TrackMaze: A Comparison of Head-Tracking, Eye-Tracking, and Tilt as Input Methods for Mobile Games

Mahdieh Abbaszadegan(✉), Sohrab Yaghoubi, and I. Scott MacKenzie

Department of Electrical Engineering and Computer Science, York University, Toronto, ON, Canada
mahdieh@eecs.yorku.ca, sohrab.y93@gmail.com, mack@cse.yorku.ca

Abstract. A user study was performed to compare three input methods (tilt, eye-tracking, head-tracking) with two gain levels (low, high) on a custom-made *TrackMaze* mobile game. The task involved maneuvering a virtual ball through a maze while trying to avoid walls. The game was developed in Swift using the ARKit framework. The TrueDepth front-facing camera of an Apple *iPhone X* was used for the eye-tracking and head-tracking conditions. We evaluated user performance (maze completion time, number of wall hits) and qualitative measures (ease of use, enjoyment, fatigue). Tilt input showed the best performance and eye-tracking showed the worst performance. The mean maze completion time was 12.3 s for tilt, 22.5 s for head-tracking, and 31.8 s for eye-tracking. High gain was 26% faster than low gain. Tilt was the most precise input method with only 1.06 wall hits per trial, compared to head-tracking (2.30) and eye-tracking (4.00). Participants preferred tilt and head-tracking over eye-tracking and noted that the eye-tracking interface was fatiguing and hard to use.

Keywords: HCI · Mobile games · Augmented reality on mobile devices ARKit · iOS · Head-tracking · Eye-tracking · Tilt-input

1 Introduction

Many computer users are switching from personal computers to smartphones or tablets for their daily computing needs. This transition has created growth in mobile application development, with many companies offering exclusive services and promotions over their mobile apps. The goal is to lure users to smartphones. And the rewards are huge. Since launching the App Store in 2008, Apple's developer community has earned $70 billion worldwide, with $20 billion in 2016 alone [12, 13]. Furthermore, gaming is the top-grossing across 25 app categories, with games accounting for 25% of all available apps [20]. This paper focuses on input methods for mobile games.

Smartphones receive input through a touchscreen, a camera, microphone, accelerometer, etc. Touch input is the primary input method on smartphones, and this is also true for mobile games. Some drawbacks of touch input for the player are lack of tactile feedback and occlusion of the display [8]. Smartphones and tablets typically

M. Kurosu (Ed.): HCI 2018, LNCS 10903, pp. 393–405, 2018.
https://doi.org/10.1007/978-3-319-91250-9_31

include an accelerometer which measures the tilting motion and the physical orientation of the device. Device orientation can be used for controlling game objects. Such control, commonly called "tilt", is widely used as an input method in mobile games. In addition, a smartphone's camera can track movement in the real world. Using the front-facing camera, we can track the user's face or eye, and even obtain their facial expression. Thus, eye-tracking and head-tracking are possible input methods for hands-free games, in addition to tilt. One benefit of hands-free input is to provide more entertainment options for people with special needs. However, using the eyes as an input method is problematic, since the eyes are also our main perception method [16]. Notably, ease of use and novelty has an impact on "perceived enjoyment" in mobile games [18].

With iOS 11, Apple introduced ARKit, a framework for creating Augmented Reality (AR) apps for the *iPhone* and *iPad*. ARKit combined with the TrueDepth front-facing camera available on the *iPhone X* enables face or eye tracking in AR apps. The TrueDepth camera gives the position, orientation, and expression of the face in real-time. It also enables Face ID, the facial recognition method for authentication on an *iPhone X*.

In this paper, we evaluate the functionality of eye-tracking and head-tracking, using ARKit and the TrueDepth camera. We compare tilt-input, head-tracking, and eye-tracking for a custom-designed game called *TrackMaze*. With *TrackMaze*, the player navigates a virtual ball through a static maze. We address several questions: Is it possible to use eye-tracking and head-tracking for mobile games that need whole-display navigation? Can eye-tracking and head-tracking result in better perceived enjoyment? Can eye-tracking or head-tracking compete with tilt-input, the natural input method for maze games? Which input method is easiest to use? Do users prefer one input method over the others?

In the next section, we review studies on mobile games related to tilt-input, eye-tracking, and head tracking. Then, we discuss the methodology and results of our user study. We then offer conclusions on user performance and enjoyment for tilt, eye-tracking, and head-tracking as input methods.

2 Related Work

2.1 Tilt-Based Interaction

Many studies have evaluated different orders of control for tilt-input [e.g., 6, 15, 24] or compared tilt-input with touch-input [e.g., 4, 17, 21, 23]. We will further examine game-related studies.

