot navigation.

CCS Concepts
•**Computer systems organization** → **Embedded and cyber-physical systems;** *Robotics;* •**Information systems** → **Information systems applications;** *Multimedia content creation;*

INTRODUCTION
The vision of *ubiquitous computing* has been emerging rapidly as the Internet of Things (IoT) based electronics are getting smaller, lower in cost, proliferating and being embedded in our everyday environment. Typically, human-IoT interactions take the form of transforming IoT data into informative knowledge, augmenting human sensory capabilities, and assisting humans to make correct and efficient decisions [5]. However, the IoT devices are mostly stationary and have limited physical interactions particularly with each other. In conjunction, the concept of Internet of Robotic Things (IoRT) has not been widely explored in practice across the IoT and robotics communities [39], and an authoring system for such robot-IoT interactive task planning is underdeveloped [44]. We envision the emergence of programmable mobile robots in a near future to serve as key medium to conduct coordinated and collaborative tasks with surrounding IoTs. In this vision, the mobile robots are combined with the embedded multiple stationary IoTs to create new types of workflows and in addition also extend humans' motor capabilities.

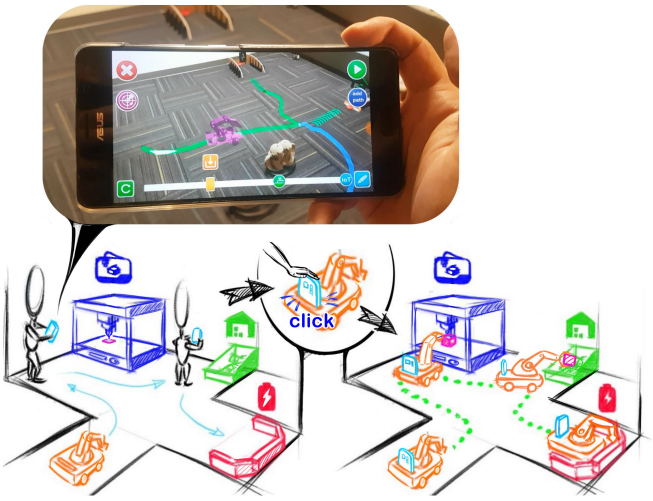

Figure 1. V.Ra workflow. Using an mobile AR mobile device, the user first spatially plans the task with the AR interface, then places the device onto the mobile robot for execution.

Current user interfaces are often designated to either IoT or robots only, without considering the robot-IoT interactions. Contemporary IoT devices allow access and control through offloaded mobile interfaces. With additional web-based services such as IFTTT [4], users can also coordinate multiple devices working with other productivity tools or social medias via active human-IoT communication [5, 10]. Even in these coordinated works, the IoT tasks are rather spatially independent. In these cases, conventional graphical user interfaces (GUI) mostly suffice the IoT-only interactions which are insensitive

*School of Electrical and Computer Engineering (by courtesy)

DIS '19, June 23–28, 2019, San Diego, CA, USA
© 2019 ACM. ISBN 978-1-4503-5850-7/19/06…$15.00
DOI: https://doi.org/10.1145/3322276.3322278

to their spatial distributions. In contrast, to command mobile robots to complete distributed tasks, the significance of spatial-awareness for authoring interfaces becomes more relevant, especially for ad-hoc tasks in less controlled environments.

The emerging augmented reality (AR) shows promise towards bridging the gap and interfacing with the physical world. In fact, AR interfaces have been introduced for IoT and robots respectively. For example, Reality Editor allows users to visually program the stationary IoT devices which are affixed with fiducial markers [24]. Similarly, robots have been attached with tags and tracked through the users' AR camera view [9, 27, 35]. However, the robots and the IoTs remain locally registered in the AR only, e.g., to resolve the spatial relationship between a robot and an IoT, a user has to *keep* both of them in the same AR camera view. To register multiple agents globally and coordinate them spatially, some alternatives including external tracking systems (e.g., infrastructured cameras [19, 22, 46, 26]) and pre-scanned and manually tagged environment maps [38, 31, 11] have been proposed. But these approaches further constrain deploying robots to ad-hoc daily tasks.

On the other hand, we leverage the advancing mobile SLAM techniques to globally associate the user, IoTs, and robots together. Users first freely examine and explore the IoT environment within a mobile AR scene, then utilize the embedded SLAM capability to seamlessly transfer their insight about the tasks to the mobile robot. These insights are tightly coupled with the environmental factors such as the path planning, as well as the semantic knowledge of the IoTs. Further, SLAM also enables a novel embodied programming modality, namely, users demonstrate a sequential tasks to the robots by physically walking the navigatory path. In addition, since both AR authoring interface and the robots' navigation share large commonalities of environmental spatial awareness, we support a smooth exchange of human knowledge to the mobile robot by simply placing the AR-SLAM device onto the robot, as its 'eyes' and 'brain'. The robot now has perceptive knowledge of the physical environment, the interactive knowledge for the IoTs, and is ready to execute the planned task from the user. We envision our system be useful for in-situ robot assistant programming in highly unstructured and ever changing situations, such as household environments, labs/offices, and fabrication workshops. Our mobile phone based interface allows for intuitive visual programming for novice user, thus greatly lowers the cognitive load and encourages broader impact. To this end, we present V.Ra (Virtual Robotic assistant), an in-situ authoring interface for robot-IoT task planning using one single mobile AR device. The key contributions of this paper are as follows:

1. **V.Ra workflow** that uses one AR device for in-situ mobile robot-IoT task authoring and execution with the on-the-fly generated SLAM map, so that the human-robot-IoT interaction is bonded together synergistically, without an external tracking infrastructure or pre-scanned map information.

2. **Authoring interface design** that utilizes contextual visual feedback and spatial awareness of AR, to enable intuitive robot programming for path planning, logic driven event scheduling, in-situ virtual simulation, as well as knowledge transferring from human to the robots.

3. **Use cases and evaluations** demonstrating and verifying that V.Ra supports robust room-scale household navigatory and fluid interactive task authoring with our system.

RELATED WORK

Workflow of Human-Robot System

Due to the limited on-board perception capabilities and underdeveloped artificial intelligence (AI), the ad-hoc tasks in our daily environment which we take for granted are still challenging for robots [28]. Thus, before it comes to an era of full autonomy and high level AI, a well-design functional human-robot interface is the key to author the domestic robots to accomplish any useful tasks. Within the authoring interface, users need to be spatially aware of the physical environment and the mobile robots. Previous works introduced an external camera system to track the robots and fed the live view to the interface [22, 34, 26, 19, 40]. However, this approach limits the authoring scene to the perspective of the camera only, which is usually fixed. In contrast, *Magic Cards* proposed an implicit command authoring workflow with human manually and spatially placing the task-representing paper tags [46]. However, tracking from an overhanging infrastructured camera not only requires heavy external system setup but also is prone to occlusion and lighting condition, especially in a cluttered scene such as a household environment. A similar concept has been explored later using tangible blocks to program robot behaviours with its intent projected onto the physical environment [41, 42]. Still, robots' perception of the programming blocks relies on its on-board camera which limits the working range to the area in front of the robot. Further, recent researches employed mobile AR interfaces and associated the robots within the AR scene, e.g., with hand-held [35, 21, 17, 29, 8, 18] or head-mounted [9] devices. Although the mobility allows users to move around and author distributed tasks from different perspectives, the limited field-of-view constrains the human-robot interaction experience, i.e. the robot must be kept within the device's camera view to maintain the tracking that guides its navigation and interaction.

