RV-XoverKit: Mixed Reality Content Creation Toolkit to Connect Real and Virtual Spaces

Yumi Fukuda[1,*], Ayumu Shikishima[1], Asako Kimura[1], Hideyuki Tamura[2] and Fumihisa Shibata[1]

[1] *Graduate School of Information Science and Engineering, Ritsumeikan University, 1-1-1 Nojihigashi, Kusatsu, 525-8577, Japan*
[2] *Research Organization of Science and Technology, Ritsumeikan University, 1-1-1 Nojihigashi, Kusatsu, 525-8577, Japan*

Abstract

We are attempting to systematize technologies used to transmit the dynamic phenomena of objects moving back and forth between real space and virtual space. We refer to this technology as R-V Crossover Rendition. The toolkit that embodies the R-V Crossover Rendition concept is called RV-XoverKit, and this paper describes the design and implementation of the RV-MessengerKit, which is a form of the RV-XoverKit toolkit. The RV-MessengerKit toolkit is designed based on LEGO® Mindstorm® machines and consists of sensors and actuators, the corresponding control units, and the corresponding Application Programming Interfaces. First, we classify the dynamic phenomena to be transmitted, and then we describe the RV-MessengerKit in detail based on the classification results. In addition, we introduce several use cases to demonstrate the practical application of the proposed RV-MessengerKit. We also prepared the implementation of RV-MessengerKit using two different methods in order to examine the difference in time delay between the two methods. The difference between the two methods was whether the values were processed in Mindstorm® or in Unity on a PC. As a result, it was found that the time delay was smaller for processing within Unity than for the Mindstorm's® internal processing.

Keywords

Mixed Reality, R-V Crossover Rendition, RV-XoverKit, RV-MessengerKit

1. Introduction

Interest in mixed reality (MR) technology has been increasing rapidly. With the virtual reality (VR) boom that began several years ago, augmented reality and MR, which are advanced forms of VR, are attracting significant attention. In line with this trend, the supply of various low-cost HMDs and developer tools has increased. Improvements in spatial positioning accuracy and CG rendering capabilities in both real and virtual spaces are driving the development of attractive and practical use cases. In addition to the improved quality of images superimposed on the real world, real-time interaction with the virtual world is also utilized effectively.

Under these circumstances, it is expected that the expressive power of MR content will be enhanced further. However, to the best of our knowledge, there have been virtually no attempts to transmit the dynamic phenomena of objects between real and virtual spaces. Here transmitting the dynamic phenomena of objects means to transmit the movement of an object in the real

APMAR'22: Asia-Pacific Workshop on Mixed and Augmented Reality, Dec. 02-03, 2022, Yokohama, Japan
*Corresponding author.
EMAIL: y-fukuda@rm2c.ise.ritsumei.ac.jp (Yumi Fukuda), sikisima@rm2c.ise.ritsumei.ac.jp (Ayumu Shikishima), asa@rm2c.ise.ritsumei.ac.jp (Asako Kimura), HideyTamura@acm.org (Hideyuki Tamura), fshibata@is.ritsumei.ac.jp (Fumihisa Shibata)

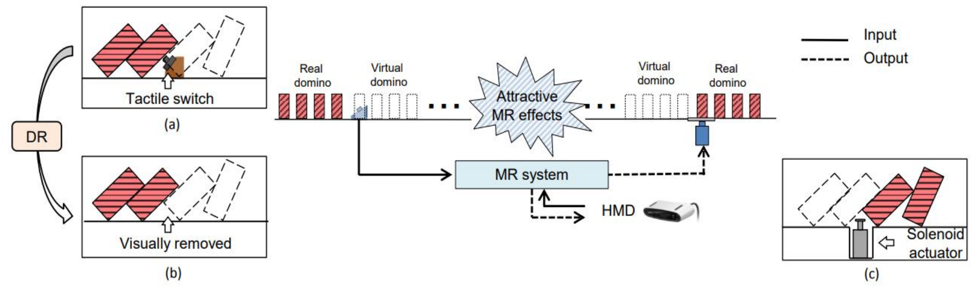


Figure 1: Mechanism to detect and actuate the domino toppling [6][7]

space, e.g., the motion of a ball rolling down a slope, to the virtual space (or vice versa). Superimposition of a moving virtual object on a stationary real scene has been introduced into MR demonstrations using a sword-shaped device with an HMD [1] and in MR attractions that use the user's hand movements as input [2]. However, in these applications, it was difficult to switch seamlessly between moving real and virtual objects. For systems that use shape displays in three different ways to mediate interaction [3] and those that superimposes CG on a small tank robot to embody a battlefield in virtual space [4], both the real and virtual spaces are synchronized to realize interactivity between the real and CG objects. However, these examples have not achieved seamless transitions between dynamic objects. There is another system [5] in which real and virtual dominoes are connected and interact via a special tunnel; however, this is not really a seamless connection between reality and virtuality because it hides the connection where reality and virtuality are interchanged. Though drawing dynamically changing phenomena in a virtual space and superimposing them onto the real space is a natural application, it is rare to represent dynamic objects that go back and forth across the reality and virtuality boundary (R-V boundary).

