BEHAVIOUR & INFORMATION TECHNOLOGY
https://doi.org/10.1080/0144929X.2020.1757760

ArmMenu: command input on distant displays with proprioception based lateral arm movements

Huawei Tu[a,b], Weiyang Huan[b], Xingdong Yang[c], Xiangshi Ren[d] and Feng Tian[e]

[a]Department of Computer Science and Information Technology, La Trobe University, Melbourne, VIC, Australia; [b]College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, People's Republic of China; [c]Computer Science, Dartmouth College, Hanover, NH, USA; [d]School of Information, Kochi University of Technology, Kochi, Japan; [e]Institute of Software, Chinese Academy of Sciences, Beijing, People's Republic of China

ABSTRACT
In this paper, we present ArmMenu, a command input approach for distant displays. ArmMenu has a circular interface like pie menus and menu selection is performed by proprioception-based lateral arm movements. We implemented ArmMenu with an off-the-shelf body tracking device (Kinect) and conducted two experiments to validate its efficacy. In the first experiment, we explored the design space of ArmMenu by varying the number of menu items, with exposed or hidden menu modes. Users can operate up to 8-item menus with high selection accuracy (>98%). ArmMenu was fast and accurate even with the hidden menu mode. The second experiment compared the performance of ArmMenu and touchless marking menus. While having similar selection accuracy, ArmMenu was faster and more preferable by users. Our studies consequently demonstrate ArmMenu's effectiveness for command input on distant displays.

ARTICLE HISTORY
Received 23 October 2018
Accepted 15 April 2020

KEYWORDS
Touchless interaction; menus; distant displays; command input; motion sensing devices; proprioception senses

1. Introduction

The availability of unobtrusive motion sensing devices (e.g. Microsoft Kinect, Sony PlayStation Camera and Asus Xtion Pro Live) has greatly changed the way of interacting with distant displays. Users can control the console or computer with their body at a distance, without physically touching or holding hardware. Therefore, this style of interaction has shown particular value in application scenarios such as gaming (Nakai et al. 2015), AR/VR (Wilson et al. 2016), medical operations (Jacob and Wachs 2014) and virtual design (Vinayak et al. 2013).

Command input is an integral task in these practical usage scenarios (e.g. selecting a sport item in the menu of Kinect Sports). Menus have been used for exploring and selecting commands on interactive remote displays. Pointing is a common menu selection technique in commercial body tracking devices (Walter et al. 2014). Such technique uses the metaphor of a cursor to point at a menu item on remote displays, usually by mapping in-air hand movements to on-screen coordinates. An example is the *point + dwell* technique in the 2010 Xbox 360 touchless interface (Kinect 2018a), which requires users to hold their hand over an item for a defined amount of time to select it. However, pointing-based methods were reported to lack accuracy on current low-cost remote sensing devices like Kinect whose tracking accuracy is still unsatisfactory for accurate mid-air interaction (Sambrooks and Wilkinson 2013) and tended to slow interactions (Schwaller and Lalanne 2013). Many menu techniques also rely on remote detection of static or dynamic gestures of the hand or body that correspond to a predefined set of commands, such as counting fingers to select menu items (Kulshreshth and LaViola 2014). But gesture learning poses significant challenges for users (Alt, Geiger, and Höhl 2018; Nacenta et al. 2013), as they generally have to learn multiple gestures, and recall one to execute its associated command. Visually guided body gestural techniques such as touchless marking menus (Bailly et al. 2011; Bossavit et al. 2014; Lenman, Bretzner, and Thuresson 2002) can be an alternative, as users can perform hand gestures following menu-item direction when needed. But touchless marking menus usually require accurate hand movements in the air to move the on-screen cursor towards a specified direction, which may be demanding for users.

In this study, we propose ArmMenu, a visually guided menu technique on remote interfaces using lateral arm movement gestures based on human proprioceptive sensations. ArmMenu provides a circular menu interface with lateral arm movement gestures (rather than extending the arm to the front with marking menus) for item selection (Figure 1). Lateral arm movements mean moving an arm away from or toward the midline of the body.

CONTACT Huawei Tu tuhuawei09@gmail.com

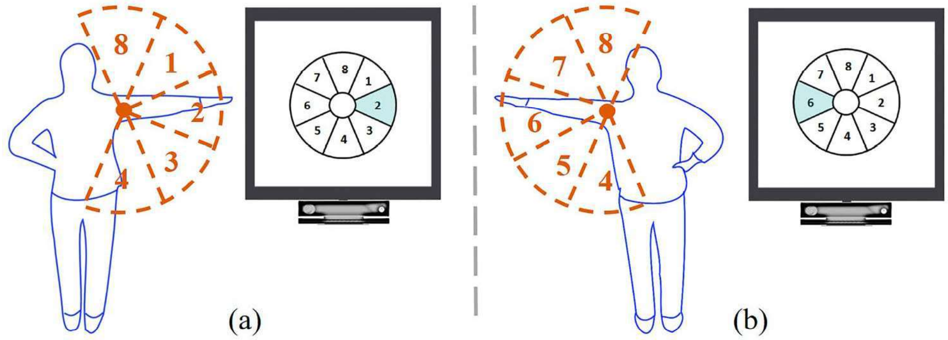


Figure 1. Menu selection with ArmMenu. The interface of ArmMenu is shown on the display. The dotted arcs represent that arm movement spaces which are evenly divided into multi-sectors by the dotted lines. Numbers in the sectors and the interface indicate interface layout is coupled with the space of lateral arm movements. For example, the user lifts (a) the right arm within sector 2 to select menu item 2 and (b) the left arm within sector 6 to select menu item 6. Note either arm can be assigned to selecting item 4 and 8.

Proprioception is 'the perception of joint and body movement as well as position of the body, or body segments, in space' (Sherrington 1952). It is one of the inherent sensations of human beings. With proprioceptive sensations, we are able to indicate our limbs' positions with reasonable accuracy, even when we are not directly looking at them. For example, we can easily stretch our arm straight out toward the horizon without visual attention. ArmMenu was proposed by taking advantage of such human ability, hence may enable users to execute it with ease and high efficiency. ArmMenu's interface is identical to pie menus (Callahan et al. 1988), as illustrated in Figure 1. In order to select a menu item, the user extends one of his arm out to the side of his body while keeping the other arm akimbo,[1] and arm direction (i.e. the direction from the shoulder joint to the wrist joint) in the frontal plane of the body can be mapped to a menu item to indicate which one to be selected (Figure 1).

We carried out two experiments to explore the design space of ArmMenu with the Kinect V2 sensor. In the first experiment, we investigated users' ability to perform discrete target selection tasks by varying the number of menu items with exposed or hidden menus. User performance differed in numbers of menu item and menu types. In the second experiment, we verified the effectiveness of ArmMenu in comparison with free-hand marking menus (Bailly et al. 2011; Bossavit et al. 2014; Lenman, Bretzner, and Thuresson 2002). ArmMenu outperformed marking menus in task time and subjective feedback while having comparable selection accuracy.

The contributions of our study are three-fold. First, we proposed a novel command input method for touchless interaction with remote displays based on the proprioception theory. Second, we systematically investigated human control ability of lateral rotation movements of arms for ArmMenu design. Third, through reporting the development and verification of ArmMenu, we demonstrated its potential as a viable command-input technique for remote display interaction.

The rest of this paper is organised as follows. After a review of related work, we describe ArmMenu design with algorithms for body gesture recognition. Then we report two experiments to examine the efficiency of ArmMenu with an analysis of experiment data. Last, we generally discuss the results, limitations and future work.

2. Related work

Command input is an indispensable task when interacting with distance displays. Users can perform the task either by operating interactive devices (Haque, Nancel, and Vogel 2015; Shoemaker et al. 2010) or by utilising body parts without physical manipulation of any devices (Vogel and Balakrishnan 2005). The latter way has gained considerable attention in the HCI field because the body possesses a rich set of abilities that permits the usage of body parts as mediators (Klemmer, Hartmann, and Takayama 2006), and also it is a natural and convenient way to interact with remote displays. ArmMenu belongs to the latter category, so we focus on this topic and review related studies as follows.

2.1. Proprioceptive sensations

ArmMenu is designed based on the theory of human proprioceptive sensations. Proprioception is a fundamental sense of the relative position of one's own parts of the body. The term of proprioception was first introduced in 1906 by Charles Scott Sherrington in his landmark book titled 'The integrative action of the nervous system' (Sherrington 1952). The sense is fundamental to our functioning. It affects our day-to-day activities and allows people to accomplish complex tasks such as driving cars where users need to keep eyes on the road and simultaneously adjust arms and hands on the steering wheel. A review of proprioception can be found in (Han et al. 2016; Proske and Gandevia 2012).

This study is focused on limb proprioception. There are many related studies in the physiological field (e.g. Fuentes and Bastian 2010; Proske and Gandevia 2012; Schofield 1976). In the HCI field, use of arm proprioception has also received much interests. For example, Uddin, Gutwin, and Lafreniere (2016) and Uddin and Gutwin (2016) designed single-handed and two-handed HandMark Menus using hands as a landmarking technique for command selection on multi-touch displays. Their work leverages the proprioceptive knowledge of users' hands for touch-based command input. Mine, Brooks, and Sequin (1997) explored object manipulation with hands in immersive virtual environments based on proprioception. Bossavit et al. (2014) designed a menu selection technique which attached virtual menu items to different parts of the body and required users to select them by reaching these zones with their hands. Li, Dearman, and Truong (2009) present Virtual Shelves, a technique to invoke shortcuts on a mobile device by orienting a spatially-aware mobile device within the circular hemisphere in front of users based on their spatial awareness and proprioceptive senses. Lopes et al. (2015) proposed a novel way of eyes-free interaction for wearable devices that offers input and output based on wrist proprioception. Our study designed ArmMenu with lateral arm movements for remote menu interaction based on proprioceptive sensations.

2.2. Touchless pointing for menu selection

Touchless pointing techniques usually use the hand to directly control a cursor on screen. Since such techniques follow the metaphor of a cursor, users who are familiar with cursor-based interfaces can learn touchless pointing quickly (Chertoff, Byers, and LaViola 2009; Vogel and Balakrishnan 2005). Therefore, touchless pointing is the most prevalent style in commercially available body tracking devices and has attracted much attention (Walter et al. 2014).

Hover-to-select is the primary form of touchless pointing on the Xbox 360 and Xbox One interfaces (Kinect 2018a). This method requires users to point the palm of their hand toward the screen and move the on-screen cursor over the target item for a defined amount of time to select it. While simple, this method has drawbacks like exacerbating 'Gorilla-Arm' fatigue issues (Hincapié-Ramos et al. 2014; Jang et al. 2017) and slowing interactions (Schwaller and Lalanne 2013).

