ScreenJump: An AR-facilitated User-centric Interaction System for Fine-grained Resource Manipulation Across Displays

Xin Zeng[1,2],XinYi Yang[1],TengXiang Zhang[1],YuKang Yan[3],YiQiang Chen[1]
[1]The Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[2]The University of Chinese Academy of Sciences, Beijing, China
[3]Tsinghua University, 30 Shuangqing Rd, Haidian Qu, Beijing, China
{zengxin18z,zhangtengxiang,yqchen}@ict.ac.cn
xyyang030@gmail.com yyk@mail.tsinghua.edu.cn

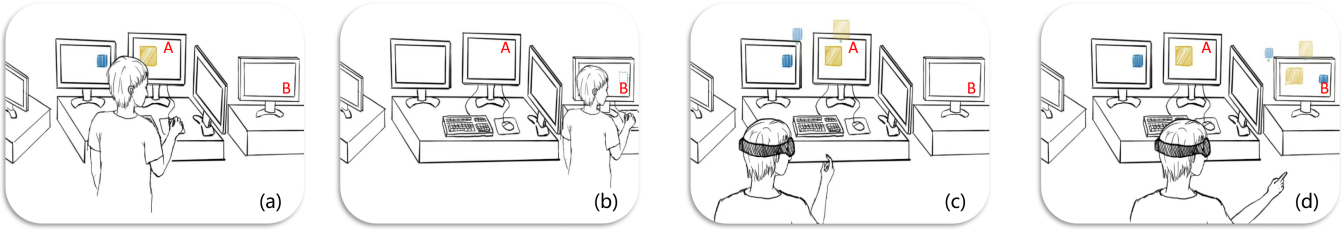

Figure 1: (a-b) Device-centric in traditional multi-device working environment. (c-d) The user-centric approach can reduce workload. Bob is writing a document on Computer A and wants to use a picture displayed by Computer B. For device-centered systems, Bob needs to move to Computer B to copy the picture into a USB flash drive, then return to Computer A to copy the picture into the document. When using our user-centered system, Bob can simply look at and drag the picture, which 'jumps' outside of the screen of Computer B and into the screen of Computer A. He can then focus on the document editing instead of moving between computers.

ABSTRACT

There is an increasing demand for remote manipulation of digital resources across different computers. In this paper, we propose ScreenJump, an AR-facilitated cross-device interaction system that enables user-centric, fine-grained resource manipulation across computer displays. Each computer encodes the identity information into screen blinks that can be detected by cameras but are invisible to human eyes. The AR headset with cameras then localizes surrounding displays and connects with the corresponding computers. The relative positions of fine-grained resources (e.g. pictures, text paragraphs, UI elements) within each display are calculated and shared with the AR headset. Users can then select and manipulate such resources by performing in-air gestures. We explain the system design and implementation in detail, as well as three application use cases that potentially benefit from ScreenJump.

KEYWORDS

Cross-device interaction, Mixed reality, Augmented reality, User Interface Distribution

UX4MDE 2021, May 9, 2021, Yokohama, Japan

ACM Reference Format:
Xin Zeng[1,2],XinYi Yang[1],TengXiang Zhang[1],YuKang Yan[3],YiQiang Chen[1]. 2021. ScreenJump: An AR-facilitated User-centric Interaction System for Fine-grained Resource Manipulation Across Displays. In . ACM, New York, NY, USA, 3 pages.

1 INTRODUCTION AND RELATED WORK

The trend of mobile computing and Internet of Things brings a more diverse working environment in our daily lives. A survey suggests that the average household has 11 connected devices including seven screens [5]. Users usually need to move between different computers to access, transfer, or edit digital resources (e.g. texts, images, videos). Such device-centered interaction systems interrupt the user's on-going workflow and occupy user's work awareness [7–9], which reduces interaction efficiency and deteriorates user experience.

