

Navigating a 3D Avatar using a Single Switch

Eelke Folmer
Player-Game-Interaction Lab
University of Nevada, Reno
efolmer@unr.edu

Fangzhou Liu
The Davidson Academy
of Nevada, Reno
fliu@davidsonacademy.unr.edu

Barrie Ellis
OneSwitch.org.uk
Essex
barrie@oneswitch.org.uk

ABSTRACT

Many users with severe motor impairments, such as quadriplegics, interact with computers using an indirect selection technique called switch access scanning. Switch scanning allows for iteratively selecting an input from a set of input options using a single switch input and which replaces the use of a keyboard or a mouse, which they may be unable to use. Navigating an avatar in a 3D virtual world using existing switch access scanning systems is slow and erroneous because these interfaces are non-linear and requires players to provide continuous (holding a key) and mixed inputs (holding two or more keys). Through the analysis of navigation behavior of eight able-bodied users, a new scanning system called hold-and-release was developed. Using simulation hold-and-release scanning was found to be significantly more efficient than existing scanning systems. Multistep selection was found to be most efficient for mixing inputs, but expanding the selection set has no approximation errors.

Keywords

Switch Access, Scanning, Motor Impairment, Virtual Worlds.

Categories and Subject Descriptors

H.5.2 [User Interface]: Input devices and strategies

General Terms

Human Factors, Performance.

1. INTRODUCTION

Software is increasingly modeled after how we interact with the real world; as such interaction is the most natural to us. However, the emergence of immersive 3D technologies such as video games and virtual worlds often raises new barriers for users with disabilities [26]. Users with severe motor or cognitive impairments often lack the degree of fine motor control required to use input devices, such as mice, joysticks or controllers, typically used to control an avatar in a 3D environment. Consequently they have to revert to using input devices, such as an eye tracker [22] or single switch



Figure 1: Hold-and-release scanning implemented in Second Life to navigate an avatar using a single switch. FORWARD and TURN RIGHT inputs are both selected and CANCEL FORWARD is highlighted.

controller [16] that have been specifically designed to accommodate their abilities. Assistive devices are constrained in the amount and types of input that can be provided, for example a single switch may only be able to provide a binary (on/off) input. A larger amount of input (at a slower rate) can be provided through *scanning* where users iteratively make a discrete selection from a predefined set of input options [16, 20]. Though existing scanning systems are efficient for linear types of input such as text entry or accessing menus, we found these to be non-optimal for providing non-linear type of inputs. Navigating a 3D avatar is an example of non-linear input as players may hold a particular key for an indeterminate amount of time, as well as hold two keys at the same time to adjust the course of their avatar. Providing these types of input using existing discrete scanning methods is inefficient and erroneous, as they have to be approximated with individual discrete key presses.

This paper presents an efficient scanning system (hold-and-release) that can be used for avatar navigation. This paper is organized as follows. Section 2 provides background and Section 3 discusses related work. Section 4 discusses the design of hold-and-release. Section 5 presents the results of a simulation with this scanning system and alternatives. Section 6 discusses results and future work and the paper is concluded in Section 7.

2. BACKGROUND

A switch is an assistive technology device that can be operated by any body part able to produce consistent and voluntary movement. Different types of switches can be identified based upon the type of action required to use them, for example, pull, push, squeeze, blink or sip or puff [6]. The smallest amount of input that can be provided with a single switch is a binary input [26] as holding down the switch may be painful for someone with arthritis or difficult to provide when using a sip and puff switch. Figure 2 shows a binary push switch that is activated by the user's head. Switches allow users with severe motor [16] or cognitive impairments [14] to access computers but they can also be used for allowing toddlers and infants to interact with computers [23].

Single switches allow for a small amount of input provision but switch users who are able to time their input, distinguish between choices, and understand the results of a particular choice, can provide a larger amount of input through an indirect selection technique called scanning [9, 20]. A scanner iterates through a set of input options (selection set) by highlighting each option for an amount of time called the scanning rate (SR). When the user presses their switch the currently highlighted option is selected. A scanning pattern refers to the way items in the selection set are presented to the user, for example, an on screen keyboard can be arranged in rows and columns (Cartesian scanning) or in a multilayer circle (Polar scanning). Multiple selection steps or switches may be required to make a selection.