Press-to-select, a variant of *hover-to-select*, has been adopted in the more recent Xbox One console (released in 2013). The two techniques are similar except for the way to confirm selection; unlike *hover-to-select*, *press-to-select* requires users to 'push' their hand directly towards the screen to select a target item. A comparative evaluation between the two techniques showed *press-to-select* was preferred by most participants despite it being less accurate than *hover-to-select* (Yoo et al. 2015). However, this style of interaction has been shown to lack accuracy (Sambrooks and Wilkinson 2013).

Overall, touchless pointing has its own pros and cons. The advantages include: (1) it is a visually guided interaction process, so searching a menu item is not highly demanding, especially menu hierarchy tends to be simple for remote interfaces; (2) it is a direct adaptation from pointing techniques for surfaces and desktop computers so it is familiar to computer users. On the other hand, the disadvantages are: (1) *Press-to-select* and *hover-to-select* are regarded to slow interaction process; (2) menu items should be large enough to compensate for inadequate accuracy of pointing input.

2.3. Body gesturing for command input

Body gestures are a motion of the body which contains information. They have been used to interaction on remote displays (e.g. counting fingers to select menu items Kulshreshth and LaViola 2014 or hand postures for multi-finger raycasting Matulic and Vogel 2018). They entail a couple of advantages for touchless interaction with remote displays. First, many of them resemble daily life actions (e.g. body gestures in Kinect games for Xbox 360 Kinect 2018a), hence are natural to use. Second, gestures can be committed to kinaesthetic memory, which helps users focus on their task (Zhai et al. 2012). Third, they provide an error-tolerant feature. In other words, even if a drawn gesture does not perfectly match its prototype, gesture recognisers should be able to judge its correctness as recognition algorithms usually rely on a set of gesture features (Rubine 1991).

Body gestures' main limitation is that users have to learn gestures and recall them before interactions, often resulting in high false-positive recall (Nacenta et al. 2013). Several methods have been proposed to alleviate this limitation, for example, user-defined gestures (Wobbrock, Morris, and Wilson 2009) or mimicry of conventions like drawing a letter (Li 2010). However, gesture memorisation is still a necessity, which renders gestures unsuitable for numerous application contexts.

Visually guided body gestural techniques have been proposed to address the above shortcoming. Carter et al. (2016) proposed PathSync, a technique for interacting with digital objects on remote displays by replicating the movement of a screen-represented pattern with their hand. This technique is suitable for multi-user interaction on distant displays. Its limitation lies in how to make it applicable for all UI designs. Touchless marking menus (Bailly et al. 2011; Bossavit et al. 2014; Lenman, Bretzner, and Thuresson 2002) or its variants (Chattopadhyay and Bolchini 2014; Ren and O'Neill 2012) are also a type of visual-guided gestural techniques for command input, but they require hand movement in a specified direction with a high level of accuracy (Chattopadhyay and Bolchini 2015). Our work is inspired by touchless marking menus and we designed ArmMenu with their limitations and strengths in mind.

3. ArmMenu design

In this section, we first present interface design and interaction design for ArmMenu. We then describe how to implement ArmMenu with the Kinect device.

3.1. Interface design

Inspired by pie menus (Callahan et al. 1988), ArmMenu's interface is a circular context menu where selection depends on arm direction (Figure 1). The main reason we adopted such design is that the angle of lateral rotation of arms can be directly mapped to a menu item thanks to the circular layout of the interface. Interface design is identical for submenus if ArmMenu has multiple levels in the hierarchy.

3.2. Interaction design

Picture a scene where a user wants to invoke a game guide menu when playing a Kinect game. With ArmMenu, the user can hold arms akimbo to do so (Figure 2(a)). This posture acts as a delimiter to tackle the challenge of separating intended body gestural input for ArmMenu from normal body motion. We used this posture as it is common in daily life and has been used in body-centric interaction design (Walter, Bailly, and Müller 2013).

For menu selection, users extends one of their arm out to the side of their body while keeping the other arm akimbo (Figure 2(b)). During selection process, users can focus on the task rather than moving their arms, as they can approximately sense the position of the extended arm based on proprioceptive sensations. Due to lateral movement restraints, each arm is allocated to selecting items in the left or right half-side of the interface, e.g. the left arm for item 4 to 8 (Figure 1(b)).

We considered two options to confirm selection. One is that the user makes a fist with the hand of the stretched arm after starting ArmMenu and opens the fist to confirm selection. The other is that the user holds the arm within the target item area for a defined amount of time (400 ms in this tudy. A pilot test showed the time was long enough for users to move their arm across menu items to reach the target one). Compared to the former method requiring a transition from moving the hand to closing/opening the fist, the latter method is a smoother interaction process. Our pilot study with three female and three male participants indicated that they could confirm menu selection easily and accurately with the latter one, and prefer it as well. So we decided to adopt the latter method.

After finishing selecting *the current-level item*, users can keep their arm posture for more than 400 ms and this does not confirm menu selection in *the next-level menu*. There are three cases users may experience for

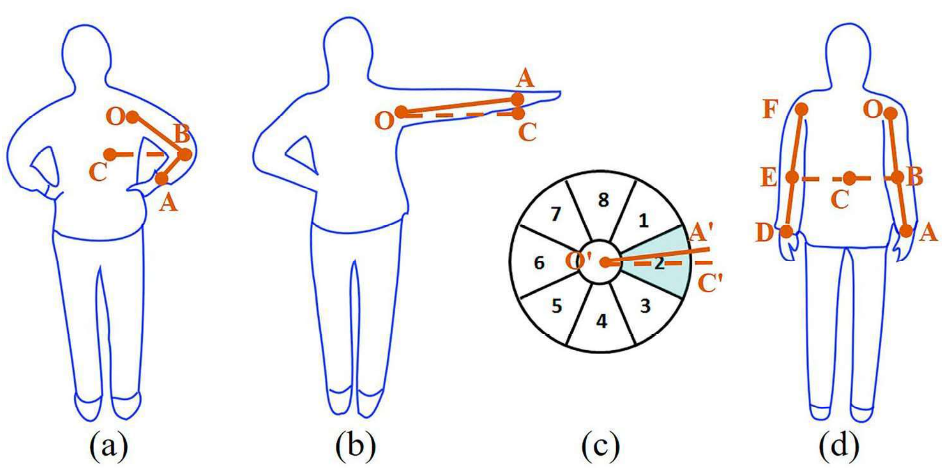


Figure 2. Point A and D represent wrist joints, B and E represent elbow joints, O and F represent shoulder joints tracked by the Kinect V2 sensor. The dotted lines represent horizontal lines (line BC, OC, EC and O'C'). Illustrations for (a) arm akimbo detection; (b) calculation of angle between the arm and the horizontal line; (c) mapping $\angle AOC$ to ArmMenu's interface; (d) posture for quitting ArmMenu.

submenu selection (Figure 3). First, *the current-level item* and *the next-level item* are in the same position of the layout (Figure 3: case 1). In this case, to select the menu, users need to move their arm out of the menu area and move it into the menu area again. A comfortable way to do this is to make the arms-akimbo posture first and then move the arm into *the next-level item* again. Second, *the current-level and the next-level items* are based in the same half-side (left or right) of their menu area but not in the same position of the layout (Figure 3: case 2). For this case, users just move their hand into *the next-level item area*. Third, *the current-level and the next-level item* are not based in the same half-side (left or right) of their menu area (Figure 3: case 3). Users retract the arm to make arms-akimbo posture again, and then move the other arm into *the next-level item*. For all three cases, the way to confirm selection is the same as described in the above paragraph.

After completing menu selection, the user would quit menu interfaces (e.g. leave game guide interfaces and continue to play games). For ArmMenu, this can be done if both arms are not akimbo (Figure 2(d) as an example).

3.3. Implementing ArmMenu with Kinect

We implemented ArmMenu with the Kinect V2 sensor. The algorithms for arm gesture recognition are detailed below.

Users posture arms akimbo to invoke ArmMenu. For an arm skeleton tracked by the Kinect V2 sensor (Figure 2(a)), this arm is akimbo if both $\angle ABC$ and $\angle OBC$ are no more than 80°. ArmMenu is activated if both arms are in this posture.

The angle between the stretched arm and the horizontal line ($\angle AOC$ in Figure 2(b)) is used to determine which menu item is selected. The angle is translated so that point O coincides with the centre of ArmMenu interface (Point O’). An item is under selection if line A’O’ crosses its area (e.g. Figure 2(c): item 2).

Users stop ArmMenu if both arms are not akimbo (Figure 2(d) as an example, where $\angle ABC$ and $\angle DEC$ are more than 80°).

It should be noted that our algorithms can also apply to ArmMenu development with other vision-based body tracking systems (e.g. Asus Xtion Pro Live or Vicon Motion Systems) if those systems can detect shoulder, elbow and wrist joints.

4. Experiment 1: user ability of performing ArmMenu

This experiment aimed to investigate human ability to perform menu selection tasks with ArmMenu. Specifically, we would like to address the following questions.

Q1. How would the number of menu items affect user performance? Intuitively, as the number of menu items increases, there is an increasingly added cognitive cost involved when users search for and select items; user performance therefore may degrade.

Q2. What would the performance of using ArmMenu be for novice and experienced users? Novice users are not familiar with menu layout. So they need to rely on visual search to find the target item. In contrast, experienced users desire a faster access to the menu items as they know menu layout well and should take less effort to search for intended items.

Q3. Could ArmMenu work well with hidden menu interfaces? We aimed to investigate the effects of human proprioceptive sensations on ArmMenu’s efficacy, so we compared ArmMenu’s performances between hidden and exposed menu interfaces. Users could only rely on proprioceptive sensations to perform tasks for the hidden menu case.

4.1. Apparatus

We used a Kinect V2 device to track participants’ body movements. The Kinect device was connected to a HP ENVY 15 Notebook PC having a Windows 10 OS, i7 CPU, 8G memory and a 15.3-inch display with 1920 × 1080 resolution. We developed an experiment program in WPF environments using Kinect V2 SDK. The experiment interface was projected on a 50-inch screen in landscape format by a Epson Home Cinema 3700 projector (resolution: 1920 × 1080, throw ratio: 1.34) which was connected to the notebook. The distance between the screen and the projector was 1.48 m.

4.2. Participants

Twelve right-handed participants (6 male and 6 female) from the local university took part in the experiment. They were 20.3 years old (SD = 2.2 years) and 169.3 cm tall (SD = 11.6 cm) on average. None had prior experience in using Kinect.

4.3. Experiment design

The experiment followed a within-subject repeated measures design with two independent and two dependent variables.

The independent variables were menu layout and operation modes. We adopted the way in (Kurtenbach, Sellen, and Buxton 1993) to design menu layout of ArmMenu. For simplicity, only one level of the menu hierarchy was considered. The menus contained 4, 5, 7, 8, 11, or 12 slices. All menu items had numbered segments, always beginning with a '1' immediately adjacent to the right of the top segment. The other slices were labelled in clockwise order with the maximum number at the top (Figure 1). For all menus, the diameter of the outer circle and the inner circle was 45 and 15 cm respectively on the projection screen. The number was labelled in Verdana 18-point bold font. All participants reported they could see the numbers clearly.