Medyrk and MacKenzie [17] tested touch input and tilt input on the *Bit.Trip Beat* game. Despite the better performance of touch input, users preferred tilt-input as it made the game more enjoyable. Cairns et al. [4] reported no significant difference between tilt and touch in performance or level of immersion in their custom designed *DoodleJump* game.

Teather and MacKenzie [23] compared two orders of control (position-control, velocity-control) for both touch input and tilt input in a custom Pong-like game. Their results showed that order of control bears more on performance than input method, with position-control showing better performance compared to velocity-control.

Constantin and MacKenzie [6] examined velocity-control and position-control for tilt-input. Their study used *TiltMaze*, an Android game to navigate a virtual ball through a simple maze. Tilt input with position-control showed better performance but 83.3% of the participants preferred velocity-control. We use velocity-control for tilt-input in *TrackMaze*.

2.2 Head-Based Interaction

Gorodnichy et al. [10] developed *Nouse 'Use your Nose as a Mouse'*, a facial tracking technique based on marking nose features. They also developed *NosePong*, a game using *Nouse* technology which employed two head-mounted web cameras. But they did not present experimental data for their applications.

Cuaresma and MacKenzie [9] compared two facial tracking methods (positional, rotational) and three selection methods (smile, dwell, blink) with an experimental Android application called *FittsFace*. *FittsFace* was developed conforming to the ISO 9241-9 standard for non-keyboard input devices [11]. Positional navigation tracks the nose position in the face and rotational navigation tracks the rotation of the face. Performance was evaluated by calculating Fitts' law throughput. Roig-Maimó et al. [22] also calculated throughput in an evaluation of head-tracking in a target-selection mobile application. They compared the effect of different phone rotations and gain levels on head-tracking. Both studies reported low values for throughput – in range of 0.5 to 1.0 bits per second – thus demonstrating the challenges for head-tracking interaction. However, these two studies establish the potential feasibility of using face- or head-tracking as an input methods in mobile applications. In the present research, we evaluate eye-tracking and head-tracking performance in *TrackMaze* and compare them with tilt input.

Cuaresma and MacKenzie [8] compared tilt-input and facial tracking on a custom Android game called *StarJelly* running on a Google *Nexus 7* tablet. They reported that the tilt-input method had significantly better performance but participants praised the novelty of facial tracking. The research noted slight fatigue using head-tracking. *StarJelly* is a vertical scrolling game and the player only navigates the avatar horizontally. In scrolling games, the camera moves with the game character but in non-scrolling games, the camera maintains a constant view at any time [23]. Our *TrackMaze* game is a non-scrolling game so the player needs to control a virtual ball in all directions. We aim to resolve the latency issue noted in *StarJelly* as ARKit promises real-time face tracking data.

2.3 Eye-Based Interaction

Eye-tracking can be used in games as an input method or as a means to observe user behaviour [2]. Peshkovskaya et al. [19] studied decision making in the *Prisoner's*

Dilemma game with eye-tracking glasses. They showed that eye-tracking can be used to understand the decision-making process.

There are many available eye-tracking studies in different areas and applications on PCs. Most use a dedicated eye-tracking apparatus or eye-tracking glasses but people are not comfortable using these devices daily [7]. Mobile eye-tracking has progressed in recent years but compared to desktop eye-tracking, "it is still in its infancy" [3, p. 280]. In *TrackMaze*, the player doesn't use a separate apparatus or glasses: The built-in mobile camera is used for the eye-tracking and head-tracking input modes.

3 Method

In this section, we represent the methodology of our user study. The goal is to compare speed, accuracy, fatigue, ease of use, and enjoyment between tilt, eye-tracking and head-tracking. The context is mobile gaming.

3.1 Participants

Twelve unpaid voluntary participants were recruited from the local university campus and a friend's workplace. Seven participants were female, five were male. Ages ranged from 23 to 52 years (mean = 30.9, *SD* = 10.3). All were regular users of smartphones, reporting 2 to 10 h daily usage (mean = 5.2 h, *SD* = 2.2 h). On overall experience using tilt input, one reported no experience; the other responses were 1–10 h (7) and >10 h (4). All but three participants reported no experience with head-tracking or eye-tracking.