Different scanning control techniques have been developed [6], for example, a scanner can automatically iterate over the set and the currently highlighted item is selected when the user presses the switch. Alternatively, the user holds the switch and the scanner iterates over the set of inputs and a selection is made when the switch is released. The scanning rate, pattern of scanning, and scanning control, are typically individualized to the physical, visual and cognitive capabilities of the user [6]. When relationships between subsequent inputs can be defined, such as letters that form a word, the rate of single switch input can be significantly increased using adaptive prediction models such as Dasher [15] or Nomon [10], but these are limited to single switch text entry.

When the switch is activated or released –depending on the control scheme– the scanner selects the input option currently highlighted and then returns to the beginning of the scanning pattern to allow the user to select the next input. Such discrete selections are efficient for activities, such as text entry or menu selection, where the set of inputs is large it can be traversed linearly and the occurrence of having to repeat an input is low. For activities where the selection set is small, and there is a high occurrence of repeating the same input, discrete selections are inefficient.

An activity that requires continuous and mixed inputs is controlling a 3D object in a game, for example, navigating an avatar in a 3D environment. Players navigate an avatar using the following set of inputs (FORWARD, TURN LEFT, TURN RIGHT, BACK) that are either activated with a keyboard or a controller. To navigate an avatar, players may hold a particular input (FORWARD) for an indeterminate amount of time but also hold two inputs simultaneously



Figure 2: Quadriplegic child playing a video game using a head activated single switch controller. The box shows the switch controller used in larger detail.

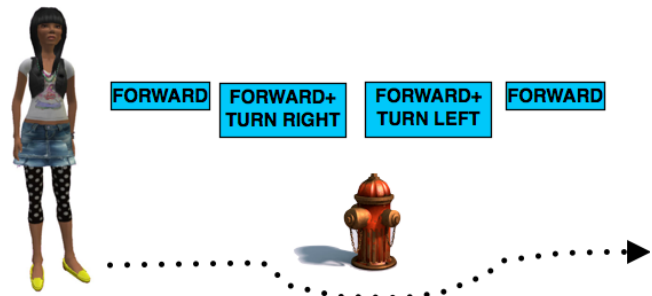


Figure 3: Inputs required to circumnavigate an obstacle with an avatar.

(FORWARD+TURN LEFT) to circumnavigate an obstacle (see Figure 3).

To provide continuous and mixed inputs using existing scanning control techniques, it must be approximated using discrete inputs. For example, let's assume the player wants to navigate towards a door, which would require providing the (FORWARD) input for 439 *ms*. A discrete input is activated for a fixed period of time each time it is activated. For example, if (FORWARD) is activated for 200 *ms* then the player must scan to this input twice to generate (FORWARD) held for 400 *ms* and there will be an approximation error of 39 *ms*, which may prevent precise navigation. Though errors could be minimized by activating the input for a lower amount of time this would significantly increase the time to generate the input, as the player has to scan to this specific input more times. Because two different inputs cannot be active at the same time using a discrete scanning control technique, mixed inputs, such as (FORWARD+TURN LEFT), have to be approximated by interleaving single discrete inputs (FORWARD, TURN LEFT), where errors further grow unbounded due to propagation. This problem is related to the problem of mouse manipulation using a single switch [9] with the fundamental difference that the length of a particular input to be provided for navigating an avatar may not be a priori known, whereas for mouse navigation this is constrained by the screen resolution.

3. RELATED WORK

Numerous arcade style games have been developed that can be played using a single switch. The one-switch website (<http://www.oneswitch.org.uk>) is a non-profit website dedicated to such games, and it lists and reviews about 80 one-switch games, most of which can be downloaded for free. Currently three games exist where players can navigate and avatar in 3D and which offer features to accommodate players with motor impairments.

POWERUP is a multiplayer educational 3D game developed by IBM [24] where players explore environments and solve various quests. Accessibility features for users with motor impairments include: (1) on screen keyboard to allow for mouse only input; (2) alternative key mappings; and (3) a toggle mode for holding down the forward key. POWERUP does not support switch access, so players with motor impairments either need to be able to use a pointing device or a keyboard.