Our research group refers to this concept as the R-V Crossover Rendition, and our goal is to systematize technology to realize R-V Crossover Rendition. The origin of this research was the production of DOMINO Toppling, which is an MR attraction based on the theme of domino toppling [6][7]. This work received high praise at the ISMAR 2015 technical exhibition; however, it was limited to domino toppling. It treated only dominoes as the target objects moving back and forth between the real and virtual spaces.

The potential targets of R-V Crossover Rendition are extremely wide. It can be deployed in the entertainment and exhibition fields, as well as urban planning and medical product design and manufacturing. Thus, as the next step, we began to generalize R-V Crossover Rendition and organize its concepts and terminology. In addition, we designed a toolkit that can be used by anyone who creates MR content. This toolkit provides an effective mechanism to transmit dynamic phenomena from the real space to the virtual space (or vice versa).

The remainder of this paper is organized as follows. In Section 2, we first organize the concept of R-V Crossover Rendition and define related terms. In Section 3, we describe the design and implementation of RV-MessengerKit, which is the toolkit to realize R-V Crossover Rendition, and in Section 4, we introduce usage examples of RV-MessengerKit and discuss a performance test. Conclusions and suggestion for future work are presented in Section 5.

2. R-V Crossover Rendition

2.1. Overview

There are two reasons why the DOMINO Toppling attraction was well received, i.e., switching between the real and virtual dominoes was seamless and not immediately apparent, and the virtual dominoes behaved without physical limitations. The flexibility in designing content in virtual space is a great attraction for the entertainment and education fields, and the repetitive movement of the R-V boundary enhanced the attractiveness of the system.

Here, the target was limited to domino toppling and was specialized to create a mechanism to detect toppled dominoes and knock down dominoes. In this system, a tactile switch was used to detect whether a real domino tile had been toppled over, and a solenoid actuator underneath the real domino was used to initiate domino toppling by applying force to the bottom of the domino tile. Figure 1 shows the mechanism to detect and actuate the domino toppling process. The information to be transmitted in DOMINO

Toppling is limited to the domino tile toppling phenomena.

To generalize R-V Crossover Rendition, we must consider the type of information that should be transmitted between the real and virtual spaces at the R-V boundary. In some cases, we want to inherit the exact shape and dynamic state of the moving objects when they are at the R-V boundary. In other cases, we want to transmit the information of a result that transforms the dynamic phenomenon of an object to the other space. Here, the transformation may be a nonlinear transformation, a geometric transformation, symbolization, or digitization. The former can be considered information transfer in similar form, and the latter can be considered information transfer in non-similar form.

Thus, we classify R-V Crossover Rendition as R-V Transition, which inherits the shape and dynamic state at the R-V boundary, and R-V Message Transmission, which converts and transmits information at the R-V boundary. In other words, R-V Crossover Rendition is the universal set, and R-V Message Transmission is the complementary set of R-V Transition. Note that the goal of R-V Transition is to transmit the dynamic phenomena of the object as accurately as possible, and R-V Message Transmission makes it possible to transform the object when crossing the R-V boundary or create a different movement.

The toolkits to realize these concepts are called RV-XoverKit, RV-TransitionKit, and RV-MessengerKit, respectively. The RV-XoverKit toolkit is a collection of the other two. Figure 2 shows the relationship among these concepts and terms. Each toolkit comprises hardware units with sensors/actuators and control computers, as well as software modules that operate the hardware units.

2.2. Target Field and Assumptions of Implementation

In the previous section, we organized the R-V Crossover Rendition concept and identified the related toolkits. However, the target field is too broad to design and implement RV-XoverKit. Thus, in this study, we narrowed down the target to chain reaction phenomena in the edutainment field and examined concrete functional design and practical implementation methods.

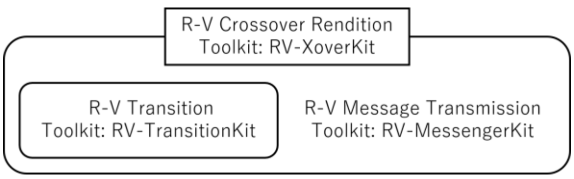


Figure 2: Relationships of the concepts and terminology

Domino toppling is a typical example of a chain reaction, and a more complex example is the “Pythagorean devices” demonstrated in a Japanese educational television program (Pythagora Switch) that has been aired by NHK since 2002. A series of tricks called the Rube Goldberg Machine [8] also falls into this category.