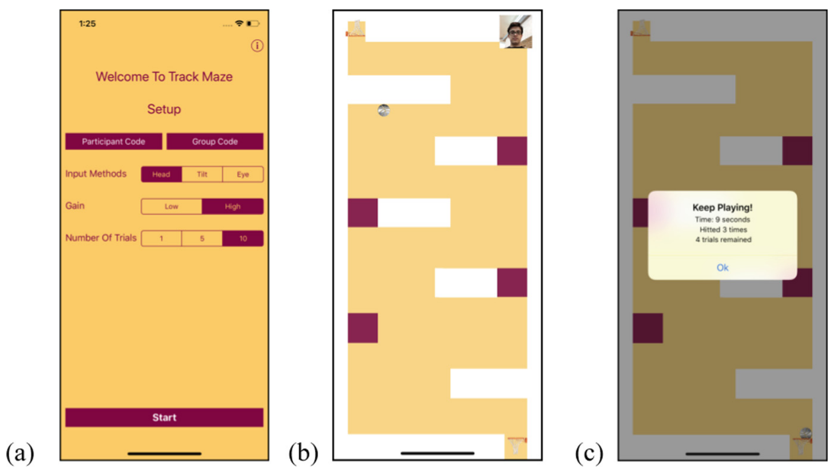
3.2 Apparatus

The experiment was conducted on an Apple *iPhone X* running iOS 11.2. The device display is 5.8 in with 2436 $\times$ 1125 pixel resolution and a density of 458 ppi. The front-facing camera of the device is a 7MP TrueDepth camera.

The experiment game application, *TrackMaze*, was developed in Swift using the iOS 11 SDK and the ARKit framework. A setup page appeared as the player opens the application. See Fig. 1a. Setup information included a participant code, a group code, the number of trials, input method, and gain.

TrackMaze uses a static maze. See Fig. 1b. The player maneuvers a virtual ball from one end of the maze to the other end. Hitting a white wall is not an error but hitting a maroon wall is. The number of wall hits, then, is the number of times the virtual ball hits a maroon wall. At the time of a wall hit, auditory and tactile feedback are provided to the player. The face of the player, via the front-facing camera, is visible at the top-right of the UI when playing with eye-tracking or head-tracking. After each trial, a result window shows the maze completion time, the number of wall hits, and the number of trials remaining. See Fig. 1c. The timing for next trial begins after clicking the OK button. The player then navigates the ball to the opposite end of the maze.

Three input methods were used: tilt, head-tracking, and eye-tracking. Obviously, there are significant implementation differences, since tilt uses the device's built-in



Input Methods: [✓ Head] Tilt Eye
Gain: Low [✓ High]
Number Of Trials: 1 5 [✓ 10]

Keep Playing!
Time: 9 seconds
Hitted 3 times
4 trials remained

Fig. 1. *TrackMaze* game sections: (a) setup page (b) maze view with face-position feedback (c) example results after a trial.

accelerometer and head- and eye-tracking use the device's front-facing camera. The eye- and head-tracking methods use services of the ARKit framework. These include SKPhysicsWorld (for tilt) and blendShapes (for head- and eye-tracking). For both head- and eye-tracking, the data provided range from 0.0 (neutral) to 1.0 (maximum) [1]. The value indicates the current position of the eyes relative to a neutral configuration (looking straight ahead). The operation of the input methods and the gain function for each are summarized in Table 1.

Table 1. Description of input methods.

Input method	Input source	Operation	Gain function
Tilt	Accelerometer	Ball velocity is controlled by the orientation of the device. The ball is stationary when the device is flat	(accelerometer data) × n n = 7.5 (low gain) n = 15 (high gain)
Head-tracking	Camera	Ball velocity is controlled by head movement as the player looks at the ball	(–1 × eye movement) × n n = 1.5 (low gain) n = 3.0 (high gain)
Eye-tracking	Camera	Ball velocity is controlled by eye movement as the head is fixed. No head movement is allowed in this mode	(+1 × eye movement) × n n = 1.5 (low gain) n = 3.0 (high gain)

3.3 Procedure

Participants were informed about the purpose of the study and given instructions on how to play the game. Their goal was to move the ball as quickly as possible to the other end of the maze without hitting the maroon walls. Participants were asked to give their consent by putting their initials on the setup page. They did the experiment in a well-lit room in a comfortable seated position so that their face was visible to the front-facing camera. They practiced the game before the data collection until they felt comfortable with the interface. Figure 2 shows a participant performing a trial using head-tracking input.

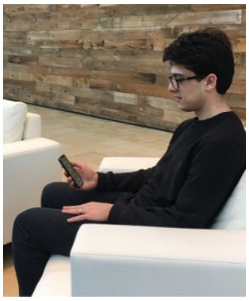


Fig. 2. A participant doing the experiment using the head-tracking input.

