Evaluating Thrust Models for Virtual Flying Game Mechanic

Samuel Van Cise

12/13/16
2.671 Measurement and Instrumentation
Monday PM
Prof. Peter So

Abstract

In an effort to determine an optimal a set of thrust parameters for a flying mechanic in a virtual reality game, players fly in a virtual environment while their performance is measured. Developers want to know what the ideal model and magnitudes are for the thrusters in this setting, so that they can create the best experiences for their players. Players are asked to fly to different target distances with the different thrust parameters, and their flight path is recorded. Their error to an optimal path was determined to compare the effectiveness of the flight path at each thrust value. It was determined that there was no significant difference in the error found between isotropic and anisotropic thrust models, or between two thrust magnitudes. This suggests that developers are free to choose what works best for their game without compromising the player's ability to control their movement.

1. Introduction

Earlier this year a new wave of consumer virtual reality devices have become available, which has brought an increasing number of game developers to platforms such as the Oculus rift and HTC Vive. When worn, these headsets allow players to see into another virtual world, which developers can use as the setting for a game. In the process of making a game in VR, developers must design and implement a way to move around in the virtual world, known as a locomotion mechanic.

Many types of locomotion mechanics exist, including teleportation, walking in place, along with others. This study describes a new type of locomotion mechanic which gives a player a thruster in each hand, allowing them to fly by pointing them at the ground. The exact physics behavior of the thrusters can be controlled, including how much force the thrusters deliver and whether or not that force is isotropic. There will be two comparisons then, between two models for thrust, isotropic and anisotropic, and two different thrust magnitudes of the anisotropic model. This experiment seeks to if determine there is a difference in players ability to control their flight across these thruster settings by measuring the accuracy of players' movements over repeated trials.

Comparing and tuning these models will be specific to this mechanic, but the general methods can apply to testing for any locomotion mechanic. This experiment will provide methods and procedures for other VR developers wanting to perform an analysis of their locomotion mechanic. Players will be asked to fly to different target distances with the different thrust values. Their will flight path is recorded and compared to an optimal path to evaluate the effectiveness of the various thrust models.

2. Background

The HTC Vive uses gyroscopes, accelerometers, and a set of photo sensors to track its position. Two base stations flood the room with an array of infrared LEDs, and a pair of lasers sweep the room at a frequency of 60 Hz. The photo sensors on the headset and controllers allow the computation of a "pose" which contains the position and orientation of the headset and each controller. [1] Using the SteamVR plugin for the Unity game engine both the player's hands and

head are tracked and mapped into a virtual environment for testing. A player sees this thruster in the virtual world, which moves with his hands via the tracked motion controllers. When the player pulls a trigger on the controller, the thruster simulates a force as a function of the orientation of the thruster.

2.1 Thruster Functionality

The existence of the thrusters in a virtual space means there is more flexibility in how they function. As a developer, it might be advantageous to create a thrust model which does not naturally fall out of standard engineering practices. In this study, an anisotropic thrust will be implemented such that horizontal force is greater than the vertical force. This could be used by a developer in a game environment in which the player is asked to navigate a space which is mostly flat, with little elevation change. There is an indication to think that isotropic thruster models are easier to control because their physical naturalness will carry into the virtual environment

The orientation for the Cartesian coordinate system for the virtual world is as follows: x is right, y is up, and z is forward. The thrusters have been implemented such that a force is exerted on the center of mass of a virtual body. The direction of the force is determined by unit vectors extending out of each of the player's fists where each component is multiplied by its respective component in a scaling vector, $\boldsymbol{v}_s$. The first part of the experiment will compare using a force which is isotropic vs one that is anisotropic using $\boldsymbol{v}_s = \left(16.67\frac{N}{kg}, 16.67\frac{N}{kg}, 16.67\frac{N}{kg}\right)$ for isotropic, and $\boldsymbol{v}_s = \left(16.67\frac{N}{kg}, 10\frac{N}{kg}, 16.67\frac{N}{kg}\right)$ for anisotropic (see Fig 1). The second experiment compares an anisotropic value of $\boldsymbol{v}_s = (25\frac{N}{kg}, 15\frac{N}{kg}, 25\frac{N}{kg})$ to the previously mentioned values. Both anisotropic vectors have the same ratio of horizontal to vertical thrust, and are only varying in total magnitude. These three sets of parameters were specifically chosen to represent all sides of the divide in player preferences when using the mechanic.

The forces exerted by the thrusters are simulated by Unity's physics engine. The position is computed by integrating acceleration from Newton's Second Law, where gravity acts in the negative y direction.

