Performance Measures of Game Controllers in a Three-Dimensional Environment

Chris Klochek, I. Scott MacKenzie

Dept. of Computer Science, York University
Toronto, Ontario, Canada M3J 1P3
{chrisk,mack}@cs.yorku.ca

Abstract

Little work exists on the testing and evaluation of computer-game related input devices. This paper presents five new performance metrics and utilizes two tasks from the literature to quantify differences between input devices in constrained three-dimensional environments, similar to "first-person"-genre games. The metrics are Mean Speed Variance, Mean Acceleration Variance, Percent View Moving, Target Leading Analysis, and Mean Time-to-Reacquire. All measures are continuous, as they evaluate movement during a trial. The tasks involved tracking a moving target for several seconds, with and without target acceleration. An evaluation between an X-Box gamepad and a standard PC mouse demonstrated the ability of the metrics to help reveal and explain performance differences between the devices.

CR Categories: H.1.2 [User/Machine Systems]: Human Factors; H.5.2 [User Interfaces]: Evaluation Methodology.

Keywords: Computer-game devices, video-game devices, performance evaluation, performance-measurement, target-tracking tasks.

1 Introduction

The computer and videogame industry is a multi-billion dollar per year interest: annual revenues recently eclipsed motion-picture box-office receipts [5]. In addition to the personal computer, there exist several different game consoles. Despite the fact that over 145 million people play video games in the United States alone [8], comparatively few studies exist that assess either the performance characteristics of video-game input devices, or that cultivate a taxonomy of interaction techniques specific to video games.

Moreover, new metrics for video-game input devices must allow researchers to characterize differences between devices beyond simple "better-or-worse" measures. For example, perhaps a prototype device has pleasing ergonomic properties, but has unsatisfactory performance when compared with an existing device. Representative tasks and performance metrics are needed to help researchers find these differences to appropriately guide design changes to an existing device.

This paper presents four new metrics to reveal performance differences between competing input devices in a first-person-genre game or simulation. We also employ two previously defined metrics to assist in performance evaluation, as well as two different tracking tasks from the literature with which to test the devices.

2 Choice of Genre

In a first-person game, the user perspective is that of an actor in the game, thus enabling a view of the action in the game from the eyes of said actor. (Other genres can have the player controlling on-screen characters from some detached perspective.) First-person games are highly popular on both personal computers and game consoles, therefore warranting more detailed study. Virtual reality simulations that also feature first-person views will also benefit from this study, as the controls in such applications often mimic game interactions.

Orienting the player in a first-person game requires methods to control both the position and the view-direction of the player in the world. These two facets of orientation are generally controlled with two different sets of device inputs. On a personal computer, players typically delegate keyboard keys to movement, and the mouse to controlling the view. This implies a binary granularity in movement speed: the user is either moving (or accelerating to top speed) or is stopped (or accelerating to all-stop). Console games have two analog joysticks to control both movement and view, thus providing the player with finer control of the player's position.

Most first-person games are action games, requiring the player to (among other objectives) attack and destroy enemy targets. Such targets appear as three-dimensional models that exist in the game world and interact with the player. A player typically orients the view of the world so that the target is visible; the player then attacks the target. Weapons for attacking typically come in two varieties: scan-hit weapons and projectile weapons. Scan-hit weapons are so-called due to the instantaneous nature of the attack. The time for such a weapon to cover any distance is zero. Therefore, the user simply acquires the target (typically in a crosshair located in the centre of the screen) and initiates the attack by pressing a key. If the user has correctly sighted the target, the attack is successful. Projectile attacks, however, require time to hit targets. If neither the target nor the player is moving, then the attack is essentially the same as a scan-hit attack. If movement of either party is allowed, then the user must lead the target. Leading is the technique of anticipating where the paths of the target and projectile will intersect in the future, based on knowledge of the present speed of both the projectile and target.

Important, however, is that scan-hit and projectile weapons are essentially the same: a scan-hit weapon is merely a projectile weapon with infinite velocity. Thus, one can treat both interactions as one. In scan-hit weapons, the user aims at the target to hit it. In the case of projectile weapons, the user aims at a mental proxy of the target.