Other works separated the authoring interface and navigation by equipping robots with on-board SLAM capabilities. This way, user referred to a scanned map of the real scene as authoring context and the robot conducted tasks using the same map [31, 11, 38]. However, the pre-scanned SLAM map, once created, remains static and cannot adapt to the changes in the environment. In fact, for an ever-changing scenario such as user's home, the system will be hampered with outdated SLAM maps. Informed by these previous works, we propose a mobile AR-SLAM authoring interface with which users can spatially author the tasks by either explicitly defining navigation paths or implicitly visiting the IoTs by just walking to each of them. Moreover, we emphasize a transparent knowledge transfer between human and the robots by allowing robots to use the same AR-SLAM device as 'eyes' and 'brain' directly. We further increase the adaptability of the robots against environment changes as we rely only on on-the-fly created and updated SLAM maps that enables in-situ task authoring.

To summarize, the limitations of previous robot task authoring system are listed as follows: **(1)** Requires external tracking infrastructure for spatial/contextual aware programming; **(2)** Requires visual focus of the interface to maintain tracking during the execution; **(3)** Requires pre-acquired map information to achieve synergistic authoring and execution. We believe these limitations greatly constrain the accessibility of the proposed system from the general population, especially the on-site and in-situ programming for novice users in a new environment. Our system V.Ra, on the other hand, aims to solve this issue by adopting a spatial and visual interface within a mobile phone device, thus breaks the cognitive barrier and is more likely to be accepted by the general population. Moreover, we take the natural initiative of human user and utilize the on-the-fly generated SLAM map to seamlessly connect together user authoring and robot execution without needing any external tracking system. This allows for broader application for domestic usage as most of the daily scenarios are unstructured and constantly changing.

Robot-IoT Ecology within AR

An AR interface is spatially and physically aware of the environment by its nature [7]. Previous works have explored accessing and controlling IoTs through the digital representations superimposed in the AR scenes [36, 32]. But in these works, the augmentation relies on keeping the IoTs in the AR camera view, thus only allow for local interactions in a limited volume. Further, leveraging the SLAM embedded in mobile AR devices [1, 2, 3], researchers also investigated spatially registered IoTs in the SLAM map to support embodied interactions in a larger space [25]. In addition, AR has been used to author and edit IoT programs in-situ [24]. Moreover, recent works further emphasized on multiple IoTs in the same environment [16], e.g., visualization of the data flow among sensors, logic programming between devices [12], and visual analytics of the fetched data [13].

For stationary industrial robot arm programming, AR motion planning allows users to preview the generated trajectories and examine potential discrepancies and collisions [14, 15]. In a robot-IoT context, the mobility of the robots is a critical complementary element, as researchers have explored using AR for robot teleoperation as well [23, 22, 34, 30]. On the other hand, we focus on authoring room scale navigatory tasks for visiting distributed IoTs and assume that the local manipulation are handled by robot itself. While simple graphical augmentation can be superimposed onto the video streamed from the external camera system [22, 34, 26, 19, 40] or projected to the physical environment [33, 43, 41], we follow a mobile AR approach because a handheld [35, 27] or head-mounted AR device [9] allow users to freely move in the environment and inspect the augmentations from multiple perspectives. Besides authoring tasks for robots, researchers further explored using AR to debug robot behaviors [37, 20], passing knowledge to robots through demonstrations [29, 8], and interacting with the embedded AI decisions [45]. Although we don't develop these specific applications, our workflow shows potential to create robust test-beds for a variety of human-robot-IoT studies.

DESIGN GOAL

We followed a user-centered design approach to derive the design goals of our system and conducted conversational style informative interviews for our study. We first explained the context of a household robot-IoT ecology to the interviewees, then asked them to think about a scenario where users author multiple tasks to the robots and reveal their considerations and requirements for the system. We interviewed 42 people totally, including students, staff, and professors in the university with various backgrounds. Each interview took 6-10 minutes with the conversation recorded in audio. After analyzing the interview records, we identified the requirements and preferences from the participants. Combining the interview analysis with our vision of robot-IoT ecology, we propose the following four Design Goals (DG).

DG1: Easy and Instant Deployment. Less dependencies on the environment is preferred so that the system can be used instantly even for a new environment. Especially for the tasks handling chores, if the preparation takes even longer than finishing the chores by users themselves, the acceptability of the robot would be severely decreased. Thus, our system should be developed in a *self-contained* and *plug-and-play* manner to avoid the environmental dependencies and allow for *in-situ* authoring.

DG2: Physical and Spatial Awareness. We aim towards leveraging users' innate knowledge of the environment to instruct the robots to accomplish tasks in a household environment which are unstructured and ever changing. A physical and spatial aware authoring interface would allow users conveniently and accurately express their intents and transfer them to the robots.

DG3: Iterative Process with Feedback. Many participants unanimously required the system to keep them informed about its operating status during the entire process with active feedback. Further, our system should support users to visually preview and iterate the authored actions so that the efficiency of a sequence of distributed tasks can be improved.

DG4: Low learning curve. Participants suggested to develop the system based on easy-to-access devices and tools so that the basic interaction modalities remain familiar to novice users. Compared to abstracted task planning tools for professionals, the system should emphasize low cognitive load by closely associating planning interactions with actions of the robots in the physical world.

V.RA ECOSYSTEM WORKFLOW

V.Ra System Walk-Through

As illustrated in Figure 1, we walk through our workflow with a typical use scenario. In a household environment, users first select a robot for the desired tasks from the available nearby ones. This allows an AR authoring interface to be specialized based on the capabilities of this particular robot. The spread IoTs can be registered into the SLAM map through a one-time QR code scanning. Users then access the embedded knowledge from the IoTs in AR view. Using our authoring interface, users formulate a group of navigation paths, IoT interactions, and other time and logic constructs to achieve

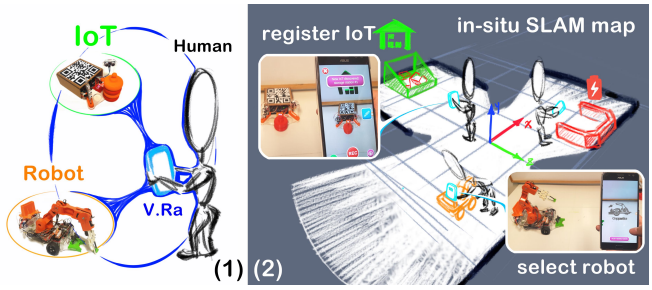

Figure 2. V.Ra ecosystem design coherently connects the three key elements of robot-IoT task planning with one AR mobile device (1), the spatial information for robot navigation and IoT interaction are stored in the on-the-fly generated SLAM map (2).

the desired robot-IoT coordination. After the authoring is finished, users physically place the authoring device onto the modular slot of the robot, and the AR system guides the robot to execute the tasks. Because of the transparency between the users' intents and robots' actions in the AR authoring phase, we achieve programming a robot in a WYDWRD fashion.

Choice of Approach

We want to develop an ecology where robots and IoTs are complementary to each other's role. As illustrated in Figure 2, Our workflow supports users to coordinate robots and IoTs temporally and spatially to accomplish multiple tasks synergistically in our daily surroundings. We deploy our AR authoring interface to a SLAM capable mobile device which is easy to access using commercial AR SDKs such as ARCore [1] and ARKit [2]. Within a mobile AR scene, users simply register IoTs with the SLAM map. By referring to the spatial distribution of the IoTs and the geometry of the environment, users then plan, preview, and iterate the robot-IoT interactions in-situ. Further, the same AR device can be employed as the the 'eyes' and 'brain' of the robot to execute the authored task. Such interchangeability between an authoring interface and robot navigation module promotes an transparent knowledge transfer from the users to the robots. As the SLAM map is constructed on-the-fly, our workflow does not rely on external tracking systems or an existing spatial map a priori, our system is therefore easy-to-install in a new environment and ready-to-use instantly.

