IMPReSS: Improved Multi-Touch Progressive Refinement Selection Strategy

Elaheh Samimi*
Carleton University, Ottawa, Canada

Robert J. Teather†
Carleton University, Ottawa, Canada

ABSTRACT

We developed a progressive refinement technique for VR object selection using a smartphone as a controller. Our technique, IMPReSS, combines conventional progressive refinement selection with the marking menu-based CountMarks. CountMarks uses multi-finger touch gestures to "short-circuit" multi-item marking menus, allowing users to indicate a specific item in a sub-menu by pressing a specific number of fingers on the screen while swiping in the direction of the desired menu. IMPReSS uses this idea to reduce the number of refinements necessary during progressive refinement selection. We compared our technique with SQUAD and a multi-touch technique in terms of search time, selection time, and accuracy. The results showed that IMPReSS was both the fastest and most accurate of the techniques, likely due to a combination of tactile feedback from the smartphone screen and the advantage of fewer refinement steps.

Keywords: Selection, progressive refinement, touchscreen.

Index Terms: Human-centered computing → Virtual reality

1 INTRODUCTION

Many selection techniques have been introduced to improve selection accuracy and speed [1]; however, 3D selection is still cumbersome [15]. Ray- and hand-based techniques are common, yet they do not address a key and ongoing challenge: designing selection techniques that perform efficiently when selecting targets in *dense* virtual environments (VEs) or for *remote/small objects*.

Progressive refinement is a relatively recent class of optimized selection techniques introduced by Kopper et al. [11], which break a target selection down into sequence smaller selections using a quad menu (a square menu consisting of four triangles placed together side by side) at each step. These techniques do not need precise selection in the first step, e.g., with a ray. Instead, they select a cluster of objects around the target without high precision. Then a quad menu opens, which contains all objects selected in the first cluster, presented on a 2D menu. The user then selects the quadrant containing the target object, refining the selection. The objects on that quadrant get reorganized onto four quadrants on the next iteration. The process repeats until just the target object remains, at which point selection is trivial.

Progressive refinement generally prioritizes accuracy over speed, but is less effective in low-density environments [11]. The primary advantage of such techniques is added precision in high-density virtual environments. Selecting the target becomes a sequence of simple 2D tasks [20, 21] rather than trying to precisely select an object in 3D, avoiding numerous distractors.

Recently, Pollack et al. devised a new multi-finger touchscreen-based selection technique called CountMarks [19]. CountMarks was developed for use with smart-glasses and/or AR displays, and uses a smartphone as a controller to select items via a menu in VR [22]. Previous work has shown that smartphones show promise as VR controllers [2, 8, 14], with recent VR devices, like the HTC Vive Flow using smartphones as a primary interaction controller[1].

CountMarks combined count menus [5], where the user touches a number of fingers to the screen corresponding to the desired item of selection, with marking menus [16], a gesture-based method where a pie menu allows users to select their desired items by swiping their finger in the correct direction (e.g., up, down, right, left). With CountMarks, multiple items were shown on each sub-menu of a radial menu. The number of fingers touched to the screen indicates which item in a menu should be selected, while swiping in a direction indicates which menu should be selected. CountMarks is good when there are many items, as it reduces the number of menus open (the level users need to dig to select the target object), improving selection time.

CountMarks' directional menu makes it complimentary with progressive refinement. We developed a new progressive refinement technique employing CountMarks to "short-circuit" the menu hierarchy in progressive refinement by skipping some menu levels for selecting the target object inside the quad menu. Our technique, IMPReSS (for **I**mproved **M**ultitouch **P**rogressive **Re**finement **S**election **S**trategy), supports "careless" selection during the first step, like other progressive refinement techniques by using sphere-casting and distributing objects into quad menus. It uses multi-touch finger input to support picking between multiple objects in a quad menu, reducing the number of refinement steps required relative to previous progressive refinement techniques. Our key contributions are the design of our new technique, a user study verifying its effectiveness.

2 RELATED WORK

According to Kopper et al. [11], selection techniques can be divided into two main categories:

- *Immediate selection*, where a target is acquired after performing an action (e.g., touch or point to it and press a button)
- *Progressive refinement,* which involves refining a subset of the objects to a smaller group until only the desired object remains.

We omit a comprehensive review of immediate selection techniques, as they are provided elsewhere [1]. We focus on progressive refinement here, since our IMPReSS technique is an example of progressive refinement.

2.1 Progressive Refinement Selection

Kopper et al. [11] first introduced progressive refinement to address selection task difficulty due to target size, distance, and density. Other issues, like hand/tracker jitter and target movement, are not discussed here but also impact VR selection tasks [15].