3 Previous Work

Performance evaluation in two-dimensional pointing tasks has been studied extensively [2, 12]. The most widely used metrics are error rate and movement time. In particular, Fitts' law [4] is frequently used to quantify the overall performance of a device in point-and-click tasks via the throughput metric, defined as:

$$Throughput = \frac{\log_2\left(D/W_e + 1\right)}{MT}$$

where D is the distance to the target, W_e is the effective width of the target (the amount the user actually used in the experiment,) and MT is the time to move the distance D. MacKenzie et al. [9] introduce several new metrics that detail why differences in device performance might exist. The tasks of interest here are target seeking, however, and do not reflect some of the unique challenges a user faces when attempting to track a target. Much work in target tracking is covered in Poulton [13]. He presents a very thorough look at one-dimensional target tracking, detailing both metrics and tasks. In particular, several error-metrics are discussed that have influenced our work, including time-based error and mean-distance metrics. The tasks studied include targets moving in various patterns (constant velocity, variable velocity, instantaneous movement) but with the motion of the target constrained to a vertical line. Tracking a target in this manner is like following a pen on a seismograph; two-dimensional tracking is not explored.

Three-dimensional target tracking is developed in Ellson [1]. Specifically, he employs the time-on-target metric for three-dimensional target tracking. We used this technique as the primary error metric by which differences in device performance are measured, albeit two-dimensionally. Central to the metric is the error between the cursor position and the target position in multiple degrees-of-freedom (DOF.)

Zhai and Milgram [15] further employ measurements along multiple DOFs by examining aspects of device performance in complex three-dimensional tasks; however, their research concerns 6-DOF devices which are more sophisticated than the 2-DOF mouse and 2-DOF analog joystick. They introduce a metric for calculating the efficiency of a device in transforming a three-dimensional target from one position and orientation to another. Coordination between DOFs in the devices is central to this metric, as it is the interaction between simultaneous rotations, translations, and rotation-translations that yields efficiency.

While not actually developing a specific error metric or task set, Mithal and Douglas [11] compare mouse and isometric joystick performance on position-based tasks. They examine the results by analyzing the structure of the raw input data from both devices. The joystick is shown to have a large amount of jitter when examining its velocity component against time. The conclusion is that the joystick is highly sensitive to small vibrations, which causes its precision to suffer greatly. Their work represents a finer look at the underlying causes of the disparity in performance between the joystick and mouse for certain tasks, and not simply noting that a difference does exist.

In their paper, Jagacinski et al. [7] extend Fitts' law for the accurate prediction intercepting moving targets. Target velocity, size, and distance from the user's stylus are all factors in the new equation. Measurements related to target tracking are not performed, however, as the research is only concerned with rapid acquisition movements associated with typical Fitts' law research. Their formulation is also a throughput measurement.

4 TASKS

4.1 Task Focus: Target Tracking

As noted, first-person games involve acquiring either a target, or a mental proxy of the target, and then attacking. Acquiring a target is a task that has already been studied extensively. The two tasks presented herein attempt to reflect what happens after target acquisition.

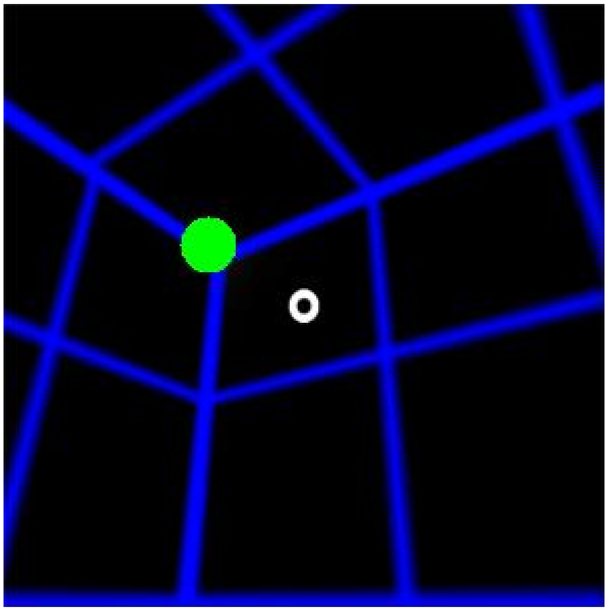

Figure 1. Image taken from the software showing the three-dimensional environment. The white circle is the crosshair. The green circle is a target.