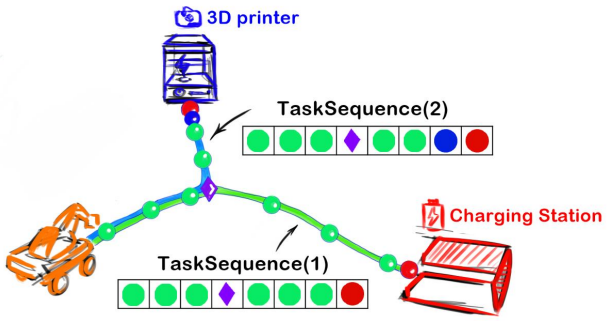

Figure 3. User authored tasks are represented by TaskSequence in V.Ra system, and they are formed by four types of Nodes. Logic driven event is represented by multiple TaskSequences.

AUTHORING INTERFACE DESIGN

Task Planning Construct

To start designing the authoring interface for mobile robot task planning, we first extract the basic elements of the task. The nature of our work is robot planning for physical tasks that involves interaction with different devices at various locations. The planned task may take a long period of time to execute, and it involves logic conditions that handle unexpected situations dynamically. By referring to previous programming protocols for IoTs and robots [4, 6, 12] and catering them to our system specifics, we develop the following ***Node***s to represent task elements and construct a task sequence.

Navigation Node ● : represents the path for the robot to travel through. It contains 3D coordinate information that can guide the robot's navigation during the Play mode.

Action Node ● : defines an action event that relates to the robot and/or the IoT device. The most common Action Node in our system is a robot-IoT interaction Node.

Time Node ● : contains information that allows the system to perform time based behaviours. For example, *keep doing this for some time,* or *wait until that happens,* etc.

Logic Node ◆ : contains a user defined check condition that allows the system to perform logic driven tasks such as *if Condition A then Action B*.

These *Nodes* are the basic abstractions that form any user authored task in V.Ra, namely, a construct array in our system, called **TaskSequence**. User can add new Nodes or manipulate the existing Nodes in the TaskSequence. When executing in the Play mode, the system guides the robot to run through each Node sequentially thus accomplish the authored task. The logic driven events are realized by multiple TaskSequences with each one representing one task line. Figure 3 illustrates a logic event with its corresponding TaskSequences. The robot checks the condition at the Logic Node and decides which path to take. If the battery is low, it will continue on TaskSequence(1) and go to the Charging Station; otherwise it will proceed on TaskSequence(2) and go pick up the 3D printed part when it is finished. Note that the *wait...until* function is realized by the blue Time Node.

V.Ra Interface and Interaction

The interface design of V.Ra system is shown in Figure 4. We exercise simple and clean style for the UI design, while exploiting the strength of AR and maintaining the accessibility of the primary features and functions. Users can create TaskSequence and preview it in the AR view and also in the ***EventLine***, which is an interactive linear abstraction of the task. To start a new task planning after selecting a robot, a user first defines the basic robot action with *AddPath* and *ScanIoT*. User then preview the authored task with EventLine, user can *Insert* new function, or *Edit* existing tasks. The user has the option to create periodic robot tasks (i.e. repeat everyday) using the *Repeat* function. When task authoring is finished, user can activate the *Play Mode* and place the mobile device onto the robot. The robot then starts the execution of the planned tasks by sequentially running all the Nodes in the TaskSequence.

Figure 4. Main interface design of V.Ra system (top). An icon reference list for interactive functions in the system (bottom).

Basic Task Generation

Add robot path. Navigation Nodes are the majority Nodes that form the TaskSequence in our system as it defines the path for the robot to navigate in the environment. There are two ways to add navigation nodes: 1) record spatial movement (REC), or 2) hand-draw the path on the screen, as illustrated in Figure 5. The hand-drawn method are suitable for path planning in a smaller area, while the REC is designed for conveniently creating large room-level navigation paths through embodied spatial movement. Each created path is broken into a series of Navigation Nodes and are sequentially added to the end of the TaskSequence. After a navigation node is added, a green path will be displayed in the AR scene giving the user active visual feedback.

Add IoT interaction. Robot-IoT interaction encompasses the majority of the Action Node in the system. Other Action Nodes include IoT-only and robot-only functions. To add a new robot-IoT interaction Node, the user first needs to register the IoT device into the AR scene, which is achieved through a one-time scan of the IoT's QR code (Figure 6 (1-2)). This not only brings an interactive 3D virtual model into the AR scene (Figure 6 (3)), but also imports semantic information into the system, like IP address and interaction protocol. After the IoT registration, user can touch on its virtual icon to access its function list and select to add an Action Node (Figure 6 (4)), to the end of the TaskSequence. When a robot-IoT interaction Action Node is added, a green arrow path appears, pointing towards the IoT device as a visual indicator (Figure 6 (5)). Other types of Action Nodes can be added using the *Insert* function, which will be described later.

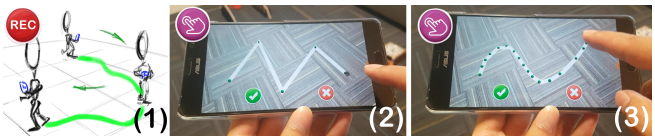

Figure 5. Authoring Navigation Node with (1) spatial movement, (2) hand-drawn segment line, and (3) hand-drawn curve.

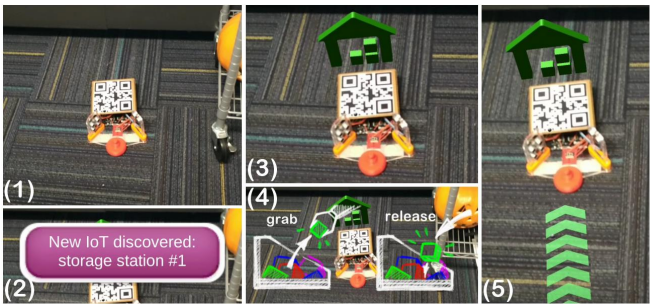

Figure 6. The process to add IoT interaction Node. (1) First scan its QR code to (2) register it into the AR scene. (3) Then touch on its virtual icon (4) to access the function list. (5) When finished, a green arrow path will appear for visual confirmation.

EventLine task visualization. While the AR view is good for spatial task visualization, it is constrained by the view of the display, which makes it difficult for user to perform global monitoring and manipulation of the entire task, especially when the task is authored in a large cross-room environment. To compensate for this on a handheld device, we introduce an abstract visualization of the task, called ***EventLine***. The design of EventLine is inspired by the timeline concept used commonly in the animation industry. The difference being that, in our case, the task is governed by events, such as robot navigation and IoT interaction. As is illustrated in Figure 7 (1), the EventLine has all the non-navigation Nodes shown on it as icons, and the user can tap on it to view its details, edit it or delete it (Figure 7 (3)). By dragging the handle on the EventLine, the user can preview the task with a virtual robot (Figure 7 (2)). This is designed to help users simulate the robot path execution to avoid unexpected errors. When multiple task lines exist, only the currently selected task line will show its EventLine on the screen, to keep the screen view clean. User can switch the selected task line by tapping on it in the AR view. The selected task line will be highlighted with the white indicator flowing through it.

