COSC411 Assignment

Dion Woolley
Department of Computer Science and
Software Engineering
University of Canterbury
dmw99@uclive.ac.nz

Andy Cockburn
Department of Computer Science and
Software Engineering
University of Canterbury
andrew.cockburn@canterbury.ac.nz

ABSTRACT

This paper explores the use of a hybrid touchpad interaction technique through the use of a physical augmentation to existing touchpads, called RubberEdge. It aims to reduce clutches and the time take for a user to navigate between pointer targets when compared to conventional touchpad transfer functions, specifically Constant Gain and Acceleration. We found that our implementation of RubberEdge was successful at reducing the number of clutch movements when compared to Constant or Acceleration functions by (14% - 42%), depending on the distance of the movement. However, there were no significant results found for selection time of targets. The data suggests that the current implementation of the RubberEdge transfer function is worse than existing Constant and Acceleration functions. This is due to flaws in the experiment implementation, with time permitting would have been resolved before the experiment was conducted.

Author Keywords

rubberedge; pointing; clutching; elastic; isotonic; COSC411

INTRODUCTION

Clutching occurs when a users required more than one finger action to perform a pointer movement. In a classic example of this a user wishes to move the pointer from one end of the screen to the other but runs out of room on their touchpad to complete the movement in a single action, so they are required to reset their finger position a number of times to complete the movement. Because multiple movements are required time is wasted between each clutch invocation, compared to if the pointer movement could be achieved in one motion.

BACKGROUND AND RELATED WORK

This work is based on the initial work done in the RubberEdge paper[13], where this paper is evaluating a more advanced version of their prototype design for RubberEdge. Conditions have changed in touchpads and screen interaction since the paper was published. Specifically the introduction of high-resolution 4k displays and display scaling as well as newer pointer acceleration curves and larger touch pads.

From the perspective of a touchpad user, we still using clutching in our everyday interactions. So despite the growing size of trackpads clutching still occurs. Whether or not clutching is bad is debatable [16] argues that pointer movements are possible without clutching but is detrimental for target selection time and induces more errors. So some clutching is beneficial, and it is not solely caused by a lack of touchpad space.

Over the years touch pads have increased in their overall size, this is likely due to the increasing availability of gestures. However, it is noted that there is no significant improvement in the usability for pointing tasks when using a larger touchpad over a smaller one[10]. The size of the touchpad may impact the type of gestures that can be used.

The density of screens has increased, with it now being common to see laptops with resolutions of 4k (3840 x 2160) or greater on the market. There is also a larger variety of screen aspect ratios, with 16:9 remaining dominate, also significant is 16:10 on MacBooks[3] and 3:2 on Microsoft's Surface[6] line of products having a significant share of the market. However, the increased resolution is offset by the use of display scaling; this means the effective resolution is a lot lower than the native resolution, often it is between 1080p and 1440p.

The current methods for user input are; fixed ratio of display movement to control movement Control-Display (CD) (a scaling factor, 1mm movement on the touchpad equates to x number of pixels moved in the same direction). The alternative is pointer cursor acceleration, which depends on the velocity of the input on the touchpad. A low velocity equates to a low CD gain this improves pointer precision and a high CD gain at high velocity, to cover a large distance while minimise clutching.

For hybrid position and rate control techniques, it was found that without haptic feedback transitioning between position and rate mode was difficult to distinguish. However, elastic feedback is well suited to rate control.

Three types of input devices, isotonic, isometric and elastic. Isotonic free moving uses position input (mouse, touchpad). Isometric devices (TrackPoint)[20] do not move and use force for input. In between these two device types are elastic devices, which use a mix of force and position for input.

RubberEdge is a hybrid between isotonic and isometric, where there is an isotonic zone for precise input, and an isometric or elastic zone, for use when faster pointer movement is required, which provides force feedback relative to the acceleration of the pointer.

