International Conference on Artificial Reality and Telexistence
Eurographics Symposium on Virtual Environments (2019)
Y. Kakehi and A. Hiyama (Editors)

ReallifeEngine: A Mixed Reality-Based Visual Programming System for SmartHomes

Ryohei Suzuki[1], Katsutoshi Masai[1] and Maki Sugimoto[1]

[1]Keio University, Japan

Abstract

The conveniences experienced by society have tremendously improved with the development of the Internet of Things (IoT). Among the affordances stemming from this innovation is an IoT concept called the SmartHome, which is already spreading even in general households. Despite this proliferation, however, ordinary users experience difficulty in performing the complex control and automation of IoT devices, thereby impeding their full exploitation of IoT benefits. These problems highlight the need for a system that enables general users to easily manipulate IoT devices.
Correspondingly, this study constructed a visual programming system that facilitates IoT device operation. The system, which was developed on the basis of data obtained from various sensors in a SmartHome, employs mixed reality(MR) in enhancing the visualization of various data, eases the understanding of the positional relationship among devices, and smoothens the checking of execution results. We conducted an evaluation experiment wherein eight users were asked to test the proposed system, and we verified its usefulness on the basis of the time elapsed until the participants completed the programming of diverse IoT devices and a questionnaire intended to derive their subjective assessments. The result indicates that the proposed system makes it easy to understand the correspondence between the real world device and the node in the MR environment, and the connection between the sensors and the home appliances. On the other hand, it is negatively evaluated for operability.

CCS Concepts
• ***Human-centered computing*** → *Ubiquitous and mobile computing; Mixed / augmented reality;*

1. Introduction

1.1. Proliferation of SmartHomes and Related Issues

The Internet of Things (IoT) is a system that controls devices that are connected to one another and exchange information via the Internet. The IoT has significantly improved societal conveniences, with the innovation already being used in every industry, such as the manufacturing, transportation, logistics, medical care, and agricultural sectors. The specific uses to which the IoT is applied include monitoring factory equipment through sensors, recording biological information via wearable devices, and analyzing the growth processes of crops. The technology has even been introduced to general households through the concept of the SmartHome, whose features support many daily life activities and enable residents to monitor their behaviors so that they can acquire data for improving system functionality.

The development of SmartHomes has been pursued by researchers for over 30 years. An early example is the Intelligent House [Sak90], which was developed by Sakamura in 1989. This invention involves installing hundreds of sensors and actuators on windows and roofs and automating the opening and closing of windows with the help of wind or rainfall. SmartHome technology began as a scholarly concept that has now penetrated common households because of the advent of the Internet and embedded technologies. The operation of networked home appliances (hereinafter referred to as IoT devices) using smartphones and smart speakers is already part of the routine of daily living.

Notwithstanding these advantages, however, ordinary users encounter difficulties in performing complex IoT control and automation tasks, such as combining data from multiple sensors and specifying detailed conditions. Such capability requires knowledge of coding and complicated settings as well as learning about a given system. Whether general users can fully capitalize on the benefits of IoT is unclear. What is evident is that the spread of SmartHome technologies has given rise to the need for a system that enables the effortless control of IoT devices by average users.

1.2. Visual Programming

For general users who do not have coding knowledge, visual programming is a promising option given that it offers simplicity in program building. With this technology, programming is performed with the use of visual objects rather than through the composition of code. An example is node-type visual programming, in which

DOI: 10.2312/egve.20191287

a user connects figures, such as rectangles and circles, as nodes using arrows, lines, and arcs onscreen [KMR91] [PDKB*08]. An alternative is block-type visual programming, wherein a user combines colored blocks in accordance with a simple rule [MRR*10] [Kay05] [CP85].

Given that such visual programming systems enable the intuitive construction of programs without requiring detailed knowledge, it is considered an effective strategy for controlling IoT devices in SmartHomes. On this basis, we developed a visual programming system that allows the straightforward operation of IoT devices by using data from various sensors in a SmartHome.

1.3. Use of Mixed Reality (MR)

MR is a technology for constructing a space where virtual objects are superimposed onto an actual environment. This attribute can be realized by attaching a stereo camera to a non-transmissive head-mounted display (HMD) or using a transmissive HMD, such as Microsoft HoloLens [Mic]. A user can operate graphical user interfaces (GUIs) and virtual objects in the real environment.

In a typical home, there are many possible automations for appliances in daily life. Here are some examples. When sitting in a chair, turn on the desk light with a pressure sensor. When somebody comes to the entrance, turn on the entrance light for 10 seconds with a human sensor. When someone goes to bed, start a calm music with a pressure sensor. Turn on a fan according to a temperature sensor, etc.