In the following, we consider the realization of such chain reaction phenomena in the MR space and describe the RV-XoverKit. Recall that the target field is edutainment; thus, simplicity is more important than strictness in information transmission at the R-V boundary. Accordingly, the goal is to design RV-MessengerKit, which is one form of the RV-XoverKit.

In this study, we attempted to realize the RV-MessengerKit toolkit to create works that interweave real and virtual spaces in the edutainment field under the R-V Message Transmission concept. For the functional design and implementation of RV-MessengerKit, we used LEGO® Mindstorms® because these devices have been used extensively in the education field, and they include development environment that is suitable for making creative content.

As one of the works that interweave real and virtual spaces, it is conceivable to create Pythagora Switch in MR space, which we refer to as the MR Pythagora Switch. In the following, we describe the design of RV-MessengerKit, which is equipped with the functions required to realize the MR Pythagora Switch. The Pythagorean devices operate only in the real world and only deal with chain reaction phenomena in fixed scenarios; our MR Pythagora Switch enables us to design attractions with branching scenarios.

3. RV-MessengerKit

3.1. Information Transmission Items

Here, we describe the functional design of RV-MessengerKit and present implementation examples. As mentioned previously, the target is narrowed down to chain reaction phenomena in the edutainment field, and we design the functions of RV-MessengerKit assuming the use of LEGO® Mindstorms® for implementation.

The basic set of Mindstorms® includes several types of sensors and actuators, which can be combined to create robots and interactive systems, and many examples that can be used with LEGO® bricks are provided. This time, we implemented our toolkit using Mindstorm®, however it can also be implemented using other small computers such as Arduino, Raspberry Pi, and so on. In that case, it is necessary to start from the assembly of the circuit to use sensors and actuators.

During the design process, we envisioned the MR Pythagora Switch as a specific use case of the RV-MessengerKit, which allows the user to experience chain reaction phenomena. The types of sensors and actuators attached to Mindstorms® are limited, and their accuracy is not relatively good thus, the RV-MessengerKit must be designed according to the types and accuracy of these devices.

As R-V Message Transmission at the R-V boundary, we consider detecting and transmitting the physical state of the target object, e.g., fallen, moved, or rotated. This makes it possible to play the role of a switch or trigger that causes a chain reaction in the other space. Tables 1–4 show the functions realized in RV-MessengerKit.

Table 1 summarizes the type of information transmitted from the real space to the virtual space at the R-V boundary. This information is referred to as the RtoV Transmission Items. When information is transmitted from real space to virtual space, the phenomena in the real space are detected using sensors. Table 1 also describes the sensors used in Mindstorms®.

There are seven types of RtoV Transmission Items. “Tilted” describes the phenomenon that a real object tilts a virtual object and “Pressed” describes the phenomenon that a real object pushes a virtual object. Similarly, “Position” is used to reflect the position of a real object to a virtual object and “Translation” is used to reflect the distance moved by a real object to a virtual object. “Rotation” is used to reflect the rotational angle of a real object to a virtual object. “Brightness” is used when you want the virtual side to reflect the object’s state according to the ambient brightness and “Color” is used to transmit the object’s color in the real to the virtual side.

Table 2 summarizes the type of information transmitted from the virtual space to the real space at the R-V boundary. This information is referred to as the VtoR Transmission Items. When information is transmitted from virtual space to the real space, some action is performed on the real space using actuators based on the state of the virtual space. The actuator that can be used in Mindstorms® is only a motor; thus, we are limited to using this motor.

There are four types of VtoR Transmission Items. “Tilted” is used to transmit tilt of a virtual object to the real and “Pressed” is used to transmit the phenomenon that a virtual object pushes the real object. “Translation” is used to transmit the distance a virtual object has traveled to the real scene and “Rotation” is used to transmit the rotation of a virtual object to the real scene.

Tables 3 and 4 show the parameters for the RtoV Transmission Items and VtoR Transmission Items, respectively. In the following, we describe the usage details of each sensor and actuator.

Touch Sensor

Touch sensors are used for RtoV’s “Tilted” and “Pressed.” The touch sensor attached to Mindstorms® can only detect the binary state of ON/OFF. Thus, RtoV’s “Tilted” does not have a value specified by the content developer and only conveys whether the target object is tilted/not tilted. RtoV’s “Pressed” only conveys whether the target object is pressed/not pressed.