ZOGAN'S LOG [4] is a one-switch 3D game, where players explore an alien planet by controlling a spaceship using a polar (rotational) scanning mechanism. Players move forward by holding the switch, shoot by tapping the switch and turn in one direction when the switch is released. Though intuitive, a previous study shows that polar scanning is significantly slower than Cartesian scanning [9]. This game cannot be played using binary switch input.

PORTEM [2] is a 3D game, where players can explore a city and play all sort of minigames. Portem can be played using regular controls such as keyboard and mouse but also using a single binary switch where a specific input option to navigate the player's avatar [FORWARD, TURN LEFT, TURN RIGHT] is held or released upon switch activation. The scanning control scheme used does not allow for mixing inputs and avatar movement is constrained in the sense that the player's avatar cannot move backwards.

4. HOLD-AND-RELEASE DESIGN

Though some users with severe motor impairments may be able to use multiple switches or hold a single switch, our research challenge was to develop an efficient scanning method that allows for providing continuous and mixed input using the smallest amount of input possible, e.g., a binary switch. The motivation behind this choice is that if we can accommodate the abilities of the most *extreme* user, this solution may be sub-solution of solutions that accommodate the abilities of less extreme users, for example, a user being able to use two binary switches or a regular switch.

We focus on navigating an avatar in the popular virtual world of Second Life [1] as Second Life offers a vast 3D world for exploration and social interaction that includes virtual communities for users with disabilities [5]. Other 3D environments such as first person shooters or action games involve combat and require quick responses from their players, while in virtual worlds these requirements do not exist and we can exclusively focus on navigation. Games may not offer free navigation in any direction to the same extent as virtual worlds. The use of virtual worlds is further motivated by their increased use as academic learning tools [18], which requires them to be made accessible because of web

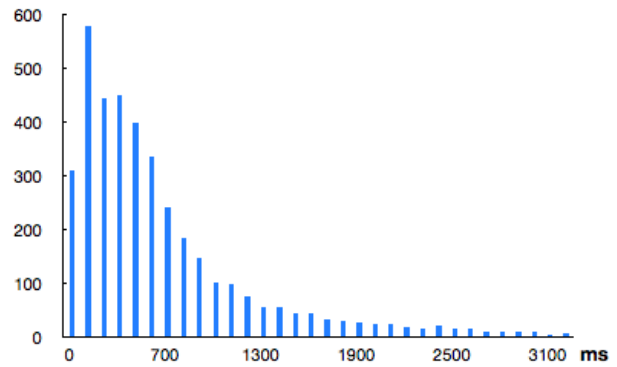


Figure 4: Histogram of keystroke length

accessibility laws such as Section 508 [25] (United States) or W3C [3] (European Union).

4.1 Characterizing Avatar Navigation

We first characterized avatar navigation to gain a better understanding about the frequency and nature of continuous and mixed input. Second Life's users can navigate their avatar (FORWARD, TURN LEFT, TURN RIGHT, BACK) with the arrow keys or a keyboard, using a controller or even using on screen keys using their mouse. As keyboard navigation is more common we focus on keyboard input.

A keystroke logger was implemented in the Second Life viewer which records which key(s) were held and for how long (in milliseconds) whenever the user releases a key or adds a key to an already held key. Mixed keystrokes that cancel each other out, such as, FORWARD+BACK were not recorded and consequently a multistroke keystroke consists of at most two keys. Eight able-bodied students (2 females, mean age 25.8) who all had experience with navigating an avatar in either a virtual world or a video game such as a first person shooter were asked to explore a random location in Second Life for 8 minutes. Though avatars are able to fly in Second Life, we disabled this as to focus exclusively on navigation using walking, which is more common in virtual environments.

```
<-- press Forward
Forward 873          <-- add Left
Forward+Left 220    <-- release Forward
Left 293            <-- add Forward
Left+Forward 167    <-- release Left
Forward 675         <-- release Forward
```

A total of 4,300 keystrokes were collected of which 37.4% were multistrokes. The smallest and largest keystrokes were: 26 ms and 30,606 ms. The average keystroke length was: 804 ms (SD= 1,309 ms). Figure 4 shows a histogram of keystroke length and it shows it does not follow a normal distribution but is right skewed. Keys were held for a total of 3,037 seconds, which means that participants were navigating 79% of the time during the 8-minute study.

