The proposed system can operate a PC using a single operation device and switch an operation target. We select a smartphone as the operation device. Users operate the smartphone through touch, and this can cause mouse events on a PC. The function of switching the operation target is based on a multi-display model. The mouse cursor on the operating PC moves off of its display, and then, the operation target switches. In addition, the proposed system has two functions. The first is to share files between PCs and the second is to share an any-application window and an operation to this window between PCs.

The purpose of our study is to develop an any-application window sharing system. A remote desktop is used to share desktop. However, it shares an entire desktop and may be inappropriate if a user wants to share a specific window. Therefore, several studies have re-ported window sharing system. However, they have versatile problems such as that there are application windows that cannot be shared due to an implementation method and a specific system must be incorporated into an application. To solve it, we develop an any-application window sharing system. Moreover, we implement an access management system to man-age multiple shared windows. It enables to deal with shared windows and requests for an any-application window sharing from browser. We evaluated real-time performance of our system [1]. In this paper, we evaluate efficiency performance of our system newly. Window sharing system is used with using other applications. Therefore, it is important to under-stand how much load our system has on other applications. We measure the memory used by our window sharing system and evaluate load our system gives to other applications and evaluate the memory measured as efficiency performance.

In this paper, we describe the support system and interface, which employ a smartphone as an operation device, for using multiple PCs. The remainder of this paper is organized as follows: In Section Ⅱ, we describe related works. In Section Ⅲ, we explain the user interface (UI) and sharing functions. In Section Ⅳ, we describe the implementation of the proposed system. In Section V, we present an experiment to evaluate its real-time operation and efficiency. Moreover, we discuss experimental results. Finally, we conclude this paper in Section VI.

2 Related Works

Several studies have reported systems that support the use of multiple computers. Ikematsu et al.[2] and Yonezawa et al.[3] presented file sharing interfaces between computers such as an iPad and a PC. Komeda et al.[4] discussed a system for controlling appliances using a single device. They used a smartphone as an operation device in their system. Nacenta et al.[5] addressed object movement in multi-display environments. Sharing content between PCs and operating them using a single device are important functions for using multiple computers efficiently, and several studies have been performed on these subjects. Based on previous research, we focus on the support for sharing content and the ability to operate multiple PCs using a single device. Therefore, we surveyed the studies on systems that em-ploy a mobile device. Baur et al.[6] and Vinayak et al.[7] used a smartphone as an operation device in their system. A smartphone can detect multiple touch gestures. In addition, the study of smartphones introduced into college classes is reported [8]. Considering a use case and an implementation case, we select a smartphone as an operation device for the proposed system.

We have developed a system that can share files and a partial screenshot of a desktop between PCs [9]. However, users can only view the other display as a static image and the

users' operations are not shared. We add window sharing function to our system. Several studies have reported window sharing system. The system reported by Hagiwara et al. [10] can share an application window and users' operations. It is used iPad as browsing device and not able to be used on PC. Ichimura et al. [11] develop the window sharing system. It enables an any-application window to share with other PCs by capturing a part of a desktop and sending the image to other PCs. Fukai et al. [12] integrated audience response system on window sharings system. In this sytem, users select a distiribution area size from predetermined sizes. In these system, appropriate distributions cannot be performed in the case that positions or sizes of windows shared are changed. Yamanoue [13] used P2P technology for window sharings system with a real-time performance. However, an application that his system isn't incorporated into cannot be shared. Moreover, existing studies are not reported a management system considered sharing multiple windows.

An any-application window sharing function could reduce the extra shared space com-pared to a remote desktop. We implemented and added an any-application window sharing function (window sharing function). Its mechanism based on WebRTC [14] to the pro-posed system. Moreover, we implemented its interface. Our system can share a window behind other windows. Therefore, the interface should enable users to select windows be-hind. Yamanaka et al. [15] developed the mouse cursor operation system enables users to operate behind windows with mouse. We use this system and User will be able to select windows behind with mouse click. However, Users need to practice the special operation. We implemented the interface used a list that seems to be familiar for users.

3 Any-Application Window Sharing Mechanism

3.1 Operation Device

The proposed system uses a smartphone as an operation device. Smartphones have many users. Therefore, special devices are not required to use the system. Moreover, we do not need to make changes to a smartphone to modify the system, such as implementing additional functions; only a program of the system is changed. Therefore, considering a use case and an implementation case, we select a smartphone as an operation device for the proposed system.

Fig. 1 shows the UI of the operation. The UI consists of the following three parts: (A) the menu bar, (B) main panel, and (C) subpanel. The menu bar is the part for selecting a view. The possible views are operation view, the view showing connection state, the view showing the information about connected PCs, and the view for setting the system. The selected view is shown under the menu bar. Fig. 1 shows the operation view. The menu bar contains four icons, and users can select a view by touching an icon.

The main panel detects touch events, i.e., touch start, move, and end, which are sent to the PCs. The subpanel detects other touch events, such as touch start, end, flick, swipe, and pinch, which are sent to the PCs. The touch start and end events on the main panel are converted to mouse down and up events. The touch move events on the main panel are converted to mouse drag if the main panel is long pressed; otherwise, they are converted to mouse move. The operation for moving the mouse cursor can be performed in two manners. The first is by touching the panel, and the second is by tilting the smartphone. The system obtains the pitch and yaw of the smartphone from its integrated gyro. Users hold the smartphone and tilt it to the right to move the mouse cursor to the right, and

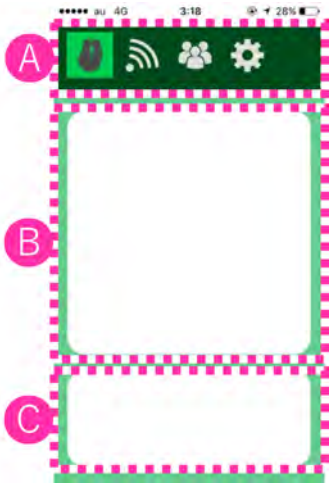

Figure 1: UI for remote device.

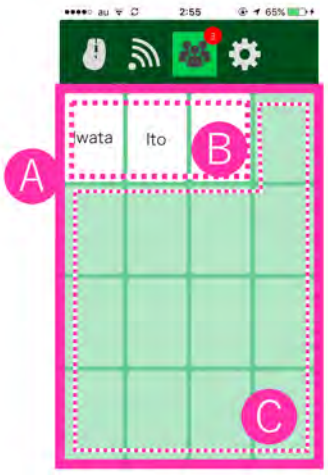

Figure 2: UI for mapping management.

similarly for the cases of left, up, and down motion. Users can select the first or the second method.

Users can change the actions caused by touching operation panels (B) and (C). In the default setting, a user touches the main panel to cause mouse down, mouse up, and mouse move events to be created on the PCs and touches the subpanel to change the scaling of the content in the active window, switch the tab of the active window where applicable, etc. However, if a user wants to create mouse events by touching the subpanel, the behavior may be switched.