Ultrasonic Sensor

Ultrasonic sensors are used for RtoV’s “Position” and “Translation.” The ultrasonic sensor can detect an object at a distance of 3–252 cm on a straight line from the sensor position. Here, the “Max distance” concept is used to specify how far the ultrasonic sensor will detect. In consideration of actual measurement accuracy, content developers can specify a “Max distance” between 5 and 250 cm. In addition, here the “Number of divisions” is a value to determine

Table 1
RtoV Transmission Items

Detection Target	Sensor	Transmission Item
Real object condition	Touch Sensor	Tilted
		Pressed
	Ultrasonic Sensor	Position
		Translation
	Motor	Rotation
Brightness and color	Color Sensor	Brightness
		Color

Table 2
VtoR Transmission Items

Target to be actuated	Actuator	Transmission Item
Real object	Motor	Tilted
		Pressed
		Translation
		Rotation

how many divisions are to be made between 5 and 250 cm specified by the content developer. When using RtoV's "Position," it is necessary to specify the "Number of divisions." As shown in Figure 3, the distance of the detected object is returned as a value according to the specified "Number of divisions." Here, a value of zero is returned if no object is detected. When using RtoV's "Translation," it is also necessary to specify the "Number of divisions." As shown in Figure 4, the moving distance of the object is returned as a value according to the specified "Number of divisions." The returned "Translation" value takes a positive or negative value. If the value detected by the ultrasonic sensor is greater than the initial position of the object, a positive value is returned. In contrast, if the value of the ultrasonic sensor is less than the initial position of the object, a negative value is returned.

Table 3
Parameters of RtoV Transmission Items

Transmission Item	Parameter	Steps	Return value
Tilted	None	None	True/False
Pressed			
Position	Max distance, number of divisions (#nd)	1–20	0 to #nd
Translation		1–10	−#nd to +#nd
Rotation (Absolute)	Number of divisions (#nd)	1–36	0 to #nd
Rotation (Relative)			−(2 × #nd) to +(2 × #nd)
Brightness (Initial Diff.)	Number of divisions (#nd)	1–5	−#nd to +#nd
Brightness (Latest Diff.)			−(2 × #nd) to +(2 × #nd)
Color	None	None	32 Colors, None

Table 4
Parameters of VtoR Transmission Items

Transmission Item	Parameter	Steps	Argument
Tilted	Rotation speed, Rotation direction	None	Speed, Positive / Negative / OFF
Pressed			
Translation			
Rotation	Rotation speed, Number of divisions (#nd)	1–36	Speed, 0 to #nd

Motor (for sensing)

A motor is originally an actuator; however, if the target object can be fixed with an appropriate attachment, the motor can also measure the rotation angle; thus, the motor is used in RtoV's "Rotation" (Figure 5). Here, the "Number of divisions" is a value that specifies how many divisions of 360° are to be made. Note that there are two types of RtoV's "Rotation," i.e., an absolute value and a relative value. When returning an absolute value, the value increases when the motor rotates clockwise, and the same value is returned when the motor rotates counterclockwise as when it rotates clockwise (Figure 6). As shown in Figure 7, when returning a relative value, the difference between before and after the rotation is returned. Here, the returned value is positive if the motor rotates clockwise and negative if the motor rotates counterclockwise.

Color Sensor

The color sensor of Mindstorms® can measure the brightness of the ambient light and the color of the observed object. Here, "Brightness" and "Color" of RtoV use the color sensor. At startup, RtoV's "Brightness" acquires the initial value and uses this value and the "Number of divisions" to determine brightness changes. The lower range between the minimum brightness value, i.e., zero, and the initial value is divided by the "Number of divisions." The upper range between the initial

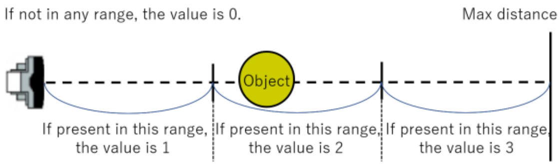


Figure 3: Return value of RtoV's Position

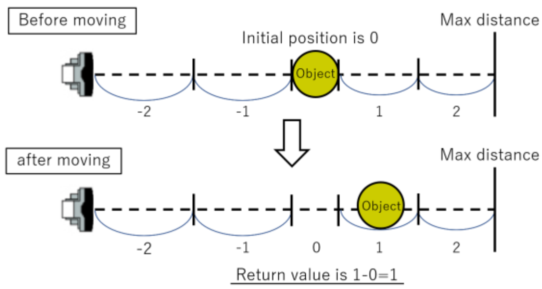
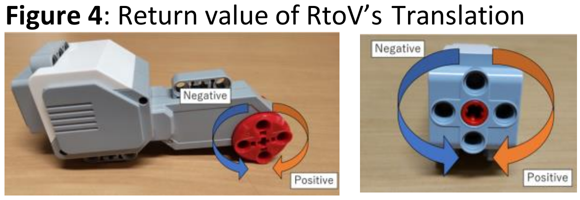