Once a target is acquired, it is usually tracked. (Most opponents are not felled easily, and require constant attacks to overcome.) The task of tracking is usually given to the user, though many console games assist the user in making small corrections to the view. Once the user has acquired the target, small motions the opponent may make are tracked automatically. Halo 2 [6], by Bungie Studios features such an assist. Other games offer more extreme forms of tracking, such as Metroid Prime [10] by Retro Studios and Nintendo, where the user actually "locks-on" to the target by pressing a key. The view then continuously tracks the target without user-input. Translation movements by the player are in a circle about the tracked target. In this fashion, the player can better dodge incoming attacks, but still maintains a centered view on the target. Without major assistance, the user relies on his or her skill in following the target, as well as on the performance of the input device. This tracking performance is the focus of the tasks and metrics presented in this paper.

These tracking tasks are variations on the velocity-ramp tracking tasks outlined in Poulton [13].

4.2 Task 1: Constant Velocity

The first task features the user in a three-dimensional world, in this case represented simply by a textured cube (see Figure 1.) A checkered background texture was chosen to supply the user with a sense of orientation in the environment. The user only controls the orientation of the view; the position is fixed.

Targets appear in the field of view. The user changes the orientation of the view until the target is bracketed by the crosshair. The user then presses a button on the controller, and the target starts to move. The path of the target is constrained to the perimeter of a circular cross-section of a sphere, with the user at the origin of this sphere. The center of the cross-section is also the origin of the sphere. Therefore, the distance between the target and the user is constant. The perceived path of the target from the perspective of the user is a straight-line. Finally, the speed of the target is set to a constant: therefore, from the perspective of the user, the targets do not undergo any acceleration.

This task was designed for target tracking in the absence of acceleration. This kind of tracking also exposes differences between zero-order and first-order input devices. An analog joystick is an example of a first-order device, as it can track a target without movement of the user's hands once the target is acquired. This is because the joystick's displacement is converted into the view's rotational velocity. A mouse still requires the user to move their hands to track the target, as the *change* in displacement being sensed by the mouse is mapped onto the view's rotational velocity. (It is possible to map a joystick into a zero-order device, but doing so renders the device nearly useless for tracking tasks.)

Targets should be distributed to test the tracking capabilities of the device in different directions; the setup in this paper has sixteen targets appearing on a circle within the field of view, with 22.5 degrees of arc separating each target. Only one target is visible at a time. During a trial, a target moves from its starting position to a position on the opposing side of the circle. Once the trial is complete, another target becomes visible, and the user then tracks that target, and so on.

Target speed must be manageable by the device and not so slow as to be overly simple. This speed depends on the context of the application the device is being measured in. Given that computer-games are the subject of this paper, a speed was empirically chosen that was considered best representative of the range of speeds a target might undergo in a typical video game.

4.3 Task 2: Variable Velocity

This task features the user in the same environmental setup as before. Targets now appear directly in the centre of the view. After the user presses a button, the target begins to oscillate along a specified axis, perpendicular to the direction the user is facing. The length of the oscillation increases as time passes; the time to complete one full oscillation, however, remains the same. Therefore, the target experiences acceleration in both direction and speed.

Unlike the first task, both zero- and first-order devices must be manipulated continuously to track the target in this task.

Targets move from the center of the screen along an axis set at integer multiples of 22.5 degrees; this is similar to the Constant Velocity task. The duration of the oscillation, the speed of the oscillation and the maximum oscillation length are application dependent.

5 Performance Metrics

We have developed several performance metrics intended to expose device idiosyncrasies. We also employ several metrics from the literature for conventional target-tracking measures.

The novel metrics described in this paper have been designed with the intention of determining both how and why devices differ in performance when used for target tracking. The development of these measures relied on examining key features of device construction, as well as the specificities of the tasks they are involved in.

The order of control of the device plays an important role in overall performance. Zero-order devices map all positional motion experienced directly to positional motion on-screen, while first-order devices convert the same motions into positional-deltas. These crucial differences in control mapping require attention from new metrics. In particular, the fact that time must elapse in order to integrate a velocity into a position for a first-order device suggests that a slight lag might be introduced into the track, as compared with a zero-order device.

