國立臺灣大學電機資訊學院資訊工程學系暨研究所

碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

Clonemator: 在虛擬實境中組合時空分身以創建互動式自動化工具

Clonemator: Composing Spatiotemporal Clones to Create Interactive Automators in Virtual Reality

林奕碩

Yi-Shuo Lin

指導教授: 鄭龍磻 博士

Advisor: Lung-Pan Cheng, Ph.D.

中華民國 112 年 6 月

June, 2023

doi:10.6342/NTU202303156

致謝

感謝鄭龍磻教授的指導，兩年來不僅在學術研究上給予我極大的幫助，更對我的人生價值觀造成很大的啟發。很幸運能夠加入實驗室，在教授的引導下順利完成有趣的研究題目，這段經歷讓我深刻體會到做研究的真諦。

感謝實驗室成員們從研究初期以來的各種幫忙，包含一起 brainstorming、協助測試系統等，以及在 group meeting 或閒聊時給予我的各種寶貴意見。

感謝口試委員陳炳宇教授、詹力韋教授、蔡欣叡教授，在口試時給予的建議與回饋，讓我能夠將研究題目發展得更完整。

感謝青邑在研究上的幫忙，每次和你討論都可以找到許多新的想法，在我懷疑自己的時候也給了我很大的信心，讓我在 UIST deadline 前有辦法堅持下去，你的加入提升了 Clonemator 的層次。

感謝亦宣常常在實驗室陪我到半夜，在水源和學新館都看過了幾次日出，幫我畫圖、測試、發想，也幫我處理了日常生活中的大小事，讓我能專心在研究中。

最後，我要感謝家人給予我的支持與鼓勵，你們確保我可以無後顧之憂地學習，謝謝你們。

doi:10.6342/NTU202303156

摘要

Clonemator 是一個虛擬實境（VR）系統，讓使用者建立分身並在時空上安排他們，以創建出自動化工具來完成複雜的任務。分身可以（1）定格使用者的身體姿勢成為靜態物體、(2）同步地模仿使用者的動作，以及（3）重播使用者在一段時間中的一連串動作。結合像是縮放、位置重新排列、群組選擇和複製等傳統技術，Clonemator 讓使用者透過將復雜的任務分解為一系列與分身的合作，來迭代開發出客製化且可重複使用的解決方案。這避免了實作專用的互動技術或程式，同時允許在 VR 應用中進行靈活的互動。我們透過多個例子來展示 Clonemator 的靈活度，並透過初步的使用者研究來驗證它的易用性和有效性。最後，我們討論 Clonemator 在遊戲機制、空間互動、多機器人控制等應用的潛力，並為未來的研究提供我們的見解。

關鍵字：分身、超現實互動、虛擬實境

doi:10.6342/NTU202303156

Abstract

Clonemator is a virtual reality (VR) system allowing users to create their avatar clones and configure them spatially and temporally, forming automators to accomplish complex tasks. In particular, clones can (1) freeze at a user's body pose as static objects, (2) synchronously mimic the user's movement, and (3) replay a sequence of the user's actions in a period of time later. Combined with traditional techniques such as scaling, positional rearrangement, group selection, and duplication, Clonemator enables users to iteratively develop customized and reusable solutions by breaking down complex tasks into a sequence of collaborations with clones. This bypasses implementing dedicated interaction techniques or scripts while allowing flexible interactions in VR applications. We demonstrate the flexibility of Clonemator with several examples and validate its usability and effectiveness through a preliminary user study. Finally, we discuss the potential of Clonemator in VR applications such as gaming mechanisms, spatial interaction techniques, and multi-robot control and provide our insights for future research.

doi:10.6342/NTU202303156

Keywords: Clone, Beyond-real Interaction, Virtual Reality

doi:10.6342/NTU202303156

Contents

doi:10.6342/NTU202303156

doi:10.6342/NTU202303156

doi:10.6342/NTU202303156

List of Figures

doi:10.6342/NTU202303156

doi:10.6342/NTU202303156

doi:10.6342/NTU202303156

Chapter 1: Introduction

Virtual reality (VR) has gained popularity with the commercialization of head-mounted displays. Over the years, many researchers have proposed several interaction techniques to enhance experiences in, including the Go-Go technique [23], the Worlds-in-Miniature technique [26], and portals [11].

In recent years, researchers have started exploring the potential of augmenting the human body in VR with duplicated body parts, such as a sixth finger [7], a third arm [4], or supernumerary hands [15]. There are even studies showing extra limbs can improve performance [24]. More recently, researchers also have tried to replicate full bodies in VR, and indicated that it can affect task performance and reduce physical movements [18].

While each of these VR techniques works well by itself, there is hardly any chance of combining them together to create new solutions to solve complex tasks as they usually are dedicated to certain types of jobs. Technically speaking, it is challenging to integrate all these VR techniques into a single system due to the complexity of development and unexpected exceptions. Also, once decided by the VR designers, users have little chance to adjust the settings or parameters of these VR techniques.

In this paper, we look into finding a more general approach that enables a user to decompose complex tasks into smaller, solvable problems by combining the user them-

doi:10.6342/NTU202303156

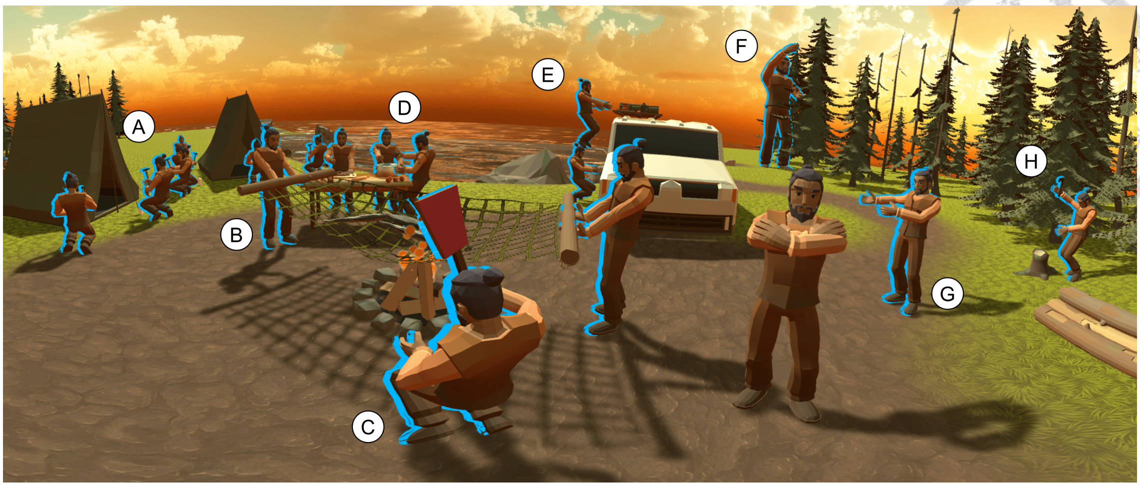

Figure 1.1: Clonemator allows a user to clone their avatars in various spatiotemporal configurations and collaborate with them to achieve complex tasks in virtual reality: (A) a group of clones synchronously follow the user to hammer the tent pegs; (B) a clone mirrors the user's movement to help spread or fold the net; (C) a clone is fanning the fire while (D) the user is chopping food with a group of self-recorded clones helping cook; (E) the user steps on a clone; (F) a giant clone; (G) a body-sign clone; and (H) a remote clone replaying the logging experience previously performed by the user.

self. We draw inspiration from previous work on automation by demonstration, such as Sikuli [31] and SUGILITE [13], and come up with an intuitive and enjoyable way to tackle dynamic challenges. We present Clonemator, a VR system that empowers users to create clones in VR. Configuring and collaborating with spatiotemporal clones offers an intuitive approach to tackle each decomposed problem, as these problems closely align with the real-world challenges that users face in their everyday lives.

Figure 1.1 depicts an overview of a user who uses Clonemator at a campsite. We use this as our example walkthrough to demonstrate how Clonemator works in the following section.

doi:10.6342/NTU202303156

Chapter 2: Example Walkthrough

At the start of the experience, the user finds himself alone at a campsite. His goals are to set up camp with the help of Clonemator, including pitching a tent, preparing food, catching fish and building a campfire.

As the user walks to the tents, he notices many loose pegs surrounding the tents that need to be hammered. The user grabs a hammer and spawns clones in front of the pegs using *Auto Spawning* (Figure 2.1A). Clones are then spawned with the same offset from the corresponding pegs, allowing the user and the clones to hammer the pegs simultaneously (Figure 2.1B). Note that all clones are outlined in blue for easy distinction.