Task Manipulation

The use of EventLine not only helps one to visualize the task in a linear abstract form, it also provides users with an editing tool to access the task details visually and manipulate them.

Insert. By dragging the handle, users can insert new Nodes into the designated position in the TaskSequence, which is illustrated by the position of the virtual robot (Figure 8 (1)).

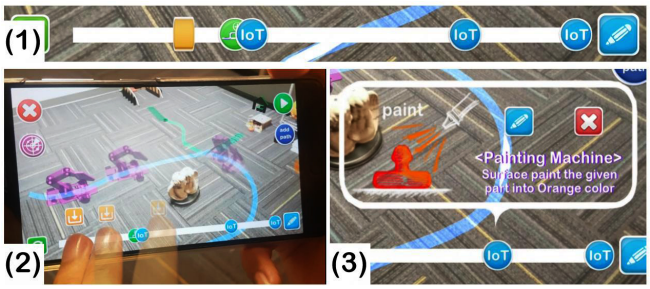

Figure 7. (1) EventLine represents the task in a linear and compact format. (2) User can drag the handlebar to preview with a virtual robot. (3) User can tap on the icon to review its detailed information, and to edit or delete it.

Figure 8. The Insert function. (1) User can drag the EventLine handle-bar and choose insert location for (2) non-robotic IoT function Action Node, (3) Time Node, or (5) Logic Node that represents logic driven event with an alternative task line. (4) It's trigger condition is defined from the working and sensing status of the connected devices.

These Nodes are 1) non-robotic IoT Action Nodes, 2) Time Nodes, and 3) Logic Nodes. To insert an IoT function, the system provides the user with a list of all the IoT devices that are connected to the system (Figure 8 (2)). Users then select from the list, access the function of that IoT, and insert it into the TaskSequence. To insert a Time Node, users either set a fixed wait time (Figure 8 (3)), or define a *wait...until* condition that is triggered by the IoT working status or sensing values. User can repeat the process and create composite AND/OR boolean conditions. In terms of the Logic Node, upon selecting, an alternative TaskSequence will be created and user will be asked to define the trigger condition, which is the same condition definer interface for the Time Node (Figure 8 (4)). The newly created TaskSequence has all the Nodes prior to the insert point copied from the original TaskSequence. This allows users to define new task line that branches from the Logic Node position (Figure 8 (5)). When executing a task with multiple TaskSequences, the system will run from the default TaskSequence (the first created TaskSequence) and decides which TaskSequence to continue at an Logic Node, based on the condition check.

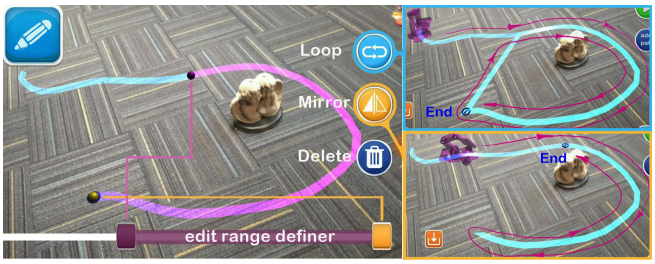

Figure 9. The Edit function for partially loop, mirror, or delete the authored task.

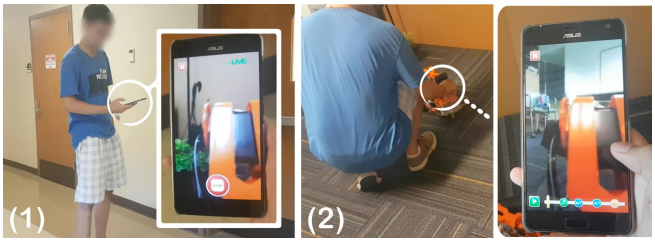

Figure 10. Post-play features of V.Ra system. (1) User can monitor the robot execution during its Play mode using an external smartphone. (2) Our system also creates video log that records the robot's execution.

Edit. By utilizing the EventLine, V.Ra allows user to edit their authored task by looping, mirroring, or deleting part of the selected EventLine. The copy and mirror functions are designed to increase the authoring efficiency for scenarios like *repeat this floor sweeping path 10 times* (loop), or *go back to where you came from* (mirror). When accessing the Edit mode, two interactive markers will appear on the EventLine with the middle part highlighted. Users can drag the markers to define the edit range, and the corresponding part in the AR view will also be highlighted (Figure 9).

Post-play Features

V.Ra's system interaction does not end at the Play mode. Guided by DG3, we want to keep the user in the loop during the entire process. Even during and after the robot execution. As illustrated in Figure 10 (1), our system allows users to live monitor the task execution using an external smartphone, by video streaming via the front camera of the authoring device (the rear camera is used for SLAM tracking). User can stop the whole operation via the STOP button if he notices something goes wrong or simply changes his mind. During the play mode, our system will automatically record the video feed from the front camera and generate an event-icon-embedded video log and stores inside the device (Figure 10 (2)). User can later access this video log to review what have happened during the Play mode, for process analysis and debugging.

IMPLEMENTATION

Software Platform

Our software interface is implemented on ASUS Zenfone AR mobile device. The AR SLAM feature is achieved using Google's software SDK (Tango Core [3]), and the application is built with Unity3D engine. The live task monitor feature is implemented with the WebRTC video stream service. It is noted that Tango Core relies a built-in depth camera to produce point cloud based user interaction. We chose this device due to the technology availability at the time of our initial development. However, our system is not limited to depth camera based Tango device. V.Ra is fully compatible with the latest AR platforms which use RGB-only cameras of the regular smart phones (e.g., ARCore [1], ARKit [2]).

Hardware Prototyping

To showcase the concept of V.Ra system, we prototyped four robots (Figure 11 (1-4)) and six different kinds of IoTs (Figure 11 (6-10)) for our use case demonstrations and preliminary user studies. All the robots and IoTs are equipped with wifi

Figure 11. Prototyped robots and IoTs. (1) TowerBot (2) GripperBot (3) SweeperBot (4) WaterBot (5) Charging Station (6) Painting Machine (7) 3D printer (8) Sorting Box (9) Storage Station (10) Water Station

communication capability using UDP protocol, which is implemented using ESP8266 and Arduino Mega microcontroller. The motor functions of some robots and IoTs are provided by the HerkuleX servo and Arduino Braccio robot arm. All of our robots and IoTs are designed to prove the concept of our proposed human-robot-IoT task authoring ecosystem, and therefore they are mockup prototypes with fairly low fidelity.

Robot Navigation and IoT Interaction

During the play mode, the authoring device instructs the robot to perform navigation and interaction activities. To navigate the robot along a user-defined path, the device constantly checks its current position and orientation in the SLAM map coordinate system, and compares with the target Node's coordinate information to guide the robot's movement. In other words, the SLAM device is the 'eyes' for the robot to navigate. To interact with an IoT, the robot first docks into the interaction position of the IoT by going through a short docking path embedded within the interaction Node. All the IoTs have similar docking target which is a red rounded object. At the end of the docking path, the robot reaches close enough to the docking target and it can finalize the docking process using the front color detection camera (Pixy CMUcam5). Once the robot is docked with an IoT device, precise manipulation (like grabbing an object from the Storage Station) can be ensured and the interaction is proceeded via a three-way communication among the authoring device, robots and IoTs. For example, to grab from the storage station, after successful docking, the robot first asks the Storage Station about how many objects are currently stacking on it, and based on the answer it grabs at different positions and then completes this Robot-IoT interaction.