Unlike immediate selection techniques, progressive refinement breaks the selection process into several steps. In the first step, the user initially selects a large set of objects, including the target

*email: elahehsamimi@cmail.carleton.ca
† email: rob.teather@carleton.ca

[1]https://www.vive.com/us/product/vive-flow/overview/

object. The technique then supports the refinement of this initial selection set by progressively reducing the set of selected objects until only the desired one remains, using either a discrete or continuous method [40]. This refinement component provides better control for users by distributing objects into a quad menu and prevents selecting the wrong object (see Figure 2).

SQUAD [11] is a discrete progressive refinement technique that employs sphere-casting and a QUAD menu to reduce the number of selected items step by step to reach the target object. All objects inside the selection sphere define the initial "cluster" of selected objects. Then, the selected objects are spread evenly across the four quadrants of a quad menu. The user selects the target quadrant using ray-casting, which discards all objects in other quadrants, thus refining the selection until only the target object remains. While SQUAD offers significantly better accuracy than ray-casting [15], it is slower than ray-casting *except* when selecting small objects or in low-density environments.

Expand [7], like SQUAD, requires first selecting a group of objects, including the target. Unlike SQUAD, the user's view zooms to the area defined in the first step rather than opening out-of-context quad menus. Cashio et al. [7] report that Expand is faster than SQUAD, especially in dense environments, but offers lower accuracy. Both SQUAD and Expand are limited by environment density – dense environments yield a proportionally higher number of objects in the initial selection step. Based on this observation, Bacim et al. [3] proposed the double bubble technique, which uses a 3D bubble cursor [9] to reduce the number of objects selected in the first step. Double bubble was found to offer better speed and accuracy compared to ray-casting.

With techniques like discrete zoom [4] and PRECIOUS [17], the desired location is zoomed in to the user, either by moving the scene closer to the user (discrete zoom) or the user closer to the scene (PRECIOUS). Unlike SQUAD, the quad menu is always visible (and transparent) with discrete zoom, preserving user presence. Both techniques improve accuracy by sacrificing speed.

Continuous zoom [4] is similar to discrete zoom, except instead of spreading the scene to a quadrant menu, it zooms to the cursor area continuously and constantly. With the intent-driven selection (IDS) technique [18], users manually adjust the selection sphere volume and zoom in and out into the environment by changing their hand and finger position, making the sphere smaller or bigger until it reaches the target object.

Flower ray [10] improves on depth ray and lock ray [10]. With flower ray, when the user wants to select an object, the items intersected with the pointing ray are selection candidates. The user then presses and holds a button first to lock the point and then to distribute the intersected items into a marking menu. Depth ray is faster than flower ray and lock ray and offers similar accuracy. However, none offer high selection speed in dense environments.

2.2 Marking Menu Selection

Marking menus [13] allow users to select items from a radial/pie menu in the desired menu's direction. A menu with more than four items, with only four items per "layer" of the menu, means users must navigate to a deeper layer of the menu to select the desired item. This reduces the number of depth layers of the menu but decreases the accuracy of selection at any given layer. Marking menus support swipe gestures for faster and more accurate selection than the linear selection menus [6]. The problem with marking menus is when there are eight items or more at levels greater than two, which increases the error rate [12]. Lepinski et al. [16] proposed a method to increase menu breadth using multi-touch input. In this method, users put their fingers in different combinations, like guitar chords, on a touch screen. Different menus pop up, and the user can select the desired menu by swiping in the desired direction on a touch screen. Finger combinations were faster than marking menus, but most combinations are too complicated for users to perform [19].

Pollock et al. [19] combined the multi-touch feature of count menus [5] with the swipe gestures of marking menus [13]. Their CountMarks technique provides one-handed selection of more menu items in fewer sub-menu layers than marking menus. The number of fingers the user touches to the screen indicates an item in each menu. Swiping in a given direction selects the indicated item from the menu in that direction. CountMarks is faster than marking menus but offers worse accuracy due to the possibility of making mistakes with the number of fingers on the screen. CountMarks may improve progressive refinement, by increasing menu breadth and potentially allowing quicker selection (based on how many fingers are touching the screen) rather than drilling down. Therefore, we combined CountMarks with progressive refinement, to initially select objects in a group, and then use CountMarks menus to choose items inside the menu.