The experiment also evaluated the effects of operation modes on menu selection performance. Operation modes had three types: *exposed menu with number indicator*, *exposed menu with slice indicator* and *hidden menu with slice indicator*, representing the transition path from novice to expert users. In the first mode, users with novice behaviour do not know the exact location of items. To simulate this behaviour, a number was presented within the centre circle to indicate which item to be selected (Figure 4); participants must navigate in the hierarchy to find and select the target. This setting was in line with (Kurtenbach, Sellen, and Buxton 1993). The second mode was similar to the first mode, except that two arms of the target slice were highlighted with purple (Figure 4) to make visually acquire the target easily. This is in accordance with the fact that experienced users know the location of the desired item and can select it quickly without much visual search effort. In the last mode, only two arms of a slice were shown to indicate the target item's position, as shown in Figure 4. Users had to rely on proprioceptive cues to accurately move their arms into the target. We would like to (1) explore the effects of proprioceptive sensation on item selection for hidden menus; (2) investigate expert users' performance without visible menu interfaces. Note in the latter two modes, ideally, participants need to select target items by retrieving command information from memory. However, in order to achieve this, participants should rehearse to assist in the development of automaticity and muscle memory (Cockburn et al. 2014). This may not be accomplished in a short amount of time for a controlled experiment like ours. Therefore, we used visual markers to aid participants to reach high levels of expert performance quickly by minimising the time of searching for target items.

The dependent variables were selection time and error rate. Selection time was defined as the duration from when users stood with arms akimbo until the selection was confirmed. An error was committed if participants selected a wrong menu item. The error rate was the ratio between the number of errors and trial numbers.

4.4. Experiment procedure

The experiment consisted of a practice phase and a test phase. During the experiment, participants stood at a distance of 1.8 m from the Kinect device (Figure 5), which

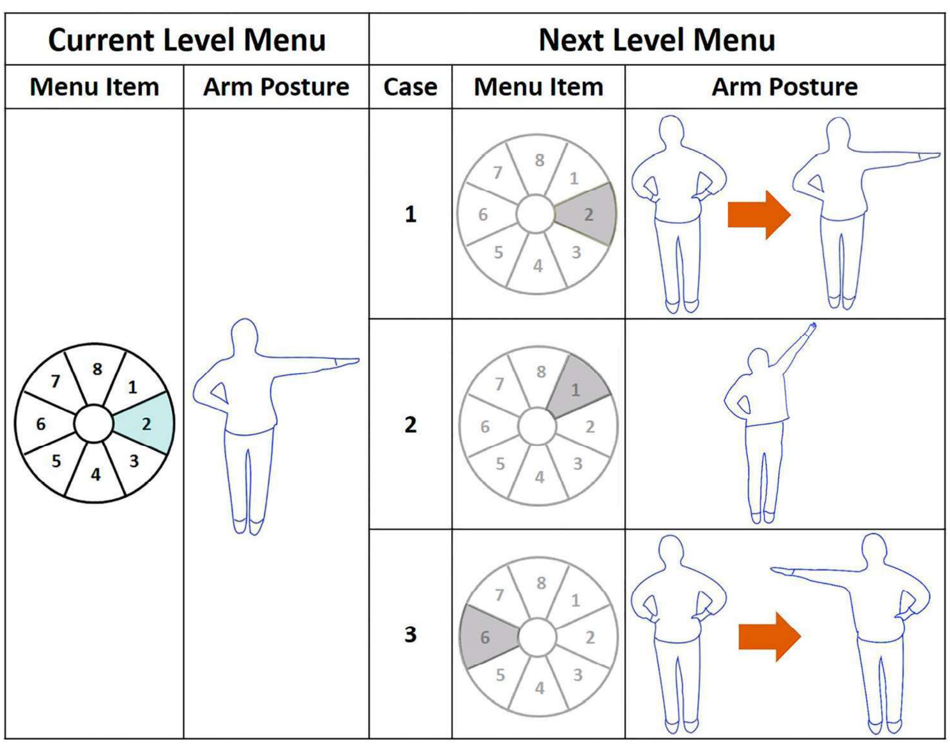


Figure 3. Three cases of submenu selection. Users have completed selection of the current level menu and start to select the next level menu.

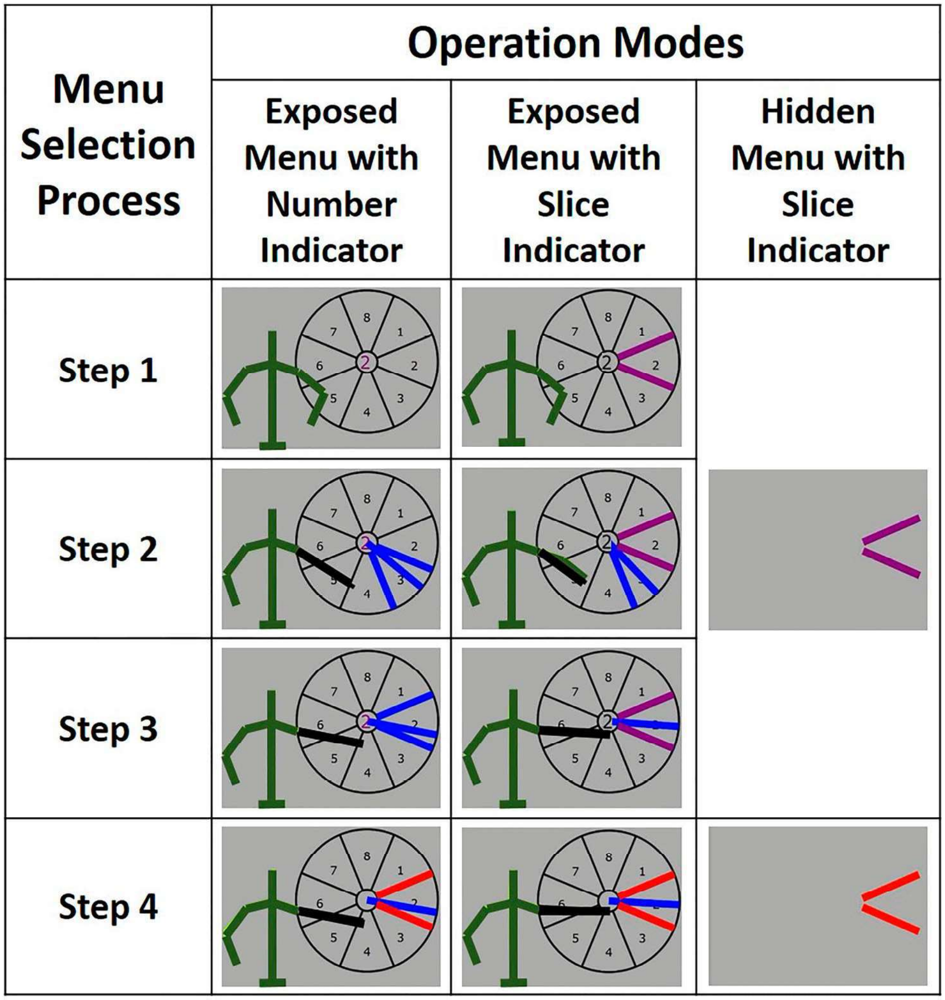


Figure 4. Experiment interfaces during menu selection process for the three operation modes. Step 1: start task; Step 2 & 3: move arm into the target slice; Step 4: confirm selection. Note for *hidden menu with slice indicator*, the interface was identical for Steps 1, 2 and 3. The green skeletons mean upper body tracked by Kinect and the black line connecting the wrist and shoulder joints indicates arm direction; they were only for illustration purposes and were not shown in the experiment process. Purple lines indicate the target item to be selected. Blue lines denote the item is under selection but not confirmed. Red lines represent the item has been selected.

was within the practical ranging limit of Kinect 2.0 (0.5–4.5 m) (Kinect 2018b). In the practice phase, the participants were instructed how to perform the task. They were asked to select 2 menu items in the six menu layouts in the three operation modes, as practice. In each trial of this practice phase, participants stood with arms akimbo to start the trial, along with the experimental interface being shown (Figure 4: step 1). Then they were required to select the target item as quickly and accurately as possible (Figure 4: step 2 and 3). Once made an error, they were asked to do the next trial. A trial ended when menu selection was confirmed (Figure 4: step 4).

In the test phase of this within-subject experiment, each participant completed three blocks of all menu items in the six menu layouts in the three operation modes. The order of operation modes was counterbalanced across participants to mitigate the effects of physical fatigue (standing and raising arms) on selection performance. For each mode, the order of six menu layouts was from simple to complex (i.e, from 4 slices to 12 slices), to allow for participants to ease gradually into the more complex layouts. And for each layout, the order of presentation of all menu items was randomised. The participants were required to rest 2 min once finishing each block. They on average took 40 min to complete the experiment (including rest). In summary, the experiment data collection consisted of (excluding practice trials): 12 participants × 3 blocks × 3 operation modes × (4 + 5 + 7 + 8 + 11 + 12) items for the 6 menu layouts = 5076 menu selection trials.

4.5. Results and analysis

We analysed experiment results using repeated measures ANOVA and *post hoc* comparisons with Bonferroni adjustment.

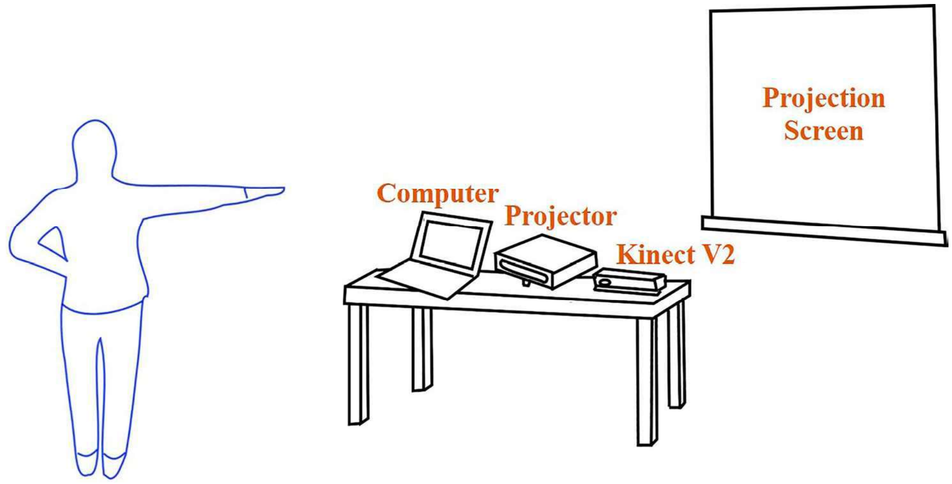


Figure 5. A participant in the experiment environment.