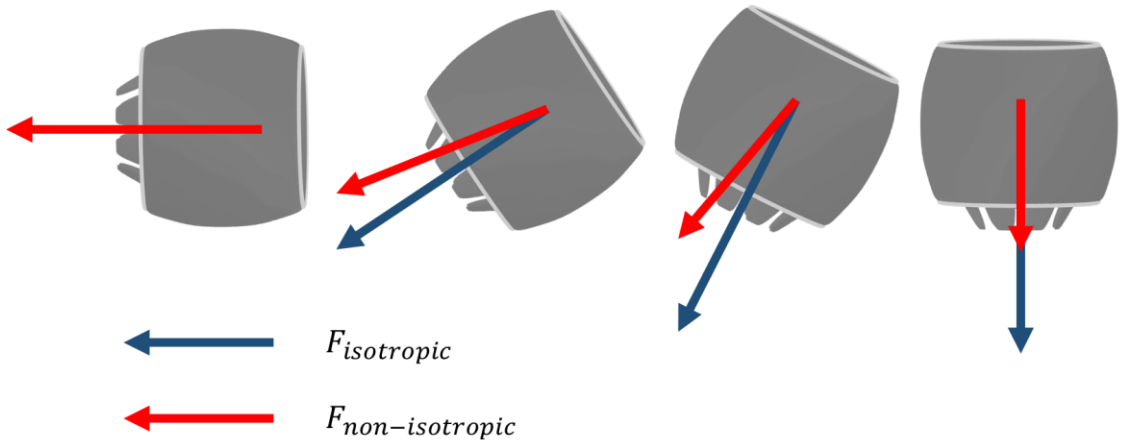

Figure 1: A player sees this thruster in the virtual world, which moves with his hands via the tracked motion controllers. When the player pulls a trigger on the controller, the thruster simulates a force in the direction depicted above, depending on whether the force is isotropic or anisotropic.

2.2 Naturalness of Other Locomotion Mechanics

A locomotion mechanic similar to the flying implemented in this study is Walking-In-Place (WIP), because they both allow the player continuous movement in the virtual space while maintaining stationary in their physical environment. This mechanic is especially suitable where spatial, technological, and financial constraints may be prominent. Distler et al. [2] has shown that individuals experience a reduction in the perceived optic flow when walking on a treadmill. That is, the walking speed appears to be faster than the visually interpreted speed. The optical flow rate can be increased, so that the virtual world moves past the user at a rate faster than their input to compensate for this. Nilsson et al. [3] showed that there exists a range of perceptually natural visual gains of 1.65 to 2.44 for WIP at a rate of 1.8 steps per second. This suggests that speeds above these gain values in our mechanic may be too unnatural to control.

A study by Slater et al. [4] indicated that individuals in a virtual environment experience a higher subjective sense of presence in that world when the move by WIP rather than flying across the ground at the push of a button. M. Usoh et al [5] confirms this, and adds actual walking in a room scale environment. This was determined by asking participants how simple, how straightforward, and how natural each of the locomotion methods were.

While the literature reflects an understanding of what is more natural in a virtual environment, it does not consider how effective the "less natural" mechanics are purely as locomotive methods. This only informs game designers as to what mechanics work well for games they want their players have a higher sense of presence in, rather than informing them on the ability of a player to effectively use the mechanic in a gameplay situation. This experiment will inform developers by giving them methods to determine the accuracy of the ability of players to move, without considering the perceived naturalness.

3. Measuring Players' Flying Abilities

Players were first introduced to the flying mechanic by a verbal description and series of images of in game characters flying around. They were given thrusters and the opportunity to learn to use them. Then, players were given a target and asked to fly to it as quickly as possible, and their position will be recorded for later analysis. The process was repeated for each set of thrust values, with the same amount of time to get adjusted to each one.