RUBBEREDGE DEVICE

This research paper aims to improve upon an existing RubberEdge implementation. The paper[13] that this research draws inspiration from only empirically examines their implementation of the RubberEdge as shown in Figure 1. The experimental results show in the paper were gathered with a Phantom Omni haptic device[1], this apparatus mocked the

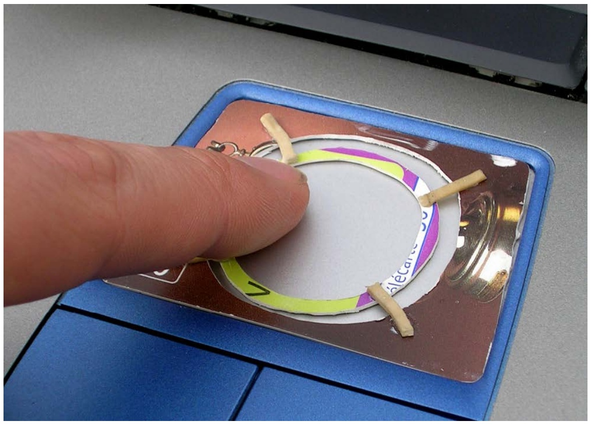

Figure 1. The RubberEdge interface from the previous study on RubberEdge [13]

feeling of RubberEdge to the participant. Although the interaction was not quite the same as it would be with a real touchpad, as the participant would not get the same feedback from physically pushing up against a ring over feeling some resistance. This research started with creating a RubberEdge apparatus using the existing design from the source paper.

Interface Design

The initial design called for a device which would sit neatly over a touchpad on a laptop. Measurements of the laptop's touchpad were 105mm x 80mm, after accounting for structural integrity of the edges of the interface, the design left a circular area with a 30mm radius, for the isotonic zone and around 7.5mm of radius for the elastic zone, the actual available area is less due to the collapsing spring arms of the final design. This left a usable isotonic area of 2827mm^2

In the first iteration of the design, an initial attempt at replicating the existing design was attempted by laser cutting an outer frame and inner ring out of acrylic. This design was abandoned in favour of a solution suggested by the mechanical technician that helped with the laser cutting and subsequent 3D printing. They suggested that a solution using spring arms might be an effective alternative, a design was quickly drafted, and cut from acrylic. The design showed promise but unfortunately, the acrylic material proved to be too rigid, and the spring arms would break under force.

An alternative method of creating the interface was to use a composite 3D printer; this printer can blend multiple polymers to change the ply-ability of the plastic. A hybrid design was used where the outer frame and inner ring was printed with a rigid material, and the spring arms printed with an elastic type polymer, this design can be seen in Figure 2. The proprieties of the elastic material used are not freely available as the technology is proprietary to the manufacturer of the 3D printer, and material. The material is simply described as Shore 85, which only indicates the hardness of the material. Further tests should be conducted to evaluate the elasticity of the material to see if it falls in line with design guidelines present in this study[12].

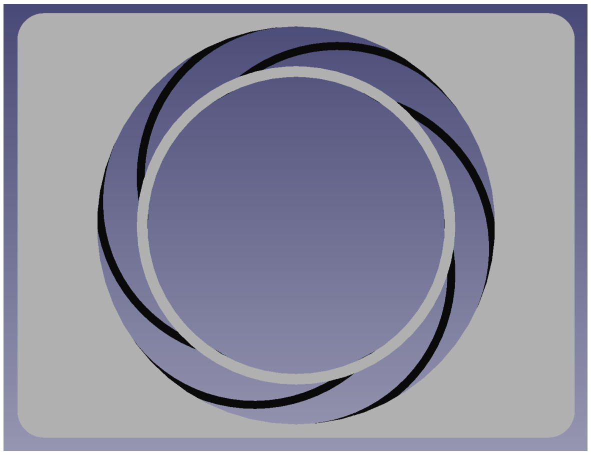

Figure 2. First design for the composite 3D printed RubberEdge interface.

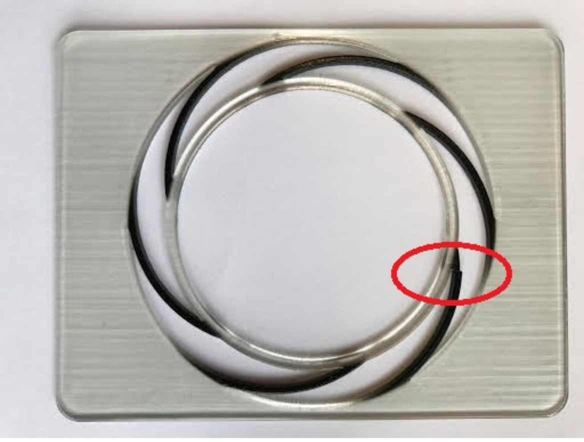

Figure 3. As shown in the red circle, a major problem of the first spring arm design was the spring arms breaking after a period of use.