4.5.1. Learning effect

We first checked the learning effect on selection time over the three blocks of trials to see if the data collected had reached a level of stability. Block had a significant main effect on selection time ($F_{2,22} = 17.60$, $p < 0.01$, $\eta_p^2 = 0.62$) and on error rate ($F_{2,22} = 9.37, p < 0.01$, $\eta_p^2 = 0.46$). The first block had a significantly longer selection time and higher error rate than the second block (p<0.01 for both time and error rate) and the third block (p<0.01 for both time and error rate), but the difference was insignificant between the second and the third blocks ($p = 1.00$ for selection time, p=0.62 for error rate). Therefore, participants had reached a steady performance from the second block and we used the data of the second and third blocks for the rest of our analysis.

4.5.2. Selection time

Operation modes had a significant main effect on selection time ($F_{2,22} = 81.06, p < 0.001, \eta_p^2 = 0.88$). *Post hoc* comparisons at the 0.001 level revealed significant increases in selection time from *hidden menu with slice indicator*, to *exposed menu with slice indicator* and *exposed menu with number indicator*. The mean time for the three modes was 1025, 1125 and 1350 ms, respectively.

Menu layouts also had a significant main effect on selection time ($F_{5,55} = 79.10, p < 0.001, \eta_p^2 = 0.88$). The average time was 1081, 1078, 1148 ms, 1170 ms, 1250 ms, and 1272 ms for 4, 5, 7,8, 11 and 12 menu items respectively. *Post hoc* comparisons showed no significant differences between 4 and 5 items (group 1) (p=1.00), between 7 and 8 items (group 2) (p=0.78), and between 11 and 12 items (group 3) (p=0.254). But there were significant differences among the three groups (p<0.01).

There was a significant interaction effect between operation modes and menu layouts on selection time ($F_{10,110} = 9.34, p < 0.001, \eta_p^2 = 0.46$). As illustrated in Figure 6(a), as number of menu items increased, the time differences between *exposed menu with number indicator* and *hidden menu with slice indicator*, and between *exposed menu with number indicator* and *exposed menu with slice indicator* also increased. Menu items had a more significant impact on the number-indicator mode than the two slice-indicator modes, as the former mode required longer time to search and choose visually among alternatives.

4.5.3. Error rate

A significant main effect was found on error rate for operation modes ($F_{2,22} = 5.96, p < 0.01, \eta_p^2 = 0.35$). *Post hoc* comparisons indicated that *hidden menu with slice indicator* (M=0.051) had higher error rate than *exposed menu with slice indicator* (M=0.016) (p<0.05). But there were no significant differences between *exposed menu with slice indicator* and *exposed menu with number indicator* (M=0.022) (p=1.00), and also between *hidden menu with slice indicator* and *exposed menu with number indicator* (p=0.16).

There was a significant main effect for menu layout ($F_{5,55} = 16.18, p < 0.01, \eta_p^2 = 0.60$). Error rate increased as a function of number of items per menu. The average error rate was 0, 0.003, 0.012, 0.016, 0.071 and 0.078 for menus having 4, 5, 7, 8, 11, 12 items, respectively. *Post hoc* comparisons showed there were no significant differences among menus having 4, 5, 7 and 8 items, and also between menus having 11 and 12 items. However, significant differences were found between each layout in the former group (4, 5, 7 and 8 items) and in the latter group (11, 12 items) ($p < 0.05$).

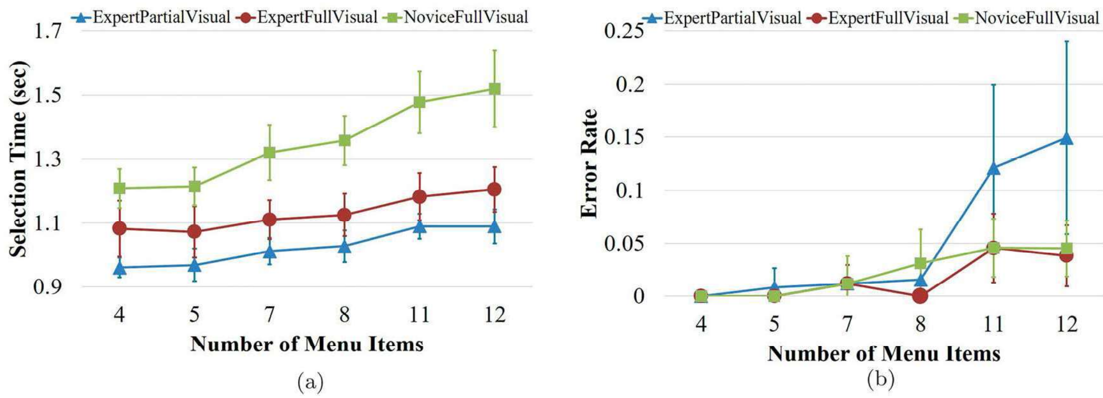


Figure 6. (a) *Selection time* and *Error rate* for each operation mode in six menu layouts. Error bars represent 0.95 confidence interval.

There was a significant interaction effect between operation modes and menu layouts on error rate ($F_{10,110} = 4.02, p < 0.001, \eta_p^2 = 0.27$). As shown in Figure 6(b), error rates for the three operation modes were similar if number of items was up to 8. If the number exceeded 8, *hidden menu with slice indicator* had significantly higher error rates than the other two (p<0.05).

4.6. Discussion

In this section, we discuss the experimental results around the questions raised before the experiment.

Q1. How does the number of menu items affect user performance?

A1. The number of menu items affected user performance in terms of selection time and error rate. Generally, selection time and error rate increased as item number increased; users tended to spend more efforts in articulating the action as menu items become smaller. However, performance on 5-item, 8-item, or 12-item menus was not significantly worse than performance on menus with one less item in both selection time and accuracy. In addition, menus having the number of items up to 8 had similar selection accuracy.

Q2. What is the performance of using ArmMenu for novice and experienced users?

A2. As expected, *exposed menu with number indicator* required novice users to visually search for item, so producing the longest selection time among the three modes. For experienced users, *hidden menu with slice indicator* led to shorter time but higher error rate than *exposed menu with slice indicator*. In the former mode, users can be more focused on the task as they can only view the target item, hence achieving a faster speed.

Q3. Could ArmMenu work well with hidden menu interfaces?

A3. *Hidden menu with slice indicator* resulted in the shortest selection time among the three operation modes, and a comparable accuracy with *exposed menu with number indicator*. This indicates proprioceptive sensation could help users select menu items accurately and quickly even limited visual feedback was provided. In such case, moving arm to a specific direction could rely only on human proprioception, which is highly practicable as shown by our experiment results. In addition, lateral arm movements is coupled to ArmMenu's interface well; both are circles with evenly divided slices, hence arm direction can directly specify which item is under selection. This may also enhance ArmMenu's performance.

5. Experiment 2: comparing ArmMenu with marking menus

Results of Experiment 1 have offered insights to ArmMenu design. In Experiment 2, we further investigated the performance of ArmMenu in comparison with touchless marking menus. Both are visually guided menu techniques with arm gesture interaction for item selection. In addition, touchless marking menus are a representative command input technique on remote displays and their efficacy has been demonstrated by previous studies (Bailly et al. 2011; Bossavit et al. 2014; Lenman, Bretzner, and Thuresson 2002). It is therefore meaningful to examine if our technique could outperform marking menus. We did not compare ArmMenu with the 'point-and-dwell' technique in Kinect, as the latter technique requires users to hold their arm still for 2 s (Yoo et al. 2015), which is longer than average menu selection time with ArmMenu (see results in Experiment 1). The following questions were the main interests of this experiment.

Q4. Would ArmMenu have an overall better performance than marking menus? Interaction gestures for ArmMenu were designed based on human proprioceptive sensations. We expected with such sensations, ArmMenu could outperform marking menus.

Q5. How would numbers of menu levels and number of items per level affect the performance of ArmMenu and marking menus? These are two important factors in menu interface design, and therefore should be considered when comparing ArmMenu and marking menus.

5.1. Apparatus

The equipment in this experiment was the same as in Experiment 1.

5.2. Participants

Twelve right-handed participants (6 male and 6 female) from the local university took part in the experiment. They were not involved in Experiment 1. They were 22.4 years old (SD = 3.4 years) and 167.6 cm tall (SD = 12.3 cm) on average. None had prior experience in using Kinect.

5.3. Experiment design

The experiment adopted a within-subject design. There were two independent variables and four dependent variables.

The independent variables were menu types and menu layouts. Menu types were ArmMenu and touchless marking menus. For both menu types, experiment tasks started as menu interfaces appeared on the screen. Experiment process with ArmMenu can be found in Figure 4: 'Exposed Menu with Slice Indicator'. Menu interfaces were presented in such mode to minimise participants' visual search cost. We designed touchless marking menus based on (Bossavit et al. 2014). The interface of the touchless marking menu was the same as ArmMenu and was located in the centre of the screen (Figure 7). Once task started, participants raised their dominant hand to the centre of the marking menu. To select an item, participants should control their hand to move the on-screen cursor towards the item's direction and also make the on-screen cursor exceed the scope of the inner circle (its diameter was 15 cm).

Menu layouts had two factors: number of levels and number of items per level. There were two kinds of levels in the hierarchy: 1 level and 2 levels. Each menu in the hierarchy had two kinds of number of items per level: 4 items and 8 items. The combinations of the two factors forms four layouts denoted as compass4-1, compass4-2, compass8-1 and compass8-2, where the first digit in these acronyms represents the number of items per level, and the second digit refers to the number of levels. For example, the compass8-2 menu has 2 levels in the hierarchy with each level having 8 items. All menus were shown in the same position on the screen with black interfaces and grey interfaces indicating the first and the second level respectively. For ArmMenu, the selection process of 2 level hierarchy can be found in Figure 3. For hierarchical marking menus, submenu selection followed step 1 and 2 in Figure 7; this is a variant of the 'inflection-free simple marks' selection method in (Zhao and Balakrishnan 2004).

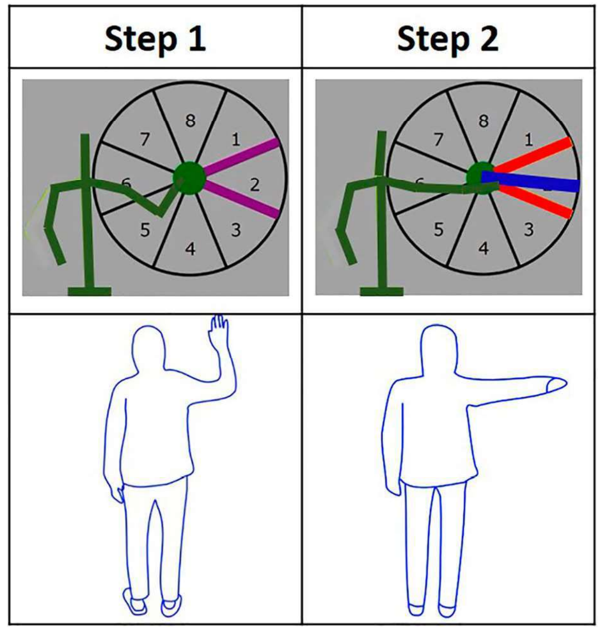


Figure 7. Experiment interfaces during selection process for marking menus. Step 1: start task and move arm into the centre; Step 2: move the on-screen cursor to exceed the scope of the inner circle for confirming selection. The skeletons represent upper body tracked by Kinect. Note they were only for illustration purposes and not shown on the experiment interfaces. In Step 1, the target item was to be selected. In Step 2, the item has been selected.