Next, the user decides to catch fish for dinner but finds it difficult to control the fishing net alone. To solve this problem, he uses *Relative Spawning* to spawn a clone in front of the handle on the other side of the net, ensuring that the offsets between the avatars and the handles are the same so that they can grab the two handles at the same time. After creating a synchronous clone, the user mirrors its movement by flipping it along the vertical axis. When the user moves backward, the clone moves backward as well, making it easier to spread the fishing net. On the other hand, when the user moves to the right, the clone moves to the left, allowing them to move in the same direction since they are facing each other. To rotate the net, the user mirrors the clone's movement again

doi:10.6342/NTU202303156

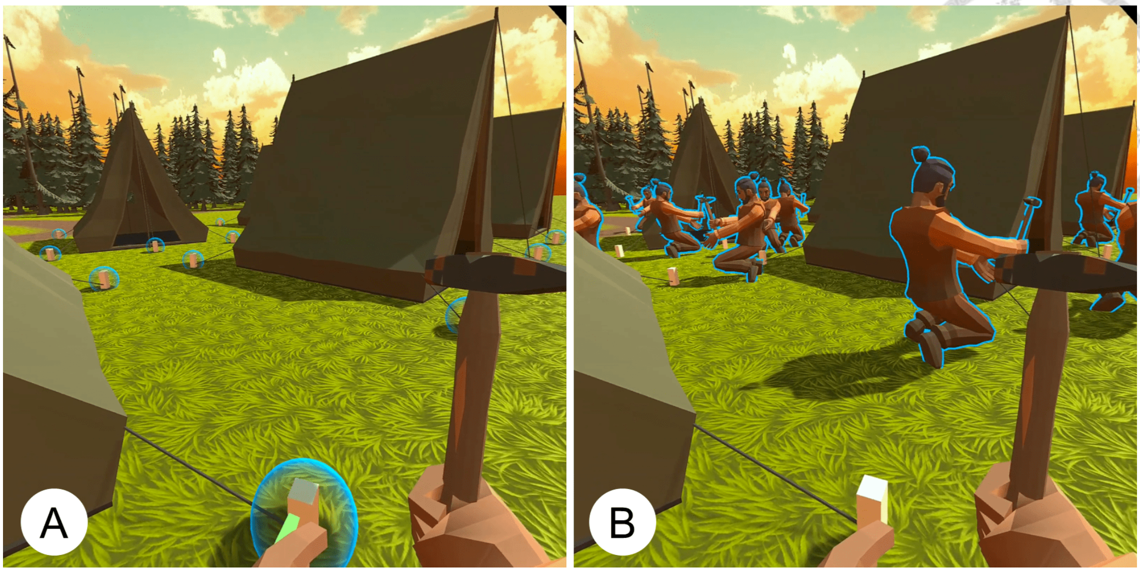

Figure 2.1: (A) When the user hovers on a pegs, other pegs will also be highlighted. (B) The clones will be spawned based on the offset between the user and the selected peg. Our system calculates this offset and applies it to each peg to spawn the corresponding clone, ensuring that all the clones can hammer the peg precisely.

to flip it back, allowing them to move synchronously and rotate in the same direction.

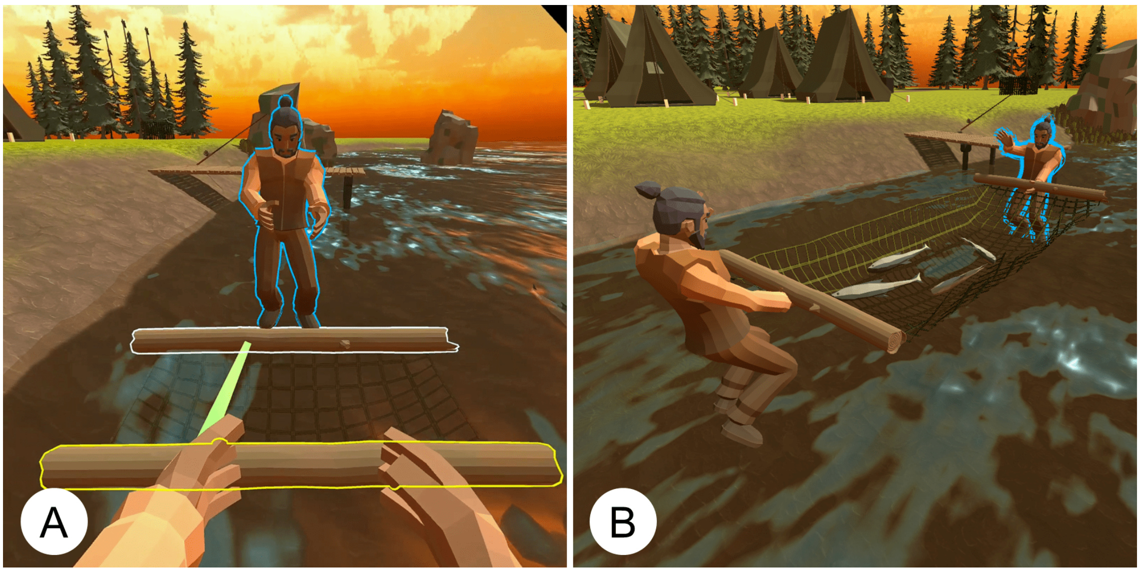

Figure 2.2: (A) The user spawns a clone using *Relative Spawning*, which ensures that the clone can grab the handle of the other side (outlined in white) when the user grabs the handle in the front (outlined in yellow). (B) The user creates a synchronous clone and mirrors its movement, enabling them to move in the same direction and catch the fish together.

Once the user catches the fish, he sets the clone to static mode and performs *Out-of-Body* to create another clone at the exact position. The two clones each help hold one

doi:10.6342/NTU202303156

handle of the fishing net. To move the fish and net near the table, the user can group the two clones first. This allows the user to directly grab one of the clones and move them together without altering the spatial relationships between the clones. Then, the user realizes that he needs to get a slice of beef on top of the van, which is too high to reach. The user crouches and performs *Out-of-Body*, leaving the original body as a clone on the ground. The clone can then serve as a stationary step stool, allowing the user to step onto it and reach the beef (Figure 2.3). Note that in our current implementation, stepping onto a clone is achieved by teleporting to the clone's head since the user's legs are not tracked.

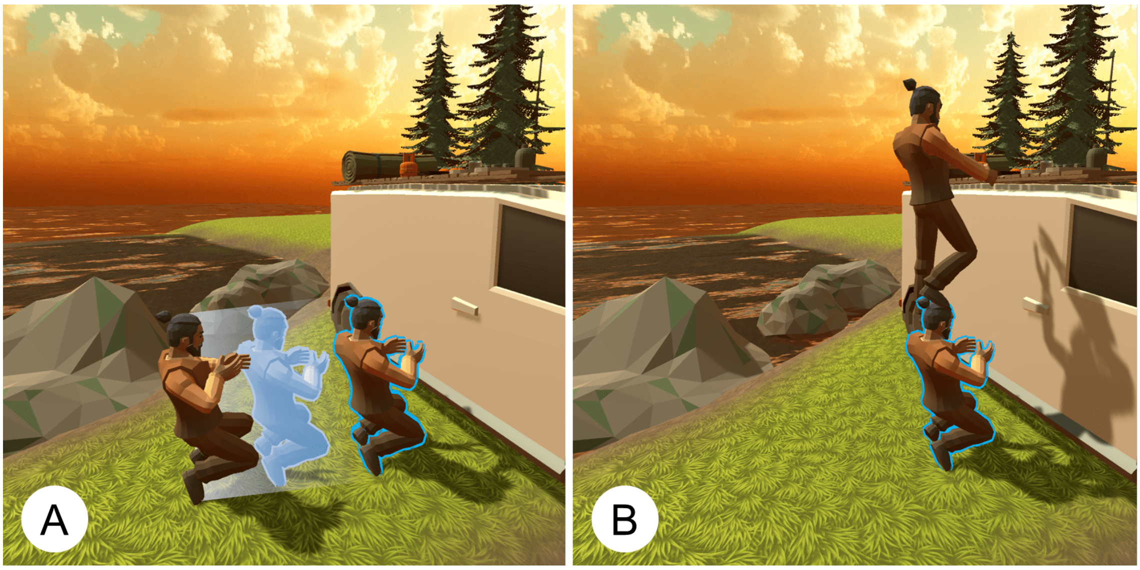

Figure 2.3: (A) The user crouches and performs *Out-of-Body* to create a static clone. Note that the figure has been edited to visualize the avatar's moving trajectory for easier understanding. (B) The clone now functions as a step stool, enabling the user to step onto it and reach the desired object.

The user brings the beef to the table and starts cutting it with a knife but notices that the campfire is almost extinguished. Since the user prefers not to wave his hands twice, he wants to cut the beef and fan the campfire at the same time. The user first spawns a clone near the campfire using *Raycast Spawning* (Figure 2.4A) and switches to it to grab the fan (Figure 2.4B). The user then switches back to the original body and grabs the knife from the table. Finally, the user updates the clone near the campfire to synchronous mode

doi:10.6342/NTU202303156

so that it can mimic the user's movement (Figure 2.4C). This allows the user to cut the beef and fan the campfire simultaneously since both interactions share similar movements (Figure 2.4D).