In the situation where many devices are deployed, there can be too many items in a conventional GUI. Therefore, users would not able to understand the correspondence relationship, and there is a possibility that the user may be at a loss in operation. Through MR, GUIs superimposed onto sensors and home appliances that are arranged in a room can be displayed. The superimposed visual information provides intuitive understanding of connections between devices and GUIs, positional relationships and execution results, and it can be expected that equipment control can be performed smoothly.

In addition, in the case of MR with mobile devices such as tablets and smartphones, it is necessary to perform position estimation using an AR marker or some other measurement device. Also, users need to hold the device by their hand to use MR GUIs. However, it is possible to reduce those issues by using a video see-through HMD.

1.4. Purpose

As previously stated, we constructed a visual programming system called ReallifeEngine, which is easy for general users to employ in controlling and automating IoT devices in a SmartHome. To achieve this purpose, we adopted MR as an environment for visual programming to facilitate the understanding of the positional relationships among devices and visualize various types of information.

To verify the usefulness of the system, we asked users to take part in an evaluation experiment where they performed diverse programming tasks. We determined performance by measuring the time elapsed until each task was achieved by the participants and administered questionnaires to ascertain the users ' subjective assessments regarding the operation of the proposed system.

2. Related Works

2.1. Programming System for SmartHome

Remarkable systems have been developed to enable residents to easily program and change the settings of IoT devices in SmartHomes. These include Push-Pin [FSF*09], created by Fukuchi et al., and Node-RED [BL14], developed by Blackstock et al. Push-Pin allows residents to modify programs in a straightforward manner by inserting a pin-type physical tag into a device. When one of the two devices connected by pins is operated or senses a change, the other device can receive a signal over the network to activate devices. The use of a physical tag aids users in viewing connection relationships and presents the advantage of enabling intuitive setting operations. The drawbacks to this research are the physical limitation in the number of applicable devices given the number of pins available and the need for a dedicated device for realizing each function.

Node-RED is a node-type visual programming system that is operated via a GUI on a monitor. The system can operate IoT devices by connecting nodes with functions such as acquiring sensor values, controlling the ON/OFF switch, and processing input values at edges. It requires no additional devices, but it is difficult for users to grasp correspondence between an actual device and a node onscreen and realize that they can control the device.

Other similar systems are those that use block-type visual programming [SSF15]; rearrange cards with words [THA04] [CDCC14], events, and actions entered into a GUI; line up written cards [DRC15]; enable touch with radio frequency identification cards [KNF08]; and use barcodes [SMF99]. No system that does not require a device and can easily understand the positional relationships of IoT devices has been developed.

2.2. Support for Device Operation via Augmented Reality (AR)

A number of researchers have inquired into supporting device operation through AR. These include Muller et al., Dongsik et al., Valentin et al., Seifried et al., and Hashimoto et al., who established Guideme [MAK13], ARIoT [JK16], Reality Editor [HHM13], CRISTAL [SRP*09], and Touchme [HIII11], respectively.

Guideme supports operations by superimposing a UI that indicates the operation method onto a home appliance through a camera image. ARIoT can operate home appliances through a GUI superimposed onto the camera image of a smartphone or tablet terminal. Reality Editor allows for information acquisition by holding up an AR marker attached to an IoT device with the camera of a smartphone. It can also be used to operate other devices. CRISTAL can control home appliances using the display and touch input interface embedded in a table, and Touchme controls robot motions through AR.

On the basis of the above-mentioned studies, we established a

SmartHome visual programming system using MR to solve physical or spatial cognitive constraints and render SmartHome programming more flexible.

3. IoT Programming in an MR Environment

3.1. Policy

To realize visual programming in an MR environment, we adopted node-based visual programming in the system put forward in this work. Representing information from each sensor or home appliance as a node makes it possible to place a node at an actual device position in the MR environment. Because comprehending the positional relationships among devices is possible in a three-dimensional space, information on device position and direction can also be used in such space.

We prepared three types of nodes for programming: an output node for controlling the operation of a home appliance, an input node for receiving information from sensors, and an intermediate node disposed between the first two nodes for additional effect such as waiting for specified seconds. The output node controls the activation/deactivation (ON/OFF) of a home appliance via a stimulus-response model. The operation of an input device or a reaction from it generates a stimulation, after which the home appliance receives the stimulus and switches the system on or off.

Note that two types of sensors are used as input nodes: one that transmits information in response to a single operation at a time and another that continuously transmits environmental information. For example, the former includes human sensors and tact switches, whereas the latter encompasses pressure and light sensors. The first type of sensor should generate stimulus directly only at the time of operation, but the second type needs to separately set up the conditions that induce stimulus from an obtained measurement value. To specify these conditions, the proposed system prepares an intermediate node that produces a stimulus when the measurement value exceeds (falls below) a threshold.