3 Design of the IMPReSS Technique

We designed a multi-touch progressive refinement technique to improve selection speed while offering precise accuracy. We dub the technique IMPReSS, short for **I**mproved **M**ultitouch **P**rogress **Re**finement **S**election **S**trategy. Like other progressive refinement techniques, the technique involves several steps to accomplish selection. We describe each step below:

Step 1: Selecting Target Area: Like SQUAD, users do not need to be precise in the first step, instead selecting the general target area. Like other progressive refinement techniques, this employs ray-casting with a selection sphere at the ray/scene intersection point. Objects in the sphere are selection candidates. See Figure 1.

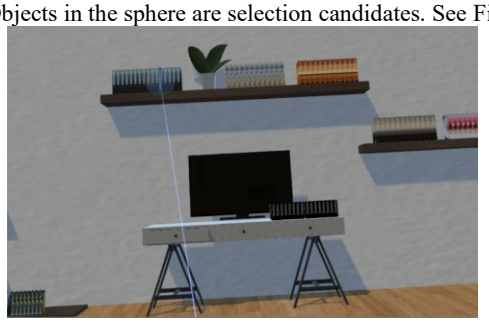

Figure 1. IMPReSS: Selecting target area via sphere-casting.

Step 2: Display Candidate Objects: The initial selection typically yields multiple candidate objects, which we distribute on a quad menu, like SQUAD. Objects are distributed evenly inside each menu quadrant. We start by arranging objects from the outside of each triangle and continue to the middle. Denser virtual environments generally have more objects in the initial selection; objects scale to fit in the quadrants. See Figure 2.

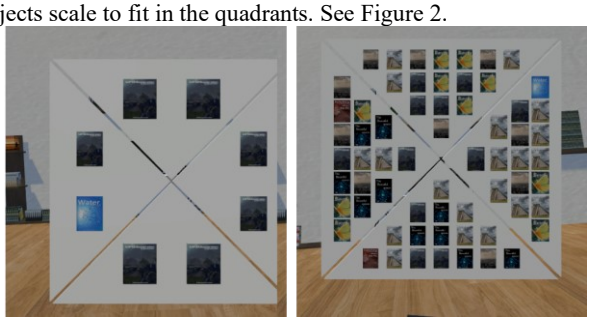

Figure 2. IMPReSS Step 2: Displaying candidate objects. (Left) distributed objects inside a quad menu in low density. (Right) distributed objects inside a quad menu in high density

To balance objects equally in quads, we dedicated one object at a time to each menu, first filling all the first positions of each quadrant, then each second position, and so on. See Figure 3.

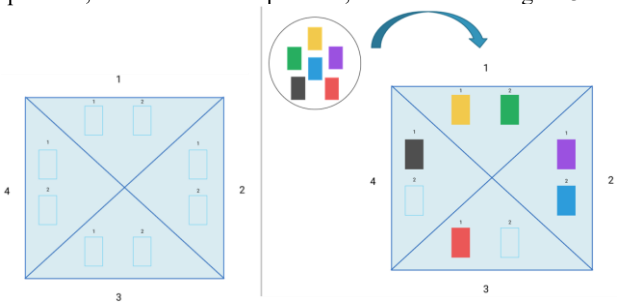

Figure 3. Ordering objects in quads. (Left) Predefined numbered positions in each quad. (Right) distributing candidates.

Finally, we display the quads. If there are four or fewer items in each quad, we also present the position number of each object on each quad, which corresponds to how many fingers would select that object with a directional swipe. See Figure 4 (left).

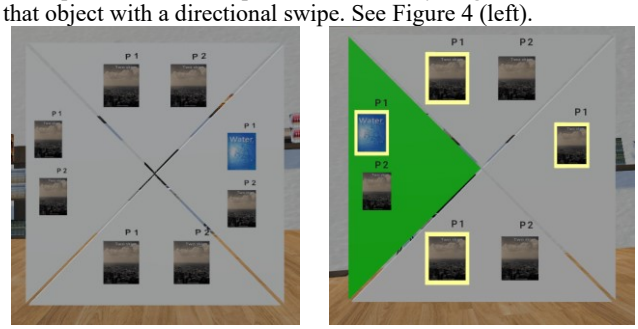

Figure 4. Left: IMPReSS quad menu, depicting finger counts. Right: Touching one finger to the screen causes all "position 1" items to highlight. Swiping left will select the blue book in this case.

The maximum number of objects in each menu is 16 based on quad and item sizes, and spacing parameters. We support up to 16 (objects) × 4 (number of menus) = 64 objects in each menu level.

Step 3: Selecting an Object in the Quad Menu: The user next refines the selection to remove undesired objects. We break this step into two sub-steps:

Step 3.1 Indicating target via multitouch: This step employs the idea of CountMarks [19]. If there are four or fewer objects in each menu, the user can select the target outright by touching the screen with the number of fingers corresponding to the target's position, for up to four fingers. Upon touching the screen, all candidate objects corresponding to the finger count are highlighted. For instance, in Figure 4 (right), the user put one finger on the screen to select the blue book in the first position.