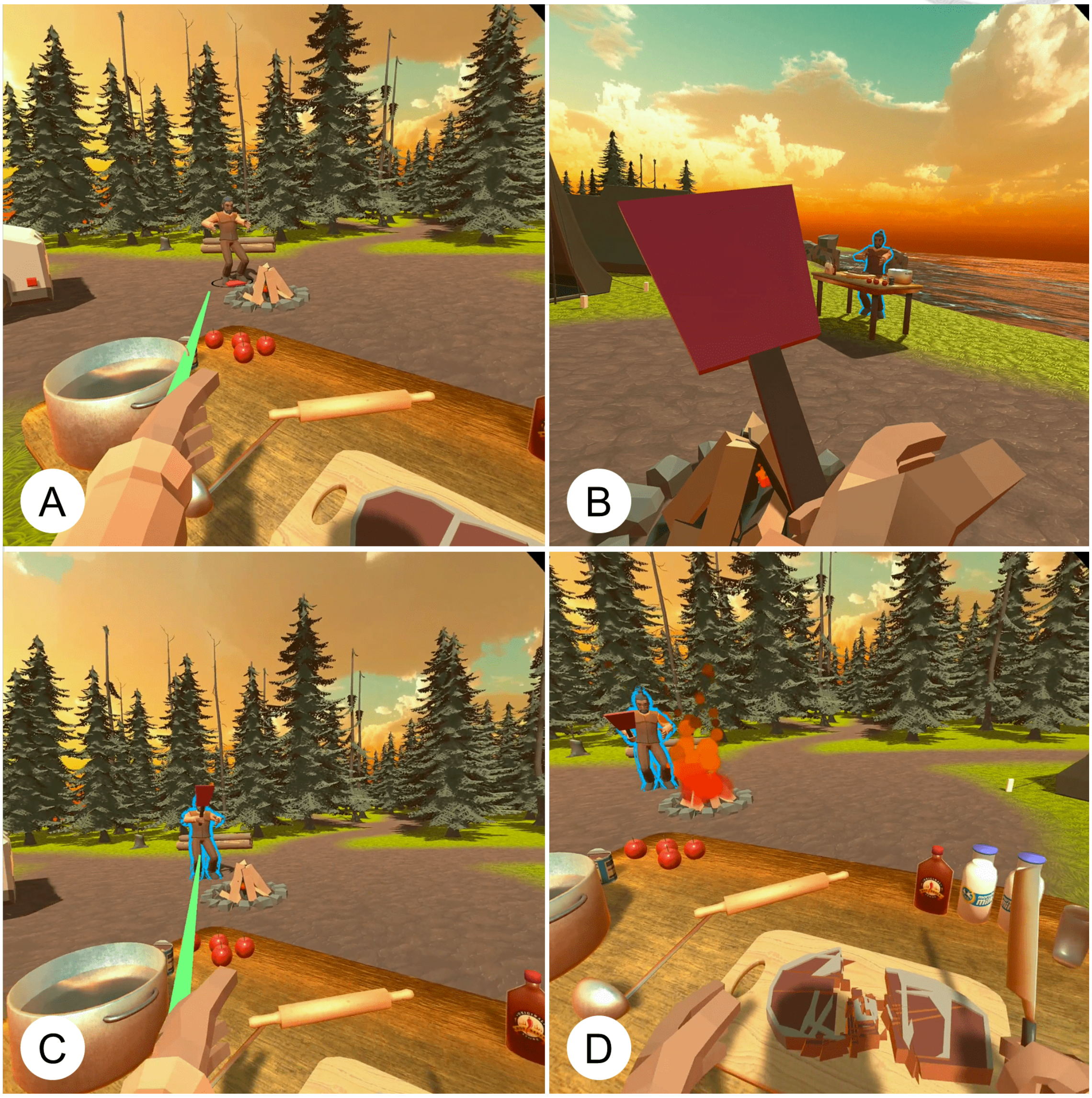

Figure 2.4: (A) The user spawns a clone at the desired location using *Raycast Spawning*. (B) The user switches to the clone to grab the fan from the ground. (C) The user changes the clone's interaction mode from static to synchronous using a ray. (D) With the assistance of the clone, the user can now cut the beef while simultaneously fanning the campfire.

Then, the user wants to add ingredients such as apples and canned food to the boiling soup while keeping it stirred to prevent burning. By recording himself stirring the pot and applying the recorded actions to a clone, the user can focus on adding ingredients while

doi:10.6342/NTU202303156

the clone continues to stir the soup (Figure 2.5).

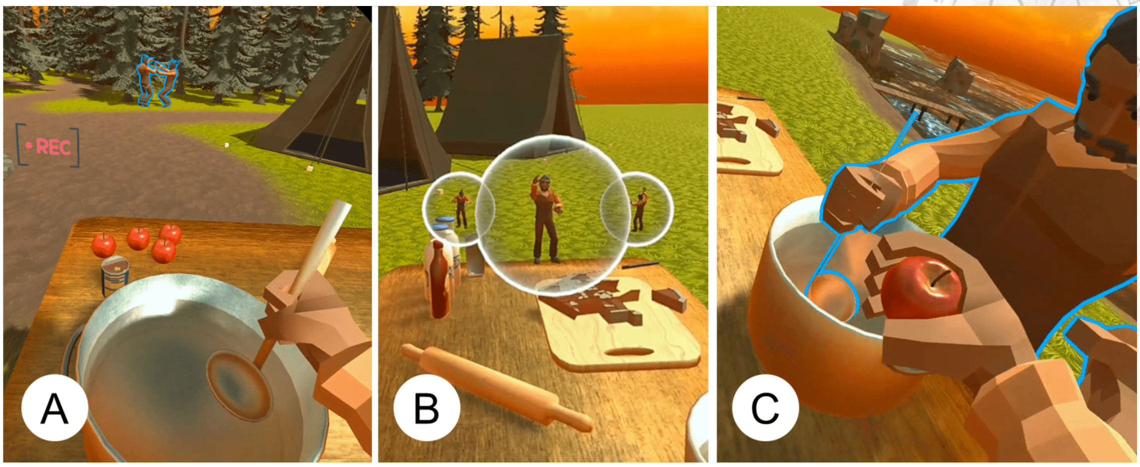

Figure 2.5: (A) The user records himself stirring the pot. During the recording stage, an icon appears in the upper left corner. (B) The user can preview and select previous recordings using a user interface inspired by [30]. (C) Once the user applies the recorded actions to the clone, he can delegate the task of stirring the pot to it and focus on other tasks such as adding ingredients.

When the user wants to collect more apples, he constructs a robotic arm by arranging multiple synchronous clones in a row and having them grab each other (Figure 2.6). By adjusting the number and size of the clones, the user can achieve different control-display ratios, enabling him to customize the arm's reach and precision to suit his needs. The flexibility of the simulated robotic arm can be further enhanced by altering the interaction modes of specific clones. For instance, changing parts of clones to static results in a joint-like movement, allowing the user to bend and twist the robotic arm at different angles.

Suddenly, a cabin near the campsite catches fire, and the user must extinguish it quickly. He stacks four clones vertically and sets up the clones so that the first and third clones move in sync while the second and fourth clones mirror their movements, forming a vertical bucket brigade to transfer water to the roof efficiently (Figure 2.7).

After extinguishing the fire, the user decides to relax by dancing. By assigning recorded actions sequentially to multiple clones, users can delay the clones' movements

doi:10.6342/NTU202303156

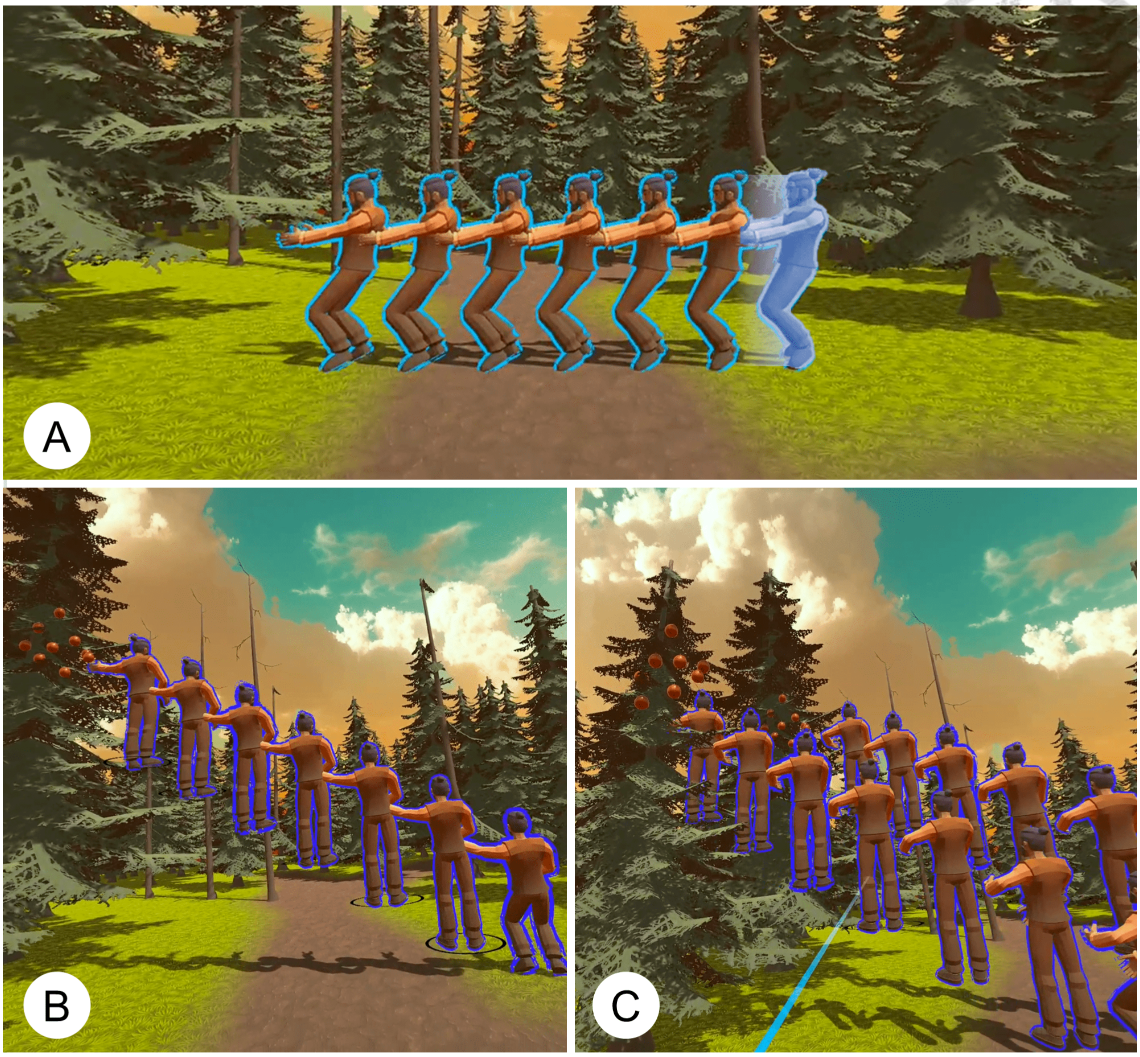

Figure 2.6: (A) The user performs *Out-of-Body* several times to form a chain of clones. (B) By having adjacent clones grab each other, the clones become a robotic arm that enables the user to reach apples on the tree. (C) The user can duplicate the robotic arm pattern to reuse it in other scenarios.

and create a phase shift between them. This allows the user to perform the Thousand-Hand Bodhisattva Dance with the clones (Figure 2.8A).

