Manipulating augmented virtual character using dynamic hierarchical pointing interface

Hyejin Kim | Jae-In Hwang

LG Electronics, Korea Institute of Science and Technology (KIST), Seoul, South Korea

Correspondence
Jae-In Hwang, Korea Institute of Science and Technology (KIST), Seoul, South Korea.
Email: hji@kist.re.kr

Funding information
Korea Institute of Science and Technology (KIST) Institutional Program; Korea Creative Content Agency

Abstract
In this paper, we introduce a bare-hand interaction method for controlling an augmented virtual character. Our method is utilizing a dynamic hierarchical structure of virtual buttons on the natural marker of the environment. Unlike the existing virtual button method, the proposed method can be used to select a very fine level. This method is adequate for controlling an augmented virtual character with a users’ various intentions. We implemented several examples of manipulating an augmented virtual character such as guiding the character, selecting part of the character, and selecting a target object for the character. We also compared the performance of full arrangement and dynamic hierarchical arrangement of virtual buttons. Our method outperforms the full arrangement method, especially in the low index of difficulty. In the user study with 15 participants, the users responded that the proposed method was significantly better than the existing full arrangement method regarding efficiency and satisfaction in the case of the fine level selection.

KEYWORDS
augmented reality, bare-hand interaction, manipulation, virtual character

1 | INTRODUCTION

Consumer-level virtual reality (VR) technologies and applications have recently been introduced through smartphone-equipped headset devices such as Google Cardboard and Samsung Gear VR. These headsets provide immersive user experience at an affordable cost for VR applications and have the potential for wearable augmented reality (AR) applications. Nevertheless, using these headsets for wearable AR applications would require a different user interface than that of typical mobile AR applications. In most AR applications, users interact with the augmented contents through a touch screen, which takes less computational power on resource-constrained mobile devices. However, touching the screen of a mobile device is not allowed when the device is worn on a user’s head or eyes. In wearable AR applications, bare-hand interaction is a more natural and appropriate user interface to point and touch the augmented contents.

With recent advances in mobile AR tracking technology, a large number of mobile AR applications are built on an image target tracker for robust and stable planar tracking. To further enrich the user experience of a mobile AR, natural user interaction, such as hand interaction, has an important and promising role. However, it is difficult to apply
a state-of-the-art bare-hand tracking in mobile AR applications mainly due to resource constraints. There also exists a lightweight alternative method known as occlusion-based hand interaction. Lee et al. demonstrated occlusion-based interaction using a set of visible markers in tangible AR environments by detecting occlusions of tracked markers. This method required much lower computational power than bare-hand tracking while providing comparable pointing performance. The virtual button is another similar concept implemented in Vuforia SDK to define rectangular regions on an image target that can be triggered for an event when occluded in a camera view. The aforementioned occlusion-based hand interaction is technically more suitable for mobile platforms and provides natural hand interaction-like user experiences. In this article, we focus on how to make a pointing interface possible for a wearable AR application, in which we can exploit only a smartphone without any other device, such as a depth camera. To this end, we use an image target to augment virtual characters for interaction with virtual buttons. If a set of multiple virtual buttons is prearranged in a grid, a user can point to any region defined by virtual buttons across the image target with his/her finger. In order to point at more precise regions of the image target, an increased number of virtual buttons need to form a densely arranged virtual button grid. However, increasing the number of virtual buttons also increases the processing time to check all virtual buttons for occlusion and event handling, which significantly decreases the interaction frame rate.

To overcome such a limitation, we propose a practical approach using dynamic hierarchical virtual buttons to support fine-granular interaction with virtual characters in wearable AR applications. We use a hierarchical structure of virtual buttons that are created and destroyed dynamically, according to different levels of proximity from the user's eyes to the image target. A user can point to precise regions covered by virtual buttons for manipulating virtual characters with motion. The goal of this approach is to provide a lightweight pointing interaction method at interactive frame rates for wearable AR systems, such as a binocular wearable AR system. With the proposed method, users can naturally interact with an animated virtual character, such as an augmented dinosaur, that continuously follows or evades the user's pointing and reacts by roaring when the user touches its tail.

2 RELATED WORK IN HAND INTERACTION FOR WEARABLE AR

With technical advancements in AR, an interface to interact with virtual contents is becoming more important for applications. An increase in wearable devices that free a user's hands, such as binocular headsets or glasses, naturally requires more hand interaction in AR.