We carefully considered the following points when designing the experiment. First, we restricted our focus to a maximum of 2 levels because menu hierarchy having over 2 levels are complex on Kinect-based interactive remote displays. In addition, previous study on pen-based marking menus has showed selection becomes error-prone for menus at depths greater than 2 (Kurtenbach and Buxton 1993). Second, we selected menu with 4 and 8 items for two reasons: the two menu layouts were commonly used in marking menu studies, e.g. (Zhao and Balakrishnan 2004), and also 4-item and 8-item menus are representative layouts of ArmMenu according to the results of Experiment 1 that user performance was best for 4-item menus and degraded rapidly for menus with more than 8 items.

The dependent variables were selection time, error rate, selection distributions and subjective evaluation. Selection time and error rate are two fundamental metrics to measure the performance of a technique. For both ArmMenu and marking menus, selection time was defined as the duration from when the menu interface appeared until the selection was confirmed. An error was committed if participants selected a wrong menu item. The error rate was the ratio between the number of errors and trial numbers. In addition, we used selection distributions to measure how well participants controlled their arm or hand to select a menu item. For ArmMenu, selection distributions mean 95% range of arm angle ($\angle A'O'C'$ in Figure 2(c)) when confirming selection for each menu slice. And for marking menus, selection distributions represent 95% range of the intersection points when the on-screen cursor exceeds the scope of the inner circle for each menu slice. To enable a fair comparison of selection distributions between ArmMenu and marking menus, we mapped the point-position distributions of marking menus to angular distributions by calculating the angle of an intersection point in polar coordinates (i.e. the angle formed by the horizontal line and the line from the centre of the marking menu to the intersection point). Generally, the narrower a distribution range is, the more easily a technique could be operated.

5.4. Experiment procedure

The experiment consisted of a practice phase and a test phase. Same as in Experiment 1, participants stood at a distance of 1.8 m from the Kinect device when performing tasks. In the practice phase, after given instructions, participants were asked to select 2 menu items for each of the four menu layouts and each of the two menu types. In each trial of this practice phase, task started with the experimental interface being shown. Participants were instructed to select the target item as quickly and accurately as possible. A trial ended when menu selection was confirmed. If an error was made, participants were asked to do the next trial.

In the test phase, each participant completed three blocks of all menu items in the four menu layouts in the two menu types. The order of the menu types was counterbalanced across participants. For each menu type, the order of the four menu layouts was from simple to complex (i.e. compass4-1, compass4-2, compass8-1 and compass8-2), to allow for participants to ease gradually into the more complex layouts. And for each layout, the order of presentation of all menu items was randomised. A 2-minute rest was required after finishing each block. We also adopted a think-aloud protocol to record participants' comments. In summary, the experiment data collection consisted of (excluding practice trials): 12 participants × 3 blocks × 2 operation modes × (4 + 8 + 16 + 64) items for the four menu layouts = 6624 menu selection trials.

At the end of the experiment, each participant was asked to fill in a questionnaire to rate ArmMenu and marking menus on 5-point Likert Scales regarding three constructs: 'perceived usefulness', 'perceived usability' and 'physical demand' (5 for strongly agree, and 1 for strongly disagree). 'Perceived usefulness' and 'perceived usability' were proposed based on the technology acceptance model. The former construct means the degree to which a participant believed that using ArmMenu would enhance command input performance for remote display interaction, and the latter construct means the degree to which a participant believed that using ArmMenu would be free from effort. For 'physical demand', we focused on the assessment of arm fatigue when interacting with ArmMenu or marking menus. Each construct had three questions (see Appendix). Participants on average took 55 min to complete the entire experiment (including rest).

5.5. Results and analysis

We analysed experiment data using repeated measures ANOVA and *post hoc* comparisons with Bonferroni adjustment.

5.5.1. Learning effect

We first checked the learning effect on selection time over the three blocks of trials for ArmMenu and marking menus respectively. There was no significant main effect on error rate for blocks for both ArmMenu ($F_{2,22} = 2.09, p = 0.15, \eta_p^2 = 0.16$) and marking menus ($F_{2,22} = 1.25, p = 0.31, \eta_p^2 = 0.10$). However, Blocks had a significant main effect on selection time for both ArmMenu ($F_{2,22} = 11.60, p < 0.001, \eta_p^2 = 0.51$) and marking menus ($F_{2,22} = 20.43, p < 0.001, \eta_p^2 = 0.65$). For both menu types, the first block resulted in a significantly longer time than the second (p<0.05) and third blocks (p<0.01). And no significant difference was found between the last two block (p=0.48 for ArmMenu and 0.09 for marking menus). Therefore, participants performed at expert levels from the second blocks and the data of the second and third blocks were used in the rest of the analysis.

5.5.2. Selection time

The main effect of menu types on selection time was significant such that ArmMenu ($M = 1831$ ms) had a shorter time than marking menus (M=1959 ms)

($F_{1,11} = 8.32, p < 0.05, \eta_p^2 = 0.39$). Number of menu levels also had a significant main effect on selection time ($F_{1,11} = 937.56, p < 0.001, \eta_p^2 = 0.99$). Unsurprisingly, 2-level menus (2353 ms) had longer average time than 1-level menus (1437 ms). However, no significant main effect was found on selection time for number of menu items per level ($F_{1,11} = 2.13, p = 0.36, \eta_p^2 = 0.102$). The average time was 1879 ms for 4-item menus and 1911 ms for 8-item menus.

There was no interaction effect of menu types × number of menu levels on selection time ($F_{1,11} = 4.12, p = 0.13, \eta_p^2 = 0.22$) (Figure 8(a)) and of menu types × number of menu items per level ($F_{1,11} = 0.16, p = 0.82, \eta_p^2 = 0.01$) (Figure 8(b)). Therefore, ArmMenu generally outperformed marking menus across the two types of menu levels and the two types of menu item numbers.

We are interested in the effects of item positions (8 positions, from item 1 to 8) on time performance in 8-item layouts for ArmMenu. For 1-level menus, item positions did not have a main effect on time ($F_{7,77} = 8.19, p = 0.12, \eta_p^2 = 0.24$). However, for 2-level menus, item positions at the top-level menu had a main effect on time ($F_{7,77} = 29.13, p < 0.001, \eta_p^2 = 0.87$). The 8 item positions can be divided into 3 groups according to selection time performance: group 1 (item 2, 3, 5 and 6), group 2 (item 1, 4 and 7) and group 3 (item 8). Within each group, there was no significant difference between each two item positions in selection time (all p>0.05); but among the three groups, the time significantly increased from item positions in group 1 to 3 (all p<0.05). Results vary between 1-level and 2-level menus, mainly because menu selection methods are different for single-level menus and hierarchical menus (see section 'ArmMenu design').

5.5.3. Error rate

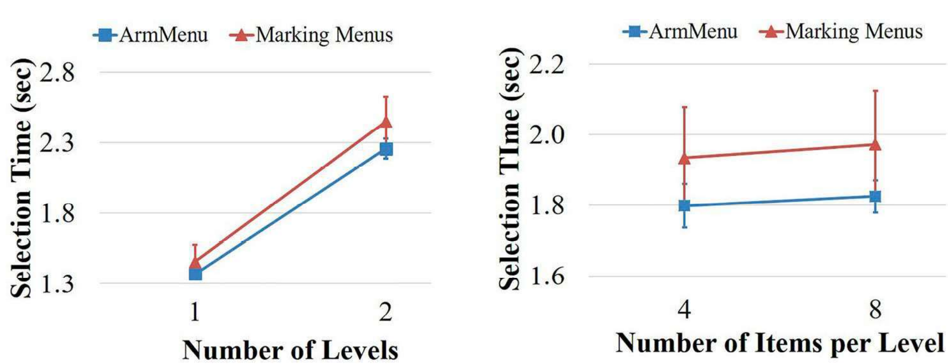
No significant main effect was found for menu types on error rate ($F_{1,11} = 2.86, p = 0.13, \eta_p^2 = 0.20$). The average error rate was 0.035 for ArmMenu and 0.056 for marking menus. A significant main effect was found for number of menu levels ($F_{1,11} = 7.12, p < 0.05, \eta_p^2 = 0.38$) and number of menu items ($F_{1,11} = 5.89, p < 0.05, \eta_p^2 = 0.33$). As expected, 2-level menus (M=0.08) resulted in higher error rate than 1-level menus (M=0.029). 8-item menus (M=0.052) had higher error rate than 4-item menus (M=0.039).

Although there was no interaction effect on error rate for menu types × number of menu levels ($F_{1,11} = 2.16, p = 0.31, \eta_p^2 = 0.11$) (Figure 9(a)), there was an interaction effect on error rate for menu types × number of menu items ($F_{1,11} = 5.71, p < 0.05, \eta_p^2 = 0.35$) (Figure 9(b)). Hence, we further calculated the simple main effects of menu types on error rate for the 4-item and 8-item menus respectively. For 4-item menus, menu types had no significant main effect on error rate ($F_{1,11} = 0.01, p = 0.96, \eta_p^2 = 0.01$), with average error rate of 0.037 for ArmMenu and 0.035 for marking menus. However, for 8-item menus, menu types had a significant main effect on error rate ($F_{1,11} = 25.34, ; p < 0.01, \eta_p^2 = 0.70$), with average error rate of 0.031 for ArmMenu and 0.072 for marking menus. ArmMenu was more accurate than marking menus when they contained 8 items.

We examined the effects of item positions (from item 1 to 8) on error rate in 8-item layouts for ArmMenu. Item positions did not have a main effect on time for both 1-level ($F_{7,77} = 0.56, p = 0.71, \eta_p^2 = 0.12$) and 2-level menus ($F_{7,77} = 1.79, p = 0.37, \eta_p^2 = 0.23$).

5.5.4. Selection distributions

There was a significant main effect for menu types on selection distributions ($F_{1,11} = 6.12, p < 0.01,$



(a) (b)

Figure 8. *Selection time* for each menu type in two kinds of (a) number of levels and (b) number of items per level. Error bars represent 0.95 confidence interval.

