Defining Rules Among Devices in Smart Environment Using an Augmented Reality Headset

Heonjin Park[1], Kyle Koh[1], Yuri Choi[2], Han Joo Chae[1], Jeongin Hwang[1], Jinwook Seo[1]

[1]Seoul National University, 1 Gwanak-ro, Gwanak-Gu, Seoul, Korea +82-2-880-7044

[2]Samsung Electronics, 56 Seongchon-gil, Seocho-gu, Seoul, Korea

{hounjini, kyle, hanjoo, jihwang}@hcil.snu.ac.kr, yuri82.choi@samsung.com, jseo@snu.ac.kr

ABSTRACT

We present a pilot study that uses optical head-mounted displays (OHMD) as an augmented reality headset to view and define rules among various augmented objects in the environment. In the traditional augmented reality techniques on mobile devices, the users must hold the devices in their hands as a viewing window to the physical world. Instead, the OHMD are attached to the users' head, allowing them to view objects and related information by turning their heads and directly gazing at the objects. In a crowded urban environment surrounded by IoT devices, OHMD can reveal the presence and capability of the devices to users. Also, free movement of hands leave an opportunity to use them for interaction, e.g. via mid-air gestures. This paper describes an early prototype and a preliminary result from a pilot study to test feasibility of interacting with virtual objects that augment physical objects using OHMD. We found that basic interaction for building new rules among them were easy to learn and use, while fine-tuning of them using the conventional GUI components left rooms for improvement.

CCS Concepts

• Human-centered computing → Mixed / augmented reality

• Human-centered computing → User interface design

Keywords

Internet of Things; Smart Cities; Augmented Reality

1. INTRODUCTION

As the use of head-mounted virtual reality displays and their technology emerge, interaction for manipulating virtual objects in a virtual world has been extensively studied. Also rapid increase in the interest in the optical head-mounted display (OHMD) and Internet of Things (IoT) in recent years has shown potential for integrating objects in the physical world with digital bits that are highly malleable to changes.

On the mixed reality [6] continuum, the use of head-mounted virtual reality displays (commonly referred to as HMD in the past) provides the visual feedback that is limited to virtual environment. On the other hand, OHMD allows a user to 'see-through' transparent glasses on which computer generated images are projected. This allows the IoT devices present in the urban environment to be manipulated via the virtual objects that augment the physical counterparts. Because managing abundant devices on a single dashboard displayed on a screen can be counterintuitive due to lack of mapping between physical and virtual objects, the benefit of AR surges as the number of devices and their interaction gets large [5].

Virtual objects can provide not only alternatives to input methods available on physical objects, but also ways to define programmable "rules" that can trigger or be triggered by other objects to build synergistic effect. For example, users may want the overhead lights' intensity to change automatically depending on a value read from an ambient light sensor. Making such rules based on a relationship requires mapping the output value of a sensor and the input of light bulbs. However, unless every object provides configurable interface on the surface, defining complex trigger-handler rules may not be possible. Therefore, an efficient interface needs to be designed for defining such rules.

To achieve this, some early prototypes have utilized augmented reality technology implemented on the mobile devices [4][5][7]. The major disadvantage of such is that the size of viewing window is limited to the viewing angle of the camera and screen size of the device. It requires users to scan through the surrounding by holding the device in multiple angles, in which case the hands are occupied. In addition, since the virtual objects are projected onto a 2D image plane, the sense of depth may fade and may cause issues when interacting with the objects hidden behind the others from the viewer's perspective. For a similar reason, it is difficult to place purely virtual objects in mid-air. Using OHMDs, users view the environment by turning their heads or eyeballs, much closer to natural behavior of exploration while reserving hands for interactions. Furthermore, the virtual objects are within the context of other physical objects with a help of the stereo depth perception. For example, advertisers may place a dynamic virtual billboard that are accustomed to each consumer in various places in the shopping mall and have them updated with data from the nearest store.

This paper describes an early prototype (Fig. 1) and a preliminary result from a pilot study to test the feasibility of defining rules among the objects using OHMD with mid-air gestures. Our method takes advantage of virtual presence of objects in the physical world to visualize the trigger-event rules with connected lines and allow users to add new rules or remove current ones. Also, when a rule between two objects cannot be established directly due to the mismatch in data type, users may create it with

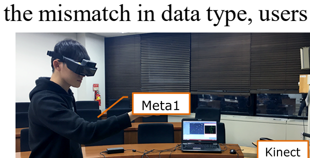

Figure 1: Meta1 [1] detects rotation and hand gestures while Kinect tracks absolute position of a user's head.