The simplest way to facilitate hand interaction in an AR environment is to use visual markers. For example, Buchmann et al. proposed fingertip-based interaction for AR by attaching fiducial markers onto fingers and hands in order to obtain a 3D position and gestures. They also applied haptic feedback devices onto two fingertips to make users feel virtual contents. Although this work requires obtrusive markers for tracking a hand, they showed the feasibility of free-hand interaction in AR. Lee et al. suggested occlusion-based hand interaction for tangible AR environments, where users interact with virtual contents augmented on the physical object by manipulating the corresponding physical object. They attached a number of markers to a single object in predefined locations and detected visual occlusion of the markers to obtain a pointing location. Because this technique is computationally light and easy to apply for AR interaction, an image target tracking solution provided by the Vuforia SDK also provides this occlusion-based implementation called a virtual button.

Misty et al. used color markers on the fingertips of the user when using a new wearable gestural interface, which consists of a small projector and a camera on a hat, to track the location of the fingertip. Hürst et al. also made use of color markers on the user's thumb and index finger for examining new interaction metaphors with mobile phones in AR. As with these studies, visual markers are still in use for mobile AR interaction because they are simple to implement and provide relatively robust tracking results.

Recent studies have aimed to interact with virtual contents by using bare-hand tracking for a natural user interface. Baldauf et al. made a fingertip detection engine on Android smartphones by using a lightweight skin color segmentation and contour-based fingertip analysis. Chun et al. developed hand interaction on a mobile phone in AR by detecting and tracking the hand with color and motion cues after marker detection. These works used a simple computer vision approach to obtain the 2D position of fingertips in the current camera image on mobile devices. Therefore, they need depth information, such as the 3D position of the hand, to support pointing and touching the specific parts of virtual contents in a 3D environment.

Several researchers have explored various ways to acquire 3D poses of bare hands in AR. Shen et al. proposed the reconstruction of a camera pose relative to a user's hand by detecting and tracking the dominant features, such as four convexity
defect points on the palm.\textsuperscript{8} Hammer et al. also tried to acquire a hand pose using a single RGB camera.\textsuperscript{9} Mueller et al. also showed robust tracking of the hand using a monocular RGB camera.\textsuperscript{10} They can interact with 3D virtual contents and point at 2D locations, but this approach is not yet available on mobile devices. Seo et al. estimated a 3D palm pose by detecting a convexity defect point between the thumb and the index finger and by acquiring a projected square model from the hand region.\textsuperscript{11} Although they tried to show the result on mobile devices, the frame rate was too slow to be applied in hand interaction for a wearable AR. In addition, Lee et al. proposed bare-hand interaction by detecting hand features and tracking frame to frame through an optical flow.\textsuperscript{12} They took an advantage of multithreaded processing for real-time performance, indicating that their approach needs more polishing before resource-constrained wearable AR platforms can use it.

Harrison et al. presented an interesting work of wearable AR interaction with a wearable depth-sensing and projection system, which projects virtual contents on any surface of the environment and detects a 2D location of fingers on the surfaces for some interactions.\textsuperscript{13} Additionally, Kurz used infrared thermography to detect a 3D fingertip location on an object.\textsuperscript{14} This approach enabled the discovery of the 3D location by detecting the residual heat generated when a user touches the colder surface of the objects with a warm fingertip.

Kim et al. also presented a bare-hand AR manipulation method.\textsuperscript{15} In the paper, they showed the feasibility of using a hierarchical virtual button interface in manipulating a virtual character in a wearable AR environment.

3 | BINOCULAR AR VIEWER

To provide an immersive user experience in an AR environment, we implemented a wearable AR system called a binocular AR viewer (see Figure 1). The viewer consists of two parts: a smartphone and a binocular glass frame. We are required to show stereographic images with a nonstereo video background because the smartphone has only a single camera. To this end, we use binocular disparity to split an input image delivered by the camera into two stereo images. At this time, because we can obtain a distance value called depth value from the camera to an image target by using a Vuforia tracker, we can also adjust the binocular disparity to the distance value. For example, when the distance has decreased, we generate one image for the left eye from the further-left area of the original video image and the other image for the right eye from the further-right area of the original video image to enhance the binocular disparity. When the distance has increased, the previously mentioned action works in an opposite way. Now, we render a virtual character model at the same location of the left- and right-eye images, as shown in Figure 1 (bottom right). In this study, we use the depicted binocular AR viewer in the user study where users hold the viewer close to their eyes with one hand while interacting with a virtual content using the other hand. The viewer can be used with a headset binocular frame, such as Samsung Gear VR, to extend our approach to support both hands.