Problems occurred with the initial 3D printed design, the key of which was the spring arms breaking after a period of use. The arms broke due to them being stretched too far when the inner ring was pushed up against the wall of the outer frame. The next and final iteration of the design solved this problem by changing the spring arm design to allow the arms to extend and contract, as well as increasing the thickness of the spring arms from 1.5mm to 2mm; the final design can be seen in Figure 4.

Interface Data

Initially, in order to read the input from a participant using the RubberEdge device on the touchpad of a laptop the libpointing library[17][11] was used. However, after an investigation into the source code and the underlying Window's Application Programming Interfaces (APIs)[5] there was no way to get the absolute position of a participants finger on the touchpad using libpointing. This is despite a method called getPosition() as part of the libpointing API.

The authors of the reference paper used a touchpad driven by Synaptic's drivers, which could be installed on the laptop. At the time, Synaptic offered an SDK to interact with their

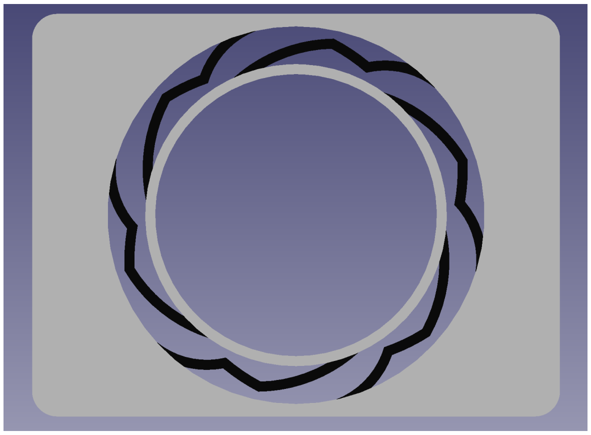

Figure 4. The final design of the RubberEdge interface used in the experiment.

drivers, however, it appears this is no longer offered on their website, and clones available on Github have not been kept up to date with the latest drivers.

The existing drivers on the laptop are for Windows precision touchpads. This driver provides two interfaces that a developer can subscribe to[4]. The first interface provides relative pointer events and is the one used by the libpointing library. The second interface positions itself as a generic touchscreen[8], this is used for gesture recognition and does provide the absolute coordinates for interactions with the touchpad.

Due to time constraints we were not able to use the newer driver interface, and instead found a compromise to the method of input as seen in Section

EXPERIMENT

Goals and Hypothesis

H1: RubberEdge will reduce clutch invocations required to traverse a pointer to a target when compared to conventional Transfer Functions, Constant Gain and Acceleration.

H2: The time taken to traverse a pointer to a target using the RubberEdge Function will be less than existing Constant and Acceleration Transfer Functions.

Apparatus

The laptop used for the experiments was a Dell XPS 15 9560[9], it featured a 15 inch display with a resolution of 3840px x 2160px, giving it 282.4 Pixels Per Inch (PPI). The Windows 10 operating system, with display scaling set to 175% for an effective resolution of 2194px x 1234px was used on the laptop.

As discussed in Section problems were encountered when getting the absolute position of a participants finger on the touchpad, due to time constraints a compromised was settled on where an iPhone 7 Plus was used, with the RubberEdge interface adhered to the screen with double sided tape, as shown in Figure 5. A separate web page was then created, that would be navigated to on the phone, from which touch input could be transmitted to the laptop hosting a web server over web sockets.

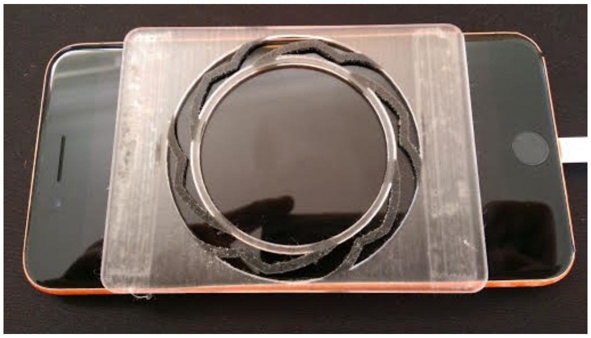

Figure 5. The RubberEdge device, adhered to the screen of an iPhone 7 Plus

Experiment Interface

An experiment based on Fitts' law was deemed to be an adequate way of testing the conditions. The interface is based off the Visualising Fitts Law[19] web page, which follows the ISO 9241-9 standard[2]

The web interface used in the experiment was built using the Polymer framework[7].