***Step 3.2 Select Target by Swipe*:** After indicating the desired object by touching the screen (Step 3.1), users finalize their selection by swiping their fingers in the desired direction. Upon moving their fingers in the desired direction, the selected quadrant highlights green. For example, in Figure 4 (right), the user is swiping left, highlighting the left quadrant. Selection occurs when the user swipes their finger(s) to the edge of the touchscreen.

The technique currently supports only four primary swiping directions. This could be extended to more (e.g., up to eight) directions, to display more selected items, or to distribute items further across each menu. If there are more than four candidate objects in each quad (i.e., more objects in each quad than the user has fingers), users must first refine the candidates further before multi-touch selection is possible. This is seen in Figure 2 (right). Users can swipe any number of fingers on the touch screen in the desired quad direction to "drill down" and refine the candidate set. The items from the selected quad then redistribute, and the process repeats from Step 2, until four or fewer items appear on each quad. Users can then select the object as described in Step 3.

4 Experiment

We experimentally compared selection performance offered by IMPReSS to SQUAD and a multi-touch technique.

4.1 Participants

We recruited 12 participants (4 men, 8 women, aged 26 to 42 years, μ = 32.75, *SD* = 4.2, all right-handed). Only a third had used VR before, the rest had no prior VR experience. All participants had a normal or corrected-to-normal stereo vision.

4.2 Apparatus

To comply with COVID-19 measures, we dropped off key hardware at participants' residences, following social distancing protocols. Participants used their own mouse and keyboard.

4.2.1 Hardware

We used a PC with an Intel Core i7-7700K CPU at 4.20GHz with 32 GB of RAM, with an Oculus Rift CV1. We used a Samsung Galaxy S8 (OS Android 9.0) as the VR controller. To ensure consistent hardware setups between participants, we included hardware setup instructions. The experiment software was pre-loaded on the provided PC and Galaxy S8 smartphone.

4.2.2 Software

We developed two applications: one running on the PC and the other on the smartphone. We developed the software in Unity3D (v2019.2.10) on Microsoft 64-bit Windows 10, with Unity Android 7.0 for the smartphone app. We used Photon[2] for networking to share data between the smartphone and PC. The PC app acted as a server, with the smartphone app acting as client. The smartphone app transmitted gyroscope data to the PC, which was used to control the 3DOF orientation of the controller in the desktop VR application. It also transmitted touchscreen touch events for the IMPReSS and multi-touch techniques.

The desktop app presented a virtual library. Upon starting the task, the software displayed a message providing instructions to participants. To advance, participants pressed any key on the keyboard after each selection. Upon pressing a key, the timer started to count task completion time. Each trial, the software placed the target blue book in a random position on one of the bookshelves. Upon pressing a key after the last selection trial, the experiment ended. The scene is depicted in Figure 5.

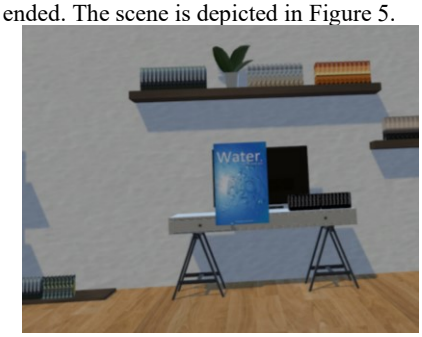

Figure 5. The scene, depicting the target blue book that users were tasked with finding and selecting from shelves with distracters.

[2] https://www.photonengine.com/pun

In addition to IMPReSS (Section 3), the software also included SQUAD, using the ray emitted by the 3DOF smartphone controller to select each quad during refinement. We included a third technique, multi-touch, which operated the same as IMPReSS, but did not use more than one finger to disambiguate targets. Instead, swiping any number of fingers on the screen simply refined the menu like SQUAD.

We included three different environment densities: *Low density* (1058 books, Figure 6, top), *Medium density* (1901 books, Figure 6, middle), and *High density* (3,055 books, Figure 6, bottom).