To address this issue, we propose ScreenJump, an AR-facilitated cross-device resource manipulation system that enables a user-centered "request to get" interaction experience. Users only need to perform in-air gestures to access and manipulate resources on different displays, which is more intuitive and efficient than the traditional device-centered "send to distribute" method (Figure 1). Our system has three parts. Each part has its own contribution: (1) *Calm and spontaneous AR-computer communication:* We developed a robust coding and decoding mechanisms for screen-camera communication to establish connections between the AR headset and computers. Each computer encodes information into screen blinks that are imperceptible to human eyes. Cameras of AR headsets

can decode such information and connect with the corresponding computer; (2) *Fine-grained on-screen resource world coordinates calculation*: We proposed a novel method to calculate the world coordinates of resources on each screen. We develop a software tool to automatically calculate the 2D coordinates of resources on the screen, then combine it with the world coordinates of the surface to locate the on-screen resource in real world for rendering digital resource in AR-world; (3) *Augmented resource display and interaction design*: We designed two resource display modes for small and large user-screen distances. We also use gaze to speed up the resource selection and use in-air gestures for intuitive and natural resource interaction.

Researchers have been leveraging AR headsets to improve cross-device interaction experience. Ubii [3] tags physical objects with QR codes for AR recognition and content rendering. However, Ubii only supports file-level transmission and the many QR codes can be visually distracting. Gluey [10] proposes a similar concept to copy resources into the HMD and then paste onto another device for cross-device resource manipulation. SurfaceFleet [2] supports fine-grained cross-device resource sharing. The user still needs to operate on different computers for cross-device operations since the widgets are still device-centered. Compared to previous work, ScreenJump achieves calm and robust AR-computer communication without introducing any visual distraction. Our system also supports manipulation of resources in finer granularity (e.g. pictures and paragraphs within a document) through automatic resource localization in world coordinates. The inside-out tracking of on-screen digital resources, gaze, and in-air gestures enables a true user-centric cross-device interaction experience.

2 SYSTEM DESIGN AND IMPLEMENTATION

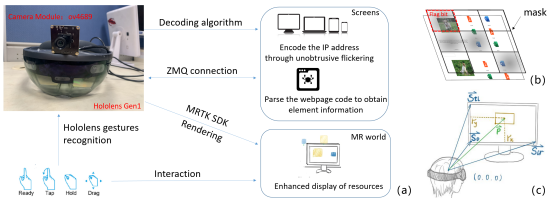

Figure 2: (a) System architecture (b) Encoding mask. The upper left corner is the flag bit, and each of the other eight blocks forms part of the IP address. When the flag bit receives four consecutive zeros or four consecutive ones, it indicates that the information has been aligned. (c) Digital resource world coordinates calculation.

ScreenJump uses an AR headset to locate the on-screen resources both in the digital (i.e. access address) and the physical (i.e. spatial position) worlds. Our prototype system used a Hololens Gen1 AR headset with a 60FPS external camera to capture screen blinks in real-time (Figure 2a). The external camera is connected to a PC via a USB cable. The PC analyzes video streams to decode the screens IP. The external camera and PC is not required if the AR headset has a high frame-rate camera and more computation power. We will explain the design and implementation of the three components of our system in detail in this section.

2.1 AR-computer Connection

ScreenJump uses hidden screen-camera communication technique [6, 12] to establish connection between the AR headset and computers. Computers encode their identity information into screen blinks that are detectable by cameras but invisible to human eyes. Compared to applying QR codes on each screen [3], our method is more aesthetic and flexible. The blinking method can also be easily scaled to resource-constrained IoT devices that only has LED outputs [1, 11].

For information encoding, We divide the screen into nine components and place a mask layer on each grid (Figure 2b) for a more robust communication performance. The brightness of each grid is Frequency-shift-keying modulated (20Hz for 1 and 30Hz for 0). The computer's IP address is encoded into a 32-bit binary string, which is used to calculate the mask's alpha value for each grid. The variation range of α is from 1% to 8%. The change of α weakens from the center to all sides to avoid obvious brightness differences between adjacent grids. For information decoding, the camera on the AR headset continuously captures photos at 60FPS to search for screens by detecting blinks. The sum of pixel's color intensity is calculated using Discreet Fourier Transform (DFT). When four consecutive 0s or 1s on flag bit are detected, the previous 24 frames and the next 48 frames are retrieved for decoding. We conducted a pilot study to evaluate the communication performance both on a 27-inch and a 13.3-inch screen. The communication takes less than 2 seconds when the headset-screen distance is less than 2 meters with a maximal 30 degrees of horizontal angle offset.