Table 1 shows the characteristics of the collected single and multistrokes as percentages of the total number of keystrokes and duration. FORWARD is the most frequently used input of single (75.25%) and multistrokes (98.92%). It is also the

Table 1: Characteristics of Avatar Navigation.

Input	Single	Multi	Total	Time
FORWARD	74.9	-	46.8	79.4
RIGHT	10.6	-	6.6	6.1
LEFT	12.7	-	8.0	7.0
BACK	1.9	-	1.2	1.8
FORWARD+RIGHT	-	46.7	17.5	10.3
FORWARD+LEFT	-	52.4	19.6	8.9
BACK+RIGHT	-	0.5	0.19	0.12
BACK+LEFT	-	0.5	0.19	0.21
total	100%	100%	100%	100%

key that is held the longest (86.92% of single & multi) of the total time participants held keys. RIGHT and LEFT are used less frequently in single strokes but comprise 46.21% and 52.71% of multistrokes. This confirms our assumption that multistrokes are primarily used to adjust the course of an avatar. BACK is rarely used. We consider the collected data to be representative of 3D navigation behavior in general and we use it to inform the design of our scanning system.

4.2 Optimization Criteria

Approximating continuous and mixed inputs with discrete inputs is inefficient and leads to approximation errors (see Section 2). Consequently to improve upon existing scanning systems we seek to minimize the following three parameters:

1. Average time (T) to generate an input.
2. Average number of switch activations (SA) required to generate an input.
3. Approximation errors (*error*).

Where T depends on SA it also depends on the length of the set of input symbols that we need to scan over (to mix inputs we may expand the input set) and the scanning rate (SR). User error, e.g., the user failing to activate the switch when the desired input option is highlighted or accidentally activating a wrong input option, is considered to be proportionally related to the number of SA's required to generate the input. User error is primarily affected by SR, which has been extensively studied in previous work [11, 13, 21]. For example, studies have shown that an optimal SR can be achieved by dividing a user's response time by 0.65 [21] and therefore we do not incorporate it in the design of our scanning method.

4.3 Continuous input

Rather than making a discrete selection when the switch is activated, we modify the scanning control method to hold the input until the user releases it. After selecting an input the scanner will keep highlighting the activated input which when the user presses their switch will release it. This inherently increases the cost of providing an input with one extra switch activation, but for inputs that are held for longer than two times the time a discrete input would be activated, holding and releasing an input is anticipated to be more efficient. This is further motivated by the observation that the smallest key press we found was 26 *ms* and the average keystroke length 804 *ms*. The set of inputs to be scanned over can also be optimized. For example, our study found FORWARD to

be the most frequently used input. The input set should be rearranged as such to allow inputs with a higher frequency of occurrence to be scanned to faster.

4.4 Mixing inputs

Adding an input to an input already activated input can be facilitated using one of the following two strategies:

1. Extending the set of inputs with symbols that represent mixed inputs, e.g., FORWARD+RIGHT (F+R).
2. Multistep selection where after selecting an input, e.g., FORWARD, we scan over a subset of inputs that can be meaningfully added to the currently active input or cancel the active input, e.g., [RIGHT, LEFT, CANCEL].

Each strategy involves different tradeoffs that affect efficiency and error of generating an input. The average time to scan to an input will increase for a larger selection set. If the user wants to mix inputs, they must cancel the currently held input and then wait for the mixed input option to become highlighted in the set. For an SR of 1 second, and a scanning set of [F, L, R, B, F+L, F+R, B+L, B+R] it would already take 5 seconds to activate F+R. Expanding the selection set stops the avatar when a different input is selected, which allows for precise but slower navigation.

The selection set can also be rearranged based on the likelihood of this input being selected, where the results of the user study can be used to inform how the selection set is optimally composed. For example, an analysis of the keystrokes collected from our user study (see Table 1) reveals that users used the input FORWARD+RIGHT (17.5%) significantly more than FORWARD (7%). By rearranging the selection set we should allow for such inputs to be scanned to faster by putting more frequently used inputs at the beginning of the selection set. The length of the selection set significantly affects the time to scan to an input. Rather than including all available inputs in the selection set, we could scan over a subset. For example we could cull single and mixed inputs involving BACK as these are rarely used. This will increase error as we must approximate these inputs using discrete inputs, e.g., similar to how currently mixed inputs must be approximated using existing scanning systems.