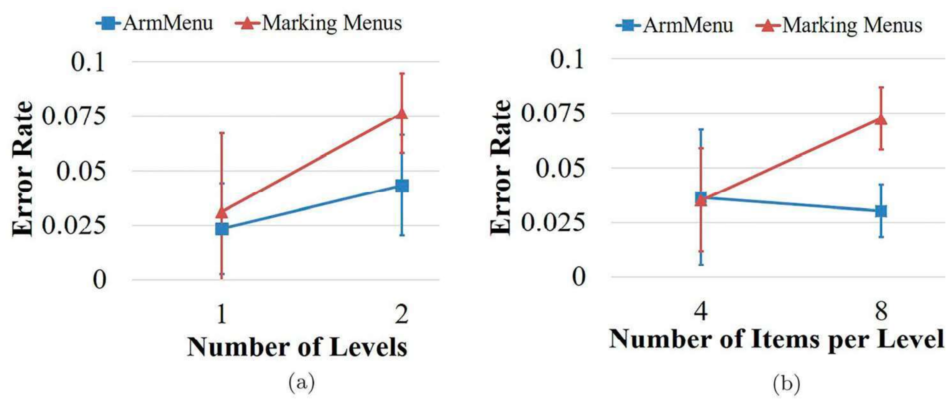


Figure 9. *Error rate* for each menu type in two (a) numbers of levels and (b) numbers of items per level. Error bars represent 0.95 confidence interval.

$\eta_p^2 = 0.76$). The mean angular range was 22.9° for ArmMenu and 40.9° for marking menus. We further calculated the simple main effects of menu types on error rate for the 4-item and 8-item menus respectively. For both 4-item and 8-item menus, menu types had a significant main effect on selection distributions ($F_{1,11} = 5.75$, $p < 0.01$, $\eta_p^2 = 0.81$ for 4-item menus, and $F_{1,11} = 6.89$, $p < 0.01$, $\eta_p^2 = 0.83$ for 8-item menus). Figure 10 illustrates the distributions of both menu types for 4-item and 8-item cases. Generally, ArmMenu had significant narrower distributions than marking menus, indicating participants were able to control their arm and hand for menu selection better with ArmMenu.

5.5.5. Subjective feedback evaluation

Mann-Whitney U test revealed that ArmMenu was rated significantly higher than marking menus in 'perceived usefulness' ($U = 11.14; p < 0.05$, $\eta_p^2 = 0.62$) (ArmMenu vs. marking menus: 4.45 vs. 4.08), 'perceived usability' ($U = 7.85; p < 0.01$, $\eta_p^2 = 0.87$) (4.68 vs. 4.08) and 'physical demand' ($U = 2.56; p < 0.01$, $\eta_p^2 = 0.82$) (4.28 vs. 3.75). Overall, participants had a higher preference for command selection with ArmMenu.

5.6. Discussion

In this section, we discuss the experimental results around the questions listed at the beginning of this experiment.

Q4. Would ArmMenu have an overall better performance than marking menus in terms of selection time and error rate?

A4. While ArmMenu and marking menus had comparable accuracy in general, ArmMenu resulted in significantly average shorter time and narrower selection distributions. In addition, ArmMenu was preferable by participants. They reported that ArmMenu was more controllable than marking menus, hence producing faster speed and higher accuracy.

The above analysis indicates the advantages of ArmMenu. However, one may still doubt if marking menu design in our study would bias the conclusion. We therefore review previous studies about touchless marking menus to compare their performance with ours. In study Kulshreshth and LaViola (2014), the mean selection time was 2.09 s for making menus in 1 level with 5 items. Also in study Bailly et al. (2011), the average time was 5.8 s for marking menus with 5 × 5 hierarchy. Compared to these studies, marking menus in our study achieved faster average speed with 1.47 s for 1-level menus and 2.58 s for 2-level menus, while having similar accuracy (mean ≈ 95%). These differences are due to many reasons like varied experiment conditions, and exploring these reasons is not the focus of our study. We would like to verify that marking menus in our studies were reasonably designed and had comparable, if not better performance than previous studies, so should not bias the conclusion.

Q5. How would number of menu levels and number of items per level affect the performance of ArmMenu and marking menus?

A5. ArmMenu resulted in significantly shorter time than marking menus across the two types of menu levels and the two types of menu item numbers. While having comparable accuracy in general, ArmMenu tended to be more accurate for 8-item menus. For marking menus, the difficulty of controlling the on-screen hand cursor to select menus rose as item number increased. However, ArmMenu seems to be immune to this problem because no significant difference was found in error rate between 4-item and 8-item menus ($F_{1,11} = 0.23, p = 0.64$, $\eta_p^2 = 0.42$). Compared to moving a hand in the air along in a direction dictated by a menu item, rotating an arm to a certain degree is easier to execute. Note we only studies menus up to 2 levels. The conclusion should

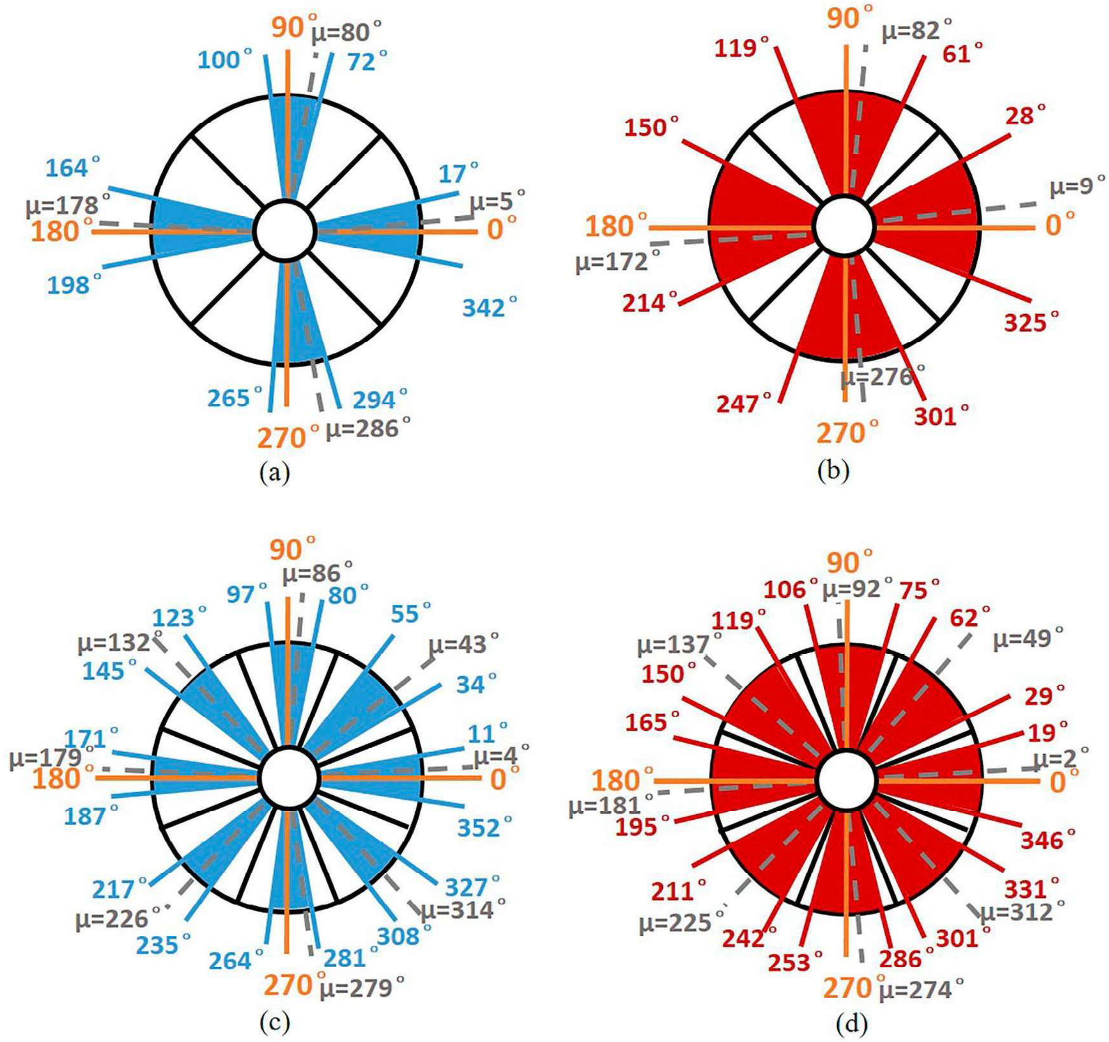


Figure 10. Selection distributions of (a) 4-item for ArmMenu; (b) 4-item for marking menus; (c) 8-item for ArmMenu; (d) 8-item for marking menus.

hold true for menus having more than 2 levels according to the analysis of the interaction effects.

6. General discussion

6.1. Advantages of ArmMenu for command input

In this study, we proposed ArmMenu, a menu technique for free-hand interaction with remote displays. Here we discuss the advantages of ArmMenu with the results of Experiment 1 and 2.

First, ArmMenu is effective for command input on distance displays. Results in Experiment 2 demonstrate ArmMenu outperformed marking menus in speed, accuracy and subjective preference. Such superior performance of ArmMenu may be because it is a visually-guided command input technique which combines pie menus' interfaces and proprioception-based arm gestures. Unlike cursor-based methods requiring accurate positioning of the hand cursor on screen, ArmMenu relies on lateral movements of arms. Users perform such body gestures based on proprioceptive senses, so can dedicate major efforts to the selection task itself rather than articulating gestures. This facilitates menu selection. In addition, there is no need to memorise these arm gestures, as menu layout provides visual cues of how to hold arms in a specific direction to invoke a command. Therefore, all interactions are immediately contextualised, permitting a large number of possible interactions at once. This is a significant difference from common gesture-based command input methods for which gesture memorisation is dispensable.

Second, ArmMenu is feasible with hidden menu interfaces. Results in Experiment 1 demonstrate that users can select menus significantly faster with *hidden menu with slice indicator* than with *exposed menu with number* *indicator*, while committing similar selection errors. A main reason for these results is lateral arm movements is coupled to ArmMenu's interface well; both are circular areas, hence arm direction can directly indicate which item is under selection (Figure 1). When visual feedback is limited, sensing and controlling arm direction can rely on human proprioception, which is highly practicable according to our experiment results. Note in Experiment 1, the target item was not completely hidden; its arms were marked by purple to indicate its position. For our experiment design, this is a 'shortcut' to mimic the situation where experienced users know the exact position of the target item. Hence, for cases where visual feedback is not present, experienced users may use ArmMenu as efficiently as shown in our experiment.

Third, ArmMenu is easy to implement with off-the-shelf APIs provided by the Kinect V2 SDK. A key step of implementing ArmMenu is to calculate arm direction. To do this, we used on-screen coordinates of shoulder and wrist joints tracked by the Kinect device, which is simple to complete. We did not evaluate arm posture tracking accuracy as done in the study (Wasenmüller and Stricker 2017), which is beyond the scope of our study. However, from the low selection error rate in Experiment 1 (mean value was 0.016 for *exposed menu with slice indicator* and 0.022 for *exposed menu with number indicator*), it is reasonable to conclude the tracking accuracy is satisfactory with Kinect V2 devices for ArmMenu interaction.

Last, ArmMenu has arm fatigue advantages over touchless marking menus according to the subjective feedback in Experiment 2. Two reasons may be attributed to this result. First, marking menus require users to perform arm movements to control the on-screen cursor in a high level of accuracy. This causes arm tiredness, as reported in (Lenman, Bretzner, and Thuresson 2002). Unlike marking menus, menu selection with ArmMenu depends on lateral arm rotation, which should be easier to execute. Second, menu selection tasks can be assigned to two arms. This may alleviate arm fatigue in comparison to touchless marking menus for which only one arm is used to select items.