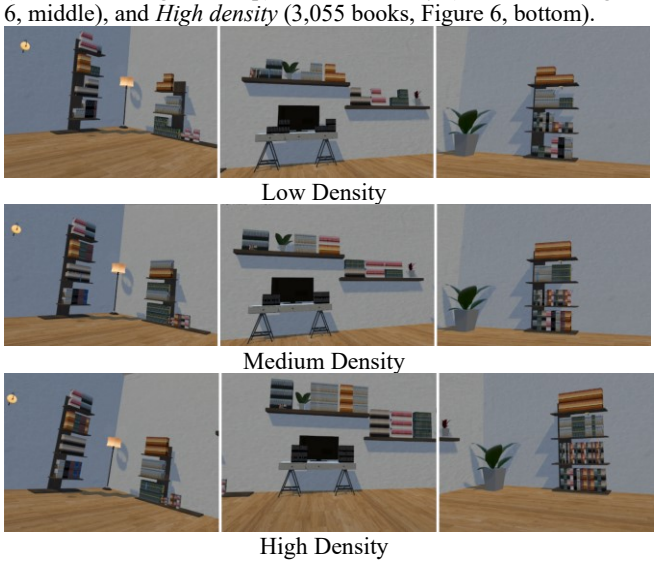

Figure 6. The environment density conditions. We scaled books to be thinner and stacked on top of each other in higher densities.

The software used participant ID to assign counterbalancing groups automatically. The software recorded search time, selection time, and error rate.

4.3 Procedure

We conducted the experiment remotely due to the COVID-19 pandemic. After pre-screening participants, we dropped off apparatus at their residence. We provided setup instructions and used virtual meetings via Zoom or Skype to assist them.

Participants first provided informed consent and a demographic questionnaire. Each participant received a unique ID via email. After starting the software, they entered some setup information, including ID and set the density as instructed. The program then started, and the software presented instructions. Pressing any key started the first trial, and put the target book in a random position. After the last trial of each density (the end of a block), the system indicated to participants that the block was complete. They repeated this process until all trials were finished for all densities.

After completing all trials for a selection technique, participants completed a device assessment and usability questionnaire. Participants could take breaks as needed between each block. At the end of the whole experiment, they completed a post-experiment questionnaire ranking the techniques. Overall, the experiment took ~45-50 minutes for the VR part, and ~10-15 minutes for setup. Participants received a $15 digital Amazon gift card as compensation for participating in the experiment.

4.4 Design

We employed a 3×3 within-subject design with the following independent variables and levels:

Selection technique: IMPReSS, SQUAD, multi-touch
Density: Low, medium, high

Participants completed 8 selection trials for each selection technique/density combination for a total of 3 selection techniques × 3 densities × 8 targets = 72 selections (864 selections in total across all 12 participants). We counterbalanced selection technique order according to a Latin square. We did *not* counterbalance density; all participants started with low and ended with high density for each selection technique. This gave participants a chance to become familiar with the technique and task with the easier low-density condition first.

We recorded three dependent variables: search time (s), selection time (s), and error rate (%). Search time was the time from a trial start until the quad menu opened. Selection time was the time from opening the quad menu until the target book was selected. Error rate was the average number of incorrect menu selections before completing a selection trial.

5 Results

We used two-way ANOVA with Bonferroni-Dunn posthocs (all at $p < .05$) are depicted as horizontal lines (●┈┈●) in bar charts.

5.1 Search Time

There was a significant main effect of density on search time ($F_{2,22} = 911.72$, $p < .0001$). However, neither the main effect of selection technique ($F_{2,\ 22} = 2.16$, $p > .05$), nor the selection technique × density interaction effect were significant ($F_{4,\ 44} = 0.32$, ns). Finding a target would clearly takes longer in a denser environment, but the selection technique – which primarily differs *after* the initial selection step – was not expected to influence search time. Mean search times and pairwise significant differences (between density levels) are seen Figure 7.

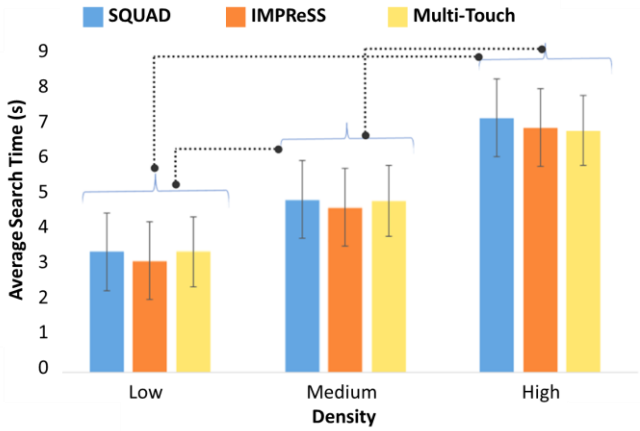

Figure 7. Mean search time by selection technique and density. Error bars show ±1 *SE*.