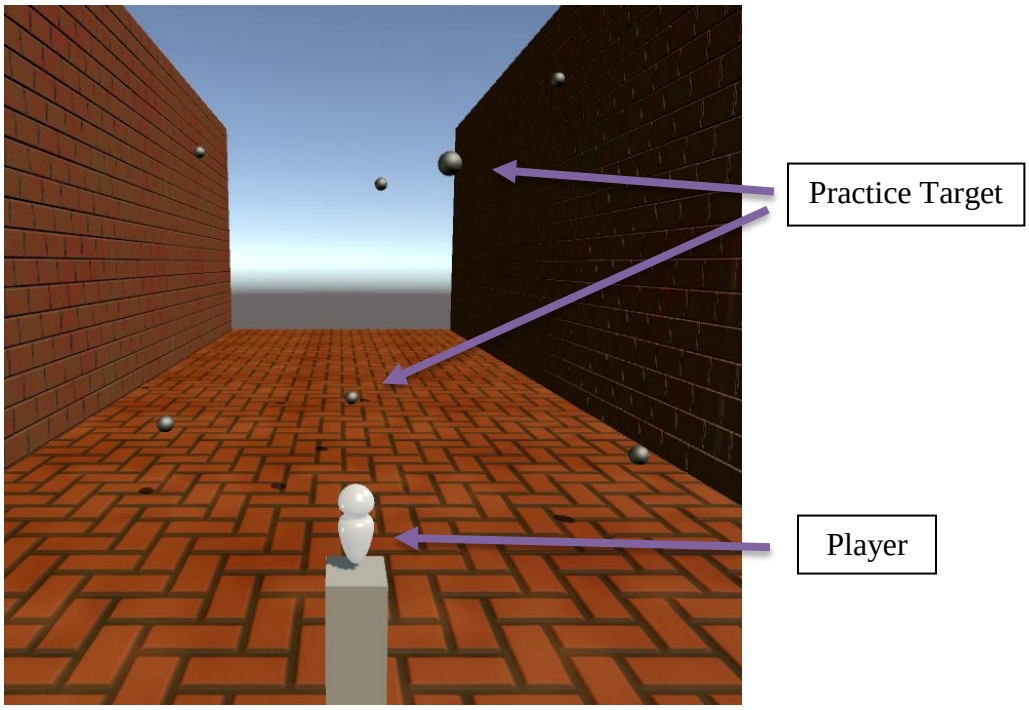

Figure 2: The room players were in to warm up and learn each new thrust value in. The floating spheres are targets sample targets so that players can practice their runs and best learn the various thrust values. The brick walls give players reference to gauge their position and speed while not distracting the player with extraneous information.

3.1 Recording Flight Path

Using the SteamVR plugin for Unity, it was possible to use the inputs from the HTC Vive in the Unity Engine. A simple scene was constructed to give players physical references for their speed, but not be distracting to the players (Fig. 2). A C# script was written to record the position of the player's head outputted by Unity's physics engine at a rate of 90 Hz. A sample rate of 90 Hz was chosen because it is also the refresh rate of the display, and therefore the maximum possible frequency at which the player can be making decisions based on the visual output of the system. The script was also responsible to placing a target in front of the player, and stopping data collection upon the collision of the player with the target. The script allows a moderator to insert the settings for the target and thrusters and turn them on and off. The player has a head of radius 10 cm, and the targets have a radius of 50 cm. This means that data collection will be stopped anytime the player reaches within 55 cm of the target distance.

3.2 Teaching Players to Fly

Test players were selected based on their fit with the target demographic of the game which this mechanic is being developed for. All players had previous virtual reality experience, but never used the flying mechanic before. Players were first given an explanation of the function of the thrusters, and a visual reference to how they work. Players were given 10 minutes in the test room to adjust to the specific thrust values, and had a set of sample targets to make practice runs. None of the practice targets were the same target distance as the official test runs, but were of the same order of magnitude.

3.3 Data Collection

Players were first given an explanation of the function of the thrusters, and a visual reference to how they work. Players had 10 minutes in the test room to adjust to the specific thrust values, and had a set of sample targets practice making runs as fast as possible. This learning time was included in the experiment to remove any bias associated with the learning curve for each individual thrust value, and improve the consistency of the results.

The target distances which were used were (0m, 10m, 20m) and (0m, 5m, 20m), which will be referred to as steep and shallow targets, respectively. The players were randomly given one of the three thrust values to learn (isotropic: $\left(16.67\frac{N}{kg}, 16.67\frac{N}{kg}, 16.67\frac{N}{kg}\right)$, low anisotropic: $(16.67\ N, 10\ N, 16.67N)$, and high anisotropic: $(25\frac{N}{kg}, 15\frac{N}{kg}, 25\frac{N}{kg})$ They flew to one of the randomly selected the target three times, and their path was recorded with the previously mentioned script. Then they flew to the remaining target at this thrust value, and the entire process was repeated for each of the thrust values.

