Proceedings of the 2nd International Conference on Human-Computer Interaction
Prague, Czech Republic, August 14-15, 2014
Paper No. XXX (The number assigned by the OpenConf System)

A Dynamic Adjustment of Control-Display Gain Based on Curvature Index

Amur Al Manji , Claire Davies
The University of Auckland, Department of Mechanical Engineering
Auckland, New Zealand
aalm759@aucklanduni.ac.nz; c.davies@auckland.ac.nz

Robert Amor
The University of Auckland, Department of Computer Science
Auckland, New Zealand
trebor@cs.auckland.ac.nz

Abstract -*The purpose of this study was to evaluate an algorithm intended to enhance the performance of point-and-click computer tasks for youths with cerebral palsy using a standard mouse. The curvature index–based algorithm for dynamic adjustment of control-display gain showed experimentally better performance during primary submovement, but worse performance during secondary correction submovements (higher number of submovements and longer movement time) for both typically developed youths and youths with cerebral palsy. It also showed better performance of average speed and maximum speed compared with Windows default settings. Furthermore, the average movement time, error rate, and overshoot rate for both typically developed youths and youths with cerebral palsy are higher using the curvature index–based algorithm.*

Keywords: Control-display gain, curvature index, point and click, target agonistic algorithm, cerebral palsy

1. Introduction

Point and click is a vital input used to access computer files and resources. Tool performance is important to evaluate the computer access efficiency. The performance is affected by many factors, such as fatigue, accuracy, position, hand-eye coordination, and so on, but a continual factor such as upper limb impairments caused by cerebral palsy (CP) must be overcome or effectively reduced. CP is a common cause of motor dysfunction affecting children and adults [1] and is an umbrella term for a group of disorders of movement and/or posture, which include spasticity, dyskinesia, ataxia, and hypotonia [2]. The prevalence is approximately 3.1 per 1,000 births [1].

1.1. HCI Design

Three major components are required to build a sophisticated computer system: interaction methods, interaction devices, and interface design [3]. A variety of interaction methods and devices have been developed to improve physical access to computers. These range from accessibility options within the operating system to enhanced input devices, such as joysticks, trackball, touchscreen, gesture recognition, and speech recognition software [4]. However, the use of assistive technology for better computer access encounters barriers. Barriers vary from insufficient funding, lack of staff training, and negative attitudes at school [5], whereas the main barrier in the home environment is cost [6]. According to the study of Davies et al. [7] on youths with upper limb impairments caused by CP, most of youths with CP prefer the typical mouse to access computers, either by hand for cursor movement with finger-clicking for target selection, or by enhanced computer access methods, such as a combination of foot with toe-clicking. In addition, the study mentioned (1) that only a small group of youths with CP use assistive technologies, although a large group is aware of existing technologies, and (2) that a number of youths with CP have used assistive devices and technologies in the past. The study suggested that there are several reasons why youths with CP return to using a standard or modified mouse, which include the lack of provided assistive technology in an education setting [8], and youths with CP are not restricted to using computers in one location; they use computers in different locations, including a friend's home and the public library, where the existence of assistive technology to access computers is rare. Therefore, the most pragmatic decision is to use a standard mouse at all locations.

We are proposing an algorithm to enhance the performance of "point and click" computer tasks using a standard mouse for youths with CP.

1.2. Related Work

The HCI interface development for point-and-click computer tasks focuses on visual- and/or motor space–categorizing algorithms known as target-aware and target-agonistic. The target-aware algorithm is based on the dimensions and location of targets, but target-agonistic dimensions are based on path evaluation measures and allow for dynamic adjustment of control-display (C-D) gain. Some target-agonistic algorithms include Angle Mouse [9], PointAssist [10, 11], and Dynamic Mouse Speed [12], and they are used to enhance the performance of point-and-click computer tasks.