3.2. System Design

ReallifeEngine consists of input/output devices, a network, a visual programming system, and an HMD (Figure.1).

The input/output devices refer to IoT home appliances controlled by the system and switches/sensors that recognize the surrounding environment. They are connected to the network via a wireless module, and they send and receive measurement values and operation information to and from a server. The visual programming system displays each input/output device information received from the server as a node. By connecting nodes through edges, we associate sensor responses with the operation of a home appliance.

The HMD attaches a stereo camera to the system to create an MR environment. The nodes and edges of the visual programming system are displayed and superimposed onto the real environment, thus enabling programming and checking the operation status of input/output devices in real time.

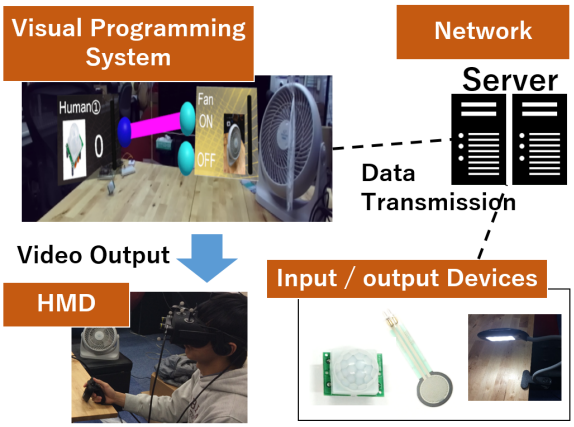


Figure 1: *System overview*

3.3. Operation Method

As mentioned earlier, a user can perform programming by connecting nodes through edges. The user can generate an edge by clicking on the sphere attached to both ends or one side of the node operated by a controller. Figure 2(a) shows how the control task for turning on a fan when a human sensor responds is established using the proposed system.

Aside from the three programming nodes that we prepared, two types of signals are handled by each node, namely, stimulus and analog signals, which are described in the succeeding sections.

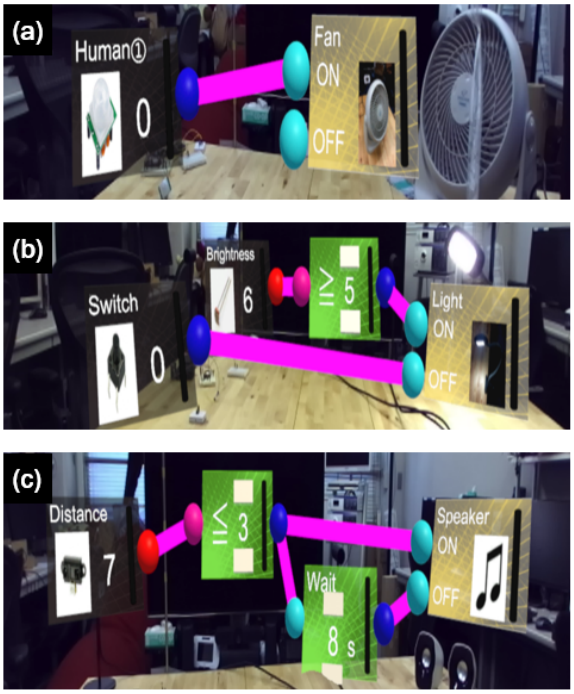
3.4. Input, Output, and Intermediate Nodes

As the programming examples in Figure 2 show, the input node (black) displays information obtained from sensors and switches and then outputs it. Examples of input nodes are human sensors, switches, pressure sensors, distance sensors, and optical sensors. The output node (yellow) receives a stimulus signal and switches a controlled home appliance on or off. Examples of such nodes include lights, fans, and speakers.

The intermediate node (green) can process given inputs and converts them into outputs. Such node can be one that waits for a specified number of seconds before it passes a signal to the next node or one that outputs the stimulus signal at the moment when the analog signal exceeds (falls below) a threshold. A user can change the threshold by clicking on the block above or below a displayed number. He/She can also place any number of intermediate nodes between input and output nodes.

3.5. Stimulus and Analog Signals

The stimulus signal conveys information only once at the moment of reaction. To illustrate, such signal is used when information is transmitted only at a single time, as in the moment a switch is pressed, the period wherein a controller button is activated, and the point at which a human sensor reacts. Conversely, the analog signal transmits value information incessantly, as is done by pressure,



(a)The fan is turned on when the human sensor responds. (b) The clip light is turned on at the moment the light sensor reaches a value of 5 or more. Turn off when the switch is pressed. (c) The speaker makes a sound as soon as the value of the distance sensor reaches 3 or less. Stop after 8 seconds.