When the user encounters an intersection in the forest, he leaves a static clone as a road sign to help him navigate more easily (Figure 2.8E). By creating clones that perform different postures, the user can create personalized gestures that are more intuitive than traditional flags or icons. In addition, the user can record and replay his body movements to create even more expressive guides, allowing him to convey more detailed information.

doi:10.6342/NTU202303156

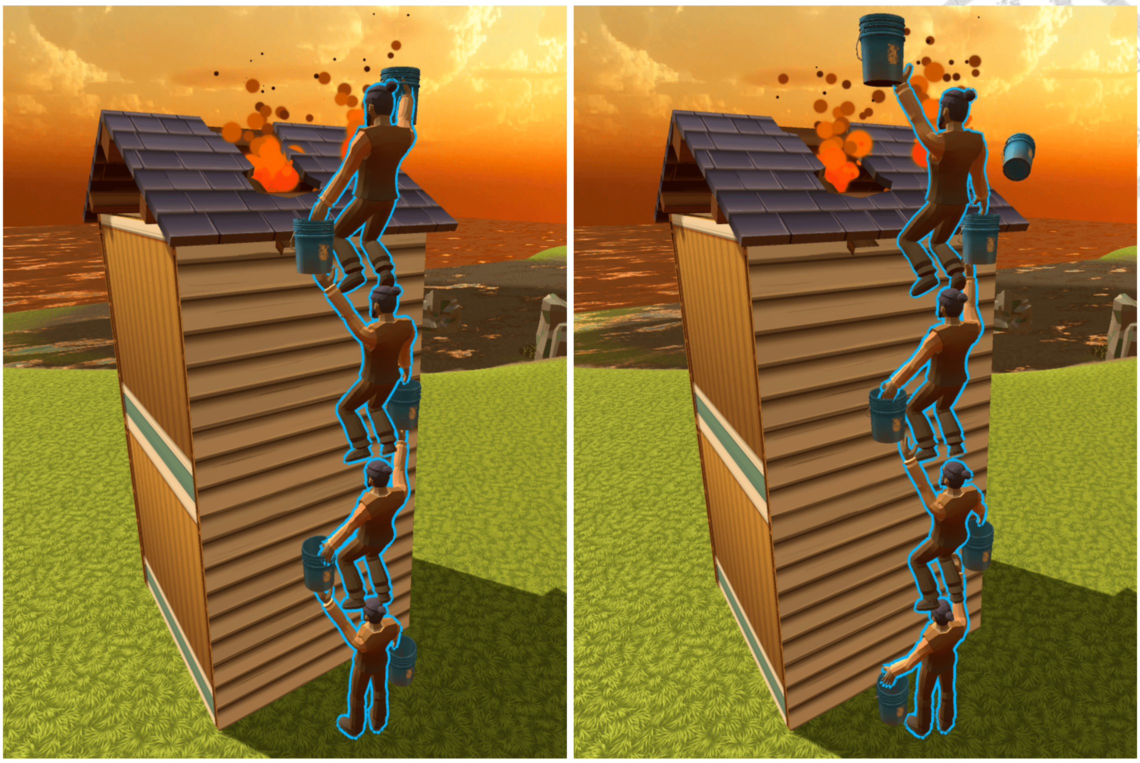

Figure 2.7: The user creates synchronous clones and forms a vertical bucket brigade by mirroring adjacent clones' movements. As one clone raises its left hand and lowers its right hand, the adjacent clone mirrors this movement by lowering its right hand and raising its left hand. It allows a single user to efficiently pass buckets to extinguish a fire.

Finally, the user takes advantage of the unique properties of clones to create painting in VR. By using a rotoscoping technique on a clone, the user can easily sketch any desired posture (Figure 2.8B). He can also create symmetrical paintings by working collaboratively with synchronous clones (Figure 2.8C, D).

2.1 Summary of the Walkthrough

While we demonstrate the effectiveness of Clonemator within a camping scenario, it is important to note that all the showcased spatiotemporal interactions are general and applicable to other scenarios and tasks as well. To provide a comprehensive overview of the interactions enabled by Clonemator, we summarize their taxonomy in Figure 2.9.

doi:10.6342/NTU202303156

Figure 2.8: (A) A group of clones with delayed movements allows the user to perform the Thousand-Hand Bodhisattva Dance. When painting in VR, the user can (B) rotoscope on a clone with any postures, (C) create symmetrical paintings by mirroring the clone's movement, and (D) create symmetrical paintings by arranging the clones' positions and orientations. (E) The user can also generate customized and expressive road signs by creating clones with different postures.

doi:10.6342/NTU202303156

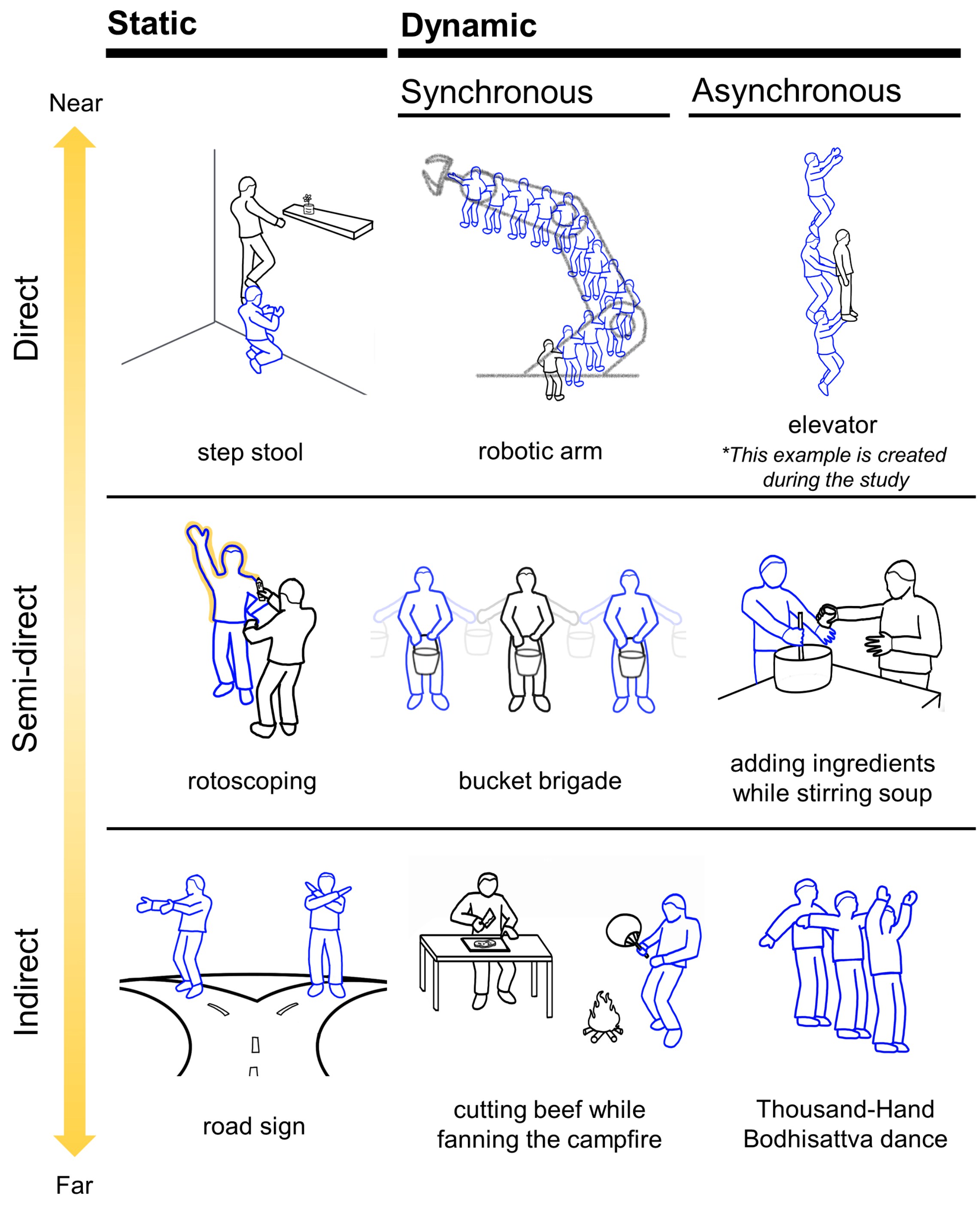

Figure 2.9: The taxonomy of Clonemator and the corresponding examples. Depending on the distance between the clones and the user, the interactions between them can be categorized as either (1) direct, (2) semi-direct, or (3) indirect. In terms of the temporal aspect, the clones can be (1) static, (2) synchronous, or (3) asynchronous. These spatial and temporal dimensions together define the possibilities of Clonemator.

doi:10.6342/NTU202303156

Chapter 3: Contributions

In this paper, we make the following contributions:

- We develop a VR system that allows users to create clones with different interaction modes, including static, synchronous, and replayed. We also integrate traditional techniques such as duplicate, mirror, and group to offer users flexibility in composing clones and building their own automators.
- We explore the possibilities and potential for interactions between clones by providing several examples and discussing our insights.
- We conduct a preliminary study to validate the usefulness and effectiveness of Clonemator.

doi:10.6342/NTU202303156

Chapter 4: Clonemator

The basic unit of Clonemator is an avatar, whether it is a clone or the user himself. This section provides an overview and information on the taxonomy of how users can spawn and manipulate a basic clone unit or a set of clones with Clonemator.

4.1 Spawning Methods

With Clonemator, users can create clones using different spawning methods, each with its unique benefits and limitations. When spawning a clone while holding an object, the object will also be duplicated and distributed to the clone's hand.

4.1.1 Out-of-Body

Clonemator allows the user to create a clone in the exact position and with the same posture as the current avatar. When triggering *Out-of-Body*, our system duplicates the current avatar and smoothly moves the user backward with a customized offset, simulating the feeling of leaving the original body. This method is particularly useful when precise spawning is required. It also comes in handy when the user wants to preserve the context of his current interaction, such as when holding an object like a fishing rod. In this case, the clone will continue to hold the fishing rod while the user can move away and perform

doi:10.6342/NTU202303156

other actions.

4.1.2 Raycast Spawning

Raycast Spawning enables the user to quickly spawn a clone at the desired location using a ray. The user can adjust the rotation of the clone along the vertical axis by pressing the thumbstick left and right. The user can also spawn clones with different sizes by pressing the thumbstick backward and forward.