Fig. 2 shows the UI of mapping management. (A) The main panel shows connected PCs and their mapping. (B) The white rectangles indicate that a connected PC is stored at that location. (C) Other rectangles indicate that there is no connected PC at that location. Users can switch an operation target or swap a connected PC for another through a touch gesture in this display. For example, the mapping data shown in Fig. 2 contain PC_a and PC_b. The current operation target is PC_a, and users want to switch it with PC_b. This could be achieved by moving the mouse cursor off of the display from the right or by tapping at

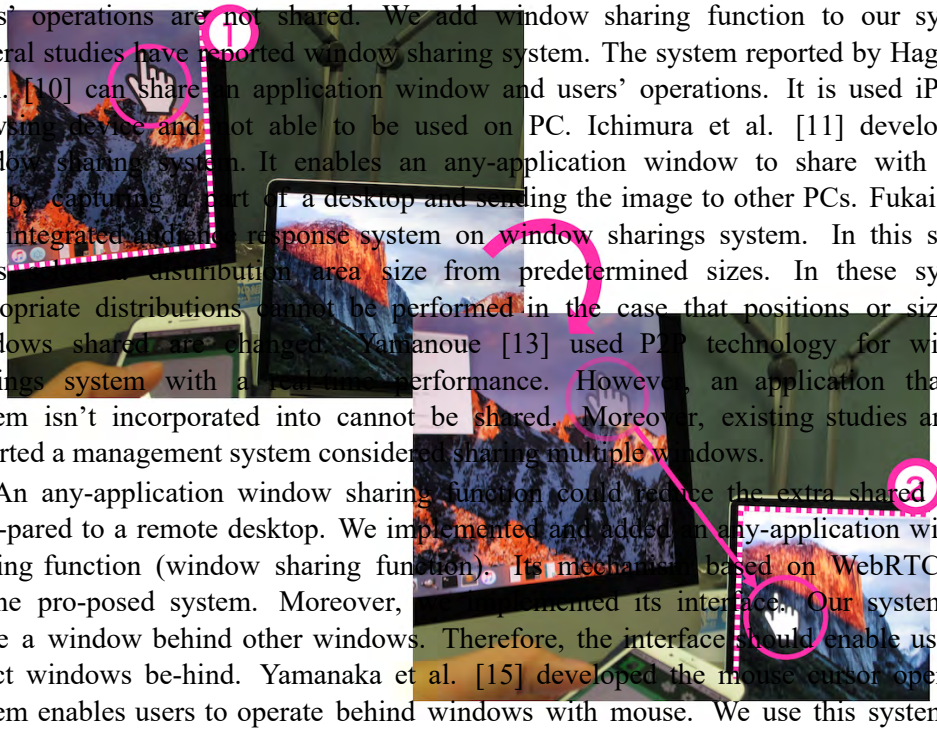

Figure 3: Example of switching operation target.

the coordinate of PC_b. To swap a connected PC for another, users touch start at a target PC's rectangle, move to the desired position, then touch end. In this view, users can set the mouse cursor icon.

3.1 Operation Device

The proposed system uses a smartphone as an operation device. Smartphones have many users. Therefore, special devices are not required to use the system. Moreover, we do not need to make changes to a smartphone to modify the system, such as implementing additional functions; only a program of the system is changed. Therefore, considering a use case and an implementation case, we select a smartphone as an operation device for the proposed system.

Fig. 4 shows the UI of the operation. The UI consists of the following three parts: (A) the menu bar, (B) main panel, and (C) subpanel. The menu bar is the part for selecting a view. The possible views are operation view, the view showing connection state, the view showing the information about connected PCs, and the view for setting the system. The selected view is shown under the menu bar. Fig. 4 shows the operation view. The menu bar contains four icons, and users can select a view by touching an icon.

The main panel detects touch events, i.e. touch start, move, and end, which are sent to the PCs. The subpanel detects other touch events, such as touch start, end, flick, swipe, and pinch, which are sent to the PCs. The touch start and end events on the main panel are converted to mouse down and up events. The touch move events on the main panel are converted to mouse drag if the main panel is long pressed; otherwise, they are converted to mouse move. The operation for moving the mouse cursor can be performed in two manners. The first is by touching the panel, and the second is by tilting the smartphone. The system obtains the pitch and yaw of the smartphone from its integrated gyro. Users hold the smartphone and tilt it to the right to move the mouse cursor to the right, and

3 Any-Application Window Sharing Mechanism

3.2 Switching of Operation Target

The proposed system can operate a PC and switch to a different operation target among multiple connected PCs. We adopt a multi-display model as a method of switching an operation target. To switch an operation target, users move the mouse cursor in the current operation target off of the display. Performing switching using the multi-display model is difficult in cases in which the next operation target is at considerable distance from the previous operation target in mapping data. In such cases, users can switch an operation target by activating the view that shows the information about connected PCs on the smartphone side or operating on the PC side. On the smartphone side, users can change the mapping data of PC screens and switch an operation target. On the PC side, the user of the next operation target PC can call the virtual mouse cursor.

Fig. 5 shows an example of switching an operation target. The upper-left PC, numbered 1, is the previous operation target. The lower-right PC is the next operation target. The circles denote the positions of the mouse cursor, which is moved off of the display toward the right. The next operation target display is to the right of the previous display. In the system, the positions of displays (mapping data) are managed by the management module for the information about connected PCs. The mapping data are in the form of a two-dimensional array. If the element the PC does not exist, the array is null.

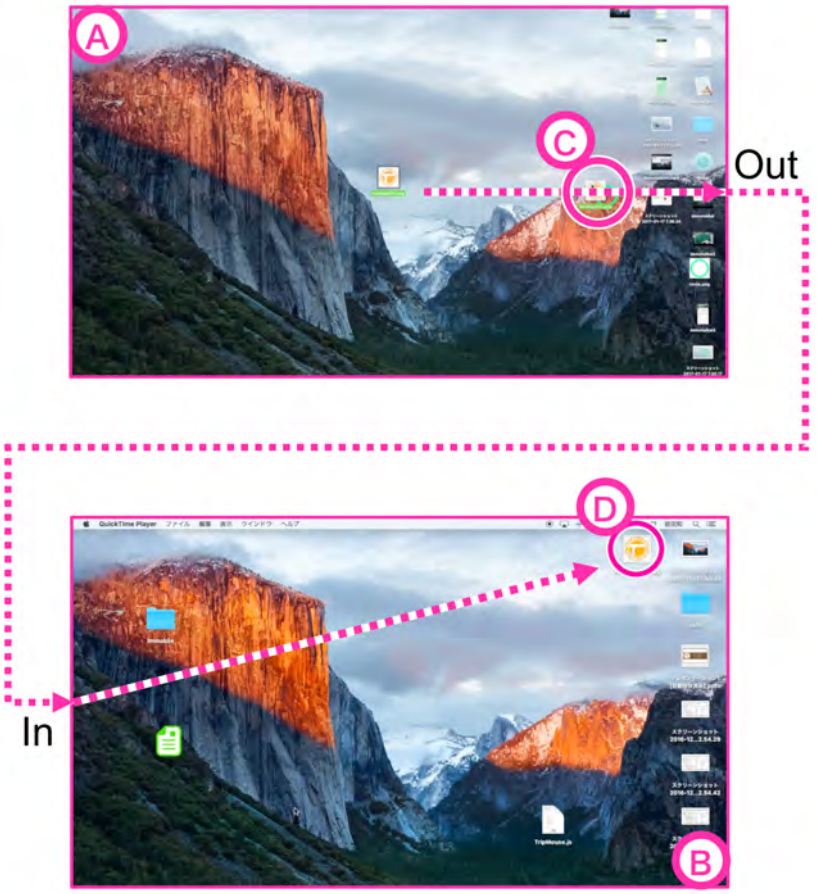

Figure 4: Example of file sharing.

3.3 File Sharing Function

The system can share files through drag and drop. Users can switch an operation target using the multi-display model by dragging files and dropping them on the next operation target. The dropped files are shared between the previous and next operation targets. The files are saved on the desktop of the next target. During target switching, the system determines whether the mouse cursor is dragging files. If it is, then the dragged files are sent to the server, and upon dropping, the files are sent to the next operation target PC. However, if the process of sending the files to the server is not completed, the files are not sent to the server soon. Then, the process is completed and the system starts sending the files to the next operation target.