Figure 2: *Programming example*

light, distance, and temperature sensors. The analog signal is used when information on a given environment is constantly desired.

In ReallifeEngine, the input/output of the stimulus signal is represented by the light blue and blue spheres, and the analog signal is represented by the pink and red spheres(Figure.2). A user cannot connect stimulus and analog signal spheres.

3.6. Behavior on Mixed Reality Environments

ReallifeEngine permits visual programming via the node connection introduced in Section 3.3. Such programming can be implemented in an MR environment appearing on the background of an image of the actual environment displayed through the HMD (Fig. 3). Each node can be arranged three-dimensionally in an MR environment, and an individual can use a handheld controller to connect nodes or change their positions.

By arranging nodes of overlapping devices in a real environment, a user can easily understand the correspondence between each device and node as well as perform programming while checking sensor input and device operation.

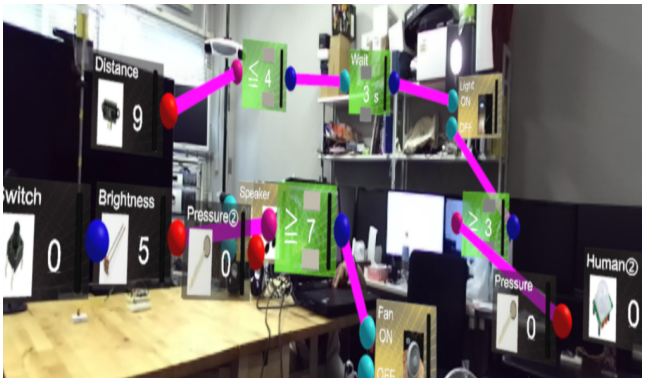


Figure 3: *State of programming in MR*

4. Implementation

4.1. Input/Output Devices

We prepared pressure sensors, light sensors (CdS cell), distance sensors, human sensors, and tact switches as input devices in the proposed system. We also prepared speakers, a fan, and clip lights as output devices. For the fans and clip lights, we connected a solid-state relay in the middle of the extension cord of an AC100 V outlet so that the power could be turned on or off from the output of the microcomputer. We also connected a fuse to prevent overcurrent. To transmit/receive device information to and from the network, we used ESP32, which is a microcomputer with a wireless function [ESP]. The speakers were connected directly to the PC running the system without the adoption of a microcomputer.

4.2. Network

We adopted message queuing telemetry transport (MQTT) as a protocol for communication between IoT devices and the visual programming system developed in this research. MQTT is one of the communication protocols available in TCP/IP networks. Compared with hypertext transfer protocol (HTTP), MQTT has a smaller header, and it adopts the publish/subscribe messaging model. Hence, MQTT exhibits its strength during the frequent and bidirectional transmission and receipt of a large amount of small data among many devices.

This communication scheme requires a "broker", which is an intermediary server that mediates communication between devices. To this broker, wireless modules and the visual programming system are connected for the purpose of transmitting and receiving sensor data and information on device operation.

4.3. Visual Programming

We used Unity(game engine) to develop this visual programming system [Uni]. Unity can set the control of three-dimensional arrangement and operation of UI in a virtual environment flexibly by script. Also, it can operate HMD easily.

In visual programming, each node holds internal variables of InputData(input storage), OutputData(output storage),

BackEdge(input source edge), FrontEgde(output destination edges). Also, it possess the StimulateOn () method, which is executed when the node receives a Stimulate signal. Each edge holds internal variables of InputNode(input source node) and OutputNode(output destination node).

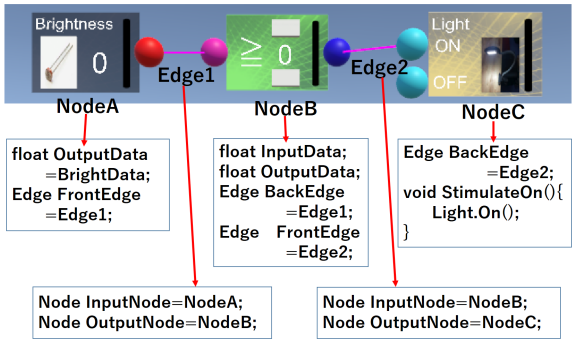


Figure 4: *Variables held by each node and edge*

Each node can obtain information from other nodes connected front and back through the use of the above-mentioned variables. As an example, the script set to NodeB can be described as follows (Fig. 4).