Although *Raycast Spawning* provides a quick way to spawn a clone at a distance, it lacks precision in determining the exact spawning position. To further ease the user's effort when positioning the clone, Clonemator offers two snapping methods that can be toggled using the controller button. *Grid Snapping* divides the world into a grid centered on the user, with a cell size equal to the user's arm length. It can ensure that the spawned clones are able to interact with each other at arm's reach if they are in two adjacent cells. This method is convenient for maintaining consistent offsets between avatars. *Nearest Object Snapping* snaps the clone to the nearest object with an arm-length offset, which is highlighted using an outline. This method is useful for situations where the spawned clones need to interact with a specific object. Both snapping methods can be further customized using a slider to adjust the offsets for different use cases.

4.1.3 Auto Spawning and Relative Spawning

With *Auto Spawning*, the user can select an object and spawn clones in front of other objects of the same kind while preserving the relative position and the rotation offset. In our current implementation, we detect objects of the same kind by searching for objects

doi:10.6342/NTU202303156

with the same tag that is manually assigned beforehand. *Relative Spawning* can be used when the user wants to spawn clones in front of arbitrary objects. When selecting a reference object, Clonemator calculates the offset between the reference object and the user. When the user selects the next object, Clonemator spawns a clone based on the position of the selected object and the calculated offset. Keeping the same offset between objects has the benefit of ensuring that the user can interact with those objects simultaneously. We also set the default interaction modes of the clones spawned by *Auto Spawning* and *Relative Spawning* to synchronous.

4.2 Clone Basics

Clonemator offers three custom interaction modes for a clone, including (1) Static, (2) Synchronous, and (3) Replayed. Different modes enable different levels of controllability and automatability, and thus offer different functionalities. Users can apply interaction modes by pointing a raycast at a clone.

4.2.1 Static Mode

In *Static Mode*, the clone remains stationary and does not move. This mode is useful when the user needs a stationary platform or needs help holding objects in place. For example, the user can hang a lantern on the clone to provide additional light in a dark forest. This can free up the user's hands and allow the user to perform another task, such as gathering woods. A static clone can also serve as a useful reference point. For example, the user can leverage a static clone's body as a measuring tool for comparing the height of a door when building a house.

doi:10.6342/NTU202303156

4.2.2 Synchronous Mode

In this mode, the movement of the clone is identical to that of the user. The user can create a group of synchronous clones to perform repetitive tasks efficiently or collaborate with them synchronously. It is also useful for activities that require synchronization between multiple clones, such as passing objects over a long distance using the bucket brigade technique. Currently, we do not synchronize the teleportation to prevent users from moving the clones unconsciously.

4.2.3 Replayed Mode

Clonemator allows a clone to perform previously recorded sequences of actions repeatedly in a loop. Clonemator records the physical movements of the user, as well as the grabbing and releasing events for each frame. Users can access a 3D carousel menu that shows previews of all previous recordings. From there, they can select the desired recording and apply it to an existing clone. This mode is useful when the user needs to repeat certain actions multiple times. For example, the user can record himself taking out the trash and delegate this chore to a clone. It allows the user to focus on other tasks or activities.

4.3 Cloning with Traditional Techniques

Clonemator supports several traditional techniques that enable users to manipulate and organize clones, thereby enhancing the system's flexibility and allowing users to tailor customized approaches to individual tasks.

doi:10.6342/NTU202303156

4.3.1 Transform

Clonemator provides two ways for users to translate, rotate, and scale a clone. The first method is to directly grab and move the clones to the desired location, rotate them by pressing the thumbstick left and right, and scale them by pressing the thumbstick up and down. This method is intuitive and simple but may not provide the highest level of precision. For more precise transformation, users can switch to a clone and control its movement from a first-person perspective, allowing for more accurate adjustments and fine-tuning of the clone's transform. By rearranging the spatial layout of the clones, users can build their own customized solutions to handle different situations.

4.3.2 Group

Users can group multiple clones by selecting them with a ray. Once the clones are grouped, they will be visually distinguished with the same outline color, allowing users to easily identify the group. Users can perform actions on them as a whole, such as moving, applying interaction modes, duplicating, and removing. Grouping is especially useful when applying the same recorded actions to multiple clones, ensuring that the replay is synchronized between them. Additionally, moving multiple clones as a group allows users to preserve existing patterns and layouts across the clones.

4.3.3 Duplicate

When duplication mode is activated, a ray will appear from the left hand of the user. Using this ray, the user can grab and pull off a clone from a distance. Although the interaction of creating a duplicate through pulling off an avatar is similar to [30], the main

doi:10.6342/NTU202303156

difference is that the duplicated clone can also interact with the virtual world. Duplication not only enables users to generate clones efficiently but also facilitates the reuse of previous patterns.

4.3.4 Mirror

Mirroring allows the user to flip a clone's movement horizontally, creating a mirror image of their own actions. For instance, when the user raises their right hand, the clone will raise its left hand, and vice versa. It provides intuitive control especially when the user and the clone are facing each other, as their movements are reflected, much like looking into a mirror. Mirroring is also helpful for activities that require symmetric movements, such as swinging a long jump rope together or performing synchronized dance moves.

4.3.5 Remove

Users can select and remove a clone using a ray. If the clone is holding an object, the object will dissolve and appear near the user. The mental model behind this feature is that the removed clone will merge back with the user. For example, the user can assign a clone to gather wood and then remove the clone to obtain the collected wood.

4.3.6 Undo

Clonemator also provides an Undo feature, which allows the user to cancel previous spawning commands. It is especially useful when the user wants to remove multiple clones created by *Auto Spawning* at the same time. Our implementation also allows users to undo grouping and duplication commands, reducing manual effort.

doi:10.6342/NTU202303156

4.4 Switching Perspectives

To switch between clones, the user can use a ray to select the clone to which they want to switch. To minimize potential disorientation after switching, we implement an interpolation method to smoothly transition the positions and rotations of the user's viewpoint between the original avatar and the clone. To ensure that the transition feels comfortable, we reduce the field of view during the transition to mitigate motion sickness.

doi:10.6342/NTU202303156

Chapter 5: Implementation

We implement Clonemator using Unity version 2020.3.18f1, running on an HP VR Backpack G2 computer equipped with a NVIDIA GeForce RTX 2080 graphics card. The VR content is streamed to a Meta Quest 2 headset via Quest Link over a USB cable. The display resolution is 1832×1920 per eye, and the refresh rate is set to 72 Hz. The user holds two Meta Quest 2 controllers as input devices.

The tracking data from the headset and controllers are distributed to both the original avatar and its clones. Specifically, we map the positions and the rotations of the headset and the controllers to the corresponding joints (i.e., the neck and the wrists) on a humanoid avatar and use Unity's built-in inverse kinematics system to animate the avatar's full body movements. Hence, one limitation of our system is that users can only record their upper body movements and can not control their legs to form more dynamic poses.

5.1 User Interface

To help users get started faster, we have implemented a 2D menu that can be toggled by pressing the controller button. The menu is mounted on the user's left hand and provides access to all the commands and functions.

doi:10.6342/NTU202303156

Additionally, Clonemator offers voice command functionality for experienced users. The voice recognition system is developed using Unity's built-in keyword recognizer, which can respond to pre-defined keywords. The voice is captured by the microphone of the headset. As the microphone is always active, users can initiate a voice command at any time. Users can also access a list with all available voice commands inside the menu. It allows users to seamlessly control Clonemator faster without breaking their immersion. In situations where a 2D menu is not feasible, such as when the user's hands are fully occupied with other tasks, voice commands become a valuable alternative.

doi:10.6342/NTU202303156

Chapter 6: Preliminary Study

Since Clonemator provides a unique and diverse range of VR interactions, it was challenging to identify a particular evaluation task with an associated metric. As a result, we conducted an initial study to gain a comprehensive understanding of the system's overall usefulness and user experience.

6.1 Design, Tasks, and Participants

The study includes 9 tasks in the VR camping scene presented earlier in Section 2, and is composed of two sessions: a first session with example solutions and a second session without.

In the first session, participants completed five tasks, including four specific tasks from the walkthrough in Section 2 (hammering multiple pegs, catching fish, cooking while adding ingredients, and fanning the campfire while cutting beef) and a free-exploration task. Prior to the five tasks, a video tutorial was given, explaining the tasks' goal, providing the sample solutions from the walkthrough, and demonstrating the necessary UI elements and actions. Participants had to replicate the sample solutions in the VR scene after viewing the video. The free-exploration task lets participants test the operation not covered in the first four tasks, such as remove, duplicate and group, and allowed partici-

doi:10.6342/NTU202303156

pants to explore the scene freely with all operations available. The first session familiarized participants with the system and provided a general understanding of Clonemator's experience.

In the second session, participants had to complete the other four tasks, including:

- Moving a table that is too heavy for a single person, so the participants have to collaborate with or utilize multiple clones
- Passing multiple basketballs as fast as possible from a starting point to a target that is 9 meters away without teleportation
- Fetching a slice of beef at 2.5 meters height from the top of a van
- Fetching an apple at 7.5 meters height from the top of a tree, where a distinct solution from fetching beef was required.

Participants were only given the task goal and requirement without any further instructions provided. These tasks are designed for utilizing more complex clone manipulation, and since participants had to come up with their own solutions, this session is aimed at testing not only the user experience but also the richness of the solutions that Clonemator can support.

We recruited 6 participants (4 male, 2 female) who had no prior knowledge of Clonemator's details, aged from 24 to 27 ($\text{M} = 24.5$, $\text{SD} = 1.4$) through word of mouth. Two had VR experience more than once a week, another two had VR experience about once a month, one had VR experience about once every three months, and the remaining one had very rare VR experience (about once a year). The apparatus used for the study is the same as mentioned earlier in Section 5.

doi:10.6342/NTU202303156

Figure 6.1: Study tasks without sample solutions. Participants have to complete four challenges without explicit instructions. These challenges included: (A) lifting a heavy table, (B) fetching a slice of beef from the top of a van, (C) fetching an apple from the top of a tree using a solution different from that used for fetching the slice of beef, and (D) passing basketballs across a distance without teleportation as fast as possible.