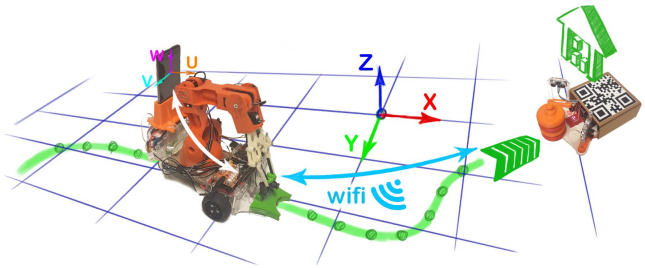

Figure 12. Communication among the robot, IoT, and the authoring device during navigation and robot-IoT interaction.

USE CASES

In this section, we demonstrate three different use cases of V.Ra system in household scenarios. For better visualization of the use cases, please refer to our demo video.

Case 1: SweeperBot for Smart Floor Cleaning. Our first use case features SweeperBot for user defined smart floor sweeping. As opposed to commercial products that try to survey the entire room, our system allows user to pinpoint the area that needs cleaning, thus greatly increase the cleaning efficiency. Before the user starts, he notices the power LED on the SweeperBot blinking, indicating a low battery status. While trying to finish the task authoring without any delay, the user programs the robot to go into the Charging Station to charge for 20 mins using the *Timer* delay function (Figure 13 (1)), then pinpoints the area for cleaning using the *SpotSweeping* robot function (Figure 13 (2)). The user also authors the curved sweeping route under the table and uses *Mirror* and *Loop* functions to repeatedly clean that area. Noted that the robot is able to successfully cruise under the table with poor lighting conditions (Figure 13 (5)), which shows the robustness of the system's navigation capability.

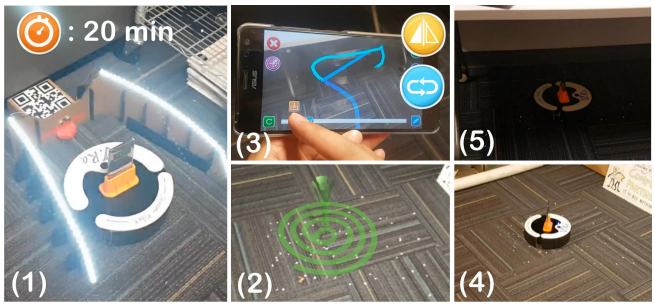

Figure 13. Use case 1, SweeperBot for household floor cleaning chore.

Case 2: TowerBot for Automated Fabrication. Our second use case features TowerBot in a large clustered room (Figure 14 (1)), helping makers with automated fabrication process. In this demo, the user wants to fabricate a few parts through the following process. Each part is 3D printed, surface coated in the Painting Machine and then placed into the Sorting Box. The entire process take several hours. To automate the above task and fabricate three parts, he first uses a triggered *Time* delay for the robot to wait until the 3D printer finishes printing the current part, picks it up (Figure 14 (2)), then authors the 3D printer to start printing another part. After that, he plans

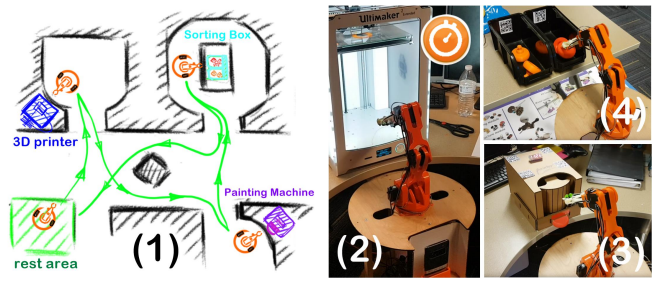

Figure 14. Use case 2, TowerBot for automated DIY fabrication.

the path for the TowerBot to navigate in the room and interact with the Painting Machine (Figure 14 (3)) and the Sorting Box (Figure 14 (4)), then comes back to the rest area to charge the battery. Before executing, the user authors a *Repeat* function upon the entire task for three time with an interval of 1 hour for battery charging.

Case 3: WaterBot for Daily Plant Watering. Our third use case features WaterBot for automatic daily watering of home plants (Figure 15). The Flower needs regular watering everyday, while the Grass needs much less water. To cater to the two plants with different watering frequency needs, the user first authors the WaterBot to water the flower (Figure 15 (2)) and then comes back to the Charging Station, then repeat the task everyday (Figure 15 (5)). On the way back to the Charging Station, he *Insert*(s) an alternate task line which is triggered by the moisture sensor planted in the Grass , to water it when needed (Figure 15 (3)). He also *Insert*(s) another alternate task line triggered by the WaterBot water level sensor, to go to the Watering Station and refill its tank when it's running out of water (Figure 15 (4)). This use case demonstrates the ability to author flexible logic driven events and shows the potential for home environment automatic plant and pet care taking.

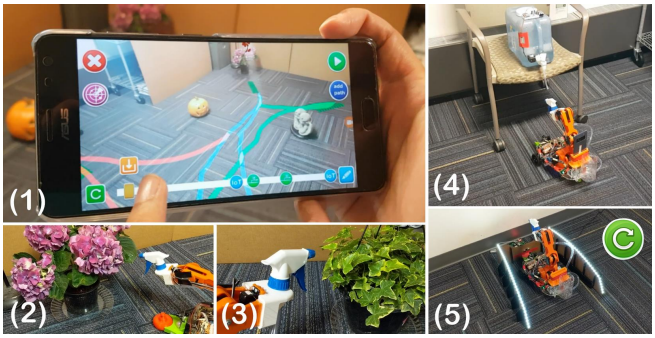

Figure 15. Use case 3, WaterBot for daily plant watering.

PRELIMINARY EXPLORATORY STUDY

To evaluate the navigation and overall usability of our system, we invited 10 users (7 male) from various backgrounds to our two-session preliminary user study. None of them had prior experiences with our system and their age ranged from 22 to 30. The two-session study was conducted in a 6.5x4.5 meter room using the GripperBot, and the entire process was video recorded for further analysis. Each user was given a 15 min tutorial before proceeding to the task in session 1. After each session, the user was given a survey to answer subjective and objective Likert-type questions. Each Likert-type item is graded by users from 1 to 5, on the usefulness of the feature and the level of agreement. A standard SUS questionnaire was also given to each user after all the study.

Session 1: Navigation Authoring Evaluation

Using a SLAM embedded AR interface, our system is capable of fast and accurate in-situ navigation authoring, which is one of the system's core features. The first session of the study is designed to evaluate this with novice users.

Procedure. As is illustrated in Figure 16 (1-2), we have drawn an 'S' shaped track (40 cm wide, total length about 13.5 m)

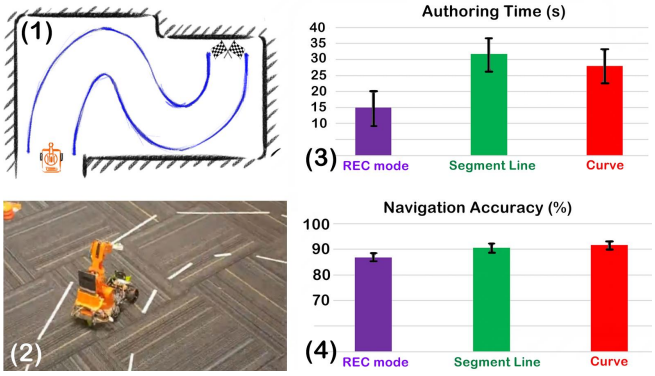

Figure 16. (1) Illustration of the ground setup for session 1. (2) User authors navigation paths for the robots to travel within the track. Result of session 1 are shown as (3) authoring time, and (4) navigation accuracy.

on the floor and asked the participants to author navigation path(s) for the robot (28 cm wide) to go through it while trying to maintain within the track. The participants were asked to use all three methods one by one (spatial RECord movement, hand-draw segment, hand-draw curve) to author the path at their normal speed. The navigation accuracy was measured as *distance the robot traveled within the track / its overall traveled distance*, this measurement is acquired from visual analysis of the video record. If any part of the robot goes out of the track, the condition was recorded as not met. Besides the accuracy, authoring time for each approach was also recorded.