Figure 4: Return value of RtoV's Translation



(a) Large Motor (b) Medium Motor

Figure 5: Direction of rotation for RtoV's Rotation

value and the maximum brightness value, i.e., one, is also divided by the "Number of divisions." The current value is determined according to which of the divided ranges the current ambient brightness is located. RtoV's "Brightness" comprises two types, i.e., one that returns the difference between the initial value and the current value, and another that returns the latest difference between the most recent value and the current value. The former, which is referred to as "Brightness (Initial Diff.)," returns the difference between the initial and current brightness values (Figure 8). The latter, which is referred to as "Brightness (Latest Diff.)," returns the difference between the most recent value and the current value (Figure 9). RtoV's "Color" returns a color name if the color is recognized from the initial state; otherwise, colorless is returned if a color is not recognized. Based on the colors listed on the official LEGO® brick purchase page [9], 32 colors can be recognized by the color sensor.

Motor (for actuating)

Only motors can be used as actuators in Mindstorms®; thus, the "Tilted," "Pressed," "Translation," and "Rotation" of VtoR must be realized using motors. When a motor is used as an actuator, it is essentially controlled by giving "Rotation speed" and "Rotation direction" parameters. Here, "Rotation speed" is given as degrees per second. For "Rotation direction," a positive value is given to rotate clockwise, a negative value is given to rotate counterclockwise, and OFF means no rotation. In the case of "Tilted" or "Pressed" of VtoR, rotary motion is converted into linear motion using a mechanism, e.g., a crank, and is connected to the motion of a real object. However, only in the case of VtoR's "Rotation," it can also be used by specifying two values, "Rotation speed" and "Number of divisions." In this case, a circle is divided according to the "Number of divisions," and the number of the divided range is given to rotate the motor.

3.2. Implementation

According to the specifications described in the previous section, we decided to use the leJOS firmware, which allows Mindstorm® to be programmed using Java rather than the standard software.

Mindstorm® has a variety of parts; thus, there are multiple options for hardware implementation of the Transmission Items described in Tables 2 and 4. For example, Figure 10 shows the hardware units used for RtoV's "Pressed." Here, Type A is designed to allow pressing from a different direction than that of the original touch sensor, and Type B is designed to be attached to an original device created by the content developer.

Figure 11 shows the hardware units used for VtoR's "Rotation." Here, Types A and B

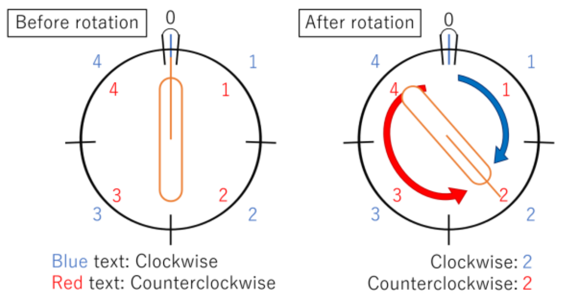


Figure 6: Return value of RtoV's Rotation (Absolute)

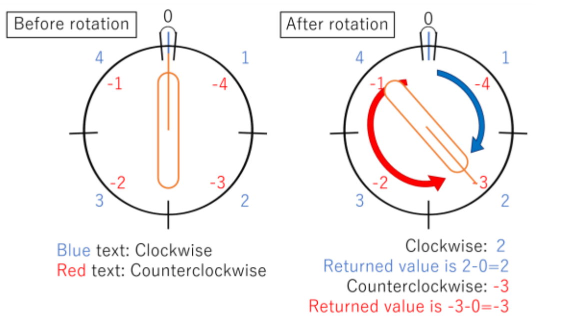


Figure 7: Return value of RtoV's Rotation (Relative)

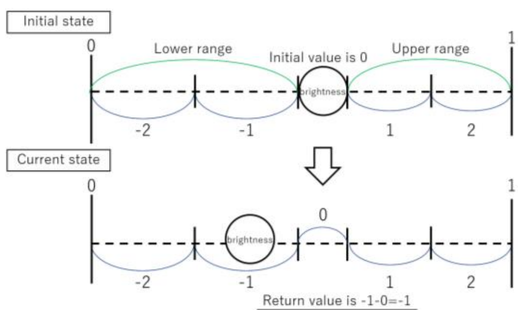


Figure 8: Return value of RtoV's Brightness (Initial Diff.)