2.2 Digital Resource World Coordinates Calculation

ScreenJump calculates the world coordinates of on-screen resources through a three-step process: 1) determines world coordinates of screens; 2) calculates 2D coordinates of the resource on the screen; 3) calculate the world coordinates of digital resources based on the above two sets of coordinates. For screen surface detection, we analyze the surface data known as "meshes" provided by Hololens spatial mapping and group these surfaces into sets. According to the number and relative position of screens in the field of view (FOV), we select the screen surfaces from the surface set and record the world coordinates of these screen's vertices.

For on-screen resource positioning, we develop an on-screen resource relative positioning tool based on HTML. We use a *"is_shared"* tag to identify shareable contents and use the selenium package to obtain the source code of the page and parse out resources with the tag. We calculate the relative positions of the resources by considering the screen resolution and the position of resources in the page. A resource queue is used to manage all shareable resources and their positions on the current screen. Existing web pages only need to add an *"is_shared"* tag to its resources to be compatible with our system. Possible future improvement is to use big data and deep learning methods to analyze on-screen resources' types and locations automatically [4].

The world coordinates of a digital resource can be calculated as follows:

$$\vec{p} = \vec{s_0} + r_x \cdot (\vec{s_{lr}} - \vec{s_0}) + r_y \cdot (\vec{s_{tl}} - \vec{s_0}) \quad (1)$$

$\vec{s_0}$, $\vec{s_{lr}}$, $\vec{s_{tl}}$ represent the world position of the lower left corner, lower right corner and top left corner of the screen respectively. $\vec{s_{lr}} - \vec{s_0}$ is the width vector and $\vec{s_{tl}} - \vec{s_0}$ is the height vector of the screen. r_x and r_y is x, y coordinate of relative position of the UI to the screen respectively. At last, $\vec{p}$ is exactly the UI world position that we want. The vector calculations ensures that our method works with different screen angles (Figure 2c).

2.3 Augmented Resource Display, Gaze Selection, and Gesture Manipulation

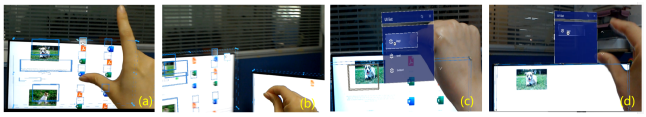

Figure 3: (a-b) The semi-transparent box mode (c-d) The list mode. In both modes, users can use gaze to select digital resources and use tap, hold, and drag gestures to manipulate resources.

We design two resource display modes for ScreenJump: a semi-transparent box overlaid on each resource (Figure 3a) and a list menu containing all resources (Figure 3c). The overlay mode support more intuitive interaction while the list mode enable direct and accurate interaction even when the on-screen resources are small or crowded. Our system creates a 3D object for each on-screen resource. The resource is selected when the gaze collides with the 3D object. The object will change its color to indicate that the corresponding resource is selected. Then user can perform *Tap* gesture (Figure 3a) to select a resource and *Hold* gesture (Figure 3b, Figure 3c) while moving selected resources in the air. When user switches from the *Hold* resource gesture to *Ready* (Figure 3d) gesture, the system detects the gaze direction and determine whether it collides with a screen. If so, the resource is copied and transferred to the same relative location on the new screen. Otherwise the copied resource becomes editable and users can directly manipulate the resource.

3 POTENTIAL APPLICATIONS

Resource Transfer ScreenJump supports calm, intuitive, and efficient resource transfer between computers. For example, during a meeting, a user can quickly assemble a slide by dragging and dropping a table and a picture displayed on two computers of different colleagues.