5.2 Selection Time

Selection time is how long participants spent navigating the quad menu after initial selection of candidate objects. The main effects on for both selection technique ($F_{2,22} = 85.76$, $p < .0001$) and density ($F_{2,22} = 25.25$, $p < .0001$) were significant as was the interaction effect between selection technique and density ($F_{4,44} = 4.40$, $p < .005$). IMPReSS offered faster selection time than the other two techniques, and was generally less affected by density than the other techniques. See Figure 8. The fact that both touchscreen-based approaches performed better than SQUAD is likely due to the lower DOFs and simpler gestural control required. IMPReSS likely performed better than multi-touch because it supported breaking out of the menu hierarchy earlier than the other techniques.

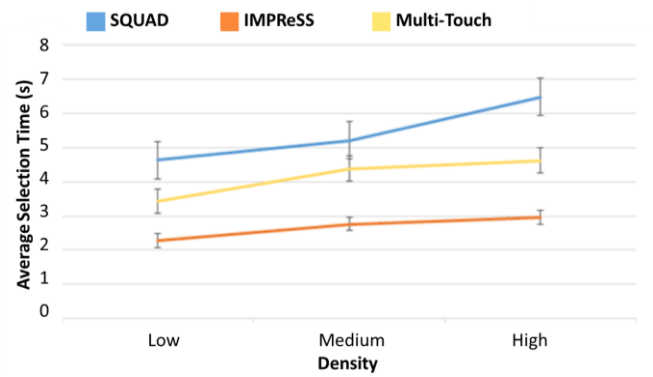

Figure 8. Mean selection time by selection technique and density. Error bars show ±1 *SE*.

5.3 Error Rate

An error occurs when selecting the wrong quad during menu navigation. Error rate was the percentage of erroneous selections out of all menu selections for a condition. There was a significant main effect of selection technique on error rate ($F_{2,22} = 11.71$, $p < .0005$). Neither the main effect of density ($F_{2,22} = 3.14$, $p > .05$), nor the selection technique/density interaction effects were significant ($F_{4,44} = 0.95$, ns). See Figure 9.

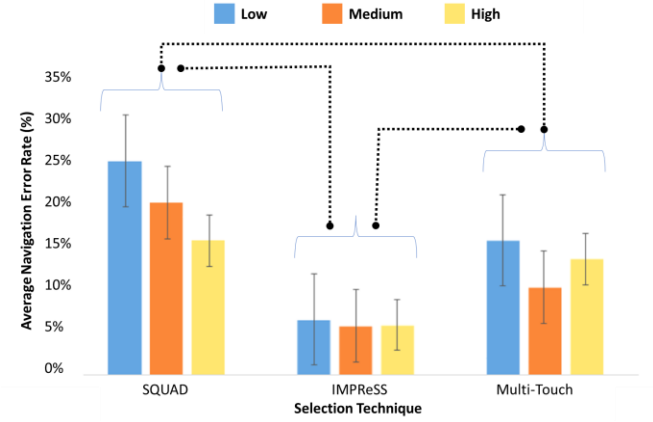

Figure 9. Mean error rate by selection technique and density. Error bars show ±1 *SE*.

We were surprised that SQUAD's error rate was much higher than two other techniques. We attribute this to imprecision in the 3DOF pointing used with the ray-based technique.

5.4 Finger Error Rate

Naturally, participants sometimes used the wrong number of fingers with IMPReSS. We recorded this "finger error rate" separate from navigation errors. Since it only exists for IMPReSS, we do not compare it to the other techniques, instead using one-way ANOVA to compare across density. The main effect for density on finger error rate was not significant ($F_{2,22} = 0.291$, ns). Finger error rate was generally low (~5%), see Figure 10.

5.5 Questionnaire Results

To assess participant experience, we also included several subjective questions. After completing each selection technique, we asked how easy, fast, and accurate the selection technique was. We also asked them to explain the reasons for their choice, so we could understand their requirements for future improvements to the technique. Results of the questionnaire are seen in Figure 11.

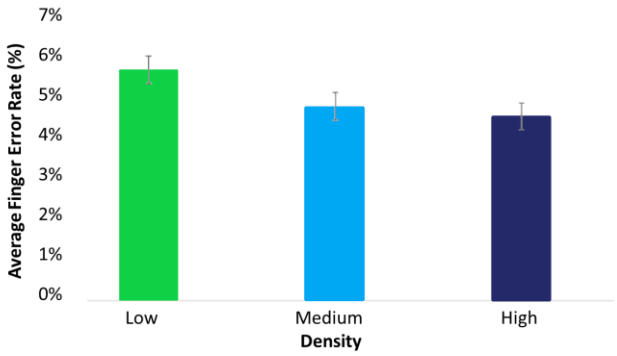

Figure 10. Average finger error rate by density for the IMPReSS technique. Error bars show ±1 *SE*.