Participants were first presented with an instructions page outlining how the RubberEdge device is intended to be used. After completing a short pre-experiment survey, they calibrate the isotonic and elastic zones for their finger. The calibration processed involved tracing with one finger around the edge of the inner ring of the RubberEdge device, without putting pressure on the ring. Calibration is important as participants had different sized fingers and would orient their hands differently relative to the device.

To calculate the isotonic and elastic zone a circle fitting algorithm[15] was applied which takes a set of points and finds the midpoint and radius of a circle that best represents those points. This circle defines the isotonic zone, points falling outside the circle are considered to be in the elastic zone.

From this entries in and out of the elastic zone can be tracked by measuring the distance between a given point and the centre of the defined isotonic zone.

Task and Stimuli

The Fitts' law task is to move the pointer in the indicated circle, upon which release the finger from the touchpad a new target would be generated. Nine targets are spaced equally around a central point, which is in the centre of the screen. Targets are selected in an opposing order, which means the user has to move the pointer and equal distance to select each target.

The distances selected based on relative pixel coordinates in the web browser were 500px, 750px, 1000px when translated to mm this works out to be 79mm, 118mm, 157mm respectively. The selected widths were 25px, 50px, 75px and translated they are 3.9mm, 7.9mm, 11.8mm respectively.

After completing the calibration phase, participants are trained on each function, Constant, Acceleration, RubberEdge with a fixed distance (118mm) and width (7.9mm) of the targets, an

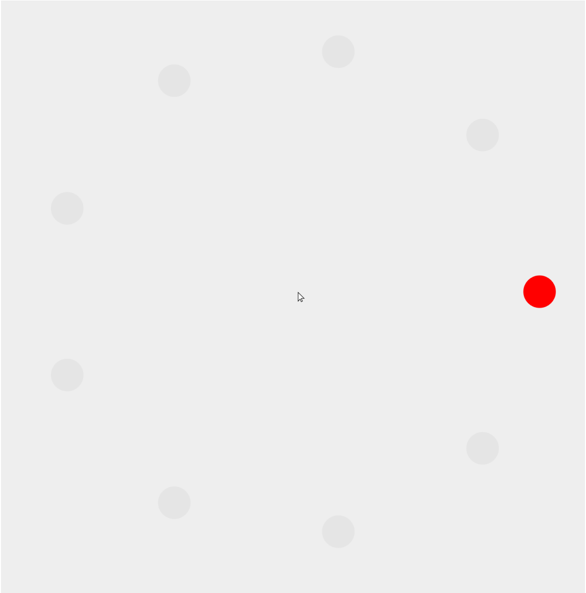

Figure 6. The Fitts' Law interface used by participants, with the target highlighted in red and possible targets in dark gray.

example of the interface presented to participants is seen in Figure 6

Once training is complete, participants are presented with one of the three transfer functions and are asked to complete the task for all distance and width combinations. After which the combinations are repeated for the other transfer functions, with a 30 second break between each transfer function. The order of the distance and width combinations are randomised for each transfer function and each participant, using the Durstenfeld shuffle algorithm.

After completing the experiment, participants are asked to complete a survey evaluating each transfer function.

Interface problems

A major flaw of the experiment was the transmission of data between the laptop and phone. Due to the instability of wireless communications, the pointer would move in a slightly staggered motion, often inducing a noticeable lag, this did affect the observed performance of the participants in the study. A solution to this problem would be changing the communication method to be over USB; however, the development time involved made this solution unfeasible.

Participants

Seven people (3 female, 4 male) participated in the experiment; participants were in the age range of (15 - 25) years, except for one in the age range of (45 - 55) years. Participants were asked the average number of hours spent using a touchpad each day; responses were varied from not using a touchpad at all to greater than 8 hours per day. Operating system use varied between participants, with five using Windows and one for each MacOS and Linux. Preference for existing pointer acceleration on touchpads also varied, four participants, preferred acceleration, one did not, and two either had no preference or did not know what pointer acceleration was.