Result and Discussion. Before inviting the novice users, the authors have run the process 3 times and chose the most fluid one as the baseline for this exploratory study, shown as follow. **REC:** 9s, 90%; **Segment:** 25s, 94%; **Curve:** 19s, 92%. When analyzing the result from the users, we found that all users were able to complete the authoring within 36 seconds using any of the three methods. As is shown in Figure 16 (3-4), all three methods were able to achieve high navigation accuracy (> 87%) with stable performance (low SD). Among the three methods, REC mode is the fastest to author navigation path (avg = 15s, sd = 3.2), while still maintaining a good accuracy (avg = 87%, sd = 1.4). Segment Line takes most time to author a new path (avg = 32s, sd = 3.7), but has the highest accuracy (avg = 93%, sd = 1.6). On the other hand, Curve takes less time (avg = 27.5s, sd = 3.6) with a small sacrifice for accuracy (avg = 91.5%, sd = 1.3). It is noted that the accuracy of the off-the-shell SLAM tracking (Tango Core) is within a few mm. However, the real navigation accuracy applied on a physical robot in V.Ra is affected highly by the driving mechanism of the robot itself. In fact, even a perfect trajectory (in the middle of the track) still only has 90%-95% accuracy when applied on the GripperBot. This study is to explore the navigation authoring efficiency and interaction intuitiveness to provide a reference preliminary accuracy test for our system.

According to the after-session interview, most participants (6/10) stated that their most handy method to author robot path is through using a Segment Line. This is because the segment method is the easiest means for a small room-scale area. "*The segment method is my favorite, I can simply tap on the screen and create a path without needing to move my body (P2).*" "*I like the segment method, it is fast and intuitive, just touching a few times on the screen (P8).*" The second favorite (3/10)

is the curve method, especially at a corner region. While its feature is beneficial to create curved trajectory with ease, some users found it hard to master. "*I think drawing on the screen to create a path is interesting, even though it needs steady hands (P7).*" It is noted that not many users (1/10) appreciate the REC mode for this task, due to the relatively small area of the study room, they preferred to avoid walking in the cluttered scene setup. However, most users (9/10) admitted that REC mode is more suitable for larger area cross-room navigation authoring, where looking at and tapping on the screen along the way would be very inconvenient.

Session 2: System Usability Evaluation

The second part of the study is to evaluate the overall usability of our system by asking the participants to complete a comprehensive task.

Procedure. The setup for this session of the study is illustrated in Figure 17. Where the Storage Station 1 (S1) is stacked with orange objects, while S2 and S3 are empty. The Painting Machine (P) can paint one object at a time, into red color, using 3 minutes. There is a Doorway (D) (represented using a Red-Green traffic light LED) in the setup that periodically opens and closes. The task is to stack *two* red objects onto S3. To achieve this, participants need to author the robot to navigate in the scene, first get the orange object from S1, paint it to red in P, then place it onto S3. Each participant was given 30 mins for this task and they were encouraged to explore as many different approaches as they like to complete the task. Participants were given full access to all the functions of the system, including the post-play features, to thoroughly experience V.Ra system.

Results and Discussion. All participants were able to successfully complete the tasks. The average authoring time for each approach is 2 min 16 s. Figure 17 (1) illustrate the most commonly used approach, which have been tried by 8 out of 10 participants. Though it is the most straightforward method, it is not the most efficient authoring approach in terms of robot travel distance and execution time for this task setup. Many participants were interested in trying different approaches after the basic solution. Figure 17 (2-4) illustrates some of the more exciting task authoring. P4 has created alternative

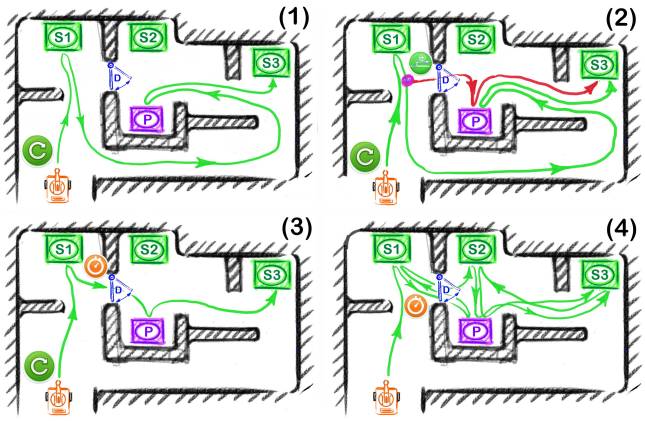

Figure 17. Results of study session 2 on a comprehensive task with different approaches from the users.

TaskSequence that allows the robot to take the shortcut if the Doorway is open, otherwise take the detour (Figure 17 (2)). While P7 determines to take the shortcut no matter what, if the Doorway is closed, the user authored the robot to wait at the entrance until it opens (Figure 17 (3)). Since this task requires two objects to be stacked onto S3, most participant use *Loop* and/or *Repeat* function to automate the authoring for the second object. However, P11 had a different and interesting approach. While the Painting Machine is processing the first object, instead of waiting idly, the user programmed the robot to go take the second object and put it onto S2 temporarily (Figure 17 (4)). It is worth mentioning that this method takes less time for robot to execute, but it did take more time for the user to author (5 min 27 s).

Observation and Feedback: Meeting the Design Goals

The Likert-type result from the two-session preliminary user study is shown in Figure 18. From our observation, participants could quickly learn the features and interactions of V.Ra, the 15 min tutorial was more than enough to shake off the cold feet for a novice user and they are excited to try V.Ra on their own. The decision of using an app-based smartphone device (Q1: avg = 4.8, sd = 0.4) greatly lowers the cognitive load for a novice user because they feel they are already familiar with the system. "*I like the idea of using my own smartphone to control the robots and IoTs, makes me feel more comfortable about using the technology (P12).*" The clean style of our UI design (Q2: avg = 4.6, sd = 0.7) and the nature of physical spatial authoring (Q4: avg = 4.8, sd = 0.6) also helps users boost up a quick start by creating a basic robot task within a minute (Q3: avg = 4.4, sd = 0.8). "*It's very easy to plan a task, just walk around to the device and click a few buttons (P3).*" This indicated that our prototype system is accessible and ready-to-use for a new user, with a fairly low skill floor to get started, which meets our DG1 and DG4.

Participants were generally receptive to the features and functions embedded within V.Ra. some of the task manipulation functions were highly appreciated by the participants. "*I really like the edit function, it's so simple and effective to create repetitive robot task. (P9)*" On the other hand, the *Insert* function received mixed feedback from the user. While most users (7/10) acknowledged that the feature of time and logic based events has the potential to increase the level of task complexity and real-life usefulness (Q7: avg = 4.1, sd = 1.3) "*I think the Insert-alternative-task function is very useful to create real-life comprehensive robotic jobs (P7)*". Two users (P4, P6) felt that the interaction flow of authoring the logic as well as the capability of the programming can be further improved. "*It took me some moment to figure out how to make an if...else task, I think the UI and interaction about this part can use some more work (P6).*" Overall, the participants agreed that the features and function of V.Ra are well integrated (Q6: avg = 4.4, sd = 0.8) and the system has a high skill ceiling that allows users to evolve through iteration and produce complex robot task authoring, therefore meeting our DG3 and DG4.