Multistep selection is potentially more efficient as users can add an input while a key is held and adjust the course of an avatar while it is moving rather than having to stop the avatar when selecting a mixed input option. This may allow for more efficient navigation, but this comes at the cost of an approximation error, e.g., when an input is held the user can either add a new input or cancel the current input or when both inputs are held the user could also cancel either of them. The problem is however that these additional input options must be made available to the user using scanning, which introduces a small delay, as the user must wait until the desired input option is highlighted. For example, if the user is navigating their avatar using FORWARD and all of a sudden encounters an obstacle and wants to add RIGHT, the user must wait until the scanner is highlighting RIGHT in the available selection set [RIGHT, LEFT, CANCEL]. Depending on the specific SR used the average time it takes to select an input from the subset is on average $1.5 \cdot \text{SR}$, which may

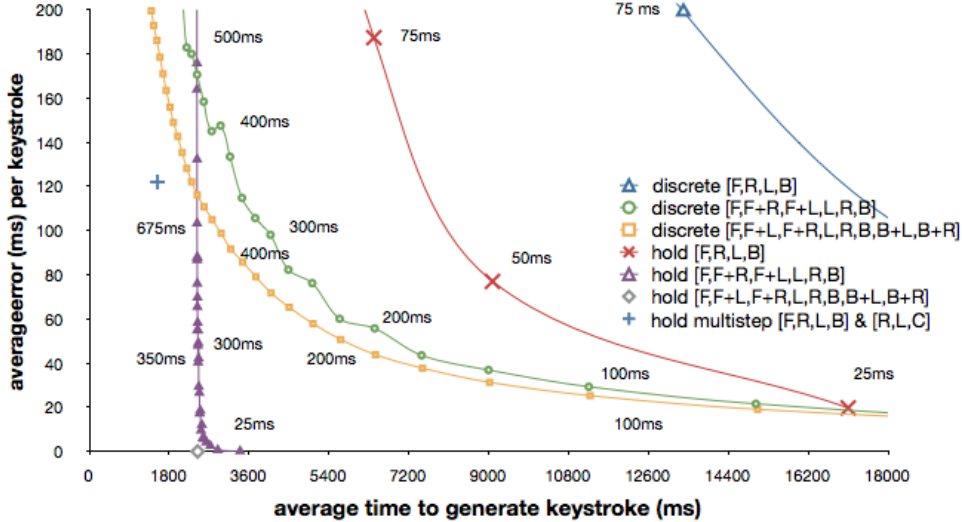


Figure 5: Results from the simulation. Labels at the data points indicate discrete input activation values.

result in the user already bumping into the obstacle with their avatar.

5. SIMULATION

The design of hold-and-release scanning elicited a large number of possible design alternatives, such as adding mixed input symbols to the selection set, using subsets in multistep selection and rearranging the selection set. Evaluating each possible design alternative and different parameters through user studies as well as comparing the performance of hold-and-release with a discrete scanning systems would require a large number of switch users, as an extended selection set that consists of the following inputs [F, L, R, B, F+L, F+R, B+L, B+R] already has 8! possible permutations.

In recent years several novel and efficient discrete scanning systems and scanning patterns have been developed where the performance of a different alternatives has been evaluated through simulation [7, 8, 17, 19]. Using simulation, sample interactions from able-bodied users are collected upon which a simulator will try to approximate these interactions using different scanning systems with different parameters where total time and number of SA are computed. The use of simulation is motivated as it not only allows for cost effectively evaluating a large number of alternative designs but also because the abilities of switch users have been extensively studied in the past decade [11, 13, 21] where results from these studies, such as what scanning rate and scanning pattern are most effective can be specified in the simulator. We already collected a large number of keystrokes from able-bodied users that represent a large set of different use cases scenarios of avatar navigation in different environments. Simulation allows for cost effectively evaluating the performance and errors of different scanning alternatives to approximate this set of keystrokes, which would be infeasible to perform using conventional user studies.