Design

A within-subjects design was used. The independent variables were Transfer Function (Constant, Acceleration, RubberEdge), target Distance (D_S - 79mm, D_M 118mm, D_L - 157mm) and target Widths (W_S - 11.8mm, W_M - 7.9mm, W_S 3.9mm) The dependent variables measured were the time between selecting targets and the number of clutches performed to select a target.

Each distance and width combination is pre-set with a sequence of nine targets for the user to select. With three different distances and three widths makes nine unique distance and width combinations. This gave seven Fitts' Indices of Difficulty (ID)[14]

For the Constant transfer function, a CD gain of 2 was used, this was to encourage clutching and was similar to the default system transfer function provided by Windows 10. The acceleration transfer function was the default provided by Windows 10. The transfer function for RubberEdge in the isotonic was set to a CD gain of 2. In the elastic zone, the average velocity of the last three data points from the isotonic zone is used as the initial velocity, from then an acceleration multiplier relative to the distance away from the edge of the isotonic and elastic zone is applied. When the participant is stationary in the elastic zone the pointer is updated at a rate of 400Hz, which is equivalent to the update rate of the touchpad of the laptop. A limit is applied on the maximum velocity that can be obtained by the pointer; this is to prevent the pointer running away from the participant and make the behaviour more predictable.

The order of the transfer functions was counterbalanced between participants using the Latin Squares method. This is to avoid any asymmetric skill transfer induced by training.

The experiment lasted approximately 15 minutes.

The total number of trials conducted are:
7 Participants × 3 Transfer Functions × 3 Distances × 3 Widths × 9 targets
= 1701 total trials

RESULTS

The dependent variables were selection time and the number of clutches.

Selection Time

Selection time is the time between each subsequent target selection. This means that the first target the participants selects for each Transfer Function is not included in the results. Outliers for trials over 20 seconds were removed from the data.

Repeated measures analysis of variance using Mauchly's Test for Sphericity, showed that the order of Transfer Functions presented to the participant had no significant effect on movement time, so there was no significant asymmetric skill transfer, this indicated that the within-subjects design was appropriate. There was a significant main effect for the for the Transfer Function on selection time ($F_{2,12} = 10.3$, $p < 0.005$) with the Constant and Acceleration outperforming the RubberEdge

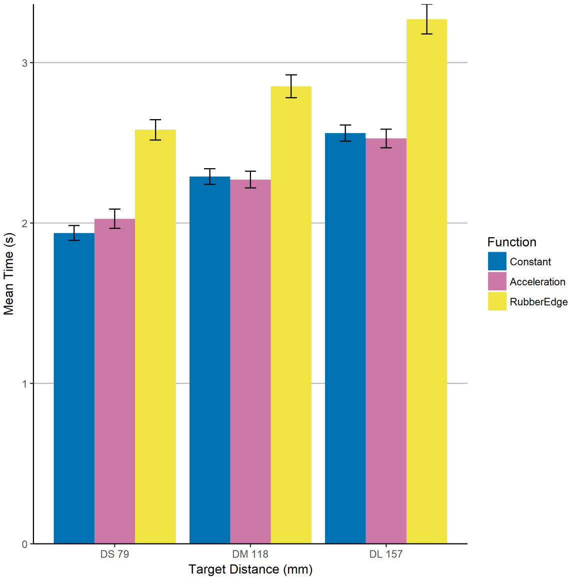

Figure 7. Selection Time, Transfer Function x Distance Interaction

Transfer Function by 21.8%, this can be seen across all distances in Figure 7. There was also a significant main effect for Distance ($F_{2,12}$ = 30.8, p < 0.0001) on selection time. The interaction between Transfer Function x Distance was not significant ($F_{2,24}$ = 0.70, p = 0.59). Therefore we cannot accept **H2**.