(Example 1) When a user wants to receive an analog value from the node connected to the input source (equivalent to acquiring output data from the input source node), the following script can be used:

this.InputData = BackEdge.InputNode.OutputData;

(Example 2) When a user wants to output a stimulus signal to the node connected to the output destination (equivalent to calling the StimulateOn () function of the output destination node), he/she can employ the script below:

this.FrontEdge.OutputNode.StimulateOn ();

In this manner, the variables and methods of previous/subsequent nodes can be called under any condition and timing on a script.

4.4. MR Support

To ensure compatibility between our visual programming system and MR, we adopted video see-through MR using an HMD and a stereo camera. We used Oculus Rift (developed by Oculus) [ocu] as the HMD and ZED Mini (developed by Stereolabs) [zed] as the stereo camera. Additionally, we used Oculus Touch [ocu] as a node controller. The three-dimensional positions of the HMD and controller can be acquired using an attached sensor.

We imported the software development kits (SDKs) of Oculus Rift and ZED Mino into Unity and then implemented these for controller input processing and spatial node movement/placement.

5. Evaluation Experiment

Using the implementation introduced in the previous section, we evaluated the effectiveness of the SmartHome MR-based visual programming system in an experiment. Participants were assigned programming tasks that required them to associate sensors with home appliances in a laboratory. We used another programming system that works with a GUI on a PC monitor for comparison with ReallifeEngine. We then measured the time taken by the participants to complete programming tasks over each of the systems. Finally, they were asked to provide subjective evaluations through questionnaires and freely describe points of ease and difficulty in operating the compared systems.

5.1. Experimental Conditions

The specifications of the PC used in the experiment are as follows.

- CPU:Intel(R) Core(TM) i7-7700 CPU 4.20GHz
- Memory: 16.00GB
- OS: Windows 10 Enterprise
- GPU: NVIDIA GeForce GTX 1080 8GB

The software programs used in the experiment are listed below.

- Unity: Version 2018.2.2f1
- ZED SDK: Version 2.7.0

The experiment involved eight participants (two females and six males) in their twenties. All of them use a computer on a daily basis.

5.2. Experimental Design

We placed a wooden table, a black desk, a chair, IoT devices, an HMD, and a PC in the experiment area (Fig. 6). The participants operated the GUI system on the PC for comparison with ReallifeEngine. As devices for control, a clip light, a fan, and a speaker were prepared, and two human sensors, two pressure sensors, seven photosensors, a distance sensor, and a tact switch served as sensors.

The components above were placed at various spots in the experiment area (Fig. 6), and the participants were instructed to proceed with associating the sensors with the home appliances.

To reduce the influence of acclimatization to the node connection operation, the participants were divided into two groups of four individuals. One group was tasked to work with the MR system first, and the other was instructed to work with the GUI system on the monitor first.

The participants could associate the operation of a device with data from the sensors by connecting the nodes displayed on the HMD or monitor. After receiving explanations regarding node connection methods and several operation examples, the participants could freely manipulate the GUI system on the monitor and the MR system on the HMD for about 2 to 3 minutes. After this, using the respective systems, the participants were directed to perform the five tasks shown in Table 1.

For all the tasks, the participants were asked to confirm that they have correctly operated the sensors and controlled the home appliances after programming. The time elapsed until task completion

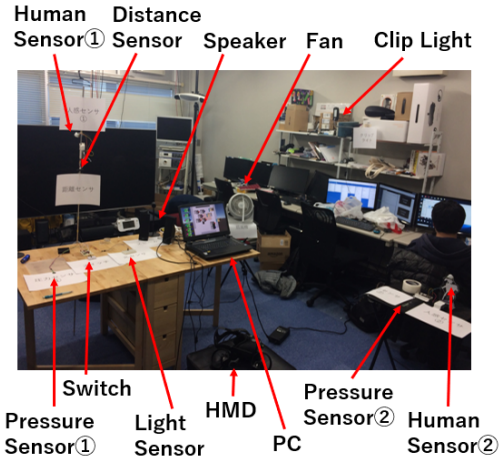


Figure 5: *Placement of each sensor and device*

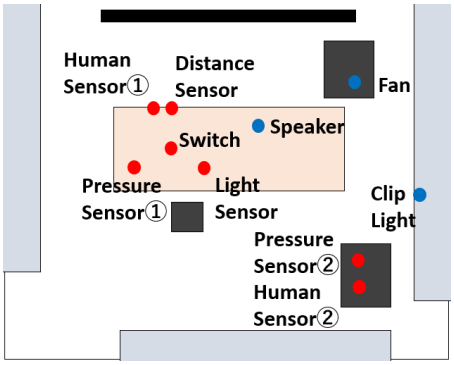


Figure 6: *Placement map*

was defined as the period running from the display of programming content on a large TV to operation confirmation. At the end of each session, the participants were asked to answer nine HMD- and PC-related questions (Table 2) on a seven-point Likert scale ranging from 1 ("strongly disagree") to 7 ("strongly agree"). As indicated earlier, the experiment ended with the participants openly describing the points that were easy/difficult to handle in the HMD system.