Angle Mouse [9] uses angular deviation to dynamically adjust C-D gain to enhance point-and-click tasks. In a study of 16 participants including 8 motion-impaired users, Angle Mouse improved the throughput of point-and-click tasks by 10.3% over the Windows default (C-D gain is static and was set to 10) for motion-impaired users. However, the Angle Mouse algorithm did not improve the throughput over the Windows default for typically developed (TD) users.

PointAssist uses C-D gain to control the precision (accuracy) of point-and-click computer tasks. PointAssist enhances accuracy measures by analyzing the features of submovements to detect when the user experiences pointing difficulty. PointAssist triggers a precision mode that slows the speed of the cursor to enhance the accuracy measures [10].

Hourcade et al. [10] assessed the efficiency of PointAssist with 20 older adults with ages ranging from 66 to 88 years. The study showed that the elderly gain better accuracy performance by using the PointAssist algorithm than the "enhance pointer precision" mode in Windows XP.

Tangand Lee [12] proposed a dynamic mouse speed (DMS) design, whereby mouse cursor speed (C-D gain) changes dynamically once they exceed predetermined 5-level speed thresholds. The DMS algorithm is based on the number of traveled pixels within the interval between mouse interrupts. If the traveled pixels' value is greater than a predetermined threshold, the system calls a particular function to change the pointer speed. The pointer speed is the C-D gain, which ranges from 1 (slowest) to 20 (fastest). Based on 2 experiments, with 11 participants in each, the DMS setting surpassed the Windows built-in settings significantly for larger distances. However, the DMS performance did not function well for all directions, especially for short distances where the performance of Windows built-in settings exceeded the DMS performance.

This prior research has shown that C-D gain has been controlled to increase the performance of cursor movement during pointing tasks. However, any increment of C-D will generally give a speed advantage over accuracy (speed accuracy trade-off); therefore, providing a dynamic algorithm that balances the speed-accuracy trade-off will certainly promote the performance of pointing tasks. If the dynamic algorithm does not balance the speed-accuracy trade-off, it will worsen the performance of both speed and accuracy of pointing tasks. This study seeks to address these issues in the development of an algorithm that better balances speed-accuracy trade-offs.

1.3. The Proposed Algorithm

We have evaluated the movement of individuals with CP in two studies. The first study showed that for individuals with CP, it is more important to focus on the development of user interfaces and algorithms to increase speed because they already appear to have sophisticated accuracy [13]. The second study examined both system and human effects that contribute to human movement of youths with CP and determined that the most significant human effect is the curvature index [14]. The curvature index is the ratio of the total distance traveled to the straight-line distance between the start and end points. A value of 1 indicates that the cursor path follows a straight line toward the target, and a larger value shows increasing deviations [15].

The movement deviation caused by a spasm can be measured by the curvature index; a higher-value curvature index causes an increase in movement time (MT).

Based on these prior studies, our proposed algorithm is based on a differential curvature index value (CI-1), as a CI value of 1 means that there is no deviation from the start-end movement line (see figure 1). The proposed formula of C-D is

$$\left\{ \begin{array}{l} C-D = 14 - ((CI-1)*4), CI <= 3 \\ C-D = 6, CI > 3 \end{array} \right\} \quad (1)$$

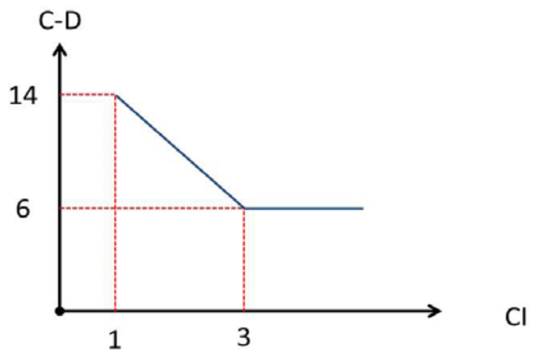

Fig. 1. CI and C-D gain relationship

The lowest value is set to 6 because the curvature index becomes high, resulting in a negative value of C-D (C-D must be greater than or equal to 1). The C-D gain is calculated based on a sequential set of 5 and 10 samples with a conditional distance of 16 pixels.