Additionally, any new measures must be keyed towards the tasks to be performed. Of primary concern is the accuracy of the track. Accuracy measures obviously include the amount of time spent correctly tracking the target. Maintaining a track, however, also requires matching the velocity and acceleration of the target. Therefore, examining the error in acceleration and velocity should yield important performance information. Smoothness and consistency are also desirable aspects to a track. Analysis of acceleration will help contribute to this information, but determining the amount of time spent trying to reacquire a target after a tracking error occurs will indicate the responsiveness of a device. A more responsive device will likely give a more consistent and smooth track.

For all metrics, N is the number of samples and t_i is the i^{th} sample of the projected screen-space position of the target. Screen-space is the (x, y) location of the three-dimensional target as projected onto the user's view, measured either in pixels, or as a pair of normalized scalars, where –1 and +1 correspond to the extremes of the screen, both horizontally and vertically. C is the screen-space position of the crosshair—a constant, as the crosshair remains in the center of the screen. These metrics are designed to elicit additional information about the possible reasons behind a given device's performance, rather than replace the more traditional error-rate and throughput measures.

5.1 Time on Target (TOT)

This metric is a continuous version of its discrete counterpart, error-rate. For each sample of data, if the cursor is within the target's screen radius, the target is counted as "in," otherwise the target is "out." The TOT for a trial is the sum of the "ins" divided by the total number of samples. This is the metric originally devised by Ellson [1]:

$$TOT = \frac{\sum_{i}^{N} In(\vec{t}_i)}{N}$$

('In' returns '1' if the target is contained within the crosshair, '0' otherwise.)

5.2 Mean Target Distance (MTD)

Mean Target Distance is the average distance between the target and the crosshair. This metric is correlated with TOT, but reveals more information about how close the user was to the given target. This metric is mentioned in Poulton [13]:

$$MTD = \frac{\sum_{i}^{N} \left|\vec{t}_i - \vec{c}\right|}{N}$$

5.3 Mean Time-to-Reacquire (MTR)

When following a target, it is likely that the participant will make a mistake and lose track of the target. The participant will have to make a correction to reacquire the target, which will take some amount of time. It is again likely that multiple such periods of target loss will occur; MTR reports the mean time it takes a participant to reacquire a target immediately after tracking is lost:

$$MTR = \frac{\sum_{i}^{T} time_i}{T},$$

where $time_i = time\_reacquired_i - time\_tracking\_lost_i$ and there are T such time-intervals.

5.4 Mean Speed Variance (MSV)

That the MTD is zero for a perfect track implies that the position of the crosshair and the position of the target are identical. It therefore follows that the perceived velocity of the target in a perfect track is zero (it occupies the same point on the screen at all times.) This gives us the Mean Speed Variance metric:

$$MSV = \frac{\sum_{i}^{N-1} \left|\vec{t}_{i+1} - \vec{t}_i\right|}{N-1}$$

5.5 Mean Acceleration Variance (MAV)

Analogous to MSV is the Mean Acceleration Variance. Given that the perceived velocity is zero in a perfect track, the perceived acceleration must also be zero. There is a similar measure of MAV devised in [3], referred to there as "smoothness" and is used to gauge the instantaneous curvature of a movement. This formulation is concerned with the overall acceleration recorded from a device:

$$MAV = \frac{\sum_{i}^{N-2} \left|\left(\vec{t}_{i+2} - \vec{t}_{i+1}\right) - \left(\vec{t}_{i+1} - \vec{t}_i\right)\right|}{N-2}$$

5.6 Target Leading Analysis (TLA)

Target Leading Analysis represents any bias of the device while tracking. Some devices might cause a user to consistently lag behind the target while tracking; others, in front of the target.

Performing this calculation in screen-space is difficult. This is due to the fact that the projected motion of the target can vary as the user adjusts the view, while the world-space motion of the target may remain constant. As a result, the calculated screen-space velocity of the target can vary wildly, which makes leading and following statistics meaningless.