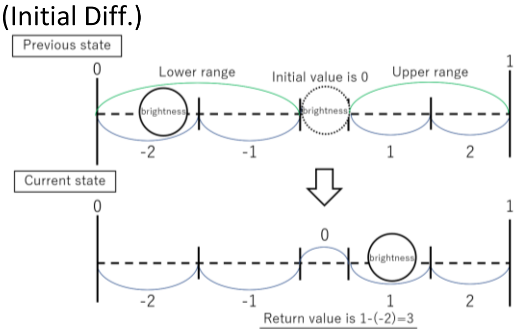


Figure 9: Return value of RtoV's Brightness (Latest Diff.)

correspond to the roll and pitch directions of rotation, respectively. Note that the gear rotation ratio can be adjusted by changing the gear components. As a result, content developers can use different hardware units to develop various use cases. In addition, content developers can use LEGO® bricks to expand their own hardware units.

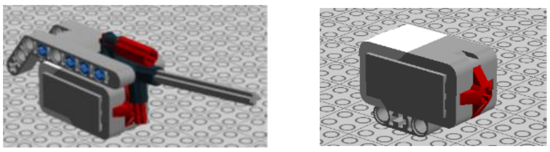
We provide RV-MessengerKit users with blueprints to facilitate the use of the hardware unit. Each blueprint was created using the LEGO® Digital Designer, which is free software that allows users to assemble LEGO® bricks on a PC. The LEGO® Digital Designer has a building guide mode that allows the user to see how to assemble a product in sequence; thus, it is easy to construct the product according to the blueprints.

We recommend using Unity to create MR content with RV-XoverKit, including RV-TransitionKit and RV-MessengerKit. The software module is implemented as a program that runs on Unity and processes the information sent from a hardware unit over network communication. Here, content developers use RV-XoverKit via the Application Programming Interfaces of the software module in Unity rather than directly using the program written on the Mindstorm®.

4. Use Cases

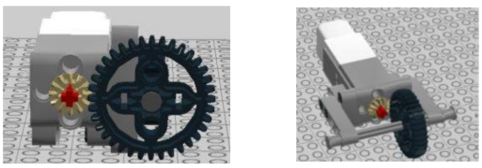
4.1. Simple Examples

In the following, we demonstrate four use case examples of the RV-MessengerKit. Here, we used an HTC VIVE Pro equipped with a Stereolabs ZED Mini to realize the MR experience. HTC VIVE Pro is a video see-through head-mounted display (HMD), and ZED mini is a stereo camera that is used to attach to an HMD.



(a) Type A (b) Type B

Figure 10: Hardware units used for RtoV's Pressed



(a) Type A (b) Type B

Figure 11: Hardware units used for VtoR's Rotation

(A) Example of RtoV's Pressed (Figure 12)

- Object in the real space: A blue marble placed in the upper left corner of Figure 12(b).
- Object in the virtual space: A brown sphere shown in the center of Figure 12(b).
- Phenomena before and after the transmission: The real marble rolls down the slope created with LEGO® bricks and pushes a touch sensor attached to the end of the slope. When the touch sensor is pressed, the information is transmitted from the real space to the virtual space, and the virtual sphere begins to move. As a result, the real marble appears to be flicking the virtual sphere.

(B) Example of VtoR's Pressed (Figure 13)

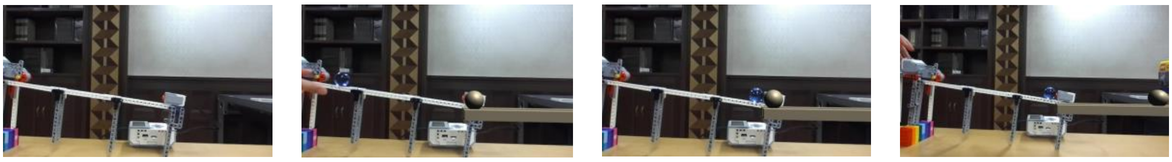
- Object in the real space: A blue marble placed in the center of Figure 13(b).
- Object in the virtual space: A brown sphere shown on the left of Figure 13(b).
- Phenomena before and after transmission: When the virtual sphere rolling on the rail touches a real marble, the information is transmitted from the virtual space to the real space, and the motor rotates to flip the real marble. As a result, the virtual sphere appears to push the real marble.

(C) Example of RtoV's Rotation (Absolute) (Figure 14)

- Object in the real space: A straight LEGO® beam attached to a motor.
- Object in the virtual space: A turntable and numbers displayed on the turntable.
- Number of divisions: four
- Phenomena before and after transmission: When the real motor is rotated manually, the information is transmitted from the real space to the virtual space, and the division range corresponding to the current angle of the straight beam is detected and output as a number. Based on the output number, a virtual number is then displayed on the virtual turntable.