2. Empirical Validation and examination of proposed algorithm

It was important to examine and validate the algorithm by carrying out a controlled experiment using standard discrete pointing tasks. This section covers three subsections of the experiments, including subjects, apparatus, and experimental design.

2.1. Subjects

A.1. Twelve TD subjects (8 males, 4 females, aged 18–32 years, all right handed) conducted the experiment.

A.2. Six subjects (aged 13–22 years) with CP (MACS I and II) conducted the “point and click” tasks.

2.2. Apparatus

The experiment was conducted on an HP laptop machine with 2.2 GHz AMD Turion™ X2 Ultra Dual-Core Mobile ZM-82 with 3GB of RAM running 32-bit Windows 7 Enterprise. The pointer speed (C-D gain) was set to 10 as normal. The screen used was at a resolution of 1280 × 800 pixels at a refresh rate of 60 Hz. The pointing device type was a standard USB optical mouse (manufactured by Dell). The resolution of the input device was at 96 pixels per inch (DPI).

2.3. Experimental Design

A multidirectional discrete pointing task was undertaken to evaluate the cursor movement between two targets with different sizes and center-to-center amplitudes. The software was developed in C# with a sampling interval of 15 milliseconds. The trial is defined as the cursor traveling from a home square located at the center of the screen to a target square located somewhere with a prespecified movement amplitude and width. The trial starts once the user clicks on the home square and ends when the user clicks on the target square. Then the user returns to the home square to initiate a new trial. The experiment type is “with prior knowledge” of the target, which means that the target is visible to the participant before the trial begins. Table 1 summarizes the experiment setup.

Table. 1. Summary of experiment configuration

Target direction	4 directions (0°, 90°, 180°, and 270°)
Number of trials	16 trials (4 in each direction)
A*W combinations	9 combinations (3 amplitudes [120, 240, 360 pixels] with target size of 20, 30, and 40 pixels
Experiment type	Default (C-D:10) CI-based algorithm (2 window sizes: 5 samples and 10 samples)

3. Results and analysis

The movement toward the target during the point-and-click computer tasks is composed of submovement components: rapid primary movement and secondary correction movements.

The main speed and accuracy performance measures are MT and error rate (ER). In addition, we compare the performance of algorithms based on average speed (AS), maximum speed (MS), and MT of primary submovement and secondary submovement.

3.1. Tukey Post Hoc Analysis

The fixed factors are experiment type (ET; default, CI-based (5 samples) [CI-based(5)] and CI-based (10 samples) [CI_based(10)], amplitude (A; 120, 240, and 360 pixels), and width (W; 20, 30, and 40 pixels). Therefore, there are six interactions of fixed factors, which are ET, A, W, ET*A, ET*W, A*W, and ET*A*W. Subjects are selected as random factors. The interactions test is conducted for the following:

Movement time (MT): The time interval from a mouse click on the home square to a mouse click inside the target square [15, 16]. The unit is milliseconds.

Average speed (AS): The average value of speed samples within a given trial. It is expected that the degree of impairment has an influence on AS. The unit is pixels/millisecond.

Maximum speed (MS): The maximum value of speed samples within a given trial. It is expected that the degree of impairment has an influence on MS. The unit is pixels/millisecond.

Primary submovement time (PST): The time interval between initial cursor movement and first cursor pause. The unit is milliseconds [17].

Primary submovement distance (PSD): The distance traveled between initial cursor movement and first cursor pause. The unit is pixels [17].

Secondary submovement time (SST): The total time of secondary submovements from the first primary submovement pause until the target click. The unit is milliseconds [17].

Secondary submovement distance (SSD): The total distance traveled by secondary submovements from the first primary submovement pause until the target click. The unit is pixels [17].

Error rate (ER): The percentage of erroneous clicks within the A*W combination [16].

Number of submovements (NS): The number of discrete movements separated by pauses, defined as a zero cursor speed. A high NS indicate more movement correction, resulting in an increase in MT. Hwang et al. [18, 19] found that TD users completed 90% of point-and-click trials in less than seven submovements, whereas 90% of trials of motion-impaired users required more than seven submovements. In our experiment, the definition of a pause was conservatively set to a cursor speed of zero because it is difficult to specifically define a submovement [20].