Instead, the calculation is performed in world-space. This is accomplished by keeping track of the position of the object with respect to the viewer, as well as recording the direction the viewer is looking. Both quantities are represented as three-dimensional vectors. Normalizing these vectors projects the values on the surface of a unit-sphere. From this point, calculating the TLA is simple. The instantaneous velocity of the target, as well as the position of the target on the unit-sphere, is used to construct a separating-plane. The viewing direction is then compared with this plane, determining both the magnitude of the displacement from the plane, as well as its sign. This value is then summed into a total, and the mean is calculated. Two such sums should be tallied, as they will be used to determine the mean displacement from both the positive and negative sides of the plane.

$$TLA_{Positive} = \frac{\sum_{i}^{N-1} Positive\_Dist(Plane(\vec{t}_{i+1}, \vec{t}_i), \vec{v})}{P}$$

P is the number of times the view vector is on the positive side of the plane. t_i is the position of the target on the unit-sphere. $\vec{v}$ is the normalized view-direction. *Plane* will construct a plane out of the two parameters. *Positive_Dist* will return either the distance from the plane to the point $\vec{v}$, or zero if the distance is less than zero.

The equation for negative distance is symmetric, and therefore omitted.

Poulton [13] originally defined an "error-in-time" metric with respect to one-dimensional tracking. The ideal and actual tracks would be super-imposed on one-another and any perceived deviations between the two would be counted as the participant leading or following the target. Thus, the error committed was considered to be one of time. This metric considers the error to be one of position.

5.7 Percentage View Moving (PVM)

The final performance metric examines the smoothness a device affords a user when tracking a target. Given that a target is always moving, keeping the target bracketed by the crosshairs implies that the view is continuously moving. If the view should cease moving, then an error in tracking has occurred, and is recorded.

To perform this analysis a threshold value is set to determine the minimum change in view required for a significant change. Any movement at or below this threshold is considered no change and is recorded. As in TLA, world-space data about the view direction is required for calculations:

$$PVM = \frac{\sum_{i}^{N-1} Significant\_Change(\vec{v}_{i+1}, \vec{v}_i)}{N-1}$$

$\vec{v}$ is again the normalized view-direction.

6 Experimental Method

The following experimental method was used to test the proposed tasks and metrics.

6.1 Participants

Ten paid participants (8 male, 2 female) were recruited. All were regular mouse users. One was an "avid" computer game player, while three were "somewhat frequent" computer game players—all four users had had some experience with console gamepads. No other participants had previous experience with gamepads before, and were not game-players. All were right-handed.

6.2 Apparatus

The experiment used an AMD Athlon XP desktop PC running Windows 2000, along with a 19" flat-panel LCD monitor with a 12 ms refresh time. Screen resolution was 1280 x 1024 pixels. The video card was an ATI Radeon 9800. The software was developed with Visual C# using the Beta 2.0 .NET framework and the Managed DirectX 9.0 API.

Input consisted of two devices: a Microsoft X-Box gamepad (Type S) controller, and a standard three-button Microsoft optical USB mouse. The X-Box controller was modified to fit a standard PC USB slot, and custom device drivers for the gamepad were acquired from the Internet [14].

The standard USB mouse sample time is 125 Hz; the gamepad was assumed to have a similar sample rate. Input was recorded from both devices at a rate of 20 Hz.

The mouse's left-right movement was mapped to control the left-right movement of the view. The up-down motion of the mouse was mapped into up-down motion of the view, but individual users were allowed to invert the mapping if they found it more natural. There were no acceleration effects added to the mouse: mouse sensitivity was fixed for all participants.

The X-Box gamepad's right analog stick was used to control view movement. The position of the stick controlled the velocity of the view. The speed of this movement was linearly proportional to the displacement of the stick from rest. Up and down motion was similarly handled: again, the option to invert this axis was presented to the participant if they felt it was more natural. Sensitivity was again fixed for all users. This ensured that the range of motion that the stick underwent would be able to allow it to track a fast-moving target.

6.3 Procedure

Each participant performed both tasks with both devices. Participants first used the mouse to perform both tasks, followed by the gamepad: learning was not assumed an issue across devices, as the two input methods were considered sufficiently dissimilar. Half the participants did the Constant Velocity task first and the other half did the Variable Velocity task first, in order to balance any possible learning effects between tasks.

Before testing with each device, participants were instructed on the purpose of the experiment, and were allowed a brief time to familiarize themselves with the devices, as well as to practice orienting themselves in the three-dimensional environment.