Fig. 4 shows an example of file sharing execution. A and B are PC screens. C shows the file before being shared, and the mouse cursor drags this file. D shows the file after being shared. File C is dragged on screen A and then out of it from the left side. File D moves onto screen B from the right side. The shared file is copied onto screen B. This function enables users to share files between PCs without using USB memory sticks or cloud storage.

3.4 Window Sharing Function

We add window sharing function to our system. Several studies have reported window sharing system. The system reported by Hagiwara et al. [10] can share an application window and users' operations. It is used iPad as browsing device and not able to be used on PC. Ichimura et al. [11] develop the window sharing system. It enables an any-application window to share with other PCs by capturing a part of a desktop and sending the image to other PCs. Fukai et al. [12] integrated audience response system on window sharings system. In this sytem, users select a distribution area size from predetermined sizes. In these system, appropriate distributions cannot be performed in the case that positions or sizes of windows shared are changed. Yamanoue [13] used P2P technology for window sharings system with a real-time performance. However, an application that his system isn't incorporated into cannot be shared. Moreover, existing studies are not reported a management system considered sharing multiple windows.

An any-application window sharing function could reduce the extra shared space compared to a remote desktop. We implemented and added an any-application window sharing function (window sharing function). Its mechanism based on WebRTC [14] to the pro-posed system. Moreover, we implemented its interface. Our system can share a window behind other windows. Therefore, the interface should enable users to select windows behind. Yamanaka et al. [15] developed the mouse cursor operation system enables users to operate behind windows with mouse. We use this system and User will be able to select windows behind with mouse click. However, Users need to practice the special operation. We implemented the interface used a list that seems to be familiar for users.

4 Implementation

4.1 System Architecture

Fig. 3 shows the architecture of our system. We implemented the system as an iOS application for the smartphone, and as a macOS application for the PCs. Our system contains three major parts: the client side (PCs), operation device side (smartphones), and server side, the client side contains two parts: guest and host. The smartphones are used as operation devices for operating multiple PCs with a single device. For files and application window sharing functions, PCs are used and a server is built. The connection between the PCs and the server is based on Wi-Fi. Wi-Fi is used for sharing files and the application window function. Our system uses the Socket.IO library for Wi-Fi connections. The connection between the smartphone and PCs bases on Bluetooth, which is used for operating multiple PCs with a single operation device. Our system uses the CoreBluetooth API for Bluetooth connections. For files and application window sharing, we use the WebRTC API.

3 Any-Application Window Sharing Mechanism

3.1 Operation Device

The proposed system uses a smartphone as an operation device. Smartphones have many users. Therefore, special devices are not required to use the system. Moreover, we do not need to make changes to a smartphone to modify the system, such as implementing additional functions; only a program of the system is changed. Therefore, considering a use case and an implementation ease, we select a smartphone as an operation device for the proposed system.

Fig. 1 shows the UI of the operation. The UI consists of the following three parts: (A) the menu bar, (B) main panel, and (C) subpanel. The menu bar is the part for selecting a view. The possible views are operation view, the view showing connection state, the view showing the information about connected PCs, and the view for setting the system. The selected view is shown under the menu bar. Fig. 1 shows the operation view. The menu bar contains four icons, and users can select a view by touching an icon.

The main panel detects touch events, i.e., touch start, move, and end, which are sent to the PCs. The subpanel detects other touch events, such as touch start, end, flick, swipe, and pinch, which are sent to the PCs. The touch start and end events on the main panel are converted to mouse down and up events. The touch move events on the main panel are converted to mouse drag if the main panel is long pressed; otherwise, they are converted to mouse move. The operation for moving the mouse cursor can be performed in two manners. The first is by touching the panel, and the second is by tilting the smartphone. The system obtains the pitch and yaw of the smartphone from its integrated gyro. Users hold the smartphone and tilt it to the right to move the mouse cursor to the right, and

The host publishes the application windows to other guests. The guest sees and operates the window shared by host. The guests get the host peer id after connecting to the server. The peer id is necessary for peer-to-peer connections via a web browser. The guests connect to the server, and they try to get the peer id of the host. The access management system of the server in Fig. 5 determines whether the clients can access the host or not. Next, their peer connects to the peer of host with the obtained peer id. The peers send and get media (file, operation, and media stream).

To operate multiple PCs with a single device, users must run the applications in the smartphone and PCs. The applications start running and their connection modules start to connect. The connection modules achieve connection and the management module stores the connected PC information. The management module also stores mapping data of the PC's display. This connection step ends and the system in the smartphone can operate the connected PCs. On the smartphone side, the user inputs operations by touch gestures. The operation information is sent to the system in a PC, which receives the information and sends it to the events management module. This then converts the information to mouse events and functions and sends them to the contents viewer, which creates the converted mouse events and views a shared window.

[7] https://nttcom.github.io/skyway/en/index.html

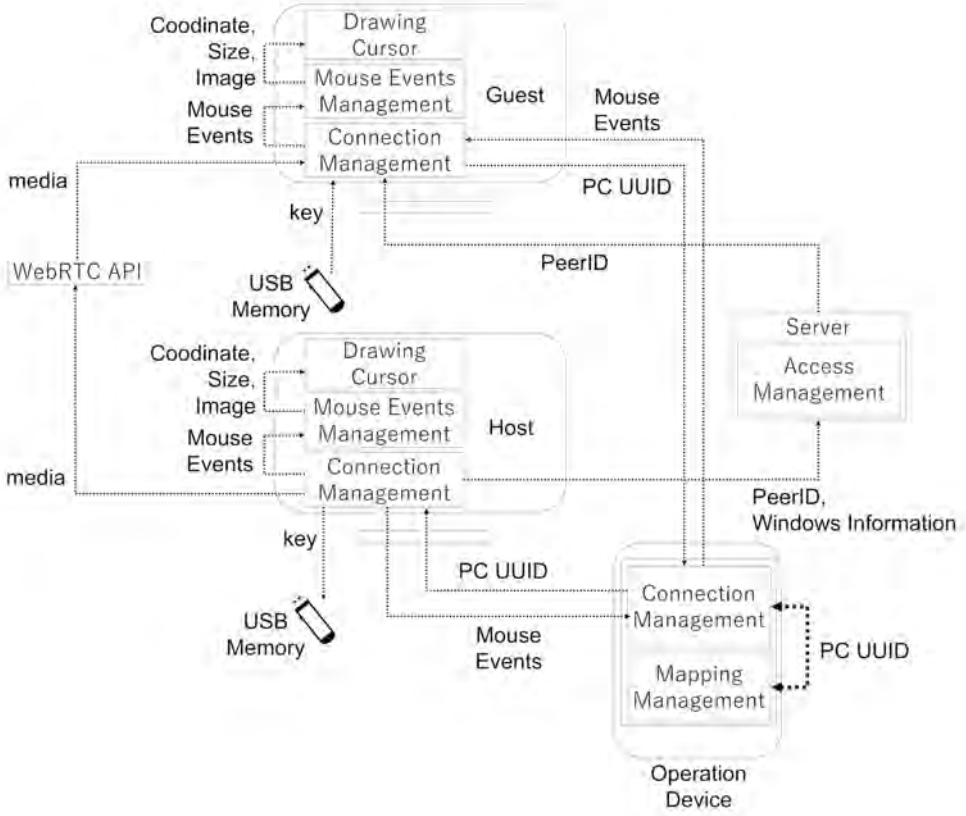

Figure 5: System Architecture.