FIGURE 1  The binocular augmented reality viewer includes a smartphone and a binocular glass frame. The smartphone generates a stereoscopic display for both eyes
4 | OCCLUSION-BASED POINTING INTERFACE

The pointing interface provides two functions: selecting an empty region on the image target and touching a 3D content augmented on the image target. To support these two functions, we created hierarchical layers of virtual buttons dynamically to switch between the layers, according to the distance between the user and the augmented contents.

4.1 | Dynamic hierarchical structure of virtual buttons

In order to make occlusion-based interaction operate in real time, we need a different and efficient approach other than placing hundreds of virtual buttons in an ad hoc manner. Therefore, we propose to use dynamic hierarchical arrangements of virtual buttons, which further divide a selected button’s neighboring regions into smaller virtual buttons, as shown in Figure 2. The full work flow consists of the following steps: First, virtual buttons are placed in a predefined grid where a pointing interaction is known to work in real time (Figure 2, bottom left). This level of the grid is considered as the first layer. When a user points at one virtual button of the first layer with his/her finger, the occluded buttons act as fingertip candidates and one final button is selected as a fingertip.

The fingertip detection is solved by choosing the top-left button (in right-handed persons) from other occluded buttons, which is similar to the method of Lee et al. After determining a fingertip button, nine first-layer virtual buttons (eight surrounding buttons and the selected button), including regions adjoining the selected virtual button, are split into quarters (Figure 2, bottom center). The split virtual buttons are considered to be placed in the second layer. Moreover, applying the same detection algorithm as the first layer then finds a more detailed position of the fingertip in the second layer, which we want to use. The benefit of this dynamic hierarchical virtual button approach is that we can point at a precise location of the fingertip at interactive frame rates on a mobile device by removing unnecessary occlusion checking and event handling.

Moreover, if a user takes a closer look at an image target, which means that the distance between the user and the image becomes closer (Figure 2 [top right]), the third-layer virtual buttons are created dynamically to enable easier access...
to more specific regions (Figure 2, bottom right). To determine when to create the third-layer virtual buttons, we use an empirically calculated distance threshold value that guarantees the feature detection of smaller regions.

### 4.2 Natural transition between layers

To find the location of a fingertip continually while moving around the image target, we need to keep the last layer of virtual buttons around the previous fingertip. One simple way to achieve this rearrangement is to delete virtual buttons of lower layers altogether and to recreate them whenever new fingertips are detected from the upper layer. However, this method results in unnecessary processing of the lower layers, which may slow down performance as fingertips are detected at each frame.

To prevent such situations, we access layers backwards when interacting, as shown in Figure 3. When a user looks at the image target, the distance between the user and the image target is estimated. The distance is compared with the predefined distance threshold value to determine the layer starting level. Here, the starting layer is generated at the center regions of the upper layer. Once the starting layer of the virtual buttons is generated, we check on the starting layer to begin when the user points with a fingertip. If there are no occluded virtual buttons on the starting layer, we go back to the upper layer and test for occluding virtual buttons. Otherwise, we detect a fingertip at the starting layer and examine whether it is in a boundary of the layer. If the fingertip is a part of the boundary in the layer, we go back to the upper layer and carry out the hierarchical arrangement of the virtual buttons. From this approach, we can switch between layers naturally (refer to the accompanying video).

---

**FIGURE 3** Flowchart for the natural transition between layers. The distance between the user and the image target determines the layer starting level. FT = fingertip
4.3 | Touching interaction

In addition to exploiting a fingertip as a pointer, we can extend this interface into touching interaction. To touch a part of a virtual character, we need to estimate a 3D point of the part to pick. This means that we need to track the 3D position of the fingertip. It is not easy with current mobile devices. To resolve this, we cast a ray from the camera to the selected fingertip and obtain an intersection point against the virtual character, as shown in Figure 4. Using this interaction, we can touch any part of the 3D object, as wanted.