Six different configurations of input method and gain were tested. Participants completed five trials for each condition. They were allowed to rest before changing the input method. Each participant took about 30 min to complete the test. Participants' performance data were saved in the device for later analysis. After completing the trials, participants were asked to complete a questionnaire for data such as age, gender, fatigue, ease of use, and enjoyment.

3.4 Design

This experiment was a 3 $\times$ 2 within-subjects design, with the following independent variables and levels:

- Input method: tilt, eye-tracking, head-tracking
- Gain: low, high

Participants were divided into six groups to counterbalance the order of the input methods and gain levels and eliminate learning effects. We used a 6 $\times$ 6 balanced Latin square for this purpose. So "Group" was a between-subject independent variable with six levels.

The dependent variables were maze completion time and wall hits per trial. Qualitative measures such as ease of use, fatigue, and user enjoyment were analyzed based on questionnaire data.

The total number of trials was 12 Participants $\times$ 3 Input Methods $\times \leftarrow$ 2 Gains $\times$ 5 Trials = 360.

4 Results and Discussion

The group effect was not statistically significant for maze completion time ($F_{5,6} = 4.05$, $p > .05$) and the number of wall hits ($F_{5,6} = 3.65$, $p > .05$). So, counterbalancing had the desired effect of eliminating order effects. All statistical analyses were performed using the GoStats package.[1] Results for both dependent variables and qualitative measurements are presented below.

4.1 Maze Completion Time

The grand mean for maze completion time was 22.2 s. Tilt was the fastest input method at 12.3 s; eye-tracking was slowest at 31.8 s (2.6 $\times \leftarrow$ longer than tilt). Head-tracking was between the other two input methods at 22.5 s. The main effect of input method on the maze completion time was statistically significant ($F_{2,12} = 97.4$, $p < .0001$). By gain, the results were 24.8 s for low gain and 19.6 s for high gain (1.26$\times$ longer than low gain). The gain effect was statistically significant ($F_{1,6} = 9.37$, $p < .05$). Figure 3 presents the maze completion time by input method and gain.

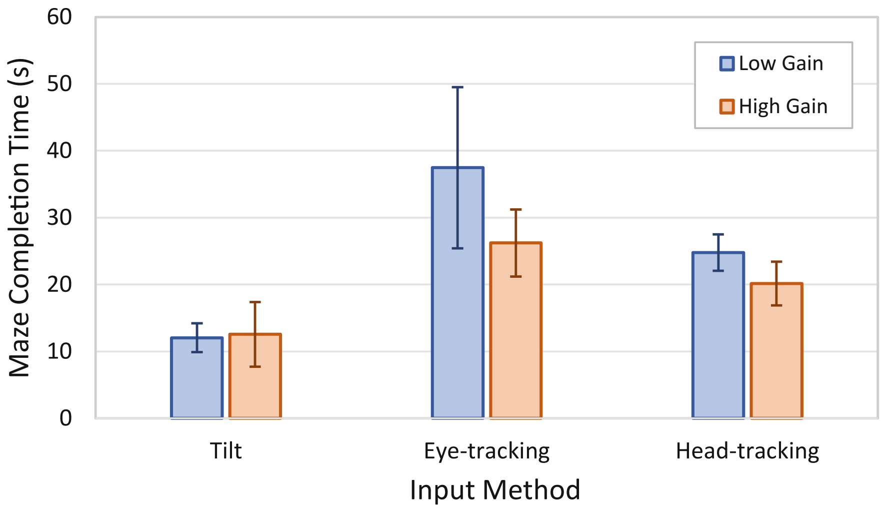


Fig. 3. Maze completion time (s) by input method and gain. Error bars show ±1 *SD*.

There was also a statistically significant Input Method $\times$ Gain interaction effect ($F_{2,12} = 4.58$, $p < .05$). The results show a considerable speed advantage for tilt-input. A Scheffé pairwise post hoc analysis showed a significant difference between all pairs except between tilt-input (low and high gain) and head-tracking high-gain.

[1] Available as a free download at www.yorku.ca/mack/HCIbook/stats_gui.

One reason for the high speed in tilt can be the presence of virtual gravity when using this input method. The vertical movement doesn't need any extra input actions and will happen automatically and correctly if the device is in the proper orientation, which is to say, tilted slightly toward the participant. The speed advantage in tilt can also be due to the participants' prior experience with mobile games using tilt input.