6.2 Solutions Generated by Participants

During the second session of the study, we observed that participants were able to come up with different solutions for the given tasks. For example, P6 used a synchronous

doi:10.6342/NTU202303156

clone to catch a basketball he threw (Figure 6.2A), while P5 recorded himself throwing a ball and applied the recording to a clone, catching the ball at the destination by himself. P2 successfully built a bucket brigade made of clones using the *Out-of-Body* technique and set their interaction modes to synchronous (Figure 6.2C). The first clone grabbed a ball with its right hand and passed it to its left hand, while the next clone used its right hand to grab the ball held by the left hand of the previous clone, forming a pipeline to pass multiple balls efficiently.

For fetching a high apple, P4 used *Relative Spawning* to spawn a clone in front of the apple by setting the reference object to the gas cylinder next to her and then switched to the clone to get the apple. P3 directly threw a clone to the top of the tree and switched to it. P7 recorded the action of grabbing and lifting objects and applied it to vertically stacked clones with different phase shifts, forming an elevator that allowed the clones to pass him to the tree top (Figure 6.2B).

Overall, participants created 5 different solutions for fetching high objects (including fetching beef and fetching an apple) and 4 different solutions for passing basketballs.

6.3 Quantitative and Qualitative Result

6.3.1 Completion Time

The average completion time for the first session of tasks with sample solutions was 79.29 seconds ($\text{SD} = 50.90$). During the free exploration, participants spent an average of 316.17 seconds ($\text{SD} = 131.05$) before they were willing to continue to the second session. The last four tasks, which did not have sample solutions, took an average of 176.46 seconds

doi:10.6342/NTU202303156

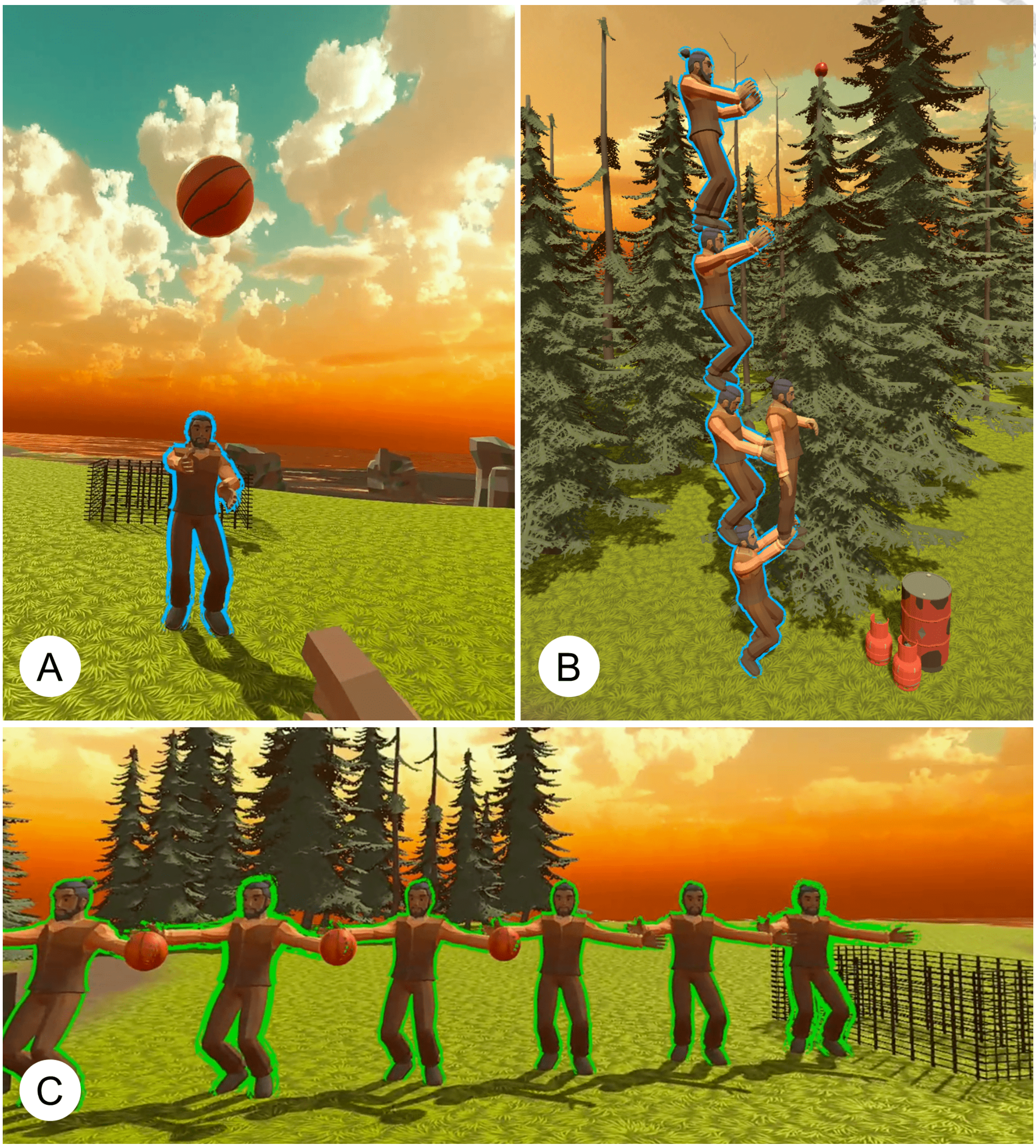

Figure 6.2: Solutions come up by participants: (A) The user throws a ball and catches it with a synchronous clone. (B) The user replays the lifting movement to multiple clones stacked vertically, forming an elevator. (C) A bucket brigade formed by a chain of clones with synchronous movement.

(SD = 237.75) to complete, and we recorded the number of clones generated and removed by participants. On average, participants generated 4.95 clones (SD = 4.61) for each task and removed an average of 1.25 clones (SD = 2.75).

doi:10.6342/NTU202303156

6.3.2 Overall Experience

Participants responded positively about the experience of completing tasks with Clonemator, including *"It's pretty cool. It's interesting and fun."* [P1] *"It's a fun game."* [P2] *"It's pretty fun, even though it's my first time using it."* [P3] *"It's quite fun. The experience is interesting."* [P5]

6.3.3 Usefulness and Learnability

Regarding learning the overall control of the system and its usefulness, participants noted *"Even without instructions, one should be able to figure it [the solution] out."* [P1] and *"It is not difficult to learn, there are many things but all the operations are basic."* [P2].

Regarding spawning the clones, participants express it's natural as *"The method of creating clones is quite intuitive."* [P3]. Specifically, *"Out-of-body is convenient and more intuitive because one can determine the position of their clone by his body."* [P1]. In terms of controlling the clones, *"simultaneously waving the wind and cutting foods [using synchronous mode] is intuitive because the movements are consistent."* [P1], and P3 mentioned her controlling metaphor, by *"to think about how to operate different bodies at the same time, it's like controlling a machine remotely."* [P3]

6.3.4 Use Cases and Further Applications

Although we make sure the keyword or notion of automation is never mentioned in our study instruction, P2 drew a comparison between Clonemator and such concept, by *"It [the system] is like writing code modules and microservices. Using the switch or replay*

doi:10.6342/NTU202303156

to trigger pre-written functions is like a game plugin." and *"similar to a macro or script but for repeated tasks in VR."*

Furthermore, participants proposed additional potential applications of Clonemator, which encompassed *"3D modeling in VR"* [P1] and *"gaming"* [P1, P2, P3]. Specifically, P1 suggested that *"It could be used in rhythm games like Beat Saber or Ragnarok, and would be especially fun if there are multi-tracks with multi-clone at once"*, while participant P2 mentioned the composition of clones *"can improve the experience of games like Story of Seasons, which involves a lot of repeated farming tasks."*

6.4 Study Summary

The preliminary study results indicate that Clonemator provides diverse and adaptable functionalities for various tasks, delivering a user-friendly and enjoyable experience with intuitive controls.

doi:10.6342/NTU202303156

Chapter 7: Related Work

The presented work draws on several areas of previous research, including Replicating Movement in the Real World, Beyond-Real Interaction, and Programming by Demonstration.

7.1 Replicating Movement in the Real World

Previous studies in teleoperation have investigated the potential of replicating control or movement in real-world scenarios. This involves remotely controlling one or multiple robots by one single user. For instance, Glas et al. [6] designed a coordination system that enables a user to operate four robots at the same time. Researchers have also explored ways of controlling two robotic arms synchronously or asynchronously through the use of EMG signals and gaze control [19]. Additionally, in Transfantome [9], users can control two robots concurrently at different scales by embodying two proxy bodies that represent each robot. Takada et al. [27] utilized parallel embodiment to allow users to operate two robot arms to play ping-pong against two opponents concurrently. These studies demonstrate the potential of duplicating control and movement in enhancing physical operation.

However, these methods are restricted by real-world hardware resources, leading to limitations in the number of clones or replications. In Clonemator, we systematically ex-

doi:10.6342/NTU202303156

plore such a concept in virtual reality environments, providing users with the capability to operate an unlimited number of clones of varying sizes, timing, and replayability, thereby enabling the full realization of the potential for movement cloning and replication.

7.2 Beyond-Real Interaction in VR

The realm of VR offers a unique platform for exploring interactions that are impossible to experience in the physical world. Such interactions have been coined as " Beyond-real interaction" [2] by Parastoo et al. Researchers have investigated a variety of beyond-real interactions, including dynamically changing arm length for reaching objects [23], scaling the body for navigation [1], and using spherical proxies to interact with distant objects [21].

Researchers have also begun exploring the possibility of multi-embodiment or duplicating body parts in VR. Studies have shown that users are able to adapt to the six-digit hand while maintaining a sense of ownership and agency with it [7]. Similar results can also be applied to having a third arm [4] and even multiple bodies [18]. Furthermore, users can develop an implicit dual motor adaptation when switching between two virtual bodies [29].