Overshoot rate (OSR): The percentage of trials that involved overshoot movement within the A*W combination [9].

The selected path evaluation measures provide clear insights and information on cursor movement and performance of point-and-click tasks by the CI-based algorithm and Windows default.

3.2. Movement Time

Average MTs for each experiment type for both youths with CP and TD youths are shown in Table 2 and Figure 2.

Youths with CP and TD youths displayed a significant increase in average MT with the proposed algorithm over the Windows default regardless of different target widths and movement amplitudes. There is no significant

difference between performances of CI_based(5) and CI_based(10).
The MT increased by 12.4% (TD) and 18.5% (youths with CP) using CI_based(10) and 17.7% (TD) and 14.8% (youths with CP) using CI_based(5).

Table. 2. Average MT

Experiment Type	MT (ms) Average (SE)	
	CP	TD
Default	2205(41)	762(8)
CI-based(10)	2613(56)	856(9)
CI-based(5)	2531(48)	897(9)

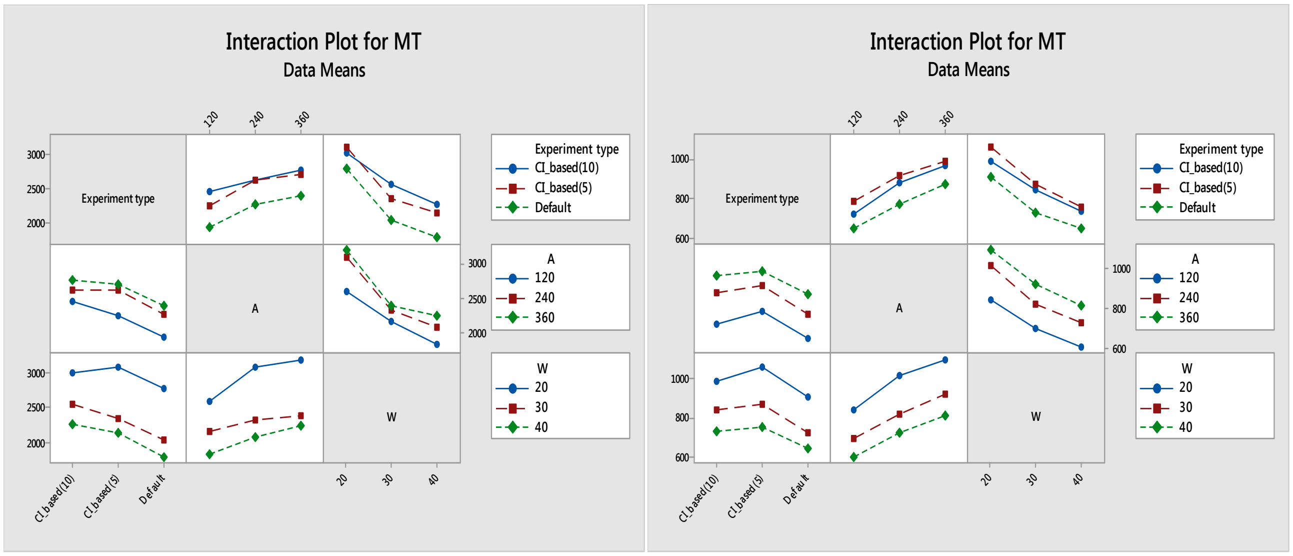

Fig. 2. Interaction plot of MT by experiment type, amplitude, and width levels; youths with CP (left), TD youths (right)

3.3. Average Speed and Maximum Speed

Means of AS for each experiment for both youths with CP and TD youths are shown in Table 3.