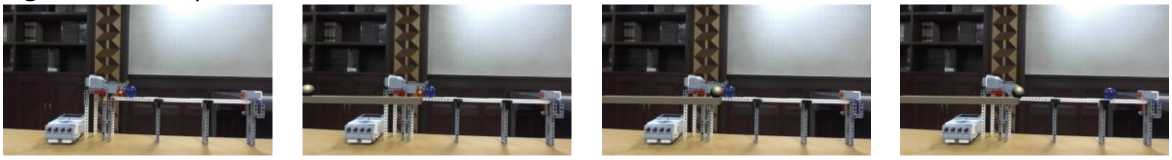
(D) Example of VtoR's Rotation (Figure 15)

- Object in real space: A straight LEGO® beam attached to a motor.
- Object in virtual space: A turntable and numbers displayed on the turntable.
- Number of divisions: four
- Phenomena before and after transmission: The rotation direction of the motor changes



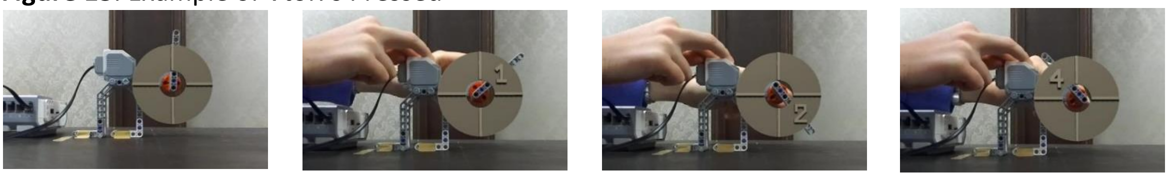
(a) Reality only (b) Initial state (c) In-progress (d) Post-transmission

Figure 12: Example of RtoV's Pressed



(a) Reality only (b) Initial state (c) In-progress (d) Post-transmission

Figure 13: Example of VtoR's Pressed



(a) Initial state (b) Division range 1 (c) Division range 12 (d) Division range 4

Figure 14: Example of RtoV's Rotation (Absolute)

(a) Initial state (b) Division range 1 (c) Division range 4 (d) Division range -3

Figure 15: Example of VtoR's Rotation

according to the numbers displayed on the turntable. If a positive number is shown, the straight beam rotates clockwise (and vice versa). In other words, when a virtual number is displayed, the straight beam moves to the same division range as the number displayed on the turntable.

In addition to these examples, we also demonstrated MR Pythagora Switch [10]. A demo video of MR Pythagora Switch 3rd is available at https://youtu.be/PRwYFJksrKo.

This is a practical example of the "Pythagorean devices" using the RV-XoverKit. Figure 16 shows the overall view of MR Pythagora Switch taken from the real space. In this figure, there were no virtual objects, thus the Pythagorean devices did not connect each other. Figure 17 shows MR Pythagora Switch as seen by a person wearing and experiencing an HMD. The figure shows the Pythagorean devices run while alternating between reality and virtuality.

When we asked the participants to use RV-MessengerKit and surveyed their impressions of its use, we received the following feedbacks. The toolkit was implemented using Mindstorm®, so it was easy to use once the sensors and actuators to be used are installed and the parameters to be used in Transmission Items are specified. Moreover, the parameter "number of divisions" was also introduced, allowing the range used for judgment to be intuitively divided and used, and some said it was easier to use than receiving the actual values of the sensors. Regarding the connection between the real and virtual spaces, we got the answer that even if there was a delay, using the toolkit makes it possible to create what appears to be a smooth connection between the real and virtual objects.

4.2. Performance Test

RV-XoverKit transmits information from the real space to the virtual space (or vice versa) for processing. As a result, the time delay in information transmission may become a problem. The current implementation of RV-MessengerKit is intended for use with Unity; thus, there may be a time delay due to the communication speed before the sensor information reaches Unity from the Mindstorm® or before the actuator's operation instructions from Unity reach the Mindstorm®.

In the following, we describe the implementation of RV-MessengerKit using two methods in order to examine which method can convey information with less time delay. The first method is the Mindstorm's® internal processing method, where the process to convert the acquired value is performed within the Mindstorm®, and only the converted values are sent to Unity. The other method involves processing within Unity. Here, all processing is performed on the Unity side.

The three Transmission Items to be examined are VtoR's "Tilted," VtoR's "Pressed," and VtoR's "Translation." We did not consider RtoV's Transmission Items because if we attempted to measure the time delay, we would need to measure the time from when the object touched the sensor until the time at which the virtual object began to move, and we decided that it would be difficult to determine the start time. Here, the measurement time is the time from the moment a virtual object contacts a specific point where it is assumed to have touched a real object to the moment Unity receives a signal that the actuator, i.e., a motor, has moved.