Our system displays a virtual mouse cursor on the desktop of the current operation target PC. The virtual mouse cursor is displayed differently from the original mouse cursor. Our system enables users to operate the PC by causing mouse events. The mouse move and drag events are caused by our system and the virtual mouse cursor moves; however the original mouse cursor does not move. Other mouse events (mouse down and up) occur at the coordinates of the virtual mouse cursor. This is displayed on top of all the other windows of the desktop and users can change the image and size. The drawing virtual mouse cursor module manages its coordinates, image, and size, and draws it. The mouse events manage module gets the mouse move and drag events, and then moves and redraws it.

Since matching many host windows and many guests are bothering, our system provides a convenient method to connect hosts and guests by using hardware. The method is based on a USB memory, which stores a key to connect hosts and guests. A host selects sharable windows, and then writes a key for sharing the windows onto a USB memory. A guest reads the key from the USB memory to connect the windows on the host. It helps users to share windows simply and easily.

Algorithm 1 Updating an Operation Target.
Input: *data*, *from*, *direction*
Output: *to*
1: $to \leftarrow from$
2: **while** *true* **do**
3: $to \leftarrow next(data, to, direction)$
4: **if** $to = from$ **then**
5: **break**
6: **end if**
7: **if** $isUpdated(to)$ **then**
8: **return** *to*
9: **end if**
10: **end while**
11: **return** *to*

Algorithm 2 Updating the Distances Traveled by the Mouse Cursor.
Input: $p, y, p_c, y_c, p_t, y_t, s$
Output: dx, dy
1: $p_d \leftarrow (p - p_c)$
2: $y_d \leftarrow function(y - y_c)$ (window sharing function).
3: **if** $p_d \geq p_t$ **then**
4: $dx \leftarrow max(p_d - p_t, 0)$
5: **else**
6: $dx \leftarrow min(p_d + p_t, 0)$
7: **end if**
8: **if** $y_d \geq 0$ **then**
9: $dy \leftarrow max(y_d - y_t, 0)$
10: **else**
11: $dy \leftarrow min(y_d + y_t, 0)$
12: **end if**
13: $dx \leftarrow dx \times s$
14: $dy \leftarrow dy \times s$
15: **return** dx, dy

3. Any-Application Window Sharing Mechanism

3.1 Operation Device

The proposed system uses a smartphone as an operation device. Smartphones have many users. Therefore, special devices are not required to use the system. Moreover, we do not need to make changes to a smartphone to modify the system, such as implementing additional functions; only a program of the system is changed. Therefore, considering a use case and an implementation case, we select a smartphone as an operation device for the proposed system.

Fig. 1 shows the UI of the operation. The UI consists of the following three parts: (A) the menu bar, (B) main panel, and (C) subpanel. The menu bar is the part for selecting a view. The possible views are operation view, the view showing connection state, the view showing the information about connected PCs, and the view for setting the system. The selected view is shown under the menu bar. Fig. 1 shows the operation view. The menu bar contains four icons, and users can select a view by touching an icon.

The main panel detects touch events, i.e. touch start, move, and end, which are sent to the PCs. The subpanel detects other touch events, such as touch start, end, flick, swipe, and pinch, which are sent to the PCs. The touch start and end events on the main panel are converted to mouse down and up events. The touch move events on the main panel are converted to mouse drag if the main panel is long pressed; otherwise, they are converted to mouse move. The operation for moving the mouse cursor can be performed in two manners. The first is by touching the panel, and the second is by tilting the smartphone. The system obtains the pitch and yaw of the smartphone from its integrated gyro. Users hold the smartphone and tilt it to the right to move the mouse cursor to the right, and

4.2 Updating an Operation Target

Algorithm 1 gives the next operation target. The input values are the mapping data of the connected PCs, *data*, the previous operation target information, *from*, and the out direction from the previous operation target display, *direction*. The possible values of *direction* are left, right, up, and down. In lines 3 and 6, the function $whereIs(data, elem)$ outputs the index of *elem* in *data*. In lines 4 and 7, the function $next(data, index, direction)$ outputs the element of *direction* from *index* in *data*. For example, if the X-index is rows of mapping data, the right element is the element of the same Y-index and 0 X-index. In lines 5 to 8, the process is searching for the next operation target.

Algorithm 2 gives dx and dy, which are the distances the mouse cursor has travelled along each axis in the current operation PC. This algorithm is used in the smartphone to determine the tilting of the smartphone. The input values are the pitch value p and yaw value y obtained from the gyro sensor in the smartphone, the pitch criteria p_c, yaw criteria y_c, threshold pitch p_t, threshold yaw y_t, and sensitivity parameter s. The values of p, y, p_c, y_c, p_t, and y_t are angles in the range of -90° to 90°. In lines 1 and 2, p_d and y_d receive the angles of the smartphone relative to p_c and y_c. In lines 3 to 11, dx and dy receive the

difference between the thresholds (p_t and y_t) and p_d and y_d. If the absolute values of p_d and y_d are lower than the thresholds, dx and dy are set to 0. Finally, the system multiplies dx and dy by s. The dx and dy values output by Algorithm 2 are floating point numbers. To decrease the length of the data sent to a PC, dx and dy are converted into Int8 numbers.

Touch start and end on the subpanel are used to select an application window for sharing with other PCs. Flick, swipe, and pinch are shortcut operations associated with certain mouse or key events. For example, users can pinch and change the scaling of content in an active window.

4.3 Connection of Window Sharing Function

The connection between the server and PCs is through Wi-Fi using Socket.IO. Socket.IO is a library for real-time communication. In order to share windows, a user must build a server on the PC containing the windows to be shared. Client users connect to the server via a web browser following which the application window images (window images) can be shared and the window names are sent to the client. The window images are images of the windows on the desktop with the server. Client users choose the window image they want to see and operate the application window. The chosen window is displayed in a magnified size and it detects mouse and key events.

This enables the users to operate the application windows in other PCs by creating mouse events on the screen of the target PC. However, the target application window must be active to operate PCs using mouse events; if the target application window is inactive, users cannot operate it. This is the same in our system. For example, the user on the host side published the application windows, A and B. A is on top of all the other windows and B is next to it. Users on the guest side who wish to share and operate B click the shared window. However, the mouse event is not sent to the original window and the user can operate only A. To avoid mouse event conflicts, the users on the guest side cannot operate the second and subsequent application windows. If the window they want to operate is not on top of all the other windows, the user on the host side must activate it. Moreover, if the user on the host side is in the process of dragging the original window, the guest side user cannot drag it simultaneously. This is to avoid conflicts. Many PCs are not designed for concurrent multiple mouse events. To operate the shared window using mouse events, we must consider conflicts, and our system restricts user operations to mouse down and mouse up.

Figs. 6, and 7 show an example of the execution of window sharing. Fig. 6 shows the entire screen on the desktop on the host side. In this screen, there are four windows (text window A, terminal B, window manager C, and web browser D). The windows and entire screen image can be shared with other users. The window images and window names can be shared with other users. Fig. 7 shows the shared window on the client side. It is displayed on a web browser. Each window image of A, B, C, and D is the same as the window images in Fig. 6. The window image E is the entire screen on the desktop on the host side. This UI can be seen on a web browser. Clients can select one window image that they want to share. In this case, the window image A has been selected. The magnified window image is displayed on a web browser. Users can operate the original window using mouse and key events.