At the end of the experiment, we asked about user preferences. IMPReSS was most preferred (58%), multi-touch was second (25%), and SQUAD was least preferred (17%). Participants mentioned that it was straightforward and faster to select objects via the IMPReSS menu. Some participants mentioned that swiping was more challenging with three or four fingers, though.

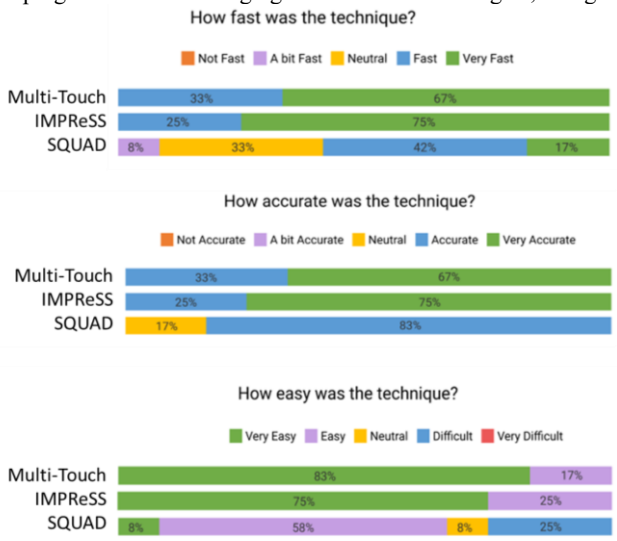

Figure 11. Subjective questionnaire results, showing percentage of participants choosing each answer.

6 Discussion

Overall, participants could select targets faster with IMPReSS due to navigating through fewer menu layers. Selection time was impacted by both selection technique and density; IMPReSS allowed faster selection than the other techniques suggesting multi-touch input to "short-circuit" out of a marking menu early is a viable option for VR selection.

The low finger error rate suggests that participants had little difficulty determining how many fingers to put on the screen during selection. Participants commented that the object position numbers in quad menus made it clear how many fingers to use. Thus, we believe that any such errors were likely accidental slips, rather than misunderstanding how many fingers to use. Although the low-density finger error rate was slightly higher than the medium and high densities, this is likely due to practice effects, rather than an effect of density itself, given the density ordering.

There was little difference in selection time between the three density levels with IMPReSS. The multi-touch technique was the second-fastest technique, with SQUAD slowest. We believe this is

due to the reduced degrees of freedom and the comparatively simple and fast finger swipe gestures required with the two touchscreen-based techniques. The effects of degrees of freedom have been extensively studied, with higher-DOF selection techniques generally performing worse [20, 21]. SQUAD required pointing the smartphone like a VR controller, to aim a ray at the various menu quadrants. Since IMPReSS performed better than multi-touch, we believe this is due to having to navigate through fewer menus, on average, than multi-touch. These effects were more pronounced in high density, especially with SQUAD, due to the additional layers of menu navigation required.

Our results show navigation error rate is affected by selection technique but not density. SQUAD had a higher navigation error rate compared to the other techniques, especially with low and medium density. With SQUAD, navigation errors occurred when the user pointed the selection ray at the wrong quad. Hence, with SQUAD, high error rates can occur due to careless selections or imprecision due to input noise and hand tremor. We note that the calibration between the smartphone and the selection ray was also imperfect, which also likely influenced SQUAD's error rate.

Finally, we also confirmed that, as expected, there was little difference in *search* time by selection technique. Unsurprisingly, search time was highly dependent on density.

6.1 Limitations

Conducting the experiment remotely potentially introduced more variability in our results. On one hand, this enhances our study's external validity, but also highlights the difficulty in conducting remote experiments. Participants had to set the equipment themselves, which required setup time, consequently limiting the number of trials and densities to keep the experiment duration reasonable. For this reason, we also excluded tutorial/practice sessions. Since density order was not counterbalanced, this likely means that there was a sharp learning curve in early trials, primarily affecting the low density conditions.

7 Conclusion

We developed IMPReSS, a VR selection technique that combines progressive refinement with the CountMarks multi-touch marking menu technique. IMPReSS was designed to improve selection time and accuracy. This work focused on enhancing the two key factors of all selection tasks: selection time and accuracy (error rate). While previous progressive refinement techniques are known to improve accuracy, our technique helped with speed by using multi-touch to reduce the required menu navigation.