5 | EVALUATION

We evaluated our interface to verify its validity and effectiveness. We implemented our method on Nexus 5 and Galaxy Note 4 smartphones and used Vuforia Unity Extension2 as an image tracker to run experiments on an image target test bed (40 × 40 cm), as shown in Figure 1.

We compared our hierarchical method against a full arrangement of virtual buttons by varying virtual button size on the same image target size as follows.

- **FULL3 × 3**: full arrangement of virtual buttons where a button is sized 3 × 3 cm. The virtual buttons are placed in a 10 × 12 array on the first layer. Therefore, the available pointing area size is 30 × 36 cm, and the total number of virtual buttons is 140.
- **DHA3 × 3**: dynamic hierarchical arrangement of virtual buttons where a button is sized 3 × 3 cm. To this end, we put virtual buttons whose size is 6 × 6 cm each into a 5 × 6 array on the first layer in advance. Then, the virtual buttons of 3 × 3 cm are dynamically placed in a 6 × 6 array as the second layer around and over the selected virtual button of the first layer. As a result, the size of the available pointing area is 30 × 36 cm, and the total number of virtual buttons is 66.
- **FULL1.5 × 1.5**: full arrangement of virtual buttons where a button is 1.5 × 1.5 cm. The virtual buttons are placed in a 20 × 24 array on the first layer. Thus, the accessible pointing area size is sized 30 × 36 cm, and the total number of virtual buttons is 480.
- **DHA1.5 × 1.5**: dynamic hierarchical arrangement of virtual buttons where a button is sized 1.5 × 1.5 cm. For this end, we put virtual buttons with a size of 6 × 6 cm each into a 5 × 6 array on the first layer in advance. Then, the virtual buttons of 3 × 3 cm are dynamically placed in a 6 × 6 array as the second layer around and over the selected virtual button of the first layer. Finally, the virtual buttons of size 1.5 × 1.5 cm are also dynamically placed in a 6 × 6 array.
Table 1 shows the frame rate results to compare the effectiveness of our dynamic hierarchical method against a full arrangement method during pointing interaction. We identified that the mean frame rate of our method is higher than that of the full arrangement, especially when the virtual button size is decreased to 1.5 × 1.5 cm. This means that as the number of virtual buttons increases, the frame rate becomes slower. Moreover, even if we implement the methods on a faster Galaxy Note 4 to improve performance, a FULL 1.5 × 1.5 condition still produces a lower frame rate, as shown in Table 1.

Because our interface is based on pointing input, we implemented a target selection task, which is widely used in Fitts's law. This task is repeated for 50 times per condition. Figure 5 shows the processing time results of four conditions. Preparation time is the time required to generate and arrange virtual buttons at the initial stage, and movement time is the time taken during the target selection task. This result shows that, if the virtual button size becomes smaller, the processing time grows larger because of the increased number of virtual buttons. This trend is more noticeable in the full arrangement method. On the other hand, our method shows the reduction of processing time, as shown in Figure 5.

In order to predict the time required for moving a target area and modeling the pointing method, we compared Fitts's law lines according to the four conditions, as shown in Figure 6. The index of difficulty of the graph is the metric that quantifies the difficulty of a target selection task. We used the index of difficulty called Shannon formulation as follows:

\[
\text{index of difficulty} = \log_2 \left( \frac{D}{W} + 1 \right),
\]

Table 1. Frame rate performance

| Virtual button size | Nexus 5  |  | Galaxy Note 4  |  |
|---------------------|----------|-----------------|-----------------|-----------------|
|                     | FULL Mean (SD) | DHA Mean (SD) | FULL Mean (SD) | DHA Mean (SD) |
| 3 × 3 cm            | 22.72(1.58)  | 28.35(2.19)  | 35.33(1.17)  | 39.04(2.55)  |
| 1.5 × 1.5 cm        | 4.87(0.25)   | 15.41(2.99)  | 10.49(0.49)  | 26.74(1.39)  |

Note. Mean (SD) of frame rate with full arrangement (FULL) and dynamic hierarchical arrangement (DHA) according to the virtual button size on Nexus 5 and Galaxy Note 4 devices (unit: fps).
where $D$ denotes the distance from the start target to the center of the stop target, and $W$ is the width of the target. Figure 6 illustrates that FULL$3 \times 3$ and DHA$3 \times 3$ can be modeled similarly to the pointing method. However, when the virtual button size becomes smaller, such as in FULL$1.5 \times 1.5$ and DHA$1.5 \times 1.5$, our method (DHA$1.5 \times 1.5$) outperforms the full arrangement method, especially in the low index of difficulty.