Table. 3. AS and MS

Experiment Type	AS(pixels/ms) Average (SE)		MS(pixels/ms) Average (SE)	
	CP	TD	CP	TD
Default	0.16 (0.002)	0.35 (0.0034)	1.1 (0.018)	1.6 (0.018)
CI_based(10)	0.18 (0.003)	0.35 (0.0035)	1.47 (0.033)	1.81 (0.024)
CI_based(5)	0.18 (0.003)	0.37 (0.0036)	1.59 (0.031)	2.01 (0.026)

Youths with CP travel with an AS that is significantly faster with the proposed algorithm (0.18 pixels/ms) than with the Windows default (0.16 pixels/ms) regardless of different target widths and movement amplitudes (the proposed algorithm increased AS by a rate of 12.5%). In addition, the interaction between amplitude and width ($P < 0.05$) has significant effects on AS, resulting in a range from a maximum AS of 0.24 pixels/ms at a 360 pixels amplitude and a 40 pixels width to a minimum AS of 0.11 pixels/ms at a 120 pixels amplitude and a 20 pixels width.
TD youths travel slightly faster with the CI_based(5) (0.37 pixels/ms) than Windows default (0.35 pixels/ms) and CI_based(10) (0.35 pixels/ms, AS increased by a rate of 5.7%). In addition, the interaction between amplitude and either experiment type or width ($P < 0.05$) has a significant effect on AS.
Youths with CP have a significantly faster speed with the proposed algorithm over the Windows default (1.1 pixels/ms) regardless of different target widths and movement amplitudes. There is a significant difference between performances of window size (5 samples vs. 10 samples). The average value of MS traveled by TD youths at the Windows default mode is equal to the average

MS traveled by youths with CP using the CI-based algorithm.

For youths with CP, there is a significant interaction effect between the experiment type and amplitude with respect to MS. CI_based results in a significantly higher MS at 240- and 360-pixel amplitudes than Windows default. In addition, the MS is significantly higher as the amplitude level increases within each experiment type.

In addition, the interaction of experiment type and width shows that the CI-based algorithm produces higher MS than Windows default across different widths.

For TD youths, MS is affected by the interaction of experiment type, amplitude, and width. The CI-based algorithm (both window sizes) was significantly faster than Windows default at each amplitude level and each width level, respectively.

3.4. Primary Submovement Time and Secondary Submovement Time

The average time of primary submovement and secondary submovements are shown in Table 4.

The proposed algorithm significantly assisted youths with CP and TD youths in traveling faster toward the target before the first pause (less PST) regardless of different target widths and movement amplitudes. There is no significant difference between performance of CI_based(5) and CI_based(10). In addition, the width has no significant effect on PST.

For TD youths, the PST using Windows default is significantly higher than the CI-based algorithm at each amplitude level or width level, respectively (significant interaction between experiment type and either amplitude or width).

Youths with CP require significantly more time with additional submovements to reach the target using the proposed algorithm compared with Windows default regardless of different target widths and movement amplitudes, but there is no significant difference between the performance of CI_based(5) and CI_based(10). The traveled SST by the Windows default across different amplitudes and widths is generally less than that of the proposed CI-based algorithm, but this is not significant.

The primary submovement performance has been enhanced by the CI-based algorithm, but the performance of secondary submovements negates the effect when compared with the Windows default system. It decreased the PST by 10.8% (youths with CP) and 12.4% (TD) using the CI-based algorithm with CI_based(10), and 14.7% (youths with CP) and 15.2% (TD) using CI_based(5). The SST increased by 27.9% (youths with CP) and 36.7% (TD) using CI_based(10), while with CI_based(5), the SST increased by 22.3% (youths with CP) and 50.1% (TD).

Table. 4. PST and SST

Experiment Type	PST (ms) Average (SE)		SST (ms) Average (SE)	
	CP	TD	CP	TD
Default	526 (11)	378 (5)	1666 (41)	384 (8)
CI_based(10)	469 (13)	331 (4)	2151 (55)	525 (9)
CI_based(5)	448 (9)	320 (4)	2082 (47)	577 (9)

3.5. Primary Submovement Distance and Secondary Submovement Distance

Youths with CP take travels significantly further in the primary submovement to reach the target using the proposed algorithm with CI_based(5) compared with Windows default and CI_based(10) regardless of different target widths and movement amplitudes.