We explain how to use a USB memory to share windows. Fig. 8 shows how to select windows to share. The left figure shows the desktop of a host. The host desktop has window A, B and C. D is a window list of the windows to select sharable windows on a

users' operations are not shared. We add window sharing function to our system. Several studies have reported window sharing system. The system reported by Hagiwara et al. [10] can share an application window and users' operations. It is used iPad as browsing device and not able to be used on PC. Ichimura et al. [11] develop the window sharing system. It enables an any-application window to share with other PCs by capturing a part of a desktop and sending the image to other PCs. Fukai et al. [12] integrated audience response system on window sharings system. In this sytem, users select a distiribution area size from predetermined sizes. In these system, appropriate distributions cannot be performed in the case that positions or sizes of windows shared are changed. Yamanoue [13] used P2P technology for window sharings system with real-time performance. However, an application that his system isn't incorporated into cannot be shared. Moreover, existing studies are not reported a management system considered sharing multiple windows.

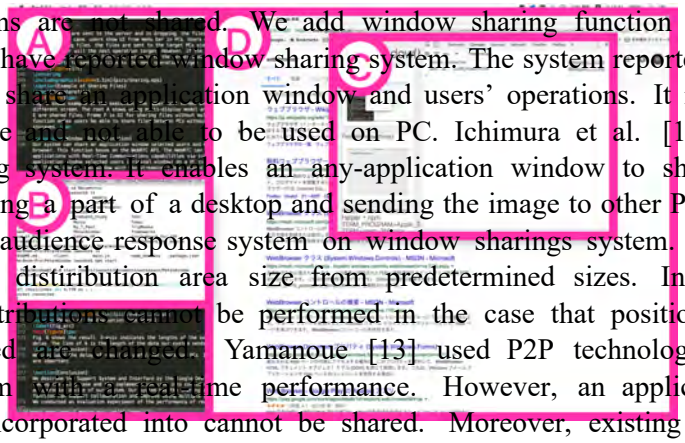

Figure 6: Sharing Application Window: Entire Screen.

An any-application window sharing function could reduce the extra shared space com-pared to a remote desktop. We implemented and added an any-application window sharing function (window sharing function). Its mechanism based on WebRTC [14] to the pro-posed system. Moreover, we implemented its interface. Our system can share a window behind other windows. Therefore, the interface should enable users to select windows be-hind. Yamanaka et al. [15] developed the mouse cursor operation system enables users to operate behind windows with mouse. We use this system and User will be able to select windows behind with mouse click. However, Users need to practice the special operation. We implemented the interface used a list that seems to be familiar for users.

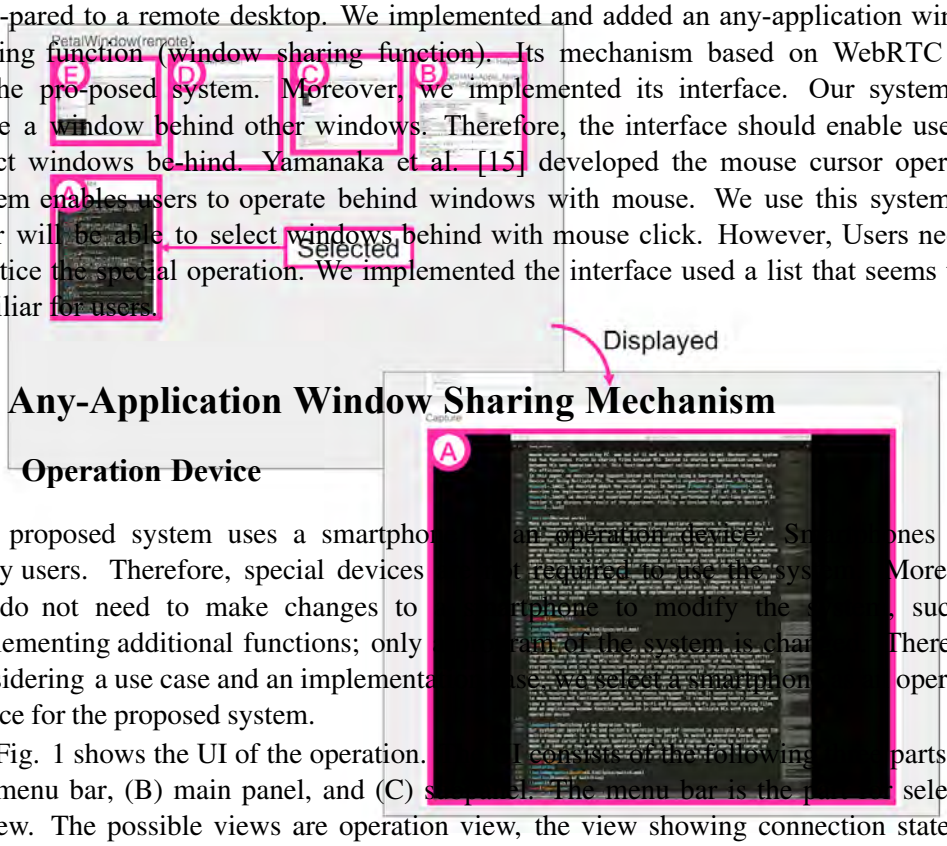

Figure 7: Sharing Application Window Using Dialog Window.

3 Any-Application Window Sharing Mechanism

3.1 Operation Device

The proposed system uses a smartpho[illegible] an operation device. Sm[illegible]ones have many users. Therefore, special devices [illegible]t required to use the sys[illegible]Moreover, we do not need to make changes to [illegible]rtphone to modify the s[illegible], such as implementing additional functions; only [illegible]ram of the system is cha[illegible]Therefore, considering a use case and an implementa[illegible]se, we select a smartphon[illegible] operation device for the proposed system.

Fig. 1 shows the UI of the operation. [illegible] consists of the following [illegible] parts: (A) the menu bar, (B) main panel, and (C) s[illegible]. The menu bar is the p[illegible] selecting a view. The possible views are operation view, the view showing connection state, the view showing the information about connected PCs, and the view for setting the system. The selected view is shown under the menu bar. Fig. 1 shows the operation view. The menu bar contains four icons, and users can select a view by touching an icon.

The main panel detects touch events, i.e. touch start, move, and end, which are sent to the PCs. The subpanel detects other touch events, such as touch start, end, flick, swipe, and pinch, which are sent to the PCs. The touch start and end events on the main panel are converted to mouse down and up events. The touch move events on the main panel are converted to mouse drag if the main panel is long pressed; otherwise, they are converted to mouse move. The operation for moving the mouse cursor can be performed in two manners. The first is by touching the panel, and the second is by tilting the smartphone. The system obtains the pitch and yaw of the smartphone from its integrated gyro. Users hold the smartphone and tilt it to the right to move the mouse cursor to the right, and

host, and is shown as the right figure. Each row consists of a check box, a thumbnail image, and an application name of a window from left to right. For example, row A in the right figure indicates window A on the desktop in the left figure. Checked windows are open for sharing. Sharable windows are highlighted in blue. A window indicated by a cell under a mouse cursor, e.g. C in the list, is highlighted in red, e.g. C in the desktop, to help uses to select the window. When a user wants to finish the selection, the user clicks a button E in the right figure. A dialog appears to select a USB memory. The host generates a key to

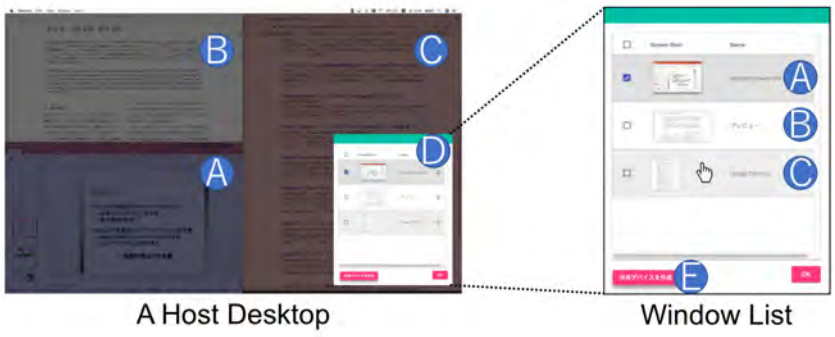

Figure 8: Sharing Application Windows Using USB Memory.