Overall, ArmMenu is a viable option for command input tasks on remote displays with advantages such as fast selection speed, high accuracy and easy to develop.

6.2. Differences between ArmMenu and existing arm-based menu techniques

In this section, we review four typical arm-based menu techniques and discuss the differences between ArmMenu and them, so as to gain a deeper understanding of the characteristics of ArmMenu. Table 1 summarises the main characteristics of each arm-based menu technique.

(1) Touchless Pointing (Chertoff, Byers, and LaViola 2009; Kinect 2018a; Sambrooks and Wilkinson 2013; Vogel and Balakrishnan 2005; Yoo et al. 2015): such technique usually uses the hand to control an on-screen cursor with a mapping of hand position to cursor position, and also confirms menu selection with hand gestures (e.g. hover-to-select or press-to-select). However, holding the hand in the air to control the cursor tends to result in a tired and slow interaction process (Sambrooks and Wilkinson 2013) and does not rely on human proprioceptive sensations. For ArmMenu, menu selection is performed by proprioception based lateral arm movements. Results indicate that such design could improve interaction efficiency and alleviate arm fatigue.

(2) Touchless marking menus (Bailly et al. 2011; Bossavit et al. 2014; Lenman, Bretzner, and Thuresson 2002) and their variants (e.g. Chattopadhyay and Bolchini 2014; Ren and O'Neill 2012): they belong to the category of visual-guided gestural techniques for command input. A major limitation is that they require hand movements in a specified direction with a high level of accuracy and such movements do not make use of human proprioceptive sensations. Inspired by touchless marking menus, we designed ArmMenu with their limitations and strengths in mind. Results show ArmMenu was faster and more preferable by users than marking menus while having similar selection accuracy.

(3) Finger-Count Menus (Kulshreshth and LaViola 2014): such technique supports mapping menu items to finger counting gestures. Users may find difficulty to identify the mapping relationship due to lacking inherent correlations. ArmMenu mitigates such limitation by establishing an 'intuitive' mapping between lateral arm movements and ArmMenu's circular interface.

Table 1. Differences between ArmMenu and other menu techniques.

	Arm-based menu techniques				
	Hover-to-Select/Press-to-Select	Touchless Marking Menu	Finger-Count Menu	BodyMenu	ArmMenu
Visually guided	Yes	Yes	No	Yes	Yes
Proprioception based	No	No	No	Yes	Yes
Menu interface	Vertical/horizontal	Radial	Vertical/radial	Items attached to body	Radial
Interaction mechanism	Using hand to control on-screen cursor	Drawing stroke with hand	Counting with fingers	Touching specific body positions	Rotating arm

(4) BodyMenu (Bossavit et al. 2014): menu selection can be accomplished by using hands to reaching body parts to which virtual menu items are attached. Such technique was designed to take advantage of human proprioceptive sensations. Unlike BodyMenu, ArmMenu has a circular interface like pie menus and menu selection is performed by proprioception based lateral arm movements.

6.3. Design implications of ArmMenu

In this section, we discuss the following two aspects related to the application of ArmMenu to interaction design with remote displays.

6.3.1. Interaction flow for ArmMenu

In practice, ArmMenu interaction could follow an activation-selection-deactivation process. Users can activate ArmMenu by holding the arm-akimbo posture, and deactivate it when not holding the posture. This is a very useful feature of ArmMenu: the menu interface does not need to be always rendered on the display and thus could avoid occluding other interface elements. The selection method is designed using lateral arm movement gestures based on human proprioceptive sensations, hence could offer a fast and accurate performance. The details of selection process can be found in the Section '3. ArmMenu Design'.

The transition path from novice to expert can be as follows. To start ArmMenu, users hold arms akimbo. If the time of holding this posture exceeds a threshold,[2] the menu interface appears to facilitate novice users to select menu items. Otherwise, users can select the desired item without the need of waiting for the pop-up menu if they have already memorised the layout (as they become expert). This transition mechanism is similar to the dwell timeout method used in marking menu design (Kurtenbach, Sellen, and Buxton 1993).

The current ArmMenu design adopts a delay method to confirm menu item selection. While this method worked well for our ArmMenu design in the lab environment, it is still meaningful to consider other mechanisms of confirming selection based on practical need, such as using a more reliable hand posture to trigger the selection instead of (or in combination of) using a delay to reduce the accidental selection for a critical menu selection. This is worthy of exploring in future.

6.3.2. Interface design for ArmMenu

Results from Experiment 1 and 2 provide implications for interface design of ArmMenu.

An important factor for ArmMenu's interface design is determining how many menu items the interface should contain. Results in Experiment 1 reveal that for 8-item menus, user performance in selection speed did not decline significantly compared to 7-item menus. In addition, selection accuracy was similar if item numbers are up to 8. Therefore, menus having 8 items can enable users to select items with high efficiency.

Another factor is determining how to arrange items so that users can access frequently-used items quickly and reliably. Taking 8-item menus as an example (Figure 3). For single level menus, items can be positioned in any slices as item position analysis in Experiment 2 indicates user performance was similar in time and accuracy across the eight slices. Item arrangement for multi-level menus is different. For the top level menu, positioning an item in menu layout should consider its effects on selection performance in this level and subsequent levels. According to the results of Experiment 2, item 2, 3, 5 or 6 should be most frequently-used menu options, then item 1, 4 or 7 should be less frequently-used options, and so on. For submenus, most frequently-used menu options should be near the position of the selected item in the previous menu level. For example, for 8-item menus, if item 2 is the previous selected slice, most frequently-used options in current level should be item 1 or 3 (Figure 3).

6.4. Beyond remote display interaction

While ArmMenu was initially proposed for command input on remote displays, its application scenarios can be extended to wearable interaction and ubiquitous interaction as well. For example, based on studies of tracking 3D arm postures using only one smartwatch (e.g. Shen, Wang, and Roy Choudhury 2016), variants of ArmMenu could be applied to command input for smartwatch interaction, which would offer benefits to smartwatch users when it comes to input commands on small screens of such devices. Taking ubiquitous environments as another example. Users can operate a remote device with ArmMenu if a body tracking system is available. Even without implicit visual appearance, ArmMenu could work well according to the results of Experiment 1. Hence it may serve as a promising technique in ubiquitous computing environments.

6.5. Multi-user capacity with ArmMenu

ArmMenu is suited for interaction with shared displays involving multiple users. Each user could operate a ArmMenu specifically assigned to him/her. However, for colocated multiuser interaction with ArmMenu, there are possibilities of posing visual and physical obstructions to others when a user stretches out his/her arm to select a menu item. To mitigate this issue, ArmMenu interface could provide signs to ask the user to move away from others (or move forward/backward) if they stand too close (this can be achieved by calculating users' distance and user's arm through motion sensors).

6.6. Mitigating hand fatigue

According to participants' feedback and our observation, arm fatigue of using ArmMenu may be attributed by two factors. One is that participants had to move their arm quickly to the target item, and the other is that they needed to control their arm accurately within the target item to confirm selection.

We observed that participants' common coping strategy for mitigating arm fatigue was they usually targeted the menu item and stretched out the arm straight to the target (instead of rotating their arm from the side of their body to the menu item, in which the arm would travel longer distance).

A potential design to mitigate arm fatigue is to predict the final item which the arm would move to based on arm kinematics in advance of confirming selection (e.g. using kinematic template matching proposed in Pasqual and Wobbrock 2014), perhaps considerably, with techniques such as target item expansion.

6.7. Limitations and future work

While our study systematically evaluated ArmMenu's performance, there are some open questions for further investigation.

First, participants in our study were right-handed. It is of interest to evaluate ArmMenu's performance with left-handed users, although handedness should not affect our findings given the structures of left and right arms are symmetrical.

Second, ArmMenu is operated by lateral rotation movements of arms. We adopted such interaction style to match ArmMenu's circular interface and also to achieve good arm-tracking effects by Kinect (arm movement space is roughly parallel to Kinect sensors and does not overlap body). Given our arms can produce many types of movements, for example moving the forearm upward at the elbow, we would further explore these movement types to enrich interaction styles of ArmMenu.

Third, our study assessed arm fatigue based on based on Likert rating. Results have gained a good understanding of arm fatigue when using ArmMenu for command input on remote displays. To further quantitatively characterise arm fatigue, future work would like to use other approaches include obtrusive measurements of bodily variables (e.g. heart-rate, oxygen level or EMG).

7. Conclusion

In this study, we designed, implemented and evaluated ArmMenu, a touchless menu selection technique for remote interfaces. This technique is easy to implement with Kinect, an off-the-shelf motion sensing device. Results in Experiment 1 indicate that users can achieve fast and accurate performance even with limited visual feedback. Also, user performance degraded rapidly if the number of menu item exceeded eight. In Experiment Two, we validated ArmMenu's efficiency in comparison to marking menus. Having comparable accuracy, ArmMenu was faster and more preferable by users. Overall, ArmMenu is an effective command input technique and can be available for current commercial body sensing devices.

Notes

1. Arm akimbo means hand on hips and elbow projecting outwards.
2. According to the time analysis in Experiment 1, participants on average took 350 ms to complete Step 1 (start task) shown in Figure 4. Hence the threshold should be larger than 350 ms.

Acknowledgments

This work is supported by the National Key R&D Program of China (No. 2016YFB1001405); Key Research Program of Frontier Sciences, CAS (No. QYZDY-SSW-JSC041); Foundation of Jiangsu Province Science & Technology Department (No. BE2017104); Natural Science Foundation of China (No. 61602236); Natural Science Foundation of Jiangsu Province (No. BK20160801); China Postdoctoral Science Foundation (No. 2016M591843); and Postdoctoral Science Foundation of Jiangsu Province (No. 1501053B).

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work is supported by the National Key Research and Development Project of China [grant number 2016YFB1001405]; Key Research Program of Frontier Sciences, Chinese Academy Sciences (CAS) [grant number QYZDY-SSW-JSC041]; Foundation of Jiangsu Province Science & Technology Department [grant number BE2017104]; Natural Science Foundation of China [grant number 61602236]; Natural Science Foundation of Jiangsu Province [grant number BK20160801]; China Postdoctoral Science Foundation [grant number 2016M591843]; and Postdoctoral Science Foundation of Jiangsu Province [grant number 1501053B].

References

Alt, F., S. Geiger, and W. Höhl. 2018. "ShapelineGuide: Teaching Mid-Air Gestures for Large Interactive Displays." In *Proceedings of the 7th ACM International Symposium on Pervasive Displays*, PerDis '18, Munich, Germany, 3:1–3:8. New York, NY: ACM.

Bailly, G. et al. 2011. "Comparing Free Hand Menu Techniques for Distant Displays Using Linear, Marking and Finger-Count Menus." In *Proceedings of the 13th IFIP TC 13 International Conference on Human-computer Interaction - Volume Part II*, INTERACT'11, Lisbon, Portugal, 248–262. Berlin: Springer-Verlag.