TD youths travel further in the primary submovement to reach the target using CI_based(5) than when using CI_based(10) and Windows default. The significant interaction between experiment type and width on the primary submovement indicates the significant difference of PSD between the CI-based algorithm with the 5-sample window size and the other two types at each width level.

Youths with CP travel further to reach the target using the proposed algorithm compared with Windows default regardless of different target widths and movement amplitudes.

For youths with CP and TD youths, the interaction between width and either experiment type or amplitude showed that SSD by the Windows default across different widths is generally less than by the proposed CI-based algorithm.

3.6. Erroneous Clicks and Error Rate

The average ER for both TD youths and youths with CP are shown in Table 5.

TD youths score the least ER using Windows default (6%), whereas their performance worsened using the CI-based algorithm (16.13% for windows size of 5 samples and 18.36% for windows size of 10 samples). For youths with CP, the ER is lower using Windows default (9.4%) and higher using the CI-based algorithm (13.21% and 10.38%) compared with TD youths. Within each trial, the youths with CP produce erroneous clicks up to 5, 6, and 6 times using Windows default, CI_based(10), and CI_based(5), respectively, whereas the TD youths produce erroneous clicks up to three times using either Windows default or the CI-based algorithm.

For youths with CP, the interaction between experiment type and width has a significant effect on ER. The Windows default and CI_based(5) sample window size produce significantly lower ERs at widths of 30 pixels and 40 pixels in comparison with a width of 20 pixels.

Table. 5. ER

Experiment Type	ER (%) Average (SE)	
	CP	TD
Default	9.4 (0.9)	6 (0.6)
CI_based(10)	13.2 (1.3)	16.1 (1)
CI_based(5)	10.4 (1)	18.4 (1)

3.7. Number of Submovements and Overshoot Rate

The average NS and OSR for both TD youths and youths with CP are shown in Table 6.

For youths with CP, the NS (including primary submovement) is significantly increased by using CI_based(10) and CI_based(5) compared with Windows default. In addition, for TD youths, the interaction between experiment type and width had a significant effect on NS. The Windows default produces significantly lower submovement correction than the CI-based algorithm at each width level; the NS at the 20-pixel width for each experiment level is significantly higher than at the 30- and 40-pixel widths. The CI-based algorithm with the 5-sample window size at the 20-pixel width level produces the highest NS corrections.

Youths with CP tend to overshoot during point-and-click computer tasks more than TD youths. The OSR of youths with CP by the Windows default mode is 40% compared with 15% for TD youths. In addition, the percentage rose considerably especially for TD youths while the OSR significantly rose, but at a lower rate for youths with CP.

For TD youths, there is significant interaction between experiment type and amplitude, which affects the OSR. The interpretation is that the impact of the experiment type level depends on the amplitude level and vice versa. Across different amplitude levels, the OSR for each experiment type is significantly different. For example, the results for an amplitude of 120 pixels are 17% for the Windows default, 41% for the CI-based algorithm with 10-sample window size, and 66% for the CI-based algorithm with 5-sample window size, but there is no significant difference between OSRs across amplitude levels within each experiment type (P = 1.06). A similar scenario was presented for the interaction effect on the overshoot between experiment type and width.

Table. 6. NS and OSR

Experiment Type	NS Average (SE)		OSR (%) Average (SE)	
	CP	TD	CP	TD
Default	8 (0.2)	3 (0.06)	40 (1.7)	15 (1)
CI_based(10)	12 (0.3)	4 (0.06)	62 (1.8)	50 (1.4)
CI_based(5)	11 (0.2)	4 (0.07)	65 (1.6)	65 (1.3)

4. Discussion

In the performance evaluation of the CI-based algorithm by conventional measures, both MT and ER show the inefficiency of the proposed algorithm overall compared with the Windows default system (higher MT and ER produced by the proposed algorithm). However, the CI-based algorithm exhibited benefits in terms of AS, MS, and the primary submovement phase.