IMPReSS offered better selection time and accuracy than other techniques, especially in dense environments. In addition, there seems to have been little confusion about how many fingers were required to select a target.

Our results add to the growing literature on smartphones as VR controllers. These devices are attractive as VR controllers, as they provide several sensors, and precise touchscreens that offer tactile feedback. In our case, we further use the device's multi-touch capabilities. Our results suggest that not only is this more performant than 3DOF ray-based control, but also user-preferred.

References

[1] Argelaguet, F. and Andujar, C. 2013. A survey of 3D object selection techniques for virtual environments. *Computers and Graphics* (2013), 121–136.

[2] Babic, T., Reiterer, H. and Haller, M. 2018. Pocket6: A 6DoF controller based on a simple smartphone application. *ACM Symposium on Spatial User Interaction*, 2–10.

[3] Bacim de Araujo Silva, F. 2015. *Increasing Selection Accuracy and Speed through Progressive Refinement*. Doctoral Dissertation, Virginia Tech.

[4] Bacim, F., Kopper, R. and Bowman, D.A. 2013. Design and evaluation of 3D selection techniques based on progressive refinement. *International Journal of Human Computer Studies*, 785–802.

[5] Bailly, G., Lecolinet, E. and Guiard, Y. 2010. Finger-count & radial-stroke shortcuts: Two techniques for augmenting linear menus on multi-touch surfaces. *ACM Conference on Human Factors in Computing Systems*, 591–594.

[6] Bailly, G., Lecolinet, É. and Nigay, L. 2017. Visual Menu Techniques. *ACM Computing Surveys,* Association for Computing Machinery.

[7] Cashion, J., Wingrave, C. and Laviola, J.J. 2012. Dense and dynamic 3D selection for game-based virtual environments. *IEEE Transactions on Visualization and Computer Graphics*, 634–642.

[8] Chen, H.-H., Lin, Y.-B. and Liou, R.-H. 2011. Direction-based wireless remote controller: A smartphone application. *Journal of Wireless Mobile Networks*, 33–45.

[9] Grossman, T. 2005. The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area. *ACM Conference on Human Factors in Computing Systems*, 281–290.

[10] Grossman, T. and Balakrishnan, R. 2006. The design and evaluation of selection techniques for 3D volumetric displays. *ACM Symposium on User Interface Software and Technology*, 3–12.

[11] Kopper, R., Bacim, F. and Bowman, D.A. 2011. Rapid and accurate 3D selection by progressive refinement. *IEEE Symposium on 3D User Interfaces*, 67–74.

[12] Kurtenbach, G. and Buxton, W. 1993. The limits of expert performance using hierarchic marking menus. *ACM Conference on Human Factors in Computing Systems*, 482–487.

[13] Kurtenbach, G.P. 1993. *The design and evaluation of marking menus*. Doctoral Dissertation, University of Toronto.

[14] Kyian, S. and Teather, R.J. 2021. Selection performance using a smartphone in VR with redirected input. *ACM Symposium on Spatial User Interaction.*

[15] Laviola, J.J., Kruijff, E., McMahan, R.P., Bowman, D.A. and Poupyrev, I. 2017. *3D User Interfaces: Theory and Practice*.

[16] Lepinski, G.J., Grossman, T. and Fitzmaurice, G. 2010. The design and evaluation of multitouch marking menus. *ACM Conference on Human Factors in Computing Systems*, 2233–2242.

[17] Mendes, D., Medeiros, D., Sousa, M., Cordeiro, E., Ferreira, A. and Jorge, J.A. 2017. Design and evaluation of a novel out-of-reach selection technique for VR using iterative refinement. *Computers and Graphics* (2017), 95–102.

[18] Periverzov, F. and Ilieş, H. 2015. IDS: The intent driven selection method for natural user interfaces. *IEEE Symposium on 3D User Interfaces*, 121–128.

[19] Pollock, J. and Teather, R.J. 2019. CountMarks: Multi-finger marking menus for mobile interaction with head-mounted displays. *International Conference on Human-Computer Interaction*, 1–20.

[20] Teather, R.J. and Stuerzlinger, W. 2013. Pointing at 3D target projections with one-eyed and stereo cursors. ACM *Conference on Human Factors in Computing Systems*, 159–168.

[21] Teather, R.J. and Stuerzlinger, W. 2011. Pointing at 3D targets in a stereo head-tracked virtual environment. *IEEE Symposium on 3D User Interfaces*, 87–94.

[22] VR controllers the good the bad and the ugly - YouTube: 2016. https://www.youtube.com/watch?v=GKkrLwQPGWM. Accessed: 2021-09-05.