5.2 | User study

To evaluate the feasibility of our method, we conducted a user study for a pointing task in wearable AR applications. To this end, we recruited 15 participants (10 men and five women) between the age of 25 and 41 (the mean was 31.67, and the standard deviation was 5.46). Eight participants had prior experience with AR interaction. Two participants reported color blindness, but it did not affect our experiment.

In order to give no priorities regarding order, we switched the order for every four participants. After the task, participants ranked each survey statement about feasibility on a scale of 1 (strongly disagree) to 7 (strongly agree) as follows.

• Efficiency: I could feel that the technique gave a natural way for a pointing interface.
• Satisfaction: I was satisfied with the technique. Therefore, I want to use this technique again.

As shown in Figure 7a, our dynamic hierarchical arrangement of virtual buttons was preferred by the participants who tried all the four conditions in two varying sizes of virtual buttons. Moreover, we have found that the smaller the virtual button size, the bigger the difference in preference (see FULL$1.5 \times 1.5$ vs. DHA$1.5 \times 1.5$). Most participants mentioned that they prefer the dynamic hierarchical arrangement for the pointing task because of the fast response rate and short preparation time. Moreover, some participants reported that, although they cannot recognize the difference between FULL$3 \times 3$ and DHA$3 \times 3$, they were able to identify FULL$1.5 \times 1.5$ from DHA$1.5 \times 1.5$ in terms of slower reaction rate.

To check the statistical significance of the results, we used a within-subjects ANOVA test on the four conditions. Through Welch’s ANOVA test, there was a significant difference between the four conditions, $F(3, 30.8632) = 22.13$, $p < 0.001$ in efficiency, and $F(3, 30.8145) = 26.22$, $p < 0.001$ in satisfaction. In order to identify which condition means are different, the Games–Howell method was used to make post hoc comparisons between the four conditions. As shown in Figure 7b, there was no significant difference of means in only the FULL$3 \times 3$ and DHA$1.5 \times 1.5$ conditions in efficiency. Accordingly, FULL$3 \times 3$ and DHA$1.5 \times 1.5$ were categorized in the same group named as “B”. This result suggests that, even if the virtual button size is smaller, the participants do not realize efficiency reduction by using a dynamic hierarchical arrangement. With regard to satisfaction, there was no significant difference of means in both FULL$3 \times 3$ versus DHA$3 \times 3$, and FULL$3 \times 3$ versus DHA$1.5 \times 1.5$ conditions, as shown in Figure 7c. Consequently, FULL$3 \times 3$, DHA$3 \times 3$, and DHA$1.5 \times 1.5$ were categorized in the same group named as “A”. From this result and Table 1, we have found that the participants are willing to use a method if the response rate of the method reaches over a certain level, such as 15 fps in DHA$1.5 \times 1.5$.

6 | DISCUSSION

Our goal is to develop a practical pointing interface for wearable AR applications, where we do not have any other devices such as a depth camera, and to enable the public to exploit their smartphone to reduce costs. Inspired by the Vuforia
occlusion-based virtual buttons, we first placed a set of multiple virtual buttons in a 5×6 array to cover the 30×36 cm parts of the image target. However, in this case, the virtual button size is 6×6 cm and does not support a more precise pointing, which is necessary to touch a specific part of a virtual character. Reducing the virtual button size is a possible solution, but the number of virtual buttons becomes larger accordingly. Our proposed dynamic hierarchical arrangement decreases not only the size but also the number of virtual buttons to support a reasonable response rate and more precise pointing than the existing full arrangement method. Moreover, we have found that our method can be applied when the virtual button size is similar to a finger (1.5×1.5 cm), and users feel that it is not inconvenient, as shown in the previous quantitative and qualitative results.