For both TD youths and youths with CP, the AS per point-and-click trial and the average MS increased by using the CI-based algorithm compared with Windows default.

The algorithm did not improve the pointing task for youths with CP; the ER and maximum number of erroneous clicks within each trial are slightly higher than Windows default. However, the algorithm enhanced the performance of the primary submovement (less time and nearly equal distance compared with Windows default). It had the opposite effect on secondary submovements; it made the user lose control when the cursor speeds reached higher levels, resulting in overshoot movements, requiring more submovements to correct the movement toward the target.

The significant interaction between experiment type and amplitude affected OSR for TD youths (higher OSR by the CI-based algorithm) and the insignificant difference of overshoot across amplitude levels for TD youths and youths with CP suggest that the CI-based algorithm deteriorates the cursor control, especially for TD youths, who have more control and stability of movement than youths with CP. This can be viewed as the rapid increase of submovement corrections and traveled SSD by the CI-based algorithm compared with Windows default.

The PointAssist study on elderly whose motor control lacks sophistication showed no significant advantages on MT, NS, and number of slip-offs (target reentry). Our results are similar to PointAssist, although PointAssist aimed to enhance accuracy, and we aimed to enhance speed according to our findings in an earlier study[13].

The DMS algorithm improved the performance (MT) at longer amplitudes, but our algorithm did not indicate the need to improve the performance of secondary submovements.

The window size had a variable influence on path measures, although the 5-sample window size produced faster modification in C-D gain than the 10-sample window size. For youths with CP, the 5-sample window size allows the user to quickly reach an MS and the target compared with the 10-sample window size. This can be seen by less PST and SST produced by the 5-sample windows size. This is because the 10-sample window size takes longer to modify, especially during the submovement corrections, whereas the user needs robust control of the cursor. The 5-sample window size provides better accuracy pointing than the 10-sample window size.

The 20-pixel width required more submovement correction and traveled SSD, suggesting that for designing small icons, it requires a pointing accuracy algorithm to be incorporated during submovement corrections similar to the PointAssist. PointAssist found that the algorithm had a significant effect on an 16-pixel width for children [11].

5. Study Limitation

The study is limited by a lack of training. People are used to using Windows default. In addition, the number of participants with CP is not large due to the difficulty of finding youths who fit our experiment criteria.

6. Conclusion

The overall performance of the CI-based algorithm, for both TD and youths with CP, is less than the Windows default system, but there is a possibility to enhance this performance by decreasing the SST. The primary submovement is faster with the algorithm than with the Windows default mode.

7. Future Work

Future work will include performance enhancements in the secondary submovements phase, such as clamping extreme speeds to provide the youth with more control with the cursor during secondary submovements, or triggering a control mode once the NS goes over the threshold NS, to provide more robust control during submovement corrections. Clamping extreme speeds and control mode triggering can be implemented by dynamic adjustments of C-D gain. Another recommendation is a switch from the CI-based algorithm mode to Windows default mode once the cursor speed reaches an upper threshold speed and a switch from Windows default to the CI-based algorithm mode once the speed crosses a lower threshold speed.