Beyond the experience of having supernumerary body parts, researchers have also investigated the resulting changes in performance. For example, it has been shown that users can increase task efficiency and reduce physical movement by leveraging additional hands [24] or bodies [25]. However, in some cases, task completion time can increase due to the increased complexity of having multiple bodies [18].

Clonemator is situated within the "duplication" category of Beyond-Real Interac-

doi:10.6342/NTU202303156

tion's taxonomy [2]. Compared to previous work, our research further explores the interaction between the user and their clones, leveraging the ability to create clones with various temporal and spatial properties to provide users with greater control over their clones. This approach builds upon previous research and takes it to the next level by enabling the possibilities for user-clone interactions.

7.3 Programming by Demonstration

Previous research has investigated Programming-by-Demonstration (PbD) [14] to enhance the accessibility of end-user programming. PbD liberates users from the constraints of dedicated programming languages or scripts by enabling them to create their automated solutions by demonstrating "an example of what they wanted it to do" directly to the system [14].

For instance, Sikuli [31] allows users to utilize screen-captured elements as their script and search input, bypassing the need for expert knowledge of the parameters or names of the corresponding UI. Additionally, SikuliBot [10] extends the same concept to physical interfaces by enabling users to take real-world images of physical buttons or touchscreen positions, and a robotic actuator automatically operates these physical UI elements based on the users' arrangement of the images. In addition to images or screenshots, recording and replaying a sequence of users' interactions further enhances the automation process with PbD. For instance, Ringer [3] allows non-coders to record a sequence of their website interaction, which generates a script that interacts with the page as demonstrated by the users. Similarly, Mau'es et al. [17] enables users to perform a smartphone task that they would like to automate, and their system generates the automation based on these

doi:10.6342/NTU202303156

latest actions. Moreover, SUGILITE [13] combines voice input and app demonstration to enhance the automation pipeline for non-programmers.

While these approaches have lowered the barrier to automation for desktop, web, and mobile applications, Clonemator explores this concept in the context of virtual reality. Specifically, Clonemator builds upon the PbD concept for task automation by allowing users to record and replay their VR demonstration, which can be rearranged, distributed, and composited with each other in the form of clones in VR. Users can create and control clones to perform specific actions as if they were controlling their own bodies, without requiring dedicated crafting of different interaction techniques for different situations.

doi:10.6342/NTU202303156

Chapter 8: Discussion and Limitations

8.1 Challenges of Alignment

Composing multiple clones to create custom automators often requires precise alignment of the clones. Clonemator addresses this by offering several alignment techniques, such as *Auto Spawning* and *Relative Spawning*, as well as *Grid Snapping* and *Nearest Object Snapping* modes when spawning a clone.

While these techniques can handle situations in our case, more advanced techniques are necessary for complex scenarios, such as when the user wants to spawn a clone in front of an object that is significantly smaller than the object in front of the user, since the required offsets may not be the same. This can lead to misalignment, making it difficult for the clone to interact with the smaller object at the same time.

8.2 Future Features of Clonemator System

Although we did not implement some more complex features in Clonemator to avoid the system becoming too complex and unintuitive, we list these features here to inspire future research on the system.

doi:10.6342/NTU202303156

Editing a replayed clone: While allowing the user to replay pre-recorded actions enables many use cases, it is still not very flexible in our current implementation. To accommodate different tasks, one possible improvement is to allow the user to edit the recordings, such as trimming non-necessary sequences or extracting part of the clips to form another recording. Another improvement would be to provide the user with more control during the replay, such as fast forwarding or rewinding.

Multi-user and collaborative use: One interesting direction for future research would be exploring the possibilities of sharing control of a clone. This would allow the user to control only specific parts of the clone while other parts are controlled by other clones (e.g., through previous recordings) or even other users.

Different Visualization of Clone: We found that, since our current tracking apparatus primarily focuses on the upper body, visualizing only the upper body of the avatar or even just the hands could reduce the visual burden when there are too many clones. Additionally, providing users with the option to change the coloring of clones would make it easier for them to systematically manage them using color labels. Moreover, visualization can improve or decrease the affordance of each clone. For instance, when using a string of static clones to build a row of fences, it would be more realistic and aesthetically pleasing to give them a wooden texture or simplified mesh. On the contrary, we can replace the humanoid avatar with other creatures or non-creature avatars to confuse the user's movement.

AI-Powered Auto Scripting: Finally, the integration of imitation learning techniques to facilitate the control of clones is also worth investigating. By embedding such techniques into a clone, the clone would be able to control itself independently, freeing

doi:10.6342/NTU202303156

the user from having to manually control every aspect of the interaction.

In general, we believe that these future directions have the potential to significantly enhance the capabilities and flexibility of the Clonemator system and open up new possibilities for users to create and interact with virtual environments in more intuitive and creative ways.

8.3 Merging a Clone's Experience

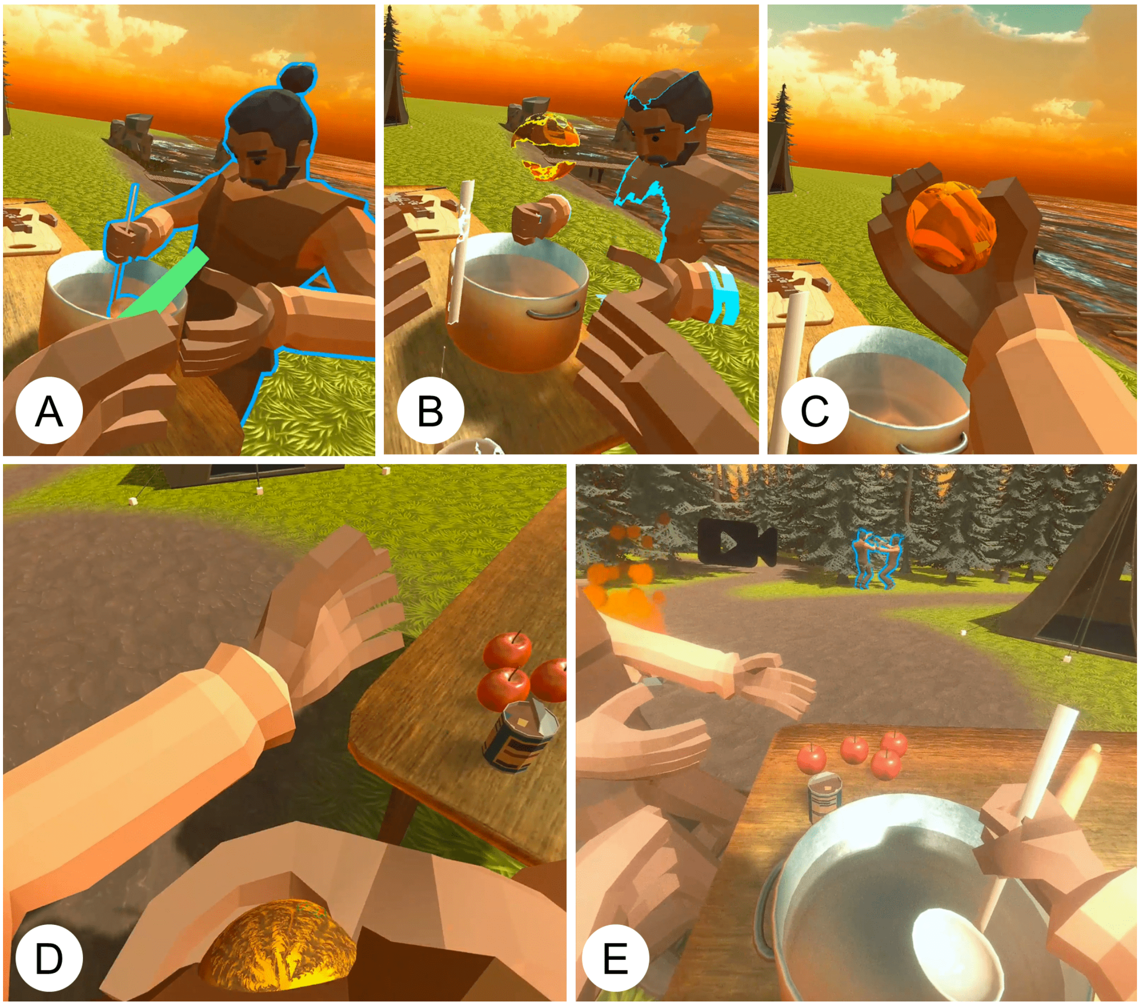

Figure 8.1: A proof-of-concept feature allowing users to relive a clone's experience: (A) The user removes a clone and (B)-(C) a memory orb appears. (D) The user places the memory orb into his chest and (E) experiences the clone's memory.

We are heavily inspired by the Shadow Clone Technique in the popular anime,

doi:10.6342/NTU202303156

Naruto. One of the key features is the user's ability to gain the clones' experiences when they disperse. In Clonemator, we have implemented a similar feature where a memory orb, inspired by the film *Inside Out*, appears in front of the user upon the removal of a clone. By grabbing the memory orb and placing it into his chest, the user can relive the clone's experience by watching a replay captured by the clone's eye (Figure 8.1).

However, a question remains: can the user experience a clone's sensations other than sight? For example, imagine removing a clone that has been stirring a pot for half an hour and instantly feeling the soreness in their arm. This presents an interesting design space for future exploration.

8.4 Distributed Perception

While most previous work has focused on visual multiplexing of viewpoints [25, 28], we believe it would be crucial to explore other sensory perceptions of the clone. It provides the user with greater awareness of the clone, which is important in many scenarios. For example, when hammering pegs together with clones, the user may not know which clone has finished or not without perceiving its sensations. Future research could investigate ways to arrange haptic feedback so that the user can accurately identify its source and provide the appropriate response.