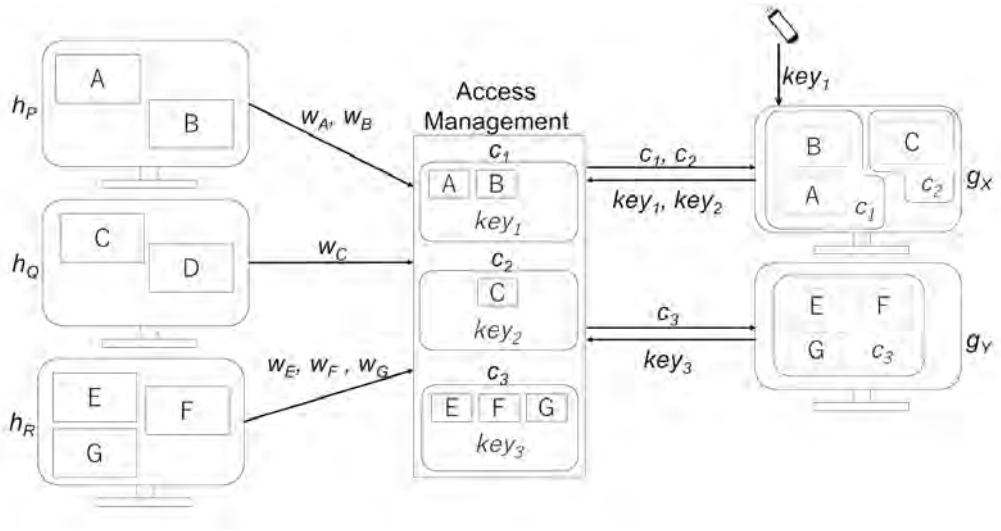

Figure 9: Access Management.

share the selected windows and stores the key into the selected USB memory. Guests read the key from the USB memory and start the sharing process to obtain the virtual windows.

Fig 9. shows the conceptual diagram of access management. In the figure, there are three hosts h_P, h_Q and h_R, and two guests g_X and g_Y. There is the access management in the server. It manages window information sent from hosts and access for window sharing from guests. Hosts send window information w_A, w_B, w_C, w_D, w_E, w_F and w_G to the access management. The access management system generates keys for each group. For example, the access management system generates key_1 for a group of w_A and w_B, constructs c_1=$\langle\{w_A, w_B\}, key_1\rangle$. Similarly for $\{w_C\}$ and $\{w_D, w_E, w_F\}$. Therefore, the access management has three groups c_1=$\langle\{w_A, w_B\}, key_1\rangle$, c_2=$\langle\{w_C\}, key_2\rangle$ and c_3=$\langle\{w_E, w_F, w_G\}, key_3\rangle$. A guest needs to get each key of groups for window sharing. For example, g_X needs key_1 to share windows w_A and w_B. There are two ways to get a key. The first is to select groups on a web browser. The second is to use a USB memory.

users' operations are not shared. We add window sharing function to our system. Several studies have reported window sharing system. The system reported by Hagiwara et al. [10] can share an application window and users' operations. It is used iPad as browsing device and not able to be used on PC. Ichimura et al. [11] develop the window sharing system. It enables an any-application window to share with other PCs by capturing a part of a desktop and sending the image to other PCs. Fukai et al. [12] integrated audience response system on window sharings system. In this sytem, users select a distiribution area size from predetermined sizes. In these system, appropriate distributions cannot be performed in the case that positions or sizes of windows shared are changed. Yamanoue [13] used P2P technology for window sharings system with a real-time performance. However, an application that his system isn't incorporated into cannot be shared. Moreover, existing studies are not reported a management system considered sharing multiple windows.

An any-application window sharing function could reduce the extra shared space com-pared to a remote desktop. We implemented and added an any-application window sharing function (window sharing function). Its mechanism based on WebRTC [14] to the pro-posed system. Moreover, we implemented its interface. Our system can share a window behind other windows. Therefore, the interface should enable users to select windows be-hind. Yamanaka et al. [15] developed the mouse cursor operation system enables users to operate behind windows with mouse. We use this system and User will be able to select windows behind with mouse click. However, Users need to practice the special operation. We implemented the interface used a list that seems to be familiar for users.

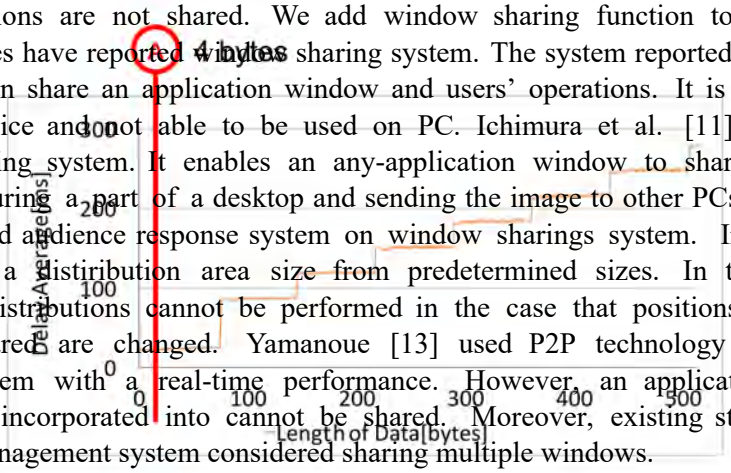

Figure 10. Result of the Evaluation Experiment about Real-Time Operation.

5 Evaluation

5.1 Experiment

The performance of real-time operation is important to efficiently operate PCs. Therefore, we conducted an experiment to evaluate the performance of real-time operation of our system by determining the delays between successive steps in the operation. First, we sent data from a smartphone to a PC and determined the delays. The lengths of the data are 0–512[bytes]. Next, we determined the delays corresponding to data of lengths one hundred times the previous lengths. Finally, we obtained the average of each delay for each length of data.

Fig. 10 shows the experimental results. The X-axis indicates the lengths of the sent data. The Y-axis indicates the average delay. The line A is the length of the data our system sends to operate PC, using mouse events, which is approximately 4 bytes. The average delay of line A is approximately 35 ms, which is the minimum.

We evaluated the efficiency. The window sharing function may be used with other native applications in hosts. Therefore, the load of the function on other applications should be small for using effectively. Then, we measured the memory used our system in a host. It may depend on the number of guests. We increased guests and measure the memory usage. First, we added a guest and a host starts window sharing between the guest and the host. Second, we measured the memory[megabytes] used our system in the host at one millisecond for one minute. Third, we repeated the first and second process until the number of the guests get 30. Finally, we obtained the average of each memory usage of the PCs. The subguests connect to the host, the sizes of the window sharing are 1920 x 1200, and 960 x 600, which are sent to the PCs. The touch start and end events on the main panel are converted to mouse down and up events. The touch move events on the main panel are converted to mouse drag if the main panel is long pressed; otherwise, they are converted to mouse move. The operation for moving the mouse cursor can be performed in two manners. The first is by touching the panel, and the second is by tilting the smartphone. The system obtains the pitch and yaw of the smartphone from its integrated gyro. Users hold the smartphone and tilt it to the right to move the mouse cursor to the right, and

3 Any-Application Window Sharing Mechanism

3.1 Operation Device

The proposed system uses a smartphone as an operation device. Smartphones have many users. Therefore, special devices are not required to use the system. Moreover, we do not need to make changes to a smartphone to modify the system, such as implementing additional functions; only a program of the system is changed. Therefore, considering a use case and an implementation case, we select a smartphone as an operation device for the proposed system.