Pair-wise comparisons using Bonferroni found that there was a significant difference between RubberEdge and the Constant and Acceleration Transfer Functions. Another pair-wise test also found a significant difference between each distance (D_S, D_M, D_L)

Fitts' Law Analysis

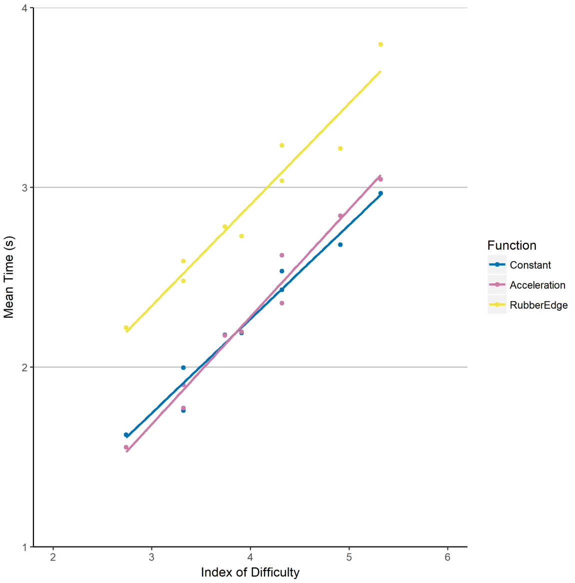

Figure 8. A comparison between the Index of difficulty and mean time, between different transfer functions.

A significant difference for Fitts' law between distances was not found (Figure 8). Whereas the previous study[13] found that the distance had a significant effect for indexes of the same difficulty. Why the results found in this study differ, is likely because the distances measured against in the previous study were much larger, 172, 344, 688mm. Compared to 79, 118, 157mm for this study. The reason that smaller distances were chosen is that they align closer to distances that users are likely to demonstrate on a laptop device. Specifically, 157mm was around the limit for the distance from the centre point of the laptop screen to the upper or lower edge, see section for details about the laptop used.

Function	a	b	r^2
Constant	0.20	0.52	0.99
Acceleration	-0.08	0.59	0.99
RubberEdge	0.64	0.56	0.95

Table 1. Fitts' Law regression values for different transfer functions, where a is the intercept of the regression line, b is the slope, r^2 is the fitness.

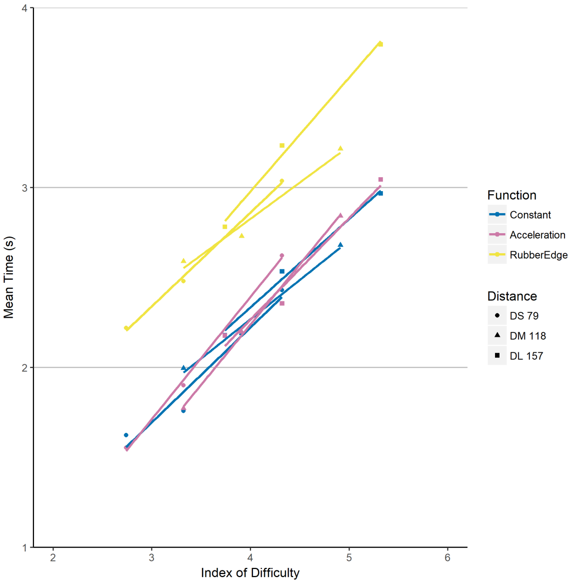

Figure 9. A comparison between the Index of difficulty and mean time, between different transfer functions, where each distance is separated into its own group.

Table 1 also shows that the r^2 values for each Transfer Function are within acceptable values. For interest, a Figure showing the different distances is also supplied (Figure 9). This shows that for each Transfer Function the distances are neatly grouped together.

Clutches

Clutches are the number of times a participant removes their finger from the touchpad to continue the task of moving the pointer to a target. This is recorded on a per trial basis.

The order of the Transfer Functions presented to the participants had no significant effect on clutch invocations as found using Mauchly's Test for Sphericity. There was a significant

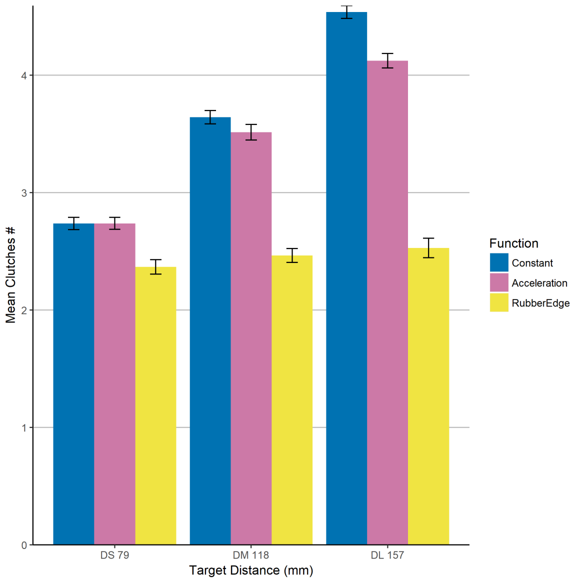

Figure 10. Transfer Function x Distance Interaction on clutch invocations.