Bossavit, B., A. Marzo, O. Ardaiz, and A. Pina. 2014. "Hierarchical Menu Selection with a Body-Centered Remote Interface." *Interacting with Computers* 26 (5): 389–402.

Callahan, J. et al. 1988. "An Empirical Comparison of Pie vs. Linear Menus." In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '88, Washington, DC, USA, 95–100. New York, NY: ACM.

Carter, M. et al. 2016. "PathSync: Multi-User Gestural Interaction with Touchless Rhythmic Path Mimicry." In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, Santa Clara, CA, USA. New York, NY: ACM.

Chattopadhyay, D., and D. Bolchini. 2014. "Touchless Circular Menus: Toward an Intuitive UI for Touchless Interactions with Large Displays." In *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces*, AVI '14, Como, Italy, 33–40. New York, NY: ACM.

Chattopadhyay, D., and D. Bolchini. 2015. "Motor-intuitive Interactions Based on Image Schemas: Aligning Touchless Interaction Primitives with Human Sensorimotor Abilities." *Interacting with Computers* 27 (3): 327–343.

Chertoff, D. B., R. W. Byers, and J.J. LaViola Jr. 2009. "An Exploration of Menu Techniques Using a 3D Game Input Device." In *Proceedings of the 4th International Conference on Foundations of Digital Games*, FDG '09, Orlando, FL, 256–262. New York, NY: ACM.

Cockburn, A., C. Gutwin, J. Scarr, and S. Malacria. 2014. "Supporting Novice to Expert Transitions in User Interfaces." *ACM Computing Surveys* 47 (2): 31:1–31:36.

Fuentes, C. T., and A. J. Bastian. 2010. "Where is Your Arm? Variations in Proprioception Across Space and Tasks." *Journal of Neurophysiology* 103 (1): 164–171.

Han, J., G. Waddington, R. Adams, J. Anson, and Y. Liu. 2016. "Assessing Proprioception: A Critical Review of Methods." *Journal of Sport and Health Science* 5 (1): 80–90.

Haque, F., M. Nancel, and D. Vogel. 2015. "Myopoint: Pointing and Clicking Using Forearm Mounted Electromyography and Inertial Motion Sensors." In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, Seoul, Republic of Korea, 3653–3656. New York, NY: ACM.

Hincapié-Ramos, J. D. et al. 2014. "Consumed Endurance: A Metric to Quantify Arm Fatigue of Mid-air Interactions." In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, Toronto, ON, Canada, 1063–1072. New York, NY: ACM.

Jacob, M. G., and J. P. Wachs. 2014. "Context-based Hand Gesture Recognition for the Operating Room." *Pattern Recognition Letters* 36: 196–203.

Jang, S. et al. 2017. "Modeling Cumulative Arm Fatigue in Mid-Air Interaction Based on Perceived Exertion and Kinetics of Arm Motion." In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, Denver, CO, USA, 3328–3339. New York, NY: ACM.

Kinect. 2018a. "Kinect Gestures." Accessed October 14, 2018. https://support.xbox.com/en-HK/xbox-360/accessories/body-controller.

Kinect. 2018b. "Kinect Hardware Key Features and Benefits." Accessed October 14, 2018. https://developer.microsoft.com/en-us/windows/kinect/hardware.

Klemmer, S. R., B. Hartmann, and L. Takayama. 2006. "How Bodies Matter: Five Themes for Interaction Design." In *Proceedings of the 6th Conference on Designing Interactive Systems*, DIS '06, University Park, PA, USA, 140–149. New York, NY: ACM.

Kulshreshth, A., and J.J. LaViola Jr. 2014. "Exploring the Usefulness of Finger-based 3D Gesture Menu Selection." In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, Toronto, ON, Canada, 1093–1102. New York, NY: ACM.

Kurtenbach, G., and W. Buxton. 1993. "The Limits of Expert Performance Using Hierarchic Marking Menus." In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, CHI '93, Amsterdam, The Netherlands, 482–487. New York, NY, USA: ACM.

Kurtenbach, G. P., A. J. Sellen, and W. A. S. Buxton. 1993. "An Empirical Evaluation of Some Articulatory and Cognitive Aspects of Marking Menus." *Human-Computer Interaction* 8 (1): 1–23.

Lenman, S., L. Bretzner, and B. Thuresson. 2002. "Using Marking Menus to Develop Command Sets for Computer Vision Based Hand Gesture Interfaces." In *Proceedings of the Second Nordic Conference on Human-Computer Interaction*, NordiCHI '02, Aarhus, Denmark, 239–242. New York, NY: ACM.

Li, Y.. 2010. "Gesture Search: A Tool for Fast Mobile Data Access." In *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, New York, NY, USA, 87–96. New York, NY: ACM.

Li, F. C. Y., D. Dearman, and K. N. Truong. 2009. "Virtual Shelves: Interactions with Orientation Aware Devices." In *Proceedings of the 22Nd Annual ACM Symposium on User Interface Software and Technology*, UIST '09, Victoria, BC, Canada, 125–128. New York, NY: ACM.

Lopes, P. et al. 2015. "Proprioceptive Interaction." In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, Seoul, Republic of Korea, 939–948. New York, NY: ACM.

Matulic, F., and D. Vogel. 2018. "Multiray: Multi-Finger Raycasting for Large Displays." In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, Montreal, QC, Canada, 245:1–245:13. New York, NY: ACM.

Mine, M. R., F. P. Brooks Jr., and C. H. Sequin. 1997. "Moving Objects in Space: Exploiting Proprioception in Virtual-environment Interaction." In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, 19–26. New York, NY: ACM Press/Addison-Wesley Publishing Co.

Nacenta, M. A. et al. 2013. "Memorability of Pre-designed and User-defined Gesture Sets." In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, Paris, France, 1099–1108. New York, NY: ACM.

Nakai, A., A. Pyae, M. Luimula, S. Hongo, H. Vuola, and J. Smed. 2015. "Investigating the Effects of Motion-based Kinect Game System on User Cognition." *Journal on Multimodal User Interfaces* 9 (4): 403–411.

Pasqual, Phillip T., and Jacob O. Wobbrock. 2014. "Mouse Pointing Endpoint Prediction Using Kinematic Template Matching." In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI'14, Toronto, ON, Canada, 743–752. New York, NY: ACM.

Proske, U., and S. C. Gandevia. 2012. "The Proprioceptive Senses: Their Roles in Signaling Body Shape, Body Position and Movement, and Muscle Force." *Physiological Reviews* 92 (4): 1651–1697.

Ren, G., and E. O'Neill. 2012. "3D Marking Menu Selection with Freehand Gestures." In *2012 IEEE Symposium on 3D User Interfaces (3DUI)*, Costa Mesa, CA, 61–68. IEEE.

Rubine, D. 1991. "Specifying Gestures by Example." In *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH'91, Las Vegas, 329–337. New York, NY: ACM.

Sambrooks, L., and B. Wilkinson. 2013. "Comparison of Gestural, Touch, and Mouse Interaction with Fitts' Law." In *Proceedings of the 25th Australian Computer-Human Interaction Conference: Augmentation, Application, Innovation, Collaboration*, OzCHI '13, Adelaide, Australia, 119–122. New York, NY: ACM.

Schofield, W. N.. 1976. "Hand Movements Which Cross the Body Midline: Findings Relating Age Differences to Handedness." *Perceptual and Motor Skills* 42 (2): 643–646.

Schwaller, M., and D. Lalanne. 2013. "Pointing in the Air: Measuring the Effect of Hand Selection Strategies on Performance and Effort." In *Human Factors in Computing and Informatics*, Maribor, Slovenia, 732–747. Springer, Berlin, Heidelberg.

Shen, S., H. Wang, and R. Roy Choudhury. 2016. "I Am a Smartwatch and I Can Track My User's Arm." In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '16, Singapore, Singapore, 85–96. New York, NY: ACM.

Sherrington, C. 1952. *The Integrative Action of the Nervous System*. Cambridge, United Kingdom: Cambridge University Press Archive.

Shoemaker, G. et al. 2010. "Body-centric Interaction Techniques for Very Large Wall Displays." In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries*, NordiCHI '10, Reykjavik, Iceland, 463–472. New York, NY: ACM.

Uddin, M. S., and C. Gutwin. 2016. "Rapid Command Selection on Multi-Touch Tablets with Single-Handed HandMark Menus." In *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces*, ISS '16, Niagara Falls, ON, Canada, 205–214. New York, NY: ACM.

Uddin, M. S., C. Gutwin, and B. Lafreniere. 2016. "HandMark Menus: Rapid Command Selection and Large Command Sets on Multi-Touch Displays." In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, San Jose, CA, USA, 5836–5848. New York, NY: ACM.

Vinayak, R. K., S. Murugappan, H. R. Liu, and K. Ramani. 2013. "Shape-It-Up: Hand Gesture Based Creative Expression of 3D Shapes Using Intelligent Generalized Cylinders." *Computer-Aided Design*, 45 (277–287.Solid and Physical Modeling 2012.

Vogel, D., and R. Balakrishnan. 2005. "Distant Freehand Pointing and Clicking on Very Large, High Resolution Displays." In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*, UIST '05, Seattle, WA, USA, 33–42. New York, NY: ACM.

Walter, R. et al. 2014. "Cuenesics: Using Mid-air Gestures to Select Items on Interactive Public Displays." In *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services*, MobileHCI '14, Toronto, ON, Canada, 299–308. New York, NY: ACM.

Walter, R., G. Bailly, and J. Müller. 2013. "StrikeAPose: Revealing Mid-air Gestures on Public Displays." In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, Paris, France, 841–850. New York, NY: ACM.

Wasenmüller, O., and D. Stricker. 2017. *Comparison of Kinect V1 and V2 Depth Images in Terms of Accuracy and Precision*, 34–45. Cham: Springer International Publishing.

Wilson, A. D. et al. 2016. Providing a tele-immersive experience using a mirror metaphor. US Patent 9,325,943.

Wobbrock, J. O., M. R. Morris, and A. D. Wilson. 2009. "User-defined Gestures for Surface Computing." In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, Boston, MA, USA, 1083–1092. New York, NY: ACM.

Yoo, S. et al. 2015. "To Dwell or Not to Dwell: An Evaluation of Mid-Air Gestures for Large Information Displays." In *Proceedings of the Annual Meeting of the Australian Special Interest Group for Computer Human Interaction*, OzCHI '15, Parkville, VIC, Australia, 187–191, New York, NY: ACM.

Zhai, S. et al. 2012. "Foundational Issues in Touch-Surface Stroke Gesture Design: An Integrative Review." *Foundations and Trends in Human–Computer Interaction* 5 (2): 97–205.

Zhao, S., and R. Balakrishnan. 2004. "Simple vs. Compound Mark Hierarchical Marking Menus." In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*, UIST '04, Santa Fe, NM, USA, 33–42. New York, NY: ACM.