Fig. 1 shows the UI of the operation. The UI consists of the following three parts: (A) the menu bar, (B) main panel, and (C) subpanel. The menu bar is the part for selecting a view. The possible views are operation view, the view showing connection state, the view showing the information about connected PCs, and the view for setting the system. The selected view is shown under the menu bar. Fig. 1 shows the operation view. The menu bar obtains four icons. The users can select a view by touching an icon.

The number of the guests is 30. Finally, we obtained the average of each memory usage of the PCs. The subguests connect to the host, the sizes of the window sharing are 1920 x 1200, and 960 x 600, which are sent to the PCs. The touch start and end events on the main panel are converted to mouse down and up events. The touch move events on the main panel are converted to mouse drag if the main panel is long pressed; otherwise, they are converted to mouse move. The operation for moving the mouse cursor can be performed in two manners. The first is by touching the panel, and the second is by tilting the smartphone. The system obtains the pitch and yaw of the smartphone from its integrated gyro. Users hold the smartphone and tilt it to the right to move the mouse cursor to the right, and

Fig. 11 shows the experimental results. The X and Y axis indicate the number of guests and the average usage, respectively. The memory usage becomes bigger as the number of guests increases. Moreover, the memory usage of larger window sharing size is bigger than the other.

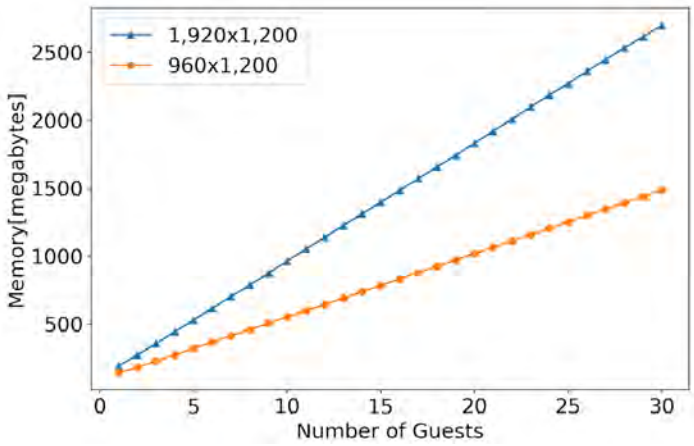

Figure 11: Result of the Evaluation Experiment about Efficiency.

5.2 Discussion

We concluded that the resulting delay of 25 ms is useful for the operation PCs. However, there are two problems related to usability: conflicts and window activation. The first is conflicts in mouse events and key events. For example, a client user operates the remote window with mouse events and key events, and the events are sent to the original window. Unexpected events can arise if the same events occur on the original window such as a simultaneous dragging operation by the server user. This is caused by a conflict of events. PCs in general are designed on the assumption that the points enabled to allow mouse events exist at one place. Therefore, our current system does not allow multiple users to operate a shared application window at the same time. Moreover, this problem occurs when using a smartphone as a single operation device. The second problem is window activation. Key events can occur on an inactive window but mouse events cannot. Therefore, a target application window must be activated to operate a shared application window for a client user. Consequently, the server user cannot activate and operate an application window other than a shared one at the same time when client users are operating a target application window. Solving the two problems would require OS modifications. However, considering the need for versatility, this would be unrealistic and inefficient. We need to survey the literature and conceive a solution for these problems.

The memory requirement of the system is reasonable for contemporary PCs. Sharing a 1,920x1,200 window with one guest requires 90MB memory of a host. For example, 20 guests sharing 4 windows of 960x600, needs about 2GB. In the case that the window sharing size is 1,920x1,200, the memory usage are more than 2 gigabytes finally. In this case, we consider that our system places big load on other applications. Moreover, delay of streaming is bigger as the number of guests increases. In the case that the window sharing size is 960x1,200, the memory usage are about half of the case that the window sharing size is 1,920x1,200. However, in this case, the memory usage are more than 1.5 gigabytes finally. Our evaluations demonstrated that the system works practically while 20 or less guests are connecting to a host PC. We consider that it is necessary to optimize our system to reduce the amount of memory used.

Moreover, we need to conduct further evaluation experiments. For example, an evalu-

Several studies have reported window sharing functions to support collaborative work. [illegible] [10] [illegible] browsing performance [illegible] PCs by capturing a part of a desktop and sending the image to other PCs. Fukui et al. [12] integrated audience response system and window sharing system. In this sytem, users select a distribution area size from predetermined sizes. In these system, appropriate distributions cannot be performed in the case that positions or sizes of windows shared are changed. Yamamoto [13] used P2P technology for window sharings system with a real-time performance. However, an application that his system isn't incorporated into cannot be shared. Moreover, existing studies are not reported a management system considered sharing multiple windows.

An any-application window sharing function could reduce the extra shared space compared to a remote desktop. We implemented and added an any-application window sharing function (Window Sharing function). Its mechanism based on WebRTC [14] to the pro-posed system. Moreover, we implemented its interface. Our system can share a window behind other windows. Therefore, the interface should enable users to select windows be-hind. Yamanaka et al. [15] developed the mouse cursor operation system enables users to operate behind windows with mouse. We use this system and User will be able to select windows behind with mouse click. However, Users need to practice the special operation. We implemented the interface used a list that seems to be familiar for users.

3 Any-Application Window Sharing Mechanism

3.1 Operation Device

We propose a smartphone as the operation device for our system. Smartphones have many [illegible]. Therefore, special devices are not required to use the system. [illegible] the operation device of the system is changed. Therefore, [illegible] smartphone as an operation device for the proposed system.

Fig. 1 shows the UI of the operation. The UI consists of the following three parts: (A) the menu bar, (B) main panel, and (C) subpanel. The menu bar is the part for selecting a view. The possible views are operation view, the view showing connection state, the view showing the information about connected PCs, and the view for setting the system. The selected view is shown under the menu bar. Fig. 1 shows the operation view. The menu bar contains four icons, and users can select a view by touching an icon.

The main panel detects touch events, i.e., touch start, move, and end, which are sent to the PCs. The subpanel detects other touch events, such as touch start, end, flick, swipe, and pinch, which are sent to the PCs. The touch start and end events on the main panel are converted to mouse down and up events. The touch move events on the main panel are converted to mouse drag if the main panel is long pressed; otherwise, they are converted to mouse move. The operation for moving the mouse cursor can be performed in two manners. The first is by touching the panel, and the second is by tilting the smartphone. The system obtains the pitch and yaw of the smartphone from its integrated gyro. Users hold the smartphone and tilt it to the right to move the mouse cursor to the right, and

[illegible] improvements in the efficiency of collaborative work [illegible] We implemented the application window sharing function [illegible] We should evaluate whether this purpose has been achieved. [illegible] real-time browsing performance [illegible] improve the window sharing system [illegible] peer-to-peer connection in our system and conduct a questionnaire survey.

The any-application window sharing system is practical in the case that there are multiple distributors and browsers. Classes, for example. In the case, there are distributors such as teachers and teaching assistants. Moreover, there are browsers such as students. In such the case, a remote desktop is often used to show windows to students. However, each application window has an appropriate size, position and so on, and it depends on a student. Our system enables a teacher to display each application window he wants to show. Each student can see each window sharing in their preferred size, scale and position on his desktop. Moreover, even if a teaching assistant wants to share application windows with students, it is possible to do without stopping windows shared with the teacher. We consider that our system is practical in the case especially the number of distributors is variable.