main effect for the Transfer Function on clutch invocations ($F_{2,12}$ = 23.4, p < 0.0001) with the RubberEdge Function outperforming the Constant and Acceleration Function at all Distances. A significant main effect was also present for Distance on clutch invocations ($F_{2,12}$ = 114.2, p < 0.0001). More importantly, there was a significant Transfer Function x Distance interaction ($F_{2,24}$ = 32.2, p < 0.0001), as shown in Figure 10. RubberEdge outperforms both the Constant and Acceleration functions at all Distances (D_S, D_M, D_L) by (13.6%, 31.1%, 41.6%) respectively. A pair-wise comparison confirms this finding with both Acceleration vs RubberEdge and Constant vs RubberEdge having p < 0.0001. We therefore accept **H1**.

Participant Feedback

Participants were asked the following question for each Transfer Function at the end of the experiment. 'Was the interface frustrating to use?', 'Was the interface easy to use?' and 'The interface felt accurate?'. Only the question about accuracy was found to be significant ($x_r^2 = 6.42$, df = 2, $p =< 0.05$). Where 5 out of 7 participants agreed that the Constant Transfer Function was accurate, 4 out of 7 agreed that Acceleration was accurate and 4 out of 7 participants found that RubberEdge not to be accurate.

Participants commented the noticeable lag in the pointer see section , which details the issue.

DISCUSSION

Our experiment confirmed the first hypotheses **H1**, but failed to confirm **H2**

Selection Time

H2 could not be confirmed: RubberEdge Transfer Function would reduce the time taken to traverse a pointer to a target, as there was no significant interaction found between Distance and Transfer Function for selection time. A Log transformation was tried on the data to reduce noise, but this did not make the results significant. This is likely caused by the lag induced by the interface. For the elastic zone, the transfer function used was a naive approach, when compared to the physics-based approach detailed in the source paper[13]. Some participants also commented on the fact that the pointer would accelerate to a peak velocity that felt as though they were not in control. This is in spite of the peak limitation of the maximum velocity and suggests that an investigation into the maximum velocity in the elastic zone is needed.

Clutches

Our results confirmed **H1**: Clutch invocations for the RubberEdge Transfer Function would be less than Constant and Acceleration. When using the RubberEdge function participants would often use a technique where they would enter the elastic zone and slightly overshoot the target, and then proceed to use the isotonic zone to perform the remaining traversal to the target.

Apparatus

One of the key issues with the physical apparatus is the limiting of the usable space for the touchpad; this is not so much of an issue for pointing tasks as 'users tend to focus input in the centre of the touchpad.'[10]. It was also found that 'increased touchpad size did not have a significant effect on the input area that people used for pointing or gestural input.' Although there is evidence that the reduced usable size will not significantly impact a user, the size of the isotonic zone does mean that four finger gestures cannot be used. A better solution for RubberEdge would have the elastic zone built into the touchpad. To induce the feeling of resistance a different texture[18] could be used, or haptic feedback which emulates a sensation of moving your finger up a slope. However, this type of targeted haptics does not currently exist.

CONCLUSION

RubberEdge enables users to reduce the clutches required to reach pointer targets but maintains the benefits of position control for precise movements. The results show that RubberEdge decreases the number of clutches (14% - 42%) for pointer targets at short and long distances. However, the implementation of RubberEdge had a negative effect on selection time; this was primarily due to an inadequate elastic function and lag in the interface. Future iterations of RubberEdge hold promise if the selection time issue can be addressed.

FUTURE WORK

If the experiment was to be repeated, consider using a Macbook Pro touchpad, where access to the API for absolute finger position on the touchpad might be easier. This would avoid the use of having to use an iPhone as a touch interface and would eliminate the lag with the existing solution; if this was not possible then consider communicating with the phone over USB. Changing the transition function of the elastic zone for RubberEdge to more closely match the function described in the source paper[13], might help to obtain results that decrease the negative effect on selection time. The study was also limited in the number of participants, increasing this would likely help obtain significant results for selection time.