5.3. GUI System on Monitor

The system that could be programmed via the GUI (Fig. 7) is a visual programming system displayed on a PC monitor, not an HMD. Instead of a controller, a mouse is used. In the background, an overview of the experiment space is displayed. Each node is exhibited at a rough position on an overview map, and a user can connect nodes by clicking on the sphere attached to the node.

5.4. Result

Figure 8 shows the mean and standard deviation of the elapsed time for each task. Both systems do not count the time required for activation and preparation completion. The results indicated that the

Table 1: *Programming content of each task*

Exercise 1	Turn on the speaker by pressing the switch. When the human sensor on the stick responds, the speaker is turned off.
Exercise 2	Turn on the clip light after 3 seconds if the human sensor on the black table responds. When the human sensor on the stick responds, the clip light is turned off.
Exercise 3	Turn on the fan when the pressure sensor on the wooden table shows 6 or more. When the human sensor on the stick responds, the fan is turned off.
Exercise 4	Turn on the clip light when the number of light sensors shows 2 or less. When the distance sensor shows 3 or less, turn on the fan.
Exercise 5	After all connections have been released, have the subjects assign as you like. Turn on/off at least 2 home appliances.

Table 2: *Questionnaire items*

Question 1	I could understand the operation method immediately.
Question 2	I was convinced that the actual devices would work correctly through the operation.
Question 3	The operation confirmation was troublesome.
Question 4	I could immediately grasp how actual sensors and devices correspond to nodes on the screen.
Question 5	I could easily understand which sensors correspond to which appliances placed in the real room.
Question 6	Working with the HMD/monitor to change node connections is awkward.
Question 7	I was confused about whether wiring during node connection would work.
Question 8	In the last experiment, I was able to quickly turn the program in mind into reality.
Question 9	I would like to use the home appliance control system with the HMD/monitor at home.

time elapsed under GUI system operation was significantly shorter than that under the HMD system operation for tasks 1 to 4 ($p = 0.071$, $p = 0.036$, $p = 0.076$, $p = 0.044$, respectively; Welch's t-test).

Figure 9 presents the findings of the subjective evaluations. The right side of the figure indicates strong agreement with questionnaire statements (7 points), whereas the left side reflects strong disagreement (1 point). With respect to Q4 and Q5 ("I could immediately grasp how actual sensors and devices correspond to nodes onscreen.", "I could easily understand which sensors correspond to which appliances placed in a real room.", respectively), the participants provided a significantly positive evaluation of the proposed system ($p = 0.002$, $p = 0.002$; Mann-Whitney-Wilcoxon test).

However, the affirmative assessment was also reflected to Q6 ("Working with the HMD to change node connections is awkward.", $p = 0.102$). No significant difference was found in the re-

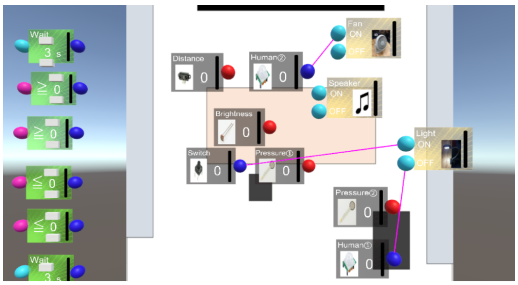


Figure 7: *GUI tool for comparison*

sponses to Q1 to Q3 and Q7 to Q9. No significant difference in any of the items was found in the comparison of the groups who completed the HMD or GUI sessions first.