8.5 Conditional Trigger

In Clonemator, users manually execute commands to trigger actions, such as applying a replay to a clone or executing a spawning method to create new clones. However, to

doi:10.6342/NTU202303156

enable more flexible automations, we propose adding conditional triggers to our system. For example, when throwing an object over a long distance, it would be convenient if the system could automatically generate a synchronous clone at the point where the ball lands, and then switch to that clone to catch the object. With this feature, users can even reproduce the teleportation technique by creating a clone at a distance, switching to it, and then removing the original body automatically.

8.6 Clonemator in Other Applications

Although implemented in VR, we believe our system has further potential to be utilized in other areas.

One such area is multi-robot system. While our system is conceptually similar to the ”Distributed Autonomous System” [20] that utilizes multiple independent robot units to work coordinately and achieve a more complex task, it has the potential to serve as a control mechanism for future swarm user interfaces [12] or multiple robotic arms. By leveraging the flexibility of Clonemator, users can control multiple elements as intuitively as controlling their clones. For example, a user can spawn a clone in VR and a robot appears in the corresponding position in the real world. It can also facilitate users in switching between multiple robots [9] or recording and replaying actions for teleoperation [16].

On the other hand, different aspects of clone-based mechanisms have been leveraged in previous video games to create novel gaming experiences. For example, Quantum League [5] and Time Rifters [8] allow players to team up with their past selves in a shooting game. More recently, The Last Clockwinder [22] enables players to create clones that replay their actions and construct a pipeline to maximize the resource harvesting through-

doi:10.6342/NTU202303156

put. While Clonemator systematically extends the possibilities of interactions between clones by providing multiple interaction modes and showcasing how they can be combined with traditional techniques for even more complicated tasks, we believe that our work will inspire game designers to explore and develop more engaging and innovative gaming mechanisms.

doi:10.6342/NTU202303156

Chapter 9: Conclusion

We have presented Clonemator, a VR system that allows a user to create and collaborate with clones to accomplish complex tasks. We have demonstrated the potential of Clonemator by showcasing concrete examples based on our systematic exploration of spawning clones, clones' properties, and interactions with traditional techniques. We also have shown from our preliminary study that participants were able to intuitively, creatively, and also enjoyably use Clonemator. With Clonemator, we see the user's avatar could be the generic automator waiting to be dispatched.

doi:10.6342/NTU202303156

References

[1] P. Abtahi, M. Gonzalez-Franco, E. Ofek, and A. Steed. I'm a giant: Walking in large virtual environments at high speed gains. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, CHI '19, page 1–13, New York, NY, USA, 2019. Association for Computing Machinery.

[2] P. Abtahi, S. Q. Hough, J. A. Landay, and S. Follmer. Beyond being real: A sensorimotor control perspective on interactions in virtual reality. In Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems, CHI '22, New York, NY, USA, 2022. Association for Computing Machinery.

[3] S. Barman, S. Chasins, R. Bodik, and S. Gulwani. Ringer: Web automation by demonstration. In Proceedings of the 2016 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications, OOPSLA 2016, page 748–764, New York, NY, USA, 2016. Association for Computing Machinery.

[4] A. Drogemuller, A. Verhulst, B. Volmer, B. H. Thomas, M. Inami, and M. Sugimoto. Remapping a third arm in virtual reality. In 2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), pages 898–899, 2019.

doi:10.6342/NTU202303156

[5] N. G. Entertainment. Quantum league, April 2021. https://store.steampowered.com/app/651150/Quantum_League/.

[6] D. F. Glas, T. Kanda, H. Ishiguro, and N. Hagita. Teleoperation of multiple social robots. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, 42(3):530–544, 2012.

[7] L. Hoyet, F. Argelaguet, C. Nicole, and A. Lécuyer. "wow! i have six fingers!" : Would you accept structural changes of your hand in vr? Frontiers in Robotics and AI, 3, 2016.

[8] P. S. Inc. Time rifters, October 2014. https://store.steampowered.com/app/270010/Time_Rifters/.

[9] A. Izumihara, T. Sasaki, M. Ogino, R. Takamura, and M. Inami. Transfantome: Transformation into bodies of various scale and structure in multiple spaces. In ACM SIGGRAPH 2019 Emerging Technologies, SIGGRAPH '19, New York, NY, USA, 2019. Association for Computing Machinery.

[10] J. Kim, M. Kasper, T. Yeh, and N. Correll. Sikulibot: Automating physical interface using images. In Adjunct Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology, UIST '14 Adjunct, page 53–54, New York, NY, USA, 2014. Association for Computing Machinery.

[11] I. Kotziampasis, N. Sidwell, and A. Chalmers. Portals: Increasing visibility in virtual worlds. In Proceedings of the 19th Spring Conference on Computer Graphics, SCCG '03, page 257–261, New York, NY, USA, 2003. Association for Computing Machinery.

doi:10.6342/NTU202303156

[12] M. Le Goc, L. H. Kim, A. Parsaei, J.-D. Fekete, P. Dragicevic, and S. Follmer. Zooids: Building blocks for swarm user interfaces. In Proceedings of the 29th Annual Symposium on User Interface Software and Technology, UIST '16, page 97–109, New York, NY, USA, 2016. Association for Computing Machinery.

[13] T. J.-J. Li, A. Azaria, and B. A. Myers. Sugilite: Creating multimodal smartphone automation by demonstration. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, CHI '17, page 6038–6049, New York, NY, USA, 2017. Association for Computing Machinery.

[14] H. Lieberman. Your wish is my command: Programming by example. Morgan Kaufmann, 2001.

[15] P. Lubos, G. Bruder, and F. Steinicke. Are 4 hands better than 2? bimanual interaction for quadmanual user interfaces. In Proceedings of the 2nd ACM Symposium on Spatial User Interaction, SUI '14, page 123–126, New York, NY, USA, 2014. Association for Computing Machinery.

[16] K. Mahadevan, Y. Chen, M. Cakmak, A. Tang, and T. Grossman. Mimic: In-situ recording and re-use of demonstrations to support robot teleoperation. In Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology, UIST '22, New York, NY, USA, 2022. Association for Computing Machinery.

[17] R. d. A. Maués and S. D. J. Barbosa. Keep doing what i just did: Automating smartphones by demonstration. In Proceedings of the 15th International Conference on Human-Computer Interaction with Mobile Devices and Services, MobileHCI '13, page 295–303, New York, NY, USA, 2013. Association for Computing Machinery.

doi:10.6342/NTU202303156

[18] R. Miura, S. Kasahara, M. Kitazaki, A. Verhulst, M. Inami, and M. Sugimoto. Multisoma: Distributed embodiment with synchronized behavior and perception. In Proceedings of the Augmented Humans International Conference 2021, AHs '21, page 1–9, New York, NY, USA, 2021. Association for Computing Machinery.

[19] Y. Nakanishi, M. Fukuoka, S. Kasahara, and M. Sugimoto. Synchronous and asynchronous manipulation switching of multiple robotic embodiment using emg and eye gaze. In Proceedings of the Augmented Humans International Conference 2022, AHs '22, page 94 – 103, New York, NY, USA, 2022. Association for Computing Machinery.

[20] J. Ota. Multi-agent robot systems as distributed autonomous systems. Advanced Engineering Informatics, 20(1):59–70, 2006.

[21] H. Pohl, K. Lilija, J. McIntosh, and K. Hornbæk. Poros: Configurable proxies for distant interactions in vr. In Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery.

[22] Pontoco. The last clockwinder, June 2022. https://www.oculus.com/experiences/quest/4837365566303714.

[23] I. Poupyrev, M. Billinghurst, S. Weghorst, and T. Ichikawa. The go-go interaction technique: Non-linear mapping for direct manipulation in vr. In Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology, UIST '96, page 79–80, New York, NY, USA, 1996. Association for Computing Machinery.

[24] J. Schjerlund, K. Hornbæk, and J. Bergström. Ninja hands: Using many hands to improve target selection in vr. In Proceedings of the 2021 CHI Conference on Human

doi:10.6342/NTU202303156

Factors in Computing Systems, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery.

[25] J. Schjerlund, K. Hornbæk, and J. Bergström. Ovrlap: Perceiving multiple locations simultaneously to improve interaction in vr. In Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems, CHI '22, New York, NY, USA, 2022. Association for Computing Machinery.

[26] R. Stoakley, M. J. Conway, and R. Pausch. Virtual reality on a wim: Interactive worlds in miniature. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '95, page 265–272, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[27] K. Takada, M. Kawaguchi, A. Uehara, Y. Nakanishi, M. Armstrong, A. Verhulst, K. Minamizawa, and S. Kasahara. Parallel ping-pong: Exploring parallel embodiment through multiple bodies by a single user. In Proceedings of the Augmented Humans International Conference 2022, AHs '22, page 121–130, New York, NY, USA, 2022. Association for Computing Machinery.

[28] K. Takada, N. Kumasaki, T. Froese, K. Shibata, J. Nishida, and S. Kasahara. Shadowclones: An interface to maintain a multiple sense of body-space coordination in multiple visual perspectives. In Proceedings of the Augmented Humans International Conference 2023, AHs '23, page 171–178, New York, NY, USA, 2023. Association for Computing Machinery.

[29] A. Verhulst, Y. Namikawa, and S. Kasahara. Parallel adaptation: Switching between two virtual bodies with different perspectives enables dual motor adaptation. In 2022

doi:10.6342/NTU202303156

IEEE International Symposium on Mixed and Augmented Reality (ISMAR), pages 169–177, 2022.

[30] H. Xia, S. Herscher, K. Perlin, and D. Wigdor. Spacetime: Enabling fluid individual and collaborative editing in virtual reality. In Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology, UIST '18, page 853–866, New York, NY, USA, 2018. Association for Computing Machinery.

[31] T. Yeh, T.-H. Chang, and R. C. Miller. Sikuli: Using gui screenshots for search and automation. In Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology, UIST '09, page 183–192, New York, NY, USA, 2009. Association for Computing Machinery.

doi:10.6342/NTU202303156