Instant Servicing ScreenJump can also enable instant servicing by leveraging semantic meaning of the resources. For example, a user can dial a phone number on a web browser by dragging and dropping it onto the 'Call' icon on a smart phone. Similarly, dropping a product picture on a banking APP can initiate the payment process.

Resource Editing Users can drag a picture out of a screen and edit the picture using gestures for a more intuitive and creative experience. It is also possible to drag a PDF page onto a paper, start writing notes on the paper, then take a picture of the notes and drag it back to the computer for future reference.

4 CONCLUSION AND FUTURE WORK

In this paper, we showed the design and implementation of an AR facilitated user-centric cross-device interaction system. We explained in detail how our system supports calm and fine-grained resource manipulation across computer displays. For future work, we plan to conduct formal studies to evaluate the AR-computer communication robustness and resource world coordinates calculation accuracy, as well as comparing the two resource display modes in terms of usability.

ACKNOWLEDGMENTS

This work was supported by National Key Research and Development Plan under Grant No. 2019YFB1404703 and Shandong Academy of Intelligent Computing Technology No. SDAICT2081060.

REFERENCES

[1] Karan Ahuja, Sujeath Pareddy, Robert Xiao, Mayank Goel, and Chris Harrison. 2019. Lightanchors: Appropriating point lights for spatially-anchored augmented reality interfaces. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 189–196.

[2] Frederik Brudy, David Ledo, Michel Pahud, Nathalie Henry Riche, Christian Holz, Anand Waghmare, Hemant Bhaskar Surale, Marcus Peinado, Xiaokuan Zhang, Shannon Joyner, et al. 2020. SurfaceFleet: Exploring Distributed Interactions Unbounded from Device, Application, User, and Time. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. 7–21.

[3] Zhanpeng Huang, Weikai Li, and Pan Hui. 2015. Ubii: Towards seamless interaction between digital and physical worlds. In *Proceedings of the 23rd ACM international conference on Multimedia*. 341–350.

[4] Ran Ju, Xingchen Zhou, Bo Xu, Weiqing Liang, Wanyi Yang, Yuan Cao, Eryan Zhang, Ronggen Li, Yinghao Li, Ning Ding, et al. 2020. DUES-Adapt: Exploring Distributed User Experience With Neural UI Adaptation. In *Proceedings of the 25th International Conference on Intelligent User Interfaces Companion*. 91–92.

[5] Dan Littmann Phil Wilson Shashank Srivastava Kevin Westcott, Jeff Loucks and David Ciampa. 2019. *Build it and they will embrace it:*.

[6] Tianxing Li, Chuankai An, Xinran Xiao, Andrew T. Campbell, and Xia Zhou. 2015. Real-Time Screen-Camera Communication Behind Any Scene. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services - MobiSys '15*. ACM Press, Florence, Italy, 197–211. https://doi.org/10/gg3hbr

[7] Roberto Martinez-Maldonado, Peter Goodyear, Judy Kay, Kate Thompson, and Lucila Carvalho. 2016. An actionable approach to understand group experience in complex, multi-surface spaces. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. 2062–2074.

[8] Stephanie Santosa and Daniel Wigdor. 2013. A field study of multi-device workflows in distributed workspaces. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*. 63–72.

[9] Stacey D Scott, Guillaume Besacier, Julie Tournet, Nippun Goyal, and Michael Haller. 2014. Surface ghosts: promoting awareness of transferred objects during pick-and-drop transfer in multi-surface environments. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces*. 99–108.

[10] Marcos Serrano, Barrett Ens, Xing-Dong Yang, and Pourang Irani. 2015. Gluey: Developing a head-worn display interface to unify the interaction experience in distributed display environments. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*. 161–171.

[11] Jackie Yang and James A Landay. 2019. InfoLED: Augmenting LED Indicator Lights for Device Positioning and Communication. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 175–187.

[12] Wenjia Yuan, Kristin Dana, Ashwin Ashok, Marco Gruteser, and Narayan Mandayam. 2012. Dynamic and invisible messaging for visual mimo. In *2012 IEEE Workshop on the Applications of Computer Vision (WACV)*. IEEE, 345–352.