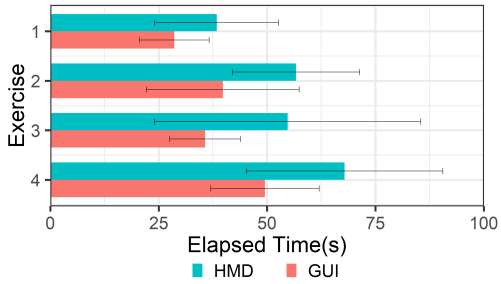


Figure 8: *Elapsed time for each task*

6. Discussion

6.1. Results of the Evaluation Experiment

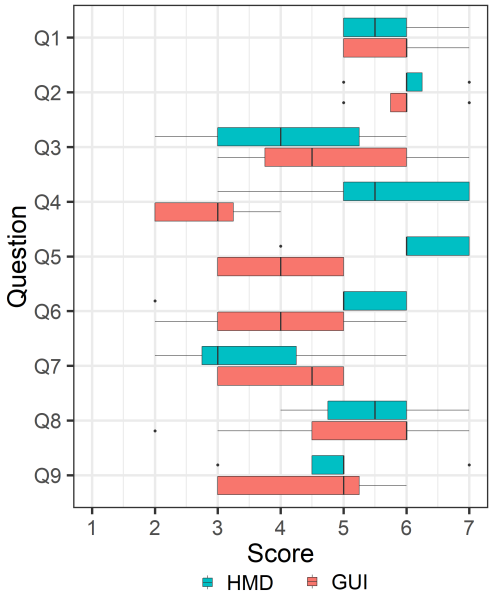
The subjective evaluation results confirmed that the proposed MR-based system was significantly superior to the GUI system in the matter of easy understanding regarding the positional relationship of IoT devices and their correspondence with nodes onscreen. The spatial arrangement of nodes was presumed to have been evaluated by the participants using MR. However, the time elapsed till task completion in the proposed system was significantly longer than that in the GUI system. Moreover, the participants negatively evaluated the proposed system in relation to Q6 (ease of operation).

As reflected in the free description section of the questionnaire, many of the participants insisted that the difficulty of operation increased in the MR system compared with the level experienced in the GUI system. Specifically, the respondents provided the following opinions on the UI:

- "It is difficult to accurately click the sphere in the three-dimensional space."
- "The nodes are overlapped each other in the mixed reality space and clicked the wrong node."

Also, regarding hardware, participants pointed out the delay of the visual information and wearing the HMD was time consuming.

The UI is expected to improve considerably when a new presentation method is designed. Such improvement can be carried out in



The right side (7 points) shows affirmative evaluation, and the left side (1 point) shows negative evaluation.

Figure 9: *Subjective evaluation results*

a number of ways: When the judgment line extending to the front of the controller hits the ball, it emits a light as a clear indication that it should be selected. Distant nodes can be expanded to prevent them from being displayed in small representations and difficult to click. A selected node can be highlighted, and nearby nodes can be shown transparently. When nodes are densely packed, they may be displayed slightly apart so that they do not overlap. Additional considerations in enhancing the UI presentation in the MR environment include the design of nodes and the approach to clicking on spheres.

In the evaluation experiment, the GUI system was highly evaluated for operability. The comments of the experiment participants suggest that the design of the UI needs to be carefully considered in order to make the MR system to be user-friendly. However, even with the current design, there was no significant difference in the result of Q9 "I would like to use the home appliance control system with the HMD/monitor at home". By improving the UI design, the MR system might have a potential to be accepted by users over the conventional system.

6.2. Future Work

Although we used various sensors as the input devices in this study, any other information on the Web can serve as input. Some examples are weather forecasts, stock prices, and traffic information. The input method can also be extended to include such functionalities as voice recognition and the processing of moving video images.

Only three types of intermediate nodes were used in this work, but nodes with various functions, such as the AND/OR circuit, the calculation of the sum or product of two or more inputs, and the

linear transformation of input signals, can be implemented. Such implementation requires further study.

Lastly, it is possible to operate a node beyond walls in case IoT devices in an entire house need to be controlled, even though the evaluation experiment did not consider this functionality. In this case, the situation occurring in other rooms cannot be viewed from the current space, pointing to the need for more devices in UI presentation. Space-related comprehension for an entire house will also be part of the directions we will pursue in the future.

7. Conclusion

We proposed a system referred to as ReallifeEngine, which can perform visual programming for a SmartHome in an MR setting. The system arranges nodes and shows information from various IoT devices three-dimensionally in the background of a real-world space. Through the connection of these devices, a user can associate sensors with the operation of home appliances.

The evaluation experiment showed that understanding the correspondence between a real world device and a node onscreen and the connection between sensors and home appliances is more easily achievable over the MR system than on the GUI system. The proposed system was negatively assessed by the participants with respect to operability as programming also entails more time with such system.

Future plans include the improvement of operability through corrections to the UI presentation method and hardware as well as operational design for an entire house. We also intend to expand the functions of the input and intermediate nodes.