Participants were comfortable using head-tracking input. But, the extra input actions needed for up and down movements can be the reason for lower speed compared to tilt. There was some confusion in using head-tracking. Even though they were instructed to rotate their head about the *y*-axis for horizontal movement, participants tended to rotate their head about the *z*-axis. This rotation didn't have the desired effect on the ball. Head rotations are seen in Fig. 4.

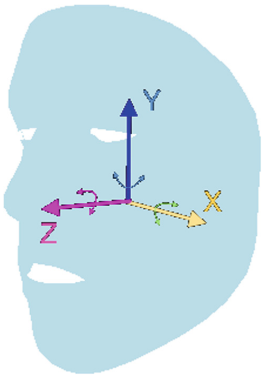


Fig. 4. Head rotation around *x*-, *y*- and *z*-axes.

The likely reasons for the poor performance in eye-tracking are as follows. The interaction was fatiguing, physically hard, and somewhat confusing. There was a simultaneous need to use the eyes for perception and control. Using eye-tracking was therefore hard for participants. Some participants couldn't move their eyes without moving their head. Head-movement while using eye-tracking input method can have an inverse effect on the ball movement. If the player keeps looking at the ball and turns their head in the desired direction, the game interprets that the player has moved their eyes in the opposite direction. The added time of using the eyes to fully see the maze environment before using the eyes to control the ball is the most likely reason for longer maze completion time in eye-tracking.

Compared to results obtained by Cuaresma and MacKenzie with the *StarJelly* game [8], our study shows a much smaller difference between head-tracking and tilt. There are perhaps two reasons: the elimination of the lag experienced in head-tracking in *StarJelly* and the practice time given to participants before doing the experiment.

We recorded the maze completion time for each trial. The chart in Fig. 5 shows the maze completion time by trial number for each input method. This chart shows that the maze completion time was less for even trials (2, 4) when using eye-tracking. This corresponds to upward ball movement. The better performance here is perhaps due to

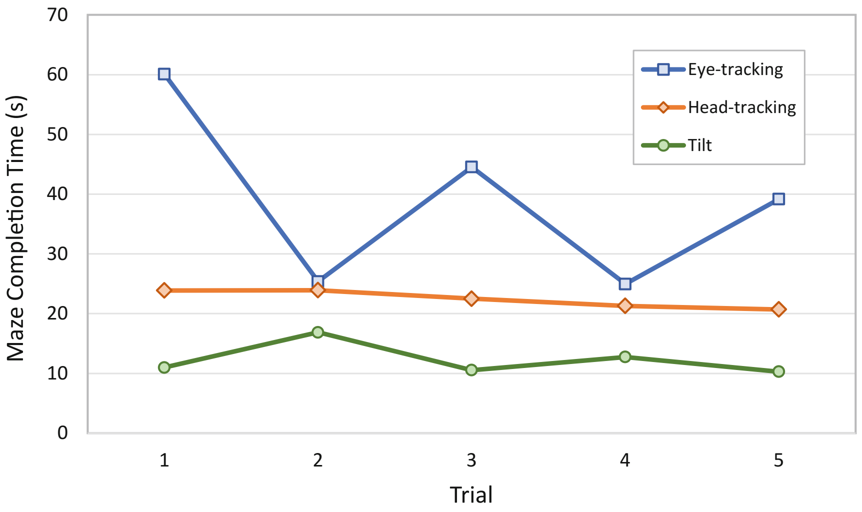


Fig. 5. Maze completion time (s) by trial. Even trials (2, 4) involve upward ball movement, odd trials (1, 3, 5) involve downward ball movement. See text for discussion.

the maze design or to participants overall tendency to hold the device lower than their eyes. So, the device camera had a more direct view while detecting the upward movement of eyes compared to downward movement.

When using the tilt input method, the maze completion time was higher for even trials. The reason can be due to the more experience with downward tilt games. Participants were supposed to rotate the device to the appropriate orientation for each trial. It can be seen that the direction of navigation in the maze didn't have an effect on head-tracking input. The maze completion time improves modestly with practice for all input methods.

4.2 Wall Hits

The grand mean for the number of wall hits per trial was 2.45. Obviously, a lower score is better. Tilt was the most accurate input method with 1.06 wall hits per trial. Eye-tracking was least accurate (4.00). Head-tracking was near the overall mean (2.30). The main effect of input method on the wall hits was statistically significant ($F_{2,12}$ = 88.3, $p < .0001$). By gain, the results were 2.39 wall hits for low gain and 2.52 wall hits for high gain. The difference here was rather small; so, unsurprisingly, statistical significance was not achieved ($F_{1,6}$ = 0.19, ns). The results by input method and gain are presented in Fig. 6. The Input Method × Gain interaction effect was not statistically significant ($F_{2,12}$ = 0.17, ns).