Figure 16: Overall view of MR Pythagora

Figure 17: HMD's point of view during MR Pythagora experience

Table 5
Results of time delay measurements in second

	VtoR's Tilted	VtoR's Pressed	VtoR's Translation
Mindstorm® internal	0.267	0.291	0.375
Processing within Unity	0.193	0.183	0.211

The results are shown in Table 5. The results represent the average times of 30 measurements for each Transmission Item (rounded to two decimal places). The results with the Mindstorm's® internal processing method are 0.267 sec for VtoR's "Tilted," 0.291 sec for VtoR's "Pressed", and 0.375 sec for VtoR's "Translation". In contrast, the results with processing within Unity method are 0.193 sec for VtoR's "Tilted," 0.183 sec for VtoR's "Pressed," and 0.211 sec for VtoR's "Translation." As can be seen, the time delay is smaller for processing within Unity than for the Mindstorm's® internal processing for all three Transmission Items. This indicates that the information can be transmitted faster when processing is performed in Unity.

However, the time delay needs to be much shorter. Because the frame rate of the ZED mini used is 60 fps, it is idealy desirable to keep it within 0.01 second. This is the goal we need to aim for in the future.

5. Conclusion

In ongoing research, we are investigating the seamless transition of moving objects between real and virtual spaces to expand the practical applications of MR technologies and realize more attractive MR-based information presentation systems. We believe that systematization of related technologies will lead to the development of new applications of MR technology.

Thus, in this paper, we organized the R-V Crossover Rendition concept, which attempts to systematize technology to transmit the dynamic phenomena of objects in real space to virtual space (and vice versa). To realize this concept, we designed and implemented the RV-MessengerKit toolkit using LEGO® Mindstorms®. In addition, we have introduced use cases of the RV-MessengerKit toolkit and performed a performance test to evaluate the time delay of different processing methods.

In the future, we would like to extend the proposed toolkit to include additional Transmission Items for electricity and magnetism, which were not implemented this time, and to create tools to support the development of MR content using the RV-XoverKit. We are also considering adding a mechanism that generates sound when a real collides with a virtual object (or vice versa), where content developers can decide what kind of sounds to generate using the mechanism.

Acknowledgments

We thank Mr. Junya Ishida, an alumnus of the Graduate School of Information Science and Engineering, Ritsumeikan University, for his guidance and cooperation in this research. We express our sincere gratitude to him.

References

[1] K. Inoue, T. Wada, K. Kitamura, S. Nishino, R. Ichikari, R. Tenmoku, T. Ohshima, and H. Tamura. Kaidan: Japanese horror experience in interactive mixed reality space, SIGGRAPH ASIA 2009, Emerging Technologies, page 75.

[2] M. Takemura, S. Haraguchi, and Y. Ohta. BLADESHIPS -An interactive attraction in mixed reality-, In Proceedings of SIGGRAPH 2004, Sketches, 101.

[3] S. Follmer, D. Leithinger, A. Olwal, A. Hogge, and H. Ishii: inFORM Dynamic physical affordances and constraints, Proc. 26th Symp. on User Interface Software and Technology (UIST 13), pp. 417 - 426 (2013)

[4] M. Kojima, M. Sugimoto, A. Nakamura, M. Tomita, H. Nii and M. Inami: Augmented coliseum: an augmented game environment with small vehicles, First IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP '06), 2006, pp. 6 pp.-, doi: 10.1109/TABLETOP.2006.3.

[5] J. Leitner, C. Koeffel, and M. Haller: Bridging the gap between real and virtual objects for tabletop games. In Workshop at ISMAR, 2007. 41–46.

[6] R. Hirata, T. Ishibashi, J. Qie, S. Mori, F. Shibata, A. Kimura, and H. Tamura: DOMINO (do mixed-reality non-stop) toppling, Proc. 14th Int. Symp. on Mixed and Augmented Reality, 2015.

[7] R. Hirata, T. Ishibashi, J. Qie, S. Mori, S. Ikeda, F. Shibata, A. Kimura, and H. Tamura: DOMINO Toppling: An MR attraction focusing on R-V continuum, Transactions of the Virtual Reality Society of Japan, Vol. 21, No. 3, pp. 463 - 472, 2016 (in Japanese).

[8] Rube Goldberg machine, https://en.wikipedia.org/wiki/Rube_Goldberg_machine (November 4, 2022)

[9] Pick a Brick, https://www.lego.com/en-us/page/static/pick-a-brick?query=&page=1 (January 10, 2022)

[10] Y. Fukuda, A. Shikishima, J. Ishida, A. Kimura, H. Tamura and F. Shibata: The third generation MR Pythagora siblings with RV-XoverKit -An example of using a tool suitable for creating edutainment works-, Proc. of the 26th Annual Conference of the Virtual Reality Society of Japan, 2D1-2, 2021 (in Japanese).