6.3.1 Constant Velocity Task

This task is as outlined in the previous section. The user's field of view was 60 degrees with a viewing window of 600 by 600 pixels. The view was first placed in "rest position," with the camera staring down the positive z-axis. A spherical green target then appeared on the screen, and the user would seek to it. Once bracketed, the user pressed a button on the controller to cause the target to move. Targets were situated 50 units from the user. Targets rotated about the origin at 20 degrees per second. The target moved for four seconds, at which point the view returned to the rest position and another target appeared. Targets appeared in a circular distribution at 22.5-degree intervals about the screen, for a total of sixteen targets. Targets moved from one extreme of the screen to the other in a perceived straight line. While the user correctly bracketed the target, the target's colour was red; otherwise it was green.

This experiment consisted of ten blocks, with each block consisting of the sixteen tracking tasks, running for approximately twenty minutes.

6.3.2 Variable Velocity Task

This task is as outlined earlier. Field of view and screen size are as in the Constant Velocity task. As with the Constant Velocity Task, the view is reset to rest position. A target then appears directly in the center of the view, bracketed by the crosshair. Targets appear 50 units from the user. The user pressed a button on the device, causing the target to oscillate along an axis. The oscillation is done in world-space using a simple cosine formula to achieve the result. Movement continued for 20 seconds. Each target moved a maximum of approximately 45 units from the start of the oscillation. Each target took approximately 1.5 seconds to move between extremes on its path. After twenty seconds, the view returned to a rest position and the next movement axis was selected. As with the Constant Velocity task, the colour of the target reflected whether or not the user was tracking it.

Eight sets of oscillations constitute one block of trials. Six trials were conducted, lasting approximately twenty minutes.

6.4 Design

The experiment was a 2 x 6 and 2 x 10 within-subjects factorial design with one between-subjects factor: namely, task-order. Within-subjects factors and levels for the tasks were:

- Device {mouse, gamepad}
- Block {Either six or ten blocks, depending on the task}

7 Results and Discussion

Counter-balancing between the two groups proved effective, as neither task showed significant group interaction. Linear regression was used to determine when learning was no longer a significant factor: with both trials, the first block of results was discarded.

(See Table 1 for a complete listing of ANOVA results for each task and metric.)

All measures except MSV revealed statistically significant differences between devices in both tasks, indicating the measures could discriminate between the devices. That MSV did not show significant differences turned out to be relevant, however.

To appreciate the differences in the device, we must first ascertain which performed with fewer errors. To wit, we utilize the TOT and MTD metrics. TOT and MTD are closely related. TOT is essentially a binary-threshold version of MTD. A user could have both a consistently non-zero value for MTD and a perfect TOT score, given that the targets are of non-zero size. These two metrics are essentially continuous analogues of error-rate in pointer-selection tasks, and are therefore the most relevant metrics when discussing the question, "Are the two devices different?" Clearly the devices show markedly different error rates, and thus it is safe to conclude that the mouse was the more accurate device when performing these tasks. This result concurs with other such research (e.g. [11].) Why these devices produce different error rates is helped explained by the remaining error metrics.

MAV and MSV together yield interesting distinctions between the device interactions. In both tasks, mean acceleration in the gamepad device was considerably lower than in the mouse device. The difference in MSV in both tasks between both devices was statistically insignificant. As previously argued, a low MSV and MAV would both be desirable. Thus, on this naive assumption, it appears the gamepad is at least as effective as the mouse. Of course, this is not the case. Given that participants using the gamepad were interested in achieving high accuracy but were unable to, the disparity between the accelerations and the similarity of velocities indicates that users were simply incapable of fine-tuning the position accurately. The low acceleration of the gamepad suggests that users were not able to quickly adjust their tracking of the target as with the mouse. In this case the *higher* acceleration afforded by the mouse was what enabled the users to perform better. That both devices delivered approximately the same results in MSV tells us that the users were able to track the velocity of the targets equally well. Seeking to the position of the target was therefore the main difference between the devices.

The differences in the ability of the devices to control the fine-bracketing of the target are reflected in the MTR metric. In both tasks, MTR was significantly lower in the mouse than in the gamepad, indicating when a user lost track of the target, reacquiring the target was accomplished faster with the mouse. The data support the notion of fine-tuning mentioned above, as reacquiring a lost target quickly implies a fast adjustment of the view, which in turn implies a quick motion of the device—a larger acceleration.