The high error for eye-tracking is explained as follows. Participants didn't accurately control the amount of eye movement and, instead, moved their eyes in extremes, albeit in the desired direction. This was also seen in head-tracking. Participants didn't accurately control the amount of rotation. They moved their head so intensely that their head was at times no longer visible to the front-facing camera.

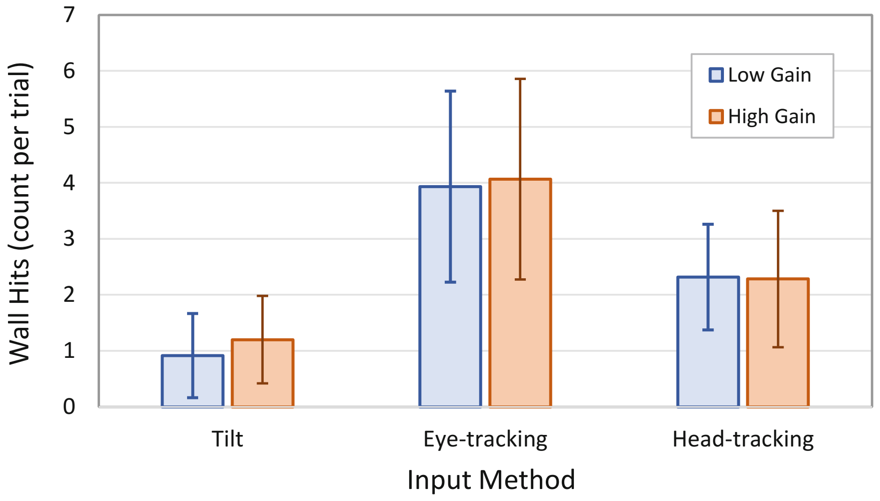


Fig. 6. Wall hits per trial by input method and gain. Error bars show ±1 *SD*.

4.3 Participant Feedback

We solicited participants' feedback on their experienced enjoyment, ease of use, and fatigue for eye-tracking, head-tracking, and tilt. Questionnaire responses were on a 5-point Likert scale, where 5 strongly agrees and 1 strongly disagrees to a statement. For ease of use and enjoyment, higher ranks are desired. For fatigue, lower ranks are favorable. A Friedman non-parametric test was used to analyze the responses, followed

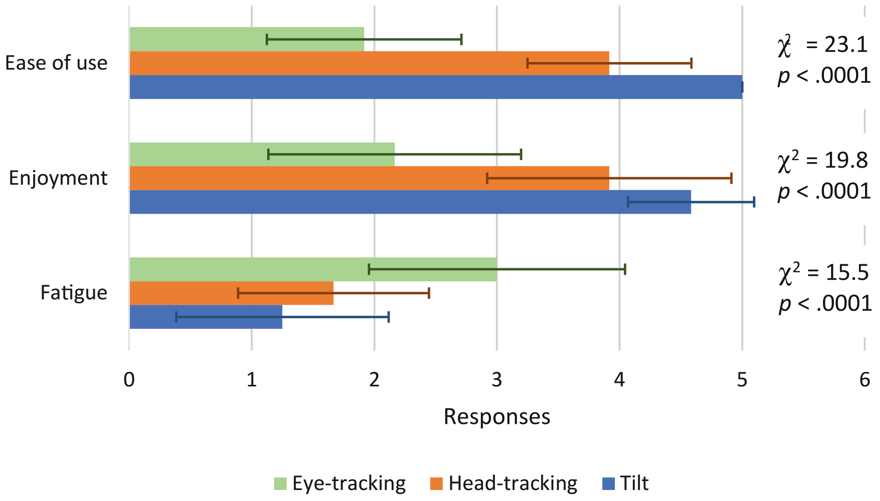


Fig. 7. Questionnaire responses. Statistical results are shown on the right. Error bars show ±1 *SD*. Higher ranks are desired for enjoyment and eye-movement but lower responses are favorable for fatigue.

by a post hoc pairwise comparisons test. All the tests showed statistically significant differences between the conditions and between pairs except for fatigue between head-tracking and tilt. Figure 7 summarizes the questionnaire responses and the statistical tests.

All participants strongly agreed that tilt was easier to use and they enjoyed it more. Participants also enjoyed head-tracking input. They found it easy to learn and amusing. They didn't experience fatigue using head-tracking and tilt.