ACRONYMS

API Application Programming Interface. 2, 3, 5

CD Control-Display. 1, 4

ID Indices of Difficulty. 4

PPI Pixels Per Inch. 3

REFERENCES

1. Geomagic Touch (formerly Geomagic Phantom Omni) Overview. (????). http://www.geomagic.com/en/products/phantom-omni/overview
2. ISO 9241-400:2007 - Ergonomics of human–system interaction – Part 400: Principles and requirements for physical input devices. (????). https://www.iso.org/standard/38896.html
3. MacBook Pro - Technical Specifications - Apple (NZ). (????). https://www.apple.com/nz/macbook-pro/specs/
4. Precision Touchpad Device Implementation Guide. (????). https://docs.microsoft.com/en-us/windows-hardware/design/component-guidelines/windows-precision-touchpad-implementation-guide
5. Raw Input (Windows). (????). https://msdn.microsoft.com/en-us/library/windows/desktop/ms645536(v=vs.85).aspx
6. Surface Laptop Technical Specs – Power &amp; Beauty. (????). https://www.microsoft.com/en-au/surface/devices/surface-laptop/tech-specs
7. Welcome - Polymer Project. (????). https://www.polymer-project.org/
8. Windows Touch Input (Windows). (????). https://msdn.microsoft.com/en-us/library/windows/desktop/dd317321(v=vs.85).aspx
9. XPS 15 9560 High Performance Laptop with InfinityEdge Display | Dell New Zealand. (????). http://www.dell.com/nz/p/xps-15-9560-laptop/pd
10. A. Avera, C. Harper, N. Russi-Vigoya, and S. Stoll. 2016. Effects of Touchpad Size on Pointing and Gestural Input Area and Performance. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 60, 1 (9 2016), 825–829. DOI: http://dx.doi.org/10.1177/1541931213601188
11. Géry Casiez and Nicolas Roussel. 2011. No More Bricolage!: Methods and Tools to Characterize, Replicate and Compare Pointing Transfer Functions. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. ACM, New York, NY, USA, 603–614. DOI: http://dx.doi.org/10.1145/2047196.2047276
12. Géry Casiez and Daniel Vogel. 2008. The effect of spring stiffness and control gain with an elastic rate control pointing device. In *Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems - CHI '08*. ACM Press, New York, New York, USA, 1709. DOI: http://dx.doi.org/10.1145/1357054.1357321
13. Géry Casiez, Daniel Vogel, Qing Pan, and Christophe Chaillou. 2007. RubberEdge. In *Proceedings of the 20th annual ACM symposium on User interface software and technology - UIST '07*. ACM Press, New York, New York, USA, 129. DOI: http://dx.doi.org/10.1145/1294211.1294234
14. I. Scott MacKenzie and I. Scott. 1992. Fitts' Law as a Research and Design Tool in Human-Computer Interaction. *Human?Computer Interaction* 7, 1 (3 1992), 91–139. DOI: http://dx.doi.org/10.1207/s15327051hci0701{_}3
15. L Maisonobe. 2007. Finding the circle that best fits a set of points. (2007). https://www.spaceroots.org/documents/circle/circle-fitting.pdf
16. Mathieu Nancel, Daniel Vogel, and Edward Lank. 2015. Clutching Is Not (Necessarily) the Enemy. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*. ACM Press, New York, New York, USA, 4199–4202. DOI: http://dx.doi.org/10.1145/2702123.2702134
17. Nicolas Roussel, Géry Casiez, and Izzatbek Mukhanov. 2017. libpointing. (2017). https://github.com/INRIA/libpointing
18. Sameerajan Suresh Babu. 2011. Effects of Laptop Touchpad Texturing on User Performance and Usability Assessments. (2011). https://repository.lib.ncsu.edu/handle/1840.16/7433
19. Simon Wallner. 2017. An Interactive Visualisation of Fitts's Law with JavaScript and D3. (2017). http://simonwallner.at/ext/fitts/
20. Shumin Zhai. Human Performance in Six Degree of Freedom Input Control. (????). http://mobile.shuminzhai.com/papers/ZhaiPhDthesis.pdf

ACKNOWLEDGEMENTS

I would like to acknowledge David Read, for help with the printing and cutting of the RubberEdge interface. Andy Cockburn for putting up with me :)

APPENDIX

Github link for source code of project https://github.com/Awarua-/RubberEdge2