Urb-IoT'16, May 24 - 25, 2016, Tokyo, Japan

ACM 978-1-4503-4204-9/16/05…$15.00
DOI: http://dx.doi.org/10.1145/2962735.2962746

a data converter. The behavior of a data converter can also be customized to meet users' need.

2. INTERFACE DESIGN

As a user wearing an OHMD looks around a place with multiple smart objects, white-colored "virtual triggers" appear on top of each corresponding objects so that the user can manage relationship among them (Fig. 2 left). Our interface includes three major functionalities: *rule creation*, *rule removal* and *rule remapping*.

Rule creation: To initiate the rule creation between objects, a user needs to grab a virtual trigger that corresponds to a real object. As soon as a user grabs it, an *agent* appears and follows his/her hand while holding. Meanwhile, other triggers turn into one of three types of *handlers* that can take events from the selected trigger based on the compatibility of the data between a trigger and their handlers. Each state is notified to users via colors. Green indicates it can directly take events from the trigger without any conversion, yellow indicates it requires type conversion and red indicates it cannot take any events from the trigger (Fig. 3). Such visual representation enables the user to easily identify objects (*handlers*) that can take events from the selected object (*trigger*).

While the user is holding an *agent*, a triangular directional edge appears from its origination to indicate the direction of the *trigger*-*handler* relationship and one can drop the agent onto a *handler* to create a rule (Fig. 3). When the rule is successfully created, the *agent* is attached to the *handler* like a 'satellite' and if the rule requires a data conversion, a magenta-colored cube appears in the middle of the edge (Fig. 3 right). When the user grabs the cube, a conversion panel appears as a head-up-display (HUD) that is always in front of his/her head, so that he/she can view or configure it (Fig. 4). During this time, all other virtual objects are hidden to prevent confusion and help user focus solely on the conversion panel. Although each edge is single directional, a user can create an additional rule with an opposite direction if bi-directional relationship is viable. Furthermore, as there is no limit on the number of inputs and outputs a single object can generate, n-to-m connections can be created with no constraints (Fig. 2 right).

Rule removal: An *agent* connected to a *handler* in the form of a satellite can be removed easily by another drag-and-drop gesture: Grab the *agent*, detach it from the handler, and then drop it on the empty space. The gesture has been selected over alternative gestures such as the cutting-the-line [4] because users can undo the removal by reattaching the *agent* back to a *handler* when it is mistakenly detached. Also, our method can be more robust in situations where a number of edges of rules are entangled in a small place as the cutting gesture can accidentally remove multiple of unintended edges.

Rule remapping: Each *agent* can be detached from a *handler* and moved to another while keeping the characteristic of the connection. Identical drag-and-drop gesture can be used for the remapping. Also, a *trigger* can be moved to another *trigger*.

3. CONVERSION TYPES

In our design, a single *trigger* or *handler* can have a single type of input or output data respectively while a single object may have multiple *triggers* and *handlers* with different types: Boolean, Integer or (JSON-like) message.

When a *trigger* and a *handler* have identical types, such as Bool2Bool, Int2Int and Msg2Msg, the connection can be generated without any additional conversion. In cases where they have different ranges, the system will automatically scale them.

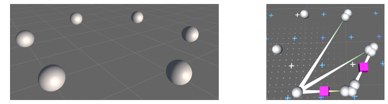

Figure 2: The initial state with no rules (Left). N-to-M rules defined (Right). The figures are designed only to illustrate the system. In the tests, they are objects within AR.

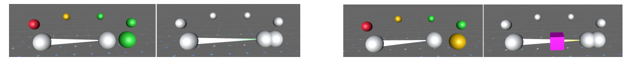

Figure 3: The objects with compatible data type are highlighted green and are directly connected (Left). The ones compatible with a converter are highlighted yellow and are connected with a converter shown as a magenta cube (Right).