Furthermore, we exploited our dynamic hierarchical virtual button-based interaction in some applications that are on display for the public to experience, as shown in Figure 8e,f. Figure 8a,b demonstrates an application in which a dinosaur follows or evades a user’s fingertip. If the user moves his/her fingertip across an image target, the second layer of virtual buttons is generated around the fingertip, as shown in Figure 8b. Brown translucent boxes represent the first layer, and yellow translucent boxes represent the second layer. After determining the fingertip through occluded virtual buttons, we make use of the selected pointer as a goal point to be followed by a dinosaur (Figure 8a). To this end, we adopted the arrival behavior of the steering behavior algorithm into a character’s movement. We can also use the selected pointer as a target point to be evaded by the dinosaur (Figure 8b). Here, we applied the flee behavior of the steering behavior algorithm into the character.
FIGURE 8  Applications. A dinosaur (a) follows or (b) evades a user's finger. (c and d) The user can also touch parts of the dinosaur with a finger. The dinosaur reacts to the parts, such as the tail and the head (refer to the accompanying video). (e and f) These applications are on display for the public to experience.

Figure 8c,d shows another application of touching specific parts of the dinosaur, such as the head, body, and tail, as well as a certain point. In order to recognize which point of the character is touched, we cast a ray from a camera to the selected fingertip against the virtual character to get an intersection point (the green-colored one). Therefore, we need a smaller size of the virtual button in this application and introduce the third layer of the virtual button, as shown in Figure 8d. Green translucent boxes indicate the third layer; other translucent boxes are the same in the previous application. If users touch the head or tail of the dinosaur, the character reacts according to the part (refer to the accompanying video).

In addition to the previously discussed applications, our proposed method can be used for such places as a virtual private keypad or work space display. For instance, if existing digital door locks have an image instead of a keypad, a virtual private keypad appears on the image through our system. Users can then enter their secret code number without worrying about who will look at it.

There are some limitations to what we have found. Because we exploit Vuforia tracking, the accuracy of the tracker influences our system. Feature points should be evenly distributed on the whole image target to apply our pointing interface. For example, when a part of the image without feature points is bigger than the desired virtual button size, selecting an exact fingertip location fails. However, due to the robust tracking solution of Vuforia, our system works well when using somewhat uneven image targets, as shown in Figure 8f. In the generalization of the hierarchy, we can extend more levels in the hierarchy, as previously stated in the occlusion-based pointing interface section, but it will slow down speed because of the time required to transit between layers. Nevertheless, we have found that constructing three layers is sufficient to deal with a fingertip-size pointing interface.
7 | CONCLUSION

In this study, we have developed a practically fast pointing interface on a wearable device for interacting with a virtual character in an AR environment. In particular, we focused on how to allow the public to use their own smartphone device without any other apparatus, such as a depth camera and a notebook computer. For this end, we proposed a dynamic hierarchical arrangement of virtual buttons to support a practical fingertip-sized pointing interface. This approach enabled the users to interact with the virtual character as if they are using bare hands without actual computation for hand tracking. Through performance measurements and user studies, we showed that our pointing interface is a feasible and efficient method for functional wearable AR applications.

In future work, we plan to develop techniques for using a wrist gesture through a smartwatch, as well as a fingertip location. This combination of wrist and finger movement tracking will facilitate various kinds of virtual character interactions.

ACKNOWLEDGEMENTS

This research is supported through the Korea Institute of Science and Technology (KIST) Institutional Program and by the Korea Creative Content Agency under the Culture Technology Research and Development Program.