A simulator was written in Java and which takes as input a set of keystrokes to be approximated. For a given scanning

control method and a selection set, the simulator computes for each keystroke: (1) the time it takes to generate (2) the number of switch activations (SA) required; and (3) any approximation errors (in *ms*). When the last keystroke is simulated it computes an average for each one of these variables. To simulate a keystroke, we look up its position in the selection set to find the number of scan iterations (SI) required for the input to become highlighted. It then it takes a response time (RT) from the user to activate the switch, e.g., $T=SI+RT$ and we also increase SA. This response time must be lower than the SR otherwise users will not have enough time to activate the switch when the desired input option is highlighted. Based on optimal values found in previous studies, we use a scanning rate (SR) of 1,000 *ms* and RT is set to 650 *ms* using the 0.65 rule [21]. The value of these variables is invariant to our analysis. User error and an initial scanning delay are not modeled.

The simulator compares discrete versus hold-and-release scanning systems using three different selection sets, e.g., [F,L,R,B], [F,F+L,F+R,L,R,B,B+L,B+R] and [F,L,R,B,F+L,F+R], which has mixed inputs with BACK removed. The simulator evaluates all permutations of a selection set. For hold-and-release scanning, multistep selection is simulated using the following subsets: [F,L,R,B], and [L,R,F,B,CANCEL] where the contents of the last set are generated dynamically based on the currently held input and also permuted.

For discrete inputs, we vary the discrete input activation (DIA) from 25 *ms* (the smallest keystroke found) to 1,000 *ms* with 25 *ms* increases. Errors are computed for discrete scanning as follows: if a single stroke is held for *t* *ms*, then $error = t - n * DIA$ where $n = \lfloor t/DIA \rfloor$. Because multistrokes are approximated using single strokes in discrete scanning, their error propagates in two dimensions and it is calculated as: $error = (t - n * DIA)^2$. For hold-and-release, if a multistroke is not available in the selection set (such as those with BACK) it is approximated using discrete inputs and error is computed as if it was generated using discrete scanning. The

error in multistep scanning is: $error = |t_{previous} - (SI * RT)|$ when $(t_{previous} - SI * RT) < 0$ which models the delay in scanning through the second set.

5.1 Results

Figure 5 show the average time to generate a keystroke versus average error for different scanning control systems and different selection sets in their ability to approximate the 4,300 keystrokes that were collected in the user study with 8 able-bodied users. Scanning systems that involve approximations using discrete inputs (all except hold-and-release with [F,F+L,F+R,L,R,B,B+L,B+R] and hold-and-release using multistep) are plotted as lines in Figure 5 with increasing values for their DIA. Figure 5 does not show the number of SA's required to generate the input as we primarily optimize over efficiency and error and SA is part of efficiency. For each technique the permutation of the selection set that yielded the smallest average time to generate a keystroke was selected.

The results from the simulation show that hold-and-release scanning control performs significantly better in average time to generate an input and average error per keystroke than discrete scanning control methods for all used selection sets and acceptable error rates ($error < 100ms$). For both scanning control types the extended set [F,F+L,F+R,L,R,B,B+L,B+R] outperforms both its subsets. Approximating multistrokes not in the selection subset using discrete keystrokes takes significantly more time, than the extra time it takes to scan through a larger set.

For hold-and-release, multistep scanning is more efficient (1,539 *ms* per stroke) than using the extended set (1,795 *ms* per stroke) ($T_{2,8576} = 31.0 p < 0.01$), as significantly fewer switch activations are required (1.51 versus 2.0 per stroke). However, the associated error of multistep scanning is significantly larger than for the extended set (122 *ms* versus 0 *ms* per keystroke) ($T_{2,8576} = 28.3 p < 0.01$). If we assume an able-bodied user is able to provide input with a response time of 650 *ms*, navigating an avatar using hold-and-release scanning is only a factor 2.4 times slower. Whether to use the extended set or multistep selection to create mixed inputs depends on the user's scanning rate and response time. Though multistep is slightly faster, the approximation error of 122 *ms* may impede the user's ability to accurately navigate their avatar. Avatar walking speed in second life is 3.0 *m/s*, which is faster than average human walking speed (1.4 *m/s*). Nevertheless an approximation error of 122 *ms* will allow switch users to navigate their avatar within 0.36 *m* distance of a specific target, which seems reasonable and will allow for going through doors that are 1 meter wide.