The conversion is necessary when a *trigger* and a *handler* have different types, such as Int2Bool, Bool2Int, Msg2Int, and Msg2Bool. In case of Int2Bool, a threshold value needs to be set. (e.g. When the value read from an ambient light sensor is below the certain level, a ceiling light will turn on.) In case of Bool2Int, true and false values from a *trigger* can be mapped to specific numbers of a *handler*. (e.g. When the ceiling light is turned on indicating the presence of a person, the room temperature is set to 25°C, otherwise it is set to 18°C to save energy.) In case of Msg2Int, the converter needs to parse an integer value mapped to a certain string. (e.g. A converter may parse and figure out a corresponding part of the message (alley light = 45) from a *trigger* and let the alley light to adjust accordingly.) In case of Msg2Bool, a converter can determine the existence of a particular string in the message and then assign Boolean value accordingly. (e.g. When the message contains strings "LED" and "Audio," only the LED and Audio turn on while other devices stay off.) We excluded the cases where other types are converted into messages because a simpler data type can be directly processed by the *handers* without relying on converted 'messages.'

4. PILOT STUDY

We recruited 4 participants (all female, M=28.3) to check the feasibility and performance of our prototype. We profiled them to have at least 20/25 normal or corrected vision with contact lens in order for them to wear an OHMD. All were informed with the concept of OHMD, but none have used such devices before.

4.1 Apparatus

We've used Meta1 [1] that has see-through display of 960x540 pixel resolution. The field-of-view was at 35° and rotation of the head movement was tracked with 9-axis Inertial Measurement Units with accelerometer, gyroscope, and compass. Since Meta1 only provides rotational information of the head, we measured the position of the participants' head using Microsoft Kinect. With the combination of both positional and rotational information, we calculated where Meta was facing at and transformed rendered virtual objects accordingly. Mid-air gestures were captured by a depth camera mounted on Meta1. In order to minimize direct

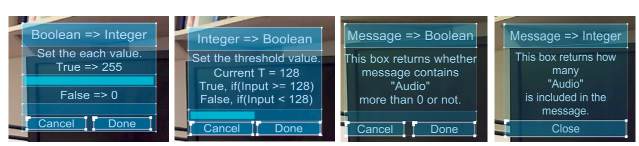

Figure 4: Conversion panels for different data converters. Each shows Boolean-to-Integer, Integer-to-Boolean, Message-to-Boolean, and Message-to-Integer (from left to right).

Table 1. Participants changed the initial thresholds to new thresholds (Left) and the initial mappings to new mappings (Right) for Rule Modification Tasks. The order was shuffled.

Integer-to-Boolean Task		
Trial	Initial Threshold	New Threshold
1	0	128
2	0	255
3	128	0
4	128	255
5	255	0
6	255	128

Boolean-to-Integer Task		
Trial	Initial Mappings	New Mappings
1	T:128 F:128	T:255 F: 0
2	T:255 F: 0	T:128 F:128
3	T:255 F: 0	T: 0 F:255

Table 2: Seeking(Left) and Completion(Right) time for trials

	T1	T2	T3
Avg.	9.91	6.45	21.10
Std.	7.64	5.32	48.96
Med.	8.71	5.18	8.82
Min	0.09	1.46	1.86
Max	63.60	50.81	289.74

	T1	T2	T3
Avg.	10.47	1.05	19.78
Std.	11.72	1.19	9.80
Med.	1.07	0.78	16.50
Min	2.00	0.06	7.68
Max	72.66	8.80	39.43

interference between depth cameras' IR blasts on Meta1 and Kinect, the Kinect was positioned 2 meters away from participants.

4.2 Experiment Design

Upon their arrival, the participants were given introduction on the purpose of the interface. Because Meta1's optimal gesture detection range was around 40cm, they were trained to walk up to virtual objects and grab them mid-air from 40cm away. A total of six virtual objects were placed around a circle so that they were 50-100cm apart from each other to prevent the participants from standing still at the center and stretching their arms to reach them all. The training session lasted as long as they felt comfortable with our prototype. Since the interaction for Rule Remapping requires a same set of gestures as Rule Creation, we only tested the latter only. There were three tasks and a five-minute break was given between each task. The entire experiment, including post experiment survey and interview, lasted for about one hour.

T1. Rule Creation: Participants were asked to create *trigger-handler* rules. For each trial, they saw one object highlighted blue (*trigger*), one in green (*handler*) and the rests in plane white grey. The task was to create a new rule between them by grabbing the blue object and drag it to the green. All combinations for both directions $(\binom{6}{2} \times 2 = 30)$ were tested for each participants. The order was randomized for each participant.

T2. Rule Removal: For each trial, the participants were always presented with two objects with a defined rule. Other objects were still present, but without any rules defined. The task was to identify the ones with a rule and their *handler* to remove the relation. Same as above, there were 30 trials for all combinations and directions for each participant.