It was observed that very little clutching occurred while tracking targets with the mouse. Clutching is a phenomenon experienced by a mouse-user when moving the mouse in an extreme direction. If the required movement is sufficiently large, then the user may run out of space to move the mouse, or their limbs may not be sufficiently long enough to complete the motion. In either case, the user must pick up the mouse and reposition it closer to themselves, away from any obstructions, and continue the motion. This causes a momentary loss of tracking as the user performs these actions, and will potentially require that the target be re-acquired. The tasks presented to the participants, however, resulted in only slight clutching at the start of the sessions. Users were observed developing coping strategies to help them avoid clutching as they became more familiar with the tasks (e.g., to track a target moving in a straight line, start with the mouse on the opposite extreme of the direction in which it will travel.)

Variable	Mouse		Gamepad		F-Statistic,
	mean	sd	mean	sd	p-value
TOT	0.9510	0.0502	0.7982	0.1007	18.46,<.001
	0.7639	0.1235	0.4136	0.1832	25.14,<.001
MTR	1.921	1.326	8.108	3.424	28.39,<.001
	5.000	1.393	14.294	5.980	22.91,<.001
MTD	13.180	3.232	21.854	4.752	22.78,<.001
	24.24	7.06	54.66	27.49	11.49,<.005
MSV	315.15	78.90	275.70	62.46	1.54,>.05
	2978	697	4383	2424	3.10,>.05
MAV	256.99	36.01	86.55	22.44	161.33,<.001
	2323.8	471.8	1272.9*	192.9*	34.69*,<.001*
TLA	12.210	4.255	17.929	3.779	10.10,<.005
	38.89	12.10	84.44	35.66	14.63,<.001
	-14.345	6.024	-23.819	8.823	7.87,<.05
	-92.92	46.25	-184.53	125.38	4.70,<.05
PVM	0.81055	0.06131	0.88378	0.04368	9.46,<.01
	NA	NA	NA	NA	NA

Table 1. Constant and Variable Velocity Task Results

(Constant Velocity results are the top numbers in each cell; Variable Velocity, the bottom.)

*(*Two outliers were removed from this computation: their scores were more than five times the computed mean.)*

For the Constant Velocity task, the PVM metric was employed, which gives further credence to the fine-tuning hypothesis. The mouse featured a lower PVM metric than the gamepad, meaning the view was in motion more often with the gamepad than with the mouse. Again, as initially argued, a larger PVM suggests a smoother, more consistent track, which ought to indicate a better TOT and MTD score. That the opposite was observed again suggests the way in which the mouse was utilized: users would not move as smoothly as with the gamepad, but would stop momentarily (perhaps to ascertain their position on the target,) and then enact a much greater acceleration of the view to correct for the new location of the target. Performing this action on the gamepad was harder, evidently.

The reasons for this difficulty in fine-tuning are likely due to the first-order control of the gamepad's analog stick. Mithal et al [11] clearly outline the difficulties in using a first-order device for a position-based task. In general, having a user subconsciously integrate a velocity and acceleration into a position is difficult to do both rapidly and accurately. Controlling orientation of the view in a three-dimensional task certainly falls under this category.

It may not seem obvious that the rotation of a view in a simulated three-dimensional space is equivalent to a positional movement. If one instead considers the environment as projected onto the interior of a sphere, which in turn is "unwrapped" and placed on a two-dimensional surface (like the projection of the Earth onto a map), then orienting the view is much like freely scrolling through this two-dimensional map, which is obviously a positional task.

Finally, TLA showed statistically significant differences between the devices, indicating that users were leading or trailing the targets more when using the gamepad than when using the mouse. Paired *t*-tests reveal statistically significant differences between the kinds of leading for both devices: unsurprisingly, users were trailing behind the targets more than they were leading them. Again, this supports the fine-tuning hypothesis, as the larger trailing in the gamepad suggests users that could not seek accurately to the new target position.

8 Conclusions and Future Work

We have presented a number of metrics to elicit differences between gaming devices in three-dimensional first-person games. These measures are continuous and yield rich data on the way a device performs a continuous movement, and can therefore help describe why differences in error-rates exist between devices.

Two tasks were utilized from the tracking literature that reflected some of the activities that users might experience in a typical first-person video game. Both featured a tracking-based testing regime, designed to mimic some of the skills a potential user would need when playing first-person games. The use of varying amounts of acceleration was useful in further illuminating differences between devices more than a simple constant-velocity test.