Next, we mention the outlook for future research. The sharing of application windows and operation functions is supported only in PCs. The function can be used via a web browser, and therefore it does not depend on the OS. For example, a Mac application can be shared with a device running on Windows OS. Our system would be more generic if we introduced this function to iPads. The screen size of iPads is generally smaller than that of PCs. A remote desktop is unsuitable for operating a PC. The function that allows sharing of each individual application window is suitable for smaller screens.

6 Conclusion

We described the support system and interface for using multiple PCs with a single device. We chose a smartphone as the operation device for our system. [illegible] any-application window sharing function [illegible] based on WebRTC [illegible] We conducted an evaluation experiment of the real-time operation performance and efficiency of the proposed system. The real-time performance was evaluated by measuring the delays between an operation device and PCs. Moreover, the efficiency was evaluated by measuring the memory used by our window sharing system. However, there are two problems regarding its usability. The first is mouse and key event conflicts. The second is window activation. We need to survey the literature and conceive a way to solve them. Moreover, we need to conduct evaluation experiments on the efficiency of improving collaborative work and performance of real-time viewing. In this paper, we discussed real-time operation with a smartphone as the operation device. We concluded that the system works practically while 20 or less guests are connecting to a host PC.

References

[1] S. Iwata, T. Ozono, T. Shintani. "Any-Application Window Sharing Mechanism based on WebRTC". 5th International Conference on Smart Computing and Artificial Intelli-

gence, pp.1-6, 2017.

[2] K. Ikematsu, I. Siio. “Memory Stones: An Intuitive Copy-and-Paste Method for Transferring Data between Multi-touch Computers and appliances”. Information Processing Society of Japan (IPSJ) Interaction 2013, pp.80-86, 2013 (in Japanese).

[3] T. Yonezawa, J. Nakazawa, H. Tokuda. “Vinteraction: Vibration-based Interaction for Smart Devices”. Proceedings of the International Conference on Mobile Computing and Ubiquitous Networking, 2015 Eighth International Conference on, Vol.54, No.4, pp.1498-1506, 2013 (in Japanese).

[4] J. Komeda, Y. Arakawa, M. Tamai, K. Yasumoto. “Intuitive Appliance Control Method Based on High-accurate Indoor Localization System”. IPSJ Transactions on Consumer Devices & Systems, Vol.5, No.1, pp.30-37, 2015 (in Japanese).

[5] M. Nacenta, C. Gutwin, D. Aliakseyeu, S. Subramanian. “Object Movement in Multi-Display Environments”. Proceedings of the Human-Computer Interaction, Vol.24, No.1, pp.170-229, 2009.

[6] D. Baur, S. Boring, S. Feiner. “Virtual Projection: Exploring Optical Projection as a Metaphor for Multi-Device Interaction”. Proceedings of the Computer-Human Interaction, pp.1693-1702, 2012.

[7] Vinayak, D. Ramanujan, C. Piya, K. Ramani. “MobiSweep: Exploring Spatial Design Ideation Using a Smartphone as a Hand-held Reference Plane”. Proceedings of the Tenth International Conference on Tangible, Embedded, and Embodied Interaction, pp.12-20, 2016.

[8] K. Ito. “Peta-gogy for Future : A Utilization of Smartphone in Higher Education”. IPSJ journal, Vol.52, No.8, pp.1026-1029, 2011 (in Japanese).

[9] S. Iwata, T. Ozono, T. Shintani. “Developing an Interface with a Smart Phone for Seamless Operation of Multiple PCs”. IPSJ Interaction 2017, pp.452-457, 2017 (in Japanese).

[10] T. Hagiwara, K. Takashima, M. Fjeld, Y. Kitamura. “A Cross-Device Application Sharing Technique using Mobile Camera”. IPSJ Interaction 2017, pp.1-9, 2017 (in Japanese).

[11] Satoshi Ichimura, Naoaki Mashita, Masahito Ito, Ryuya Uda, Kazuya Tago, Yutaka Matsushita. “A Compression Scheme Suitable for Sending Out Desktop Screen to Web Browsers”. IPSJ journal, Vol.46. No.1, pp.70-79, 2005 (in Japanese).

[12] Yuji Fukai, Hiroaki Kawai, Masayuki Kudo. “Development and Evaluation of ARS Integrated Real-time Screen Distribution System”. The Institute of Electronics, Information and Communication Engineers (IEICE) Transactions, Vol.J99-D, No.12, pp.1120-1131, 2016 (in Japanese).

[13] Yamanoue Takashi. “A System Which Shares the Common Operation on a Distributed System in Realtime Using P2P Technology”. IPSJ journal, Vol.46, No.2, pp.392-402, 2005 (in Japanese).

[14] Alvestrand, H. T.: Overview: Real Time Protocols for Browser-based Applications, IETF Internet-Draft draft-ietf-rtcweb-overview-11 (2015).
[15] Shota Yamanaka, Homei Miyashita. "Mouse Cursor Operation for Overlapped Windowing and Its Evaluation", the transaction of Human Interface Society, Vol.15, No.3, pp.313-326, 2013 (in Japanese).

Several studies have reported window sharing systems. Hagiwara et al. [10] can share an application window and users' operations. It is used iPad as browsing device and not able to be used on PC. Ichimura et al. [11] develop the window sharing system. It enables an any-application window to share with other PCs by capturing a part of a desktop and sending the image to other PCs. Fukai et al. [12] integrated audience response system on window sharings system. In this sytem, users select a distiribution area size from predetermined sizes. In these system, appropriate distributions cannot be performed in the case that positions or sizes of windows shared are changed. Yamanoue [13] used P2P technology for window sharings system with a real-time performance. However, an application that his system isn't incorporated into cannot be shared. Moreover, existing studies are not reported a management system considered sharing multiple windows.

An any-application window sharing function could reduce the extra shared space com-pared to a remote desktop. We implemented and added an any-application window sharing function (window sharing function). Its mechanism based on WebRTC [14] to the pro-posed system. Moreover, we implemented its interface. Our system can share a window behind other windows. Therefore, the interface should enable users to select windows be-hind. Yamanaka et al. [15] developed the mouse cursor operation system enables users to operate behind windows with mouse. We use this system and User will be able to select windows behind with mouse click. However, Users need to practice the special operation. We implemented the interface used a list that seems to be familiar for users.

3 Any-Application Window Sharing Mechanism

3.1 Operation Device

The proposed system uses a smartphone as an operation device. Smartphones have many users. Therefore, special devices are not required to use the system. Moreover, we do not need to make changes to a smartphone to modify the system, such as implementing additional functions; only a program of the system is changed. Therefore, considering a use case and an implementation case, we select a smartphone as an operation device for the proposed system.

Fig. 1 shows the UI of the operation. The UI consists of the following three parts: (A) the menu bar, (B) main panel, and (C) subpanel. The menu bar is the part for selecting a view. The possible views are operation view, the view showing connection state, the view showing the information about connected PCs, and the view for setting the system. The selected view is shown under the menu bar. Fig. 1 shows the operation view. The menu bar contains four icons, and users can select a view by touching an icon.

The main panel detects touch events, i.e., touch start, move, and end, which are sent to the PCs. The subpanel detects other touch events, such as touch start, end, flick, swipe, and pinch, which are sent to the PCs. The touch start and end events on the main panel are converted to mouse down and up events. The touch move events on the main panel are converted to mouse drag if the main panel is long pressed; otherwise, they are converted to mouse move. The operation for moving the mouse cursor can be performed in two manners. The first is by touching the panel, and the second is by tilting the smartphone. The system obtains the pitch and yaw of the smartphone from its integrated gyro. Users hold the smartphone and tilt it to the right to move the mouse cursor to the right, and