8. REFERENCES

1. Yeargin-Allsopp, M., Braun, K. V. N., Doernberg, N. S., Benedict, R. E., Kirby, R. S., and Durkin, M. S., *Prevalence of cerebral palsy in 8-year-old children in three areas of the United States in 2002: a multisite collaboration.* Pediatrics, 2008. **121**(3): p. 547-54.
2. Access Economics Pty Limited, *The economic impact of cerebral palsy in Australia in 2007*. April, 2008.
3. Benyon, D., Crerar, A., and Wilkinson, S., *Individual differences and inclusive design.*, in *User Interfaces for All: Concepts, Methods, and Tools*, C. Stephanidis, Editor. 2000, CRC Taylor and Francis. p. 21-46.
4. Davies, T.C., Mudge, S., Ameratunga, S. and Stott, N.S., *Enabling self-directed computer use for individuals with cerebral palsy: a systematic review of assistive devices and technologies.* Developmental Medicine & Child Neurology, 2010. **52**(6): p. 510-516.
5. Copley, J. and Ziviani, J., *Barriers to the use of assistive technology for children with multiple disabilities.* Occup Ther Int, 2004. **11**(4): p. 229-43.
6. Ellis, J.B. *The digital divide in special education*. International Study Association on Teachers and Teaching's Totems and taboos: Risk and relevance in research on teachers and teaching, 2007.
7. Davies, T.C., Chau, T., Fehlings, D., Ameratunga, S., and Stott, N.S., *Youth With Cerebral Palsy With Differing Upper Limb Abilities: How Do They Access Computers?* Archives of Physical Medicine and Rehabilitation, 2010. **91**(12): p. 1952-1956.
8. Michaels, C. A., Prezant, F. P., Morabito, S. M., and Jackson, K., *Assistive and instructional technology for students with disabilities: A national snapshot of postsecondary service providers.* Journal of Special Education Technology, 2002. **17**(1): p. 8.
9. Wobbrock, J.O., Fogarty, J., Liu, S., Kimuro, S., Harada, S., *The angle mouse: target-agnostic dynamic gain adjustment based on angular deviation*, in *Proceedings of the 27th international conference on Human factors in computing systems*. 2009, ACM: Boston, MA, USA.
10. Hourcade, J.P., Nguyen, C.M., Perry, K.B., and Denburg, N.L, *Pointassist for older adults: analyzing sub-movement characteristics to aid in pointing tasks*, in *Proceedings of the 28th international conference on Human factors in computing systems*. 2010, ACM: Atlanta, Georgia, USA. p. 1115-1124.
11. Hourcade, J.P., Perry, K.B., and Sharma, A., *PointAssist: helping four year olds point with ease*, in *Proceedings of the 7th international conference on Interaction design and children*. 2008, ACM: Chicago, Illinois. p. 202-209.
12. Tang, K.-H. and Y.-H. Lee, *Dynamic mouse speed scheme design based on trajectory analysis*, in *Proceedings of the 2007 international conference on Ergonomics and health aspects of work with computers*. 2007, Springer-Verlag: Beijing, China. p. 329-338.
13. Almanji, A., Davies, T.C., and Stott, N.S, *Using cursor measures to investigate the effects of impairment severity on cursor control for youths with cerebral palsy.*

International Journal of Human-Computer Studies, 2014. **72**(3): p. 349-357.

14. Almanji, A., Davies, T.C., Payne, A. and Amor, R., *A Nonlinear model for mouse pointing task movement time analysis based on both system and human effects.* IEEE Transactions on Neural Systems and Rehabilitation Engineering.
15. Keates, S., Hwang, F., Langdon, P., Clarkson, P.J., Robinson, P., *Cursor measures for motion-impaired computer users*, in *Proceedings of the fifth international ACM conference on Assistive technologies*. 2002, ACM: Edinburgh, Scotland. p. 135-142.
16. MacKenzie, I.S., Tatu, K., and Miika, S., *Accuracy measures for evaluating computer pointing devices*, in *Proceedings of the SIGCHI conference on Human factors in computing systems*. 2001, ACM: Seattle, Washington, United States.
17. Balakrishnan, R., *"Beating" Fitts' law: virtual enhancements for pointing facilitation.* International Journal of Human-Computer Studies, 2004. **61**(6): p. 857-874.
18. Hwang, F., Keates, S.,Langdon, P.,Clarkson, J., *Mouse movements of motion-impaired users: a submovement analysis.* ACM SIGACCESS Accessibility and Computing, 2003(77-78): p. 102.
19. Hwang, F., Keates, S.,Langdon, P.,Clarkson, J., *A submovement analysis of cursor trajectories.* Behaviour & Information Technology, 2005. **24**(3): p. 205-217.
20. Chapuis, O., Blanch, R. and Beaudouin-Lafon, M., *Fitts' law in the wild: A field study of aimed.* 2007