In particular, the MSV and MAV metrics helped describe why performance differences exist between the mouse and gamepad. Both devices allowed the participants to track the target's velocity equally well; the mouse, however, allowed the participants to enact a much greater acceleration in order to correct for errors in position. The much smoother acceleration of the gamepad, while theoretically desirable, did not benefit users when making small positional corrections. This performance difference is almost certainly due to the first-order control of the analog stick. Enacting an acceleration of large magnitude but short duration requires an extremely rapid movement of the thumb to position the stick at an extreme point on the gamepad, quickly followed by moving the stick back to the previously-held position. This is a highly sensitive and therefore error-prone maneuver, and is likely too difficult to perform consistently and accurately by most users.

The small range of motion of the analog-stick makes these kinds of motions even more difficult. If an individual is tracking at a constant non-zero velocity, then the stick must be held fixed at some off-centre position. Enacting a large acceleration to compensate for a lagging track (for example) would therefore be even more difficult, as the maximum acceleration possible would be limited by the reduced distance of the displaced stick from the extreme range of motion of the stick. This problem is somewhat analogous to performing a clutching maneuver with a mouse: there is insufficient physical range to perform a sufficiently large displacement of the stick. Future tasks for gaming devices that focus on first-person games will need to account for translational motion, in addition to the view-orientation tasks presented in this paper. It is likely that such tasks may reveal interactions across the translation and orientation domains, as suggested by Zhai and Milgram [15].

What was not explored, however, was the relationship (if any) between these metrics and the standard measurement of pointing devices, throughput. The throughput of the devices could not be measured during these trials, as throughput is defined in terms of both error and movement-time. Movement time was a constrained quantity in this experiment, due to the time-based nature of the tasks. Relating these metrics and throughput would no doubt place them in a more secure role when used to evaluate video-game input devices.

References

[1] Ellson, D.C., The independence of tracking in two and three dimensions with the B-29 pedestal sight. In *Report TSEAA-694-2G,* 1947, Aero Medical Laboratory.

[2] Ergonomic requirements for office work with visual display terminals (VDTs) - Part 9 - Requirements for non-keyboard input devices. In *ISO ISO/TC 159/SC4/WG3 N147*, May 25, 1998. International Organisation for Standardisation.

[3] Fischer, C. A. and Kondraske, G. V. A new approach to human motion quality measurement. In *IEEE/EMBS conference,* 1997, pages 1701-1704.

[4] Fitts, P.M. The information capacity of the human motor system in controlling the amplitude of movement. In *Journal of Experimental Psychology*, 1954. **47**: pages 381-391.

[5] Gaming goes to Hollywood, in The Economist. 2004.

[6] *Halo 2*. 2005, Microsoft Game Studios: http://www.bungie.net/Games/Halo2/.

[7] Jagacinski, R.J., Repperger, D. W., Ward, S. L. and Moran, M. S. A test of Fitts' law with moving targets. In *Human Factors,* 1980. **22**: pages 225-233.

[8] *Life is Just a Game.* The New Atlantis, Winter 2004.

[9] MacKenzie, I.S., Kauppinen, T. and Silfverberg, M. Accuracy measures for evaluating computer pointing devices. In *ACM CHI,* 2001, pages 9-16.

[10] *Metroid Prime*. 2002, Nintendo: http://www.metroid.com/prime.

[11] Mithal, A.K. and Douglas, S. A. Differences in movement microstructure of the mouse and the finger-controlled isometric joystick. In *ACM CHI,* 1996, pages 300-307.

[12] Murata, A. An experimental evaluation of mouse, joystick, joycard, lightpen, trackball, and touchscreen for pointing: Basic study on human interface design. In *Fourth International Conference on Human-Computer Interaction,* 1991, pages 123-127.

[13] Poulton, E.C. *Tracking skill and manual control*. 1974, New York: Academic Press.

[14] *XBCD X-Box Windows Device Drivers*, http://www.redcl0ud.com/xbcd.html.

[15] Zhai, S. and Milgram, P. Quantifying coordination in multiple DOF movement and its application to evaluating 6 DOF input devices. In *ACM CHI,* 1998, pages 320-327.