Survey responses were positive about the visual feedback of the planned task provided by our system. The AR view of planned robot path with IoT interaction was highly appreciated

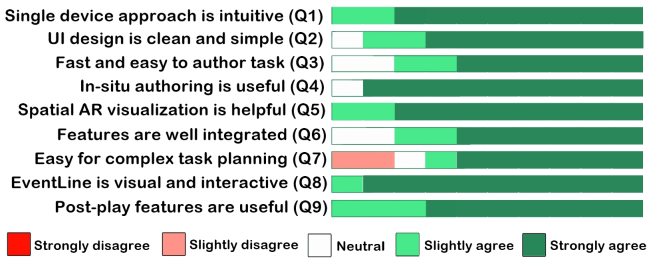

Figure 18. Likert-type result after the two-session study.

by all users (Q5: avg = 4.8, sd = 0.4), "*I think it's a really cool idea to use augmented reality to visualize the robot task in 3D environment, helps me to simulate what's going to happen and remove my doubt (P2).*" the same appreciation was received for other task visualization features like the EventLine (Q8: avg = 4.9, sd = 0.3) and post-play feature (Q9: avg = 4.7, sd = 0.5). "*My favorite thing about V.Ra is that I can know what's going on through the entire process, even after the programming (P11).*" We believe these feedbacks have confirmed that our system has engaged the user-in-the-loop during the robot-IoT task planning lifecycle (DG2 and DG3). The System Usability Scale (SUS) survey was also deployed after the second session, and the response result is 85.25 with a standard deviation of 6.6, which indicated high usability.

LIMITATION AND FUTURE WORK

SLAM tracking. The current system relies solely on SLAM to navigate at room-scale level during the execution. Though fairly robust, the tracking can be lost when the camera view lacks sufficient features, i.e. facing towards a white wall. Current system has no way of recovery in the event of lost tracking, and is therefore handicapped. To deal with this issue, a future system can embed a lost-tracking response protocol that allows the robot to automatically restore the tracking.

Logic interaction. Current system allows users to create logic driven events based on the boolean value from the robot's or IoT's working status and sensing value. We believe this limitation is partly due to the mobile platform, which is better suited for touch-based toggle interaction. Future endeavors should explore other interaction modalities, with different platforms like head-mounted ones, to achieve high level of complex programming with intuitive interactions.

Robotic system capability. The robots demonstrated in this work are proof-of-concept prototypes and are very limited in their functional capabilities (i.e. no auto obstacle avoidance or fault detection and correction, etc). To develop robotic system for practical real-world applications, multiple sensors can be fused with the robot to improve its perception capability. This can potentially enable the robots to automatically solve local issues independently, such like the lost-tracking problem. With more advanced robots, we can expect future robotic authoring systems to focus more on high-level user intent authoring, and less on the low-level execution process details.

IoT registration. Instead of using a QR code for IoT registration (current approach), future work can automate this process by introducing other IoT localization methods and cloud robot/IoT database. Further, our application is not limited to active IoT devices, but also passive objects, such as door handle, light switch, trash cans, and more chores in kitchens and living rooms.

Single smartphone approach. The nature of the current approach limits the task to only one mobile robot. However, our system can easily adapt to multi-robot collaborative task flow. In this case, each robot needs individual navigation capability and their map needs to be synchronized. Any individual authoring information can be shared through a cloud server and visualized on other user's handheld AR view, for collaborative task authoring of multi-branched complex logic driven tasks. The smartphone approach is used for our concept prototype. In general, the authoring interface can be any specially designed handheld devices. It does not need to be someone's smartphone when using in public location.

Human-Robot-IoT ecosystem. For future endeavor to enrich the ecology, a cloud based data management system needs to be setup with a new protocol created for the human-robot-IoT communication. This enables better handling of more complex tasks that requires coordinated scheduling among multiple devices, through efficient distribution of available resource, and intelligent control of information flow. The expandability of the system also needs to be considered. For example, new devices and can easily be developed to be connected into the ecosystem in a plug-and-play manner.

CONCLUSION

This paper has presented V.Ra, a spatially situated visual programming system for household robot task planning. We have explained our design rationale and demonstrated our user-oriented system design process. We adopted a workflow approach of one single mobile AR device for task authoring and robot execution, and developed our authoring interface for robot-IoT in-situ visual programming. We have shown three different use cases for household applications, featuring floor cleaning chores, DIY maker fabrication, and daily plant watering. Finally, the promising results from our 2-session preliminary user study have validated the navigation robustness and the system useability, showing that the prototype system has reached our design goals. In V.Ra, humans and smartthings enhance each other's capability within the fluidly connected ecology, such that spatially oriented collaborative tasks can be operated with lightweight system requirements. V.Ra highlights the initiative of human user and emphasize its position in the human-robot-IoT ecology. By shifting the design priority from automation to human interaction, Vra has demonstrated a new direction for HRI with the promise to deliver timely real-world solution to a broader population. We believe that V.Ra opens an inspiring perspective for researchers to reconsider human's role in the coming era of Internet-of-Robotic-Things.

ACKNOWLEDGEMENT

This work was partially supported by the NSF under grants FW-HTF 1839971, IIS (NRI) 1637961 and IIP (PFI:BIC) 1632154. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agency.

REFERENCES

[1] 2017. ARcore. (2017). https://developers.google.com/ar/.

[2] 2017. ARkit. (2017). https://developer.apple.com/arkit/.

[3] 2017. TangoCore. (2017). https://en.wikipedia.org/wiki/Tango_(platform).

[4] 2018. IFTTT. (2018). https://ifttt.com/.

[5] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. 2015. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials* 17, 4 (2015), 2347–2376.

[6] Sonya Alexandrova, Zachary Tatlock, and Maya Cakmak. 2015. Roboflow: A flow-based visual programming language for mobile manipulation tasks. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 5537–5544.

[7] Mark Billinghurst, Adrian Clark, Gun Lee, and others. 2015. A survey of augmented reality. *Foundations and Trends® in Human–Computer Interaction* 8, 2-3 (2015), 73–272.

[8] Peter Birkenkampf, Daniel Leidner, and Christoph Borst. 2014. A knowledge-driven shared autonomy human-robot interface for tablet computers. In *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on*. IEEE, 152–159.

[9] Yuanzhi Cao, Zhuangying Xu, Terrell Glenn, Ke Huo, and Karthik Ramani. 2018. Ani-Bot: A Modular Robotics System Supporting Creation, Tweaking, and Usage with Mixed-Reality Interactions. In *Proceedings of the Twelfth International Conference on Tangible, Embedded, and Embodied Interaction*. ACM, 419–428.

[10] Yaliang Chuang, Lin-Lin Chen, and Yoga Liu. 2018. Design Vocabulary for Human–IoT Systems Communication. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 274.

[11] Matei Ciocarlie, Kaijen Hsiao, Adam Leeper, and David Gossow. 2012. Mobile manipulation through an assistive home robot. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 5313–5320.

[12] Barrett Ens, Fraser Anderson, Tovi Grossman, Michelle Annett, Pourang Irani, and George Fitzmaurice. 2017. Ivy: Exploring spatially situated visual programming for authoring and understanding intelligent environments. In *Proceedings of the 43rd Graphics Interface Conference*. Canadian Human-Computer Communications Society, 156–162.