T3. Rule Modification: While the prototype supported four types of data conversion, we tested for two that involved modification: Int2Bool and Bool2Int. From our internal test, we found that Rule Modification takes longer time than the previous two. Therefore, we tested each trial once on a randomly picked object pair instead of testing against all permutations.

For Int2Bool task, the participants were asked to open a converter window for a predefined rule and change initial threshold to new threshold (Table 1 left). Because an integer value maybe set at any arbitrary value, we limit our experiment to using "clean cases" and changing to each other; 0, 128, and 255 to denote minimum, middle, and maximum values respectively on our slider (Fig. 4). Unlike previous two tasks where participants had to perform the same tasks repeatedly, a conductor observed the debug console and verbally delivered instructions for each trial. For Bool2Int task, we used the same "clean values" as before for each trial: from both mapped to a middle value to other extremes; from extremes to the same middle value; swap mapping (Table 1 right).

5. RESULT

We measure the time it took for participants to find objects for each trial (Seeking time) and the time to complete an actual task (Completion time) (Table 2). Our average SUS [3] score was 59.4. The scores for the questions I would imagine that most people would learn to use this system very quickly (avg.=3.5, std.=1.5), and I needed to learn a lot of things before I could get going with this system (avg.=2.25, std.=1.1), indicate that our prototype did not have steep learning curve.

6. DISCUSSION

Where our system was most strongly criticized in SUS was consistency. The participants sometimes took tens of seconds to successfully grab an object to initiate the action. We used Meta1's gesture recognition engine, which often failed to recognize the grab gesture and interfered with the Kinect's IR blast. Extreme maximum values (Table 2) far from its average and deviation shows that there were cases where the gesture recognition continued to fail. In an urban environment, wearable devices such as smartwatches can be used as input devices to reduce the interference. Also advances in pure computer vision method may improve accuracy for gesture recognition.

Performance for Rule Modification was not on par with the other tasks. Participants mainly complained about using the conventional GUI (Fig. 4) using mid-air gesture was sub-par experience, and other modals would be more useful. "Using voice or wearable devices like smart watches may be more feasible in this case (P2)."

Participants also have expressed issues regarding the OHMD. All participants expressed that the device must get lighter in order for it to be useful and socially accepted to be worn outside. "*The device was heavy to carry around in head mount. (P1)*" Also, they complained about the cables connected to the computer and it significantly limited its portability. We wonder if such issues can be alleviated if some other devices with smaller form factors like Hololens [2] become available.

On the other hand, participants expressed the grabbing and dragging gesture for making connections was intuitive and fun. Also the removing gesture, similar to the actual behavior of throwing rubbish, was learned quickly. P2 expressed that "*even the number of IoT devices increase around our everyday environment, the method using OHMD using AR technologies may help organize their behaviors and various rules*" which supports our intention.

7. CONCLUSION

We present a pilot study on how optical head-mounted displays (OHMD) with augmented reality techniques can be used in a smart environment with various IoT Devices to view and define rules among virtual objects. We provided mid-air gestures for managing rules. The result of the pilot study demonstrates our early prototype were easy to learn and use, although may not be suitable to use conventional GUI components for manipulation, and the accuracy of gesture recognition could be improved.

8. ACKNOWLEDGEMENTS

This paper was supported by Samsung Electronics. Jinwook Seo is the corresponding author.

9. REFERENCES

[1] Meta Company “Meta 1 Developer Kit” URL: https://www.getameta.com/ (2015)

[2] Microsoft “Hololens” https://www.microsoft.com/microsoft-hololens/ (2016)

[3] Brooke, John. "SUS-A quick and dirty usability scale." Usability evaluation in industry 189, no. 194 (1996): 4-7.

[4] Heun, Valentin, Shunichi Kasahara, and Pattie Maes. "Smarter objects: using AR technology to program physical objects and their interactions." In CHI'13 Extended Abstracts on Human Factors in Computing Systems, pp. 961-966. ACM, 2013.

[5] Heun, Valentin, James Hobin, and Pattie Maes. "Reality editor: Programming smarter objects." In Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication, pp. 307-310. ACM, 2013.

[6] Milgram, Paul, and Fumio Kishino. "A taxonomy of mixed reality visual displays." IEICE TRANSACTIONS on Information and Systems 77, no. 12 (1994): 1321-1329.

[7] Rüth, Jan, Hanno Wirtz, and Klaus Wehrle. "Demo: Ubiquitous interaction with smart objects." In Proceedings of the 12th annual international conference on Mobile systems, applications, and services, pp. 355-356. ACM, 2014.