ORCID

Jae-In Hwang http://orcid.org/0000-0002-8748-9077

REFERENCES

1. Lee GA, Billinghurst M, Kim G. Occlusion based interaction methods for tangible augmented reality environments. Paper presented at: VRCAI ’04. Proceedings of the 2004 ACM SIGGRAPH International Conference on Virtual Reality Continuum and its Applications in Industry; 2004 Jun 16–18; Singapore. New York: Association for Computing Machinery; 2004. p. 419–426.
2. Qualcomm. Vuforia [online]. Available from: http://www.qualcomm.com/products/vuforia
3. Buchmann V, Violich S, Billinghurst M, Cockburn A. FingARtips: gesture based direct manipulation in augmented reality. Paper presented at: GRAPHITE ’04. Proceedings of the 2nd International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia; 2004 Jun 15–18; Singapore. New York: ACM; 2004. p. 212–221.
4. Mistry P, Maes P, Chang L. UWU - wear Ur world: A wearable gestural interface. Paper presented at: CHI ’09. Proceedings of the 27th International Conference on Extended Abstracts on Human Factors in Computing Systems; 2009; Boston, MA. New York: Association for Computing Machinery; 2009. p. 4111–4116.
5. Hürst W, van Wezel C. Gesture-based interaction via finger tracking for mobile augmented reality. Multimed Tools Appl. 2013;62(1):233–258.
6. Baldauf M, Zambanini S, Fröhlich P, Reichl P. Markerless visual fingertip detection for natural mobile device interaction. Paper presented at: MobileHCI ’11. Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services; 2011 Aug 30–Sep 2; Stockholm, Sweden. New York: Association for Computing Machinery; 2011. p. 539–544.
7. Chun WH, Höllerer T. Real-time hand interaction for augmented reality on mobile phones. Paper presented at: IUI ’13. Proceedings of the 2013 International Conference on Intelligent User Interfaces; 2013; Santa Monica, CA. New York: Association for Computing Machinery; 2013. p. 307–314.
8. Shen Y, Ong SK, Nee AYC. Vision-based hand interaction in augmented reality environment. Int J Hum Comput Interact. 2011;27(6):523–544.
9. Hammer JH, Beyerer J. Robust hand tracking in realtime using a single head-mounted RGB camera. Paper presented at: MobileHCI ’13. Proceedings of the 15th International Conference on Human-Computer Interaction: Interaction Modalities and Techniques; 2013 Jul 21–26; Las Vegas, NV. Berlin, Germany: Springer; 2013. p. 252–261.
10. Mueller F, Bernard F, Sotnychenko O, et al. GANerated hands for real-time 3D hand tracking from monocular RGB. 2017. arXiv:1712.01057.
11. Seo B-K, Choi J, Han J-H, Park H, Park J. One-handed interaction with augmented virtual objects on mobile devices. Paper presented at: VRCAI ’08. Proceedings of the 7th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry; 2008 Dec 8–9; Singapore. New York: Association for Computing Machinery; 2008. p. 8.1–8.6.
12. Lee T, Höllerer T. Hybrid feature tracking and user interaction for markerless augmented reality. Paper presented at: Virtual Reality Conference; 2008; Reno, NE.
13. Harrison C, Benko H, Wilson AD. Omnitouch: Wearable multitouch interaction everywhere. Paper presented at: UIST ’11 Proceedings of the 24th annual ACM Symposium on User Interface Software and Technology; 2011 Oct 16–19; Santa Barbara, CA. New York: Association for Computing Machinery; 2011. p. 441–450.
14. Kurz D. Thermal touch: Thermography-enabled everywhere touch interfaces for mobile augmented reality applications. Paper presented at: IEEE International Symposium on Mixed and Augmented Reality; 2014 Sept 10–12; Munich, Germany.
Kim H, Widjojo EA, Hwang JI. Dynamic hierarchical virtual button-based hand interaction for wearable AR. Paper presented at: IEEE Virtual Reality; 2015 Mar 23–27; Arles, France. Piscataway, NJ: Institute of Electrical and Electronics Engineers; 2015.

Fitts PM. The information capacity of the human motor system in controlling the amplitude of movement. J Exp Psychol. 1954;47(6):381–391.

Reynolds CW. Steering behaviors for autonomous characters. Paper presented at: Game Developers Conference; 1999; San Jose, CA.

Hyejin Kim is a senior research engineer in Software Platform R&D Lab. at LG Electronics. She received the B.S. degree in IT and electronics engineering at Soongsil University, Korea, in 2006. She received her M.S. and Ph.D. degrees from Gwangju Institute of Science and Technology (GIST), Korea, in 2007 and 2014, respectively. She was a postdoctoral researcher in Image Media Research Center at Korea Institute of Science and Technology (KIST), Korea (2014~2015). Her research interests include virtual/augmented reality, human-computer interaction and data-driven character animation.

Jae-In Hwang is an associate professor at KIST School and a senior research scientist in Imaging Media Research Center at Korea Institute of Science and Technology. He received the B.S., M.S, and PhD. degree in computer science and engineering at Pohang University of Science and Technology, Korea, in 1998, 2000 and 2007. He was a visiting scholar at ICT, University of Southern California during 2003~2004, 2012.

How to cite this article: Kim H, Hwang J-I. Manipulating augmented virtual character using dynamic hierarchical pointing interface. Comput Anim Virtual Worlds. 2018;29:e1824. https://doi.org/10.1002/cav.1824