[13] Barrett Ens and Pourang Irani. 2017. Spatial Analytic Interfaces: Spatial User Interfaces for In Situ Visual Analytics. *IEEE computer graphics and applications* 37, 2 (2017), 66–79.

[14] HC Fang, SK Ong, and AYC Nee. 2012. Interactive robot trajectory planning and simulation using augmented reality. *Robotics and Computer-Integrated Manufacturing* 28, 2 (2012), 227–237.

[15] HC Fang, SK Ong, and AYC Nee. 2014. A novel augmented reality-based interface for robot path planning. *International Journal on Interactive Design and Manufacturing (IJIDeM)* 8, 1 (2014), 33–42.

[16] Jared Alan Frank, Sai Prasanth Krishnamoorthy, and Vikram Kapila. 2017. Toward Mobile Mixed-Reality Interaction With Multi-Robot Systems. *IEEE Robotics and Automation Letters* 2, 4 (2017), 1901–1908.

[17] Jared A Frank, Matthew Moorhead, and Vikram Kapila. 2016. Realizing mixed-reality environments with tablets for intuitive human-robot collaboration for object manipulation tasks. In *Robot and Human Interactive Communication (RO-MAN), 2016 25th IEEE International Symposium on*. IEEE, 302–307.

[18] Jared A Frank, Matthew Moorhead, and Vikram Kapila. 2017. Mobile Mixed-reality interfaces That enhance human–robot interaction in shared spaces. *Frontiers in Robotics and AI* 4 (2017), 20.

[19] Richard Fung, Sunao Hashimoto, Masahiko Inami, and Takeo Igarashi. 2011. An augmented reality system for teaching sequential tasks to a household robot. In *RO-MAN, 2011 IEEE*. IEEE, 282–287.

[20] Fabrizio Ghiringhelli, Jérôme Guzzi, Gianni A Di Caro, Vincenzo Caglioti, Luca M Gambardella, and Alessandro Giusti. 2014. Interactive augmented reality for understanding and analyzing multi-robot systems. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE, 1195–1201.

[21] Jesús Gimeno, Pedro Morillo, Juan Manuel Orduña, and Marcos Fernández. 2013. A new AR authoring tool using depth maps for industrial procedures. *Computers in Industry* 64, 9 (2013), 1263–1271.

[22] Sunao Hashimoto, Akihiko Ishida, Masahiko Inami, and Takeo Igarashi. 2011. Touchme: An augmented reality based remote robot manipulation. In *21st Int. Conf. on Artificial Reality and Telexistence, Proc. of ICAT2011*.

[23] Hooman Hedayati, Michael Walker, and Daniel Szafir. 2018. Improving Collocated Robot Teleoperation with Augmented Reality. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 78–86.

[24] 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.

[25] Ke Huo, Yuanzhi Cao, Sang Ho Yoon, Zhuangying Xu, Guiming Chen, and Karthik Ramani. 2018. Scenariot: Spatially Mapping Smart Things Within Augmented Reality Scenes. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 219.

[26] Kentaro Ishii, Yoshiki Takeoka, Masahiko Inami, and Takeo Igarashi. 2010. Drag-and-drop interface for registration-free object delivery. In *RO-MAN, 2010 IEEE*. IEEE, 228–233.

[27] Shunichi Kasahara, Ryuma Niiyama, Valentin Heun, and Hiroshi Ishii. 2013. exTouch: spatially-aware embodied manipulation of actuated objects mediated by augmented reality. In *Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction*. ACM, 223–228.

[28] Charles C Kemp, Aaron Edsinger, and Eduardo Torres-Jara. 2007. Challenges for robot manipulation in human environments [grand challenges of robotics]. *IEEE Robotics & Automation Magazine* 14, 1 (2007), 20–29.

[29] Jens Lambrecht, Martin Kleinsorge, Martin Rosenstrauch, and Jörg Krüger. 2013. Spatial programming for industrial robots through task demonstration. *International Journal of Advanced Robotic Systems* 10, 5 (2013), 254.

[30] Benoit Larochelle and Geert-Jan M Kruijff. 2012. Multi-view operator control unit to improve situation awareness in usar missions. In *RO-MAN, 2012 IEEE*. IEEE, 1103–1108.

[31] Adam Leeper, Kaijen Hsiao, Matei Ciocarlie, Leila Takayama, and David Gossow. 2012. Strategies for human-in-the-loop robotic grasping. In *Human-Robot Interaction (HRI), 2012 7th ACM/IEEE International Conference on*. IEEE, 1–8.

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

[33] Natan Linder and Pattie Maes. 2010. LuminAR: portable robotic augmented reality interface design and prototype. In *Adjunct proceedings of the 23nd annual ACM symposium on User interface software and technology*. ACM, 395–396.

[34] Kexi Liu, Daisuke Sakamoto, Masahiko Inami, and Takeo Igarashi. 2011. Roboshop: multi-layered sketching interface for robot housework assignment and management. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 647–656.

[35] Stéphane Magnenat, Morderchai Ben-Ari, Severin Klinger, and Robert W Sumner. 2015. Enhancing robot programming with visual feedback and augmented reality. In *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education*. ACM, 153–158.

[36] 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.

[37] Alan G Millard, Richard Redpath, Alistair Jewers, Charlotte Arndt, Russell Joyce, James A Hilder, Liam J McDaid, and David M Halliday. 2018. ARDebug: an augmented reality tool for analysing and debugging swarm robotic systems. *Frontiers Robotics AI* (2018).

[38] Hai Nguyen, Matei Ciocarlie, Kaijen Hsiao, and Charles C Kemp. 2013. Ros commander (rosco): Behavior creation for home robots. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 467–474.

[39] Partha Pratim Ray. 2016. Internet of Robotic Things: Concept, Technologies, and Challenges. *IEEE Access* 4 (2016), 9489–9500.

[40] Daisuke Sakamoto, Yuta Sugiura, Masahiko Inami, and Takeo Igarashi. 2016. Graphical instruction for home robots. *Computer* 49, 7 (2016), 20–25.

[41] Yasaman S Sefidgar, Prerna Agarwal, and Maya Cakmak. 2017. Situated tangible robot programming. In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 473–482.

[42] Yasaman S Sefidgar, Thomas Weng, Heather Harvey, Sarah Elliott, and Maya Cakmak. 2018. RobotIST: Interactive Situated Tangible Robot Programming. In *Proceedings of the Symposium on Spatial User Interaction*. ACM, 141–149.

[43] Masanori Sugimoto, Tomoki Fujita, Haipeng Mi, and Aleksander Krzywinski. 2011. RoboTable2: a novel programming environment using physical robots on a tabletop platform. In *Proceedings of the 8th International Conference on Advances in Computer Entertainment Technology*. ACM, 10.

[44] Ovidiu Vermesan, Arne Bröring, Elias Tragos, Martin Serrano, Davide Bacciu, Stefano Chessa, Claudio Gallicchio, Alessio Micheli, Mauro Dragone, Alessandro Saffiotti, and others. 2017. Internet of robotic things: converging sensing/actuating, hypoconnectivity, artificial intelligence and IoT Platforms. (2017).

[45] Michael Walker, Hooman Hedayati, Jennifer Lee, and Daniel Szafir. 2018. Communicating Robot Motion Intent with Augmented Reality. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 316–324.

[46] Shengdong Zhao, Koichi Nakamura, Kentaro Ishii, and Takeo Igarashi. 2009. Magic cards: a paper tag interface for implicit robot control. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 173–182.