6. DISCUSSION AND FUTURE WORK

Though the simulator allows for cost effectively comparing a large number of scanning system alternatives it currently has some shortcomings. The simulator does not model user errors and hence assumes the user has perfect timing skills. Though user error is directly related to the number of SA, humans do make errors, though a study [21] with proficient switch users shows they have fairly high selection accuracies (>90%). However avatar navigation using scanning is significantly different from text entry where a mistake can be fairly easily corrected using a delete input option. When a

user navigating their avatar using hold-and-release fails to select the option when it is highlighted it must wait until the scanner has traversed the entire selection set, which for the extended set maybe be very time consuming. Users may also activate the wrong input option. Because an input is continuously held in hold-and-release, correcting the mistake will take more time than using a discrete scanning control method as the user has to cancel the activated input and then active and cancel the inverse input. To evaluate the efficiency of hold-and-release pertaining user error we could inject Bayesian noise and implement user error correction strategies as to simulate more realistic switch user behavior.

Hold-and-release was found to be most accurate in being able to generate those keystrokes typically used by able bodied users that navigate virtual worlds. Consequently, to develop hold-and-release we leveraged data from able-bodied users but it can be argued that users with disabilities may approach problems from a different perspective and may employ different strategies. For example, switch users may prefer navigating their avatar with a selection set that does not contain the BACK input, as they can move their avatar backwards by turning their avatar 180°, move forward and then turn back 180°, such as it has been implemented in PORTEM [2]. Figure 1 shows how hold-and-release scanning has been implemented in the Second Life viewer by modifying the built-in on screen keyboard. We plan to distribute our hold-and-release as an avatar attachment and collect experiences with hold-and-release from switch users.

Though 3D games and virtual worlds share similar interaction mechanisms, games require fast responses from their players as they often involve combat or some form of competition. A modification of Half Life 2 exists called Gordon's Trigger Finger [12] where players can play a multiplayer game using a single switch. Interaction is limited to being able to fire the player's gun where the AI takes care of navigation as it automatically chases enemies or finds ammo, which resembles the gameplay of a rail shooter. Future research will investigate whether hold-and-release scanning control can be used to navigate characters in first person shooters and whether it can meet stringent input requirements that are required by video games. Hold-and-release could possibly be implemented as an overlay HUD on game consoles offering switch access to 3D games on consoles.

7. CONCLUSION

The research in this paper to answer a fundamental interaction design problem: how do you control a 3D avatar using the smallest amount of input? We describe hold-and-release, a novel scanning control method that is able to provide continuous and mixed inputs using a single switch and of which navigating a 3D avatar is an example activity that requires providing such inputs. Hold-and-release was developed by leveraging data from able-bodied users and we assessed the efficiency and accuracy of hold-and-release using simulation. Hold-and-release scanning control is significantly more efficient than existing scanning systems. To provide mixed inputs hold-and-release using a selection set that contains mixed inputs yields no approximation errors, though using multistep selection to add a mixed input to an already held input is faster but has an associated error of 122 *ms* per keystroke. As 3D interaction increasingly defines how we in-

teract with software, hold-and-release scanning control may allow for users with severe motor impairments to access the educational, social and entertainment opportunities offered by these technologies.

8. ACKNOWLEDGEMENTS

This work is supported by NSF Grant IIS-0738921.