References

[BL14] BLACKSTOCK M., LEA R.: Toward a distributed data flow platform for the web of things (distributed node-red). In *Proceedings of the 5th International Workshop on Web of Things* (2014), ACM, pp. 34–39. 2

[CDCC14] COUTAZ J., DEMEURE A., CAFFIAU S., CROWLEY J. L.: Early lessons from the development of spok, an end-user development environment for smart homes. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication* (New York, NY, USA, 2014), UbiComp '14 Adjunct, ACM, pp. 895–902. URL: http://doi.acm.org/10.1145/2638728.2641559, doi:10.1145/2638728.2641559. 2

[CP85] CARDELLI L., PIKE R.: Squeak: A language for communicating with mice. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1985), SIGGRAPH '85, ACM, pp. 199–204. URL: http://doi.acm.org/10.1145/325334.325238, doi:10.1145/325334.325238. 2

[DRC15] DE RUSSIS L., CORNO F.: Homerules: A tangible end-user programming interface for smart homes. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems* (New York, NY, USA, 2015), CHI EA '15, ACM, pp. 2109–2114. URL: http://doi.acm.org/10.1145/2702613.2732795, doi:10.1145/2702613.2732795. 2

[ESP] ESP32 Modules and Boards. https://docs.espressif.com. (Accessed on 01/15/2019). 4

[FSF*09] FUKUCHI K., SUGIMOTO M., FERNANDO C., ZHAO S., INAMI M., IGARASHI T.: Push-pins: design-by-user approach of home automation programming. *International Conference on Ubiquitous Computing, workshop 5: ArchiBots 2009, ubicomp 2009, Orlando, Florida.* (2009). 2

[HHM13] HEUN V., HOBIN J., MAES P.: Reality editor: programming smarter objects. In *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication* (2013), ACM, pp. 307–310. 2

[HIII11] HASHIMOTO S., ISHIDA A., INAMI M., IGARASHI T.: Touchme: An augmented reality based remote robot manipulation. In *21st Int. Conf. on Artificial Reality and Telexistence, Proc. of ICAT2011* (2011). 2

[JK16] JO D., KIM G. J.: Ariot: scalable augmented reality framework for interacting with internet of things appliances everywhere. *IEEE Transactions on Consumer Electronics 62*, 3 (August 2016), 334–340. doi:10.1109/TCE.2016.7613201. 2

[Kay05] KAY A.: Squeak etoys, children & learning. *online article 2006* (2005). 2

[KMR91] KODOSKY J., MACCRISKEN J., RYMAR G.: Visual programming using structured data flow. In *Proceedings 1991 IEEE Workshop on Visual Languages* (1991), IEEE, pp. 34–39. 2

[KNF08] KAWSAR F., NAKAJIMA T., FUJINAMI K.: Deploy spontaneously: Supporting end-users in building and enhancing a smart home. In *Proceedings of the 10th International Conference on Ubiquitous Computing* (New York, NY, USA, 2008), UbiComp '08, ACM, pp. 282–291. URL: http://doi.acm.org/10.1145/1409635.1409673, doi:10.1145/1409635.1409673. 2

[MAK13] MÜLLER L., ASLAN I., KRÜSSEN L.: Guideme: A mobile augmented reality system to display user manuals for home appliances. In *Advances in Computer Entertainment*. Springer, 2013, pp. 152–167. 2

[MRR*10] MALONEY J., RESNICK M., RUSK N., SILVERMAN B., EASTMOND E.: The scratch programming language and environment. *ACM Transactions on Computing Education (TOCE) 10*, 4 (2010), 16. 2

[PDKB*08] PRADAL C., DUFOUR-KOWALSKI S., BOUDON F., FOURNIER C., GODIN C.: Openalea: a visual programming and component-based software platform for plant modelling. *Functional plant biology 35*, 10 (2008), 751–760. 2

[Sak90] SAKAMURA K.: Tron-concept intelligent house. *The Japan Architect 65* (1990). 1

[SMF99] SIIO I., MASUI T., FUKUCHI K.: Real-world interaction using the fieldmouse. In *Proceedings of the 12th annual ACM symposium on User interface software and technology* (1999), ACM, pp. 113–119. 2

[SRP*09] SEIFRIED T., RENDL C., PERTENEDER F., LEITNER J., HALLER M., SAKAMOTO D., KATO J., INAMI M., SCOTT S. D.: Cristal, control of remotely interfaced systems using touch-based actions in living spaces. In *ACM SIGGRAPH 2009 Emerging Technologies* (2009), ACM, p. 6. 2

[SSF15] SERNA M. A., SREENAN C. J., FEDOR S.: A visual programming framework for wireless sensor networks in smart home applications. In *2015 IEEE Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)* (2015), IEEE, pp. 1–6. 2

[THA04] TRUONG K. N., HUANG E. M., ABOWD G. D.: Camp: A magnetic poetry interface for end-user programming of capture applications for the home. In *International Conference on Ubiquitous Computing* (2004), Springer, pp. 143–160. 2

[Uni] Unity. https://unity3d.com. (Accessed on 01/15/2019). 4