Participants had difficulty using eye-tracking input, with which they experienced the highest fatigue and the lowest ease of use and enjoyment. Evidently, they didn't enjoy the eye-tracking interface!

5 Conclusion

In this paper, we presented a custom-made maze-like game, *TrackMaze*, for evaluating head-tracking and eye-tracking input methods and comparing them with tilt-input. *TrackMaze* tracks the user's eyes and head using the front-facing camera of the device. We implemented this game for iOS using Apple's ARKit. Each input method was tested with two gain levels. We conducted our user study with twelve participants on an *iPhone X*. Speed was assessed with maze completion time and accuracy was assessed with the number of wall hits.

Tilt had the lowest maze completion time with 12.3 s and eye-tracking had the highest maze completion time with 31.8 s (2.6 $\times$ higher than tilt). Maze completion time was 22.5 s for head-tracking. High gain, at 19.6 s, was 26% faster than low gain, at 24.8 s.

Tilt was most accurate with an average of 1.06 wall hits per trial. Eye-tracking had the worst accuracy with 4.00 wall hits per trial. Head-tracking was between eye-tracking and tilt (2.30). The number of wall hits was approximately the same for both gain levels (2.39 for low gain, 2.52 for high gain).

Participants found the eye-tracking interface fatiguing and hard, and they didn't enjoy it. They did not report fatigue using head-tracking or tilt. Overall, participants expressed a preference for tilt, then head-tracking.

We conclude that tilt is the fastest and most accurate input method for this game. But, head-tracking is a satisfactory and novel alternative.

6 Future Work

Eye tracking and heading tracking interfaces have potential for use in muscle training, physical therapy, or rehabilitation. We present these as opportunities for future work.

Vision therapy is a physical exercise for the eyes and brain and can alleviate visual problems such as lazy eyes, crossed eyes, and double vision [5]. Amblyopia, or lazy eyes, happens when the eyes have large vision differences and the brain ignores the picture from the weaker eye. This disorder can be fixed in younger children with vision therapy. Implementing vision therapy in mobile games through an eye tracking input method has potential to make vision therapy interesting for children, and perhaps adults as well.

A similar case exists for a head-tracking interface. Physical therapy or rehabilitation exercises are sometimes used to alleviate neck pain, with head movements playing a role [14]. A head-tracking mobile interface, perhaps a game, can impose certain patterns of head movement on the user, an example being the pattern required in *TrackMaze*. The pattern can be customized for a specific therapeutic goal. In the context of a game, such exercise may even be fun, adding motivation to an otherwise tedious task.

Gain – the relationship between the input control and the output display – plays an interesting role in the possible use of eye tracking or head tracking for therapy or rehabilitation. Under a high-gain setting, a desired pattern of cursor or object manipulation on a screen requires relatively little movement of the eyes or head. This is useful for training, practice, or perhaps in the initial interactions where only minimal movements are desired. As interaction (i.e., exercise) continues, it may be desired to impose greater eye or head movement on the individual seeking therapy or rehabilitation. This is achieved by lowering the gain setting. A lower gain requires correspondingly more movement in the eyes or head to achieve the same on-screen movement in a cursor or object. Lower the gain even more, and even more movement is required. Here, we see an interesting interaction between eye or head tracking and an interface's gain setting for therapeutic or rehabilitation purposes.

Acknowledgments. We thank all participants who took part voluntarily in this experiment.

References

1. Apple, Inc., ARFaceAnchor. https://developer.apple.com/documentation/arkit/arfaceanchor/. Accessed 23 Jan 2018
2. Almeida, R., Veloso, A., Roque, L., Mealha, O.: The eyes and games: a survey of visual attention and eye tracking input in video games. In: Proceedings of the 2011 Brazilian Symposium on Games and Digital Entertainments - SBGames 2011, pp. 1–10. IEEE, New York (2011)
3. Bergstrom, J.R., Schall, A.: Eye Tracking in User Experience Design. Elsevier, Amsterdam (2014)
4. Cairns, P., Li, J., Wang, W., Nordin, A.I.: The influence of controllers on immersion in mobile games. In: Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems - CHI 2014, pp. 371–380. ACM, New York (2014)
5. Carvelho, T., Allison, R.S., Irving, E. L.: Computer gaming for vision therapy. In: Proceedings of Virtual Rehabilitation, pp. 198–204. IEEE, New York (2008)
6. Constantin, C., MacKenzie, I.S.: Tilt-controlled mobile games: velocity-control vs. position-control. In: Proceedings of the 6th IEEE Consumer Electronics Society Games, Entertainment, Media Conference - IEEE-GEM 2014, pp. 24–30. ACM, New York (2014)
7. Corcoran, P.M., Nanu, F., Petrescu, S., Bigioi, P.: Real-time eye gaze tracking for gaming design and consumer electronics systems. Consum. Electron. **58**, 347–355 (2012)
8. Cuaresma, J., MacKenzie, I.S.: A comparison between tilt-input and facial tracking as input methods for mobile games. In: Proceedings of the 6th IEEE Consumer Electronics Society Games, Entertainment, Media Conference - IEEE-GEM 2014, pp. 70–76. IEEE, New York (2014)