References

- [1] Linden research, secondlife, <http://www.secondlife.com>, access date: 1-14-2010.
- [2] Portem, <http://www.mirrorgames.com/en/portem.php>, access date: 11-12-2010.
- [3] Web content accessibility guidelines 1.0, <http://www.w3.org/TR/WCAG10/>, access date: 11-12-2010.
- [4] Zogan's log, <http://www.gamevial.com/playgames.php?game=zoganslog>, access date: 11-12-2010.
- [5] Virtual ability, <http://virtualability.org>, access date: 9-2-2009, 2008.
- [6] BEUKELMAN, D., AND MIRENDA, P. Augmentative and alternative communication: Supporting children and adults with complex communication needs. P. H. Brookes, Ed.
- [7] BISWAS, P., AND ROBINSON, P. Simulation to predict performance of assistive interfaces. In *Assets '07: Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility* (New York, NY, USA, 2007), ACM, pp. 227–228.
- [8] BISWAS, P., AND ROBINSON, P. Automatic evaluation of assistive interfaces. In *IUI '08: Proceedings of the 13th international conference on Intelligent user interfaces* (New York, NY, USA, 2008), ACM, pp. 247–256.
- [9] BLACKSTIEN-ADLER, S., SHEIN, F., QUINTAL, J., BIRCH, S., AND WEISS, P. L. T. Mouse manipulation through single-switch scanning. *Assistive Technology* 16, 1 (2004), 28–42.
- [10] BRODERICK, T., AND MACKAY, D. J. C. Fast and flexible selection with a single switch. *PLoS ONE* 4, 10 (10 2009), e7481.
- [11] GHEDIRA, S., PINO, P., AND BOURHIS, G. Conception and experimentation of a communication device with adaptive scanning. *ACM Trans. Access. Comput.* 1 (February 2009), 14:1–14:23.
- [12] HOANG, R., FRANKLIN, C., AND FOLMER, E. Gordon's trigger finger, one switch first person shooter, <http://www.eelke.com/oneswitchgames.html>, 2008.
- [13] LESHER, G. W., HIGGINBOTHAM, D. J., D, P., D, P., AND MOULTON, B. J. Techniques for automatically updating scanning delays. In *Annual Conference on Rehabilitation Technology (RESNA)* (2000), RESNA Press, pp. 85–87.
- [14] LEWIS, C. Hci and cognitive disabilities. *interactions* 13, 3 (2006), 14–15.
- [15] MACKAY, D. J. C., BALL, C. J., AND DONEGAN, M. Efficient communication with one or two buttons. In *Proceedings of Maximum Entropy and Bayesian Methods*, edited by, pp. 207–218.
- [16] PERKINS, W., AND STENNING, B. Control units for operation of computers by severely physically handicapped persons. *Journal of Medical Engineering Technology* 10, 1 (1986), 21–23.
- [17] PRADIPTA BISWAS, P. R. A new screen scanning system based on clustering screen objects. *Journal of Assistive Technologies* 2, 3 (2008).
- [18] ROBBINS, S. S. Immersion and engagement in a virtual classroom: Using second life for higher education. In *EDUCAUSE Learning Initiative Spring 2007 Focus Session* (2007).
- [19] ROBINSON, P. Performance comparison of different scanning systems using a simulator. In *In Proceedings of the 9th European Conference for the Advancement of the Assistive Technologies in Europe (AAATE'07)*.
- [20] SIMPSON, R., AND KOESTER, H. Adaptive one-switch row-column scanning. *Rehabilitation Engineering, IEEE Transactions on* 7, 4 (Dec 1999), 464–473.
- [21] SIMPSON, R., KOESTER, H., AND LOPRESTI, E. Selecting an appropriate scan rate: the ".65 rule". *Assistive Technology* 19, 2 (2007).
- [22] SMITH, J. D., AND GRAHAM, T. C. N. Use of eye movements for video game control. In *ACE '06: Proceedings of the 2006 ACM SIGCHI international conference on Advances in computer entertainment technology* (New York, NY, USA, 2006), ACM, p. 20.
- [23] SWINTH, Y., ANSON, D., AND DEITZ, J. Single-switch computer access for infants and toddlers. *American Journal of Occupational Therapy* 47, 11 (Nov 1993), 1031–8.
- [24] TREWIN, S., LAFF, M., HANSON, V., AND CAVENDER, A. Exploring visual and motor accessibility in navigating a virtual world. *ACM Transactions on Accessible Computing* 2, 2 (2009), 1–35.
- [25] U.S.GOVERNMENT. 1998 amendment to section 508 of the rehabilitation act. *SEC. 508. Electronic and information Technology* (1998).
- [26] YUAN, B., FOLMER, E., AND HARRIS, F. C. Game accessibility; a survey. *Universal Access in the Information Society* 10, 1 (2010).