9. Cuaresma, J., MacKenzie, I.S.: exploring navigation and selection methods for facial tracking. In: Antona, M., Stephanidis, C. (eds.) UAHCI 2017. LNCS, vol. 10278, pp. 403–416. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-58703-5_30
10. Gorodnichy, D., Malik, S., Roth, G.: Nouse 'use your nose as a joystick or a mouse'. In: Proceedings of the International Conference on Vision Interface - VI 2002, pp. 1–11. National Research Council of Canada, Ottawa (2002)
11. ISO: Ergonomic requirements for office work with visual display terminals (VDTs) - part 9: Requirements for non-keyboard input devices (ISO 9241-9), International Organisation for Standardisation Report Number ISO/TC 159/SC4/WG3 N147 (2000)
12. Israel, L., Neumayr, T.: Developer earnings from the App Store top 70 billion. https://www.apple.com/newsroom/2017/06/developer-earnings-from-the-app-store-top-70-billion/. Accessed 10 Nov 2017
13. Israel, L., Saffer, S.: App Store shatters records on new year's day. https://www.apple.com/newsroom/2017/01/app-store-shatters-records-on-new-years-day.html/. Accessed 10 Nov 2017
14. Jull, G.A., Falla, D., Vicenzino, B., Hodges, P.W.: The effect of therapeutic exercise on activation of the deep cervical flexor muscles in people with chronic neck pain. Manual Ther. **14**, 696–701 (2009)
15. MacKenzie, I.S., Teather, R.J.: FittsTilt: the application of Fitts' law to tilt-based interaction. In: Proceedings of the 7th Nordic Conference on Human-Computer Interaction - NordiCHI 2012, pp. 568–577. ACM, New York (2012)
16. Majaranta, P., Bulling, A.: Eye tracking and eye-based human–computer interaction. In: Fairclough, S.H., Gilleade, K. (eds.) Advances in Physiological Computing. HIS, pp. 39–65. Springer, London (2014). https://doi.org/10.1007/978-1-4471-6392-3_3
17. Medryk, S., MacKenzie, I.S.: A comparison of accelerometer and touch-based input for mobile gaming. In: Proceedings of the International Conference on Multimedia and Human-Computer Interaction - MHCI 2013, pp. 117.1–117.8. ASET Inc., Ottawa (2013)
18. Nguyen, D.: Understanding perceived enjoyment and continuance intention in mobile games, Ph.D. thesis, Aalto University, Espoo, Finland (2015)
19. Peshkovskaya, A.G., Babkina, T.S., Myagkov, M.G.: The socialization effect on decision making in the Prisoner's Dilemma game. PLoS ONE **12**, e0175492 (2017)
20. PocketGamer, Apple: Most popular App Store categories (2017). https://www.statista.com/statistics/270291/popular-categories-in-the-app-store/. Accessed 10 Nov 2017
21. Ramcharitar, A., Teather, R.J.: A Fitts' law evaluation of video game controllers: thumbstick, touchpad, and gyrosensor. In: Proceedings of the ACM SIGCHI Conference on Human Factors in Computer Systems - CHI 2017, pp. 2860–2866. ACM, New York (2017)
22. Roig-Maimó, M.F., MacKenzie, I.S., Manresa, C., Varona, J.: Evaluating Fitts' law performance with a non-ISO task. In: Proceedings of the 18th International Conference of the Spanish Human-Computer Interaction Association, pp. 51–58. ACM, New York (2017)
23. Teather, R.J., MacKenzie, I.S.: Comparing order of control for tilt and touch games. In: Proceedings of the 10th Australasian Conference on Interactive Entertainment - IE 2014, pp. 1–10. ACM, New York (2014)
24. Teather, R.J., MacKenzie, I.S.: Position vs. velocity control for tilt-based interaction. In: Proceedings of Graphics Interface 2014 - GI 2014, pp. 51–58. Canadian Information Processing Society, Toronto (2014)