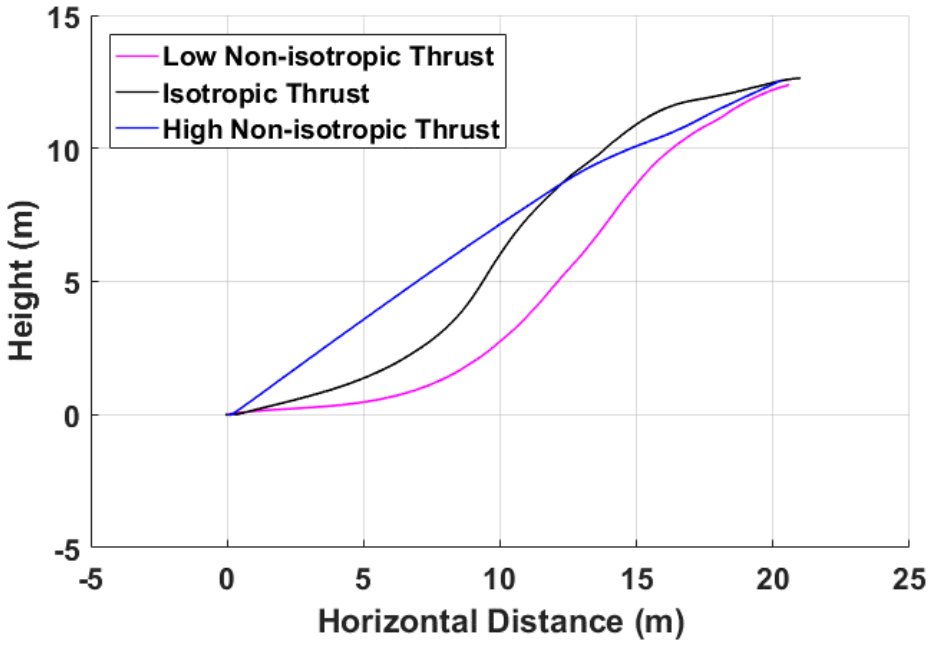

Figure 3: Individual sample flight path at each of the three thrust values

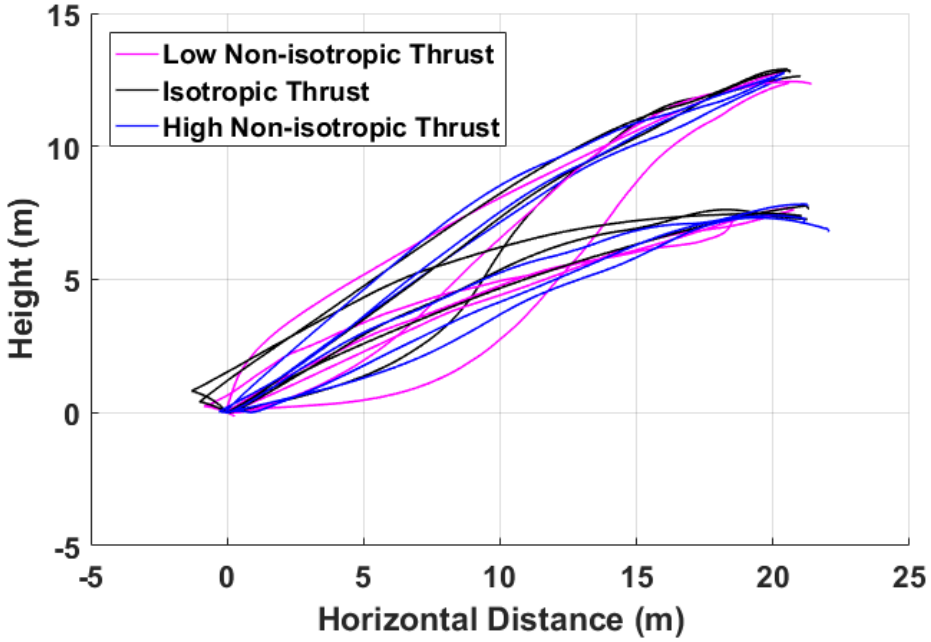

Figure 4: All flight paths of a single person, over each set of thrust parameters and target distance.

3.4 Error Definition

To evaluate how accurate a player's flight path is, an optimal path was defined and paths will be compared based on their error to that optimal path. For the case tested in this experiment, where a player is flying from one point in space to another, the ideal path is simply a straight line connecting the two points. Additionally, to normalize for each distance, the error will be divided by the error of a "worst path" for that given target. For this experiment, the worst path is defined as moving only horizontally until beneath the target, and then straight up. This is depicted in Fig 5 below. The definition of error of $y_{path}(x)$ from $(x_1\,, y_1)$ to $(x_2\,, y_2)$ is:

$$error = \frac{\int_{x_1}^{x_2} \left| y_{path}(x) - \left(\frac{y_2 - y_1}{x_2 - x_1} x + y_1\right) \right| dx}{\int_{x_1}^{x_2} |y_{path}(x) - y_1 \,| dx} \tag{1}$$

This makes error a dimensionless quantity, and normalized such that errors can be compared for any distance traveled. It can be thought of as a percentage, where having an error of 0 is being on the ideal path, while an error of 1 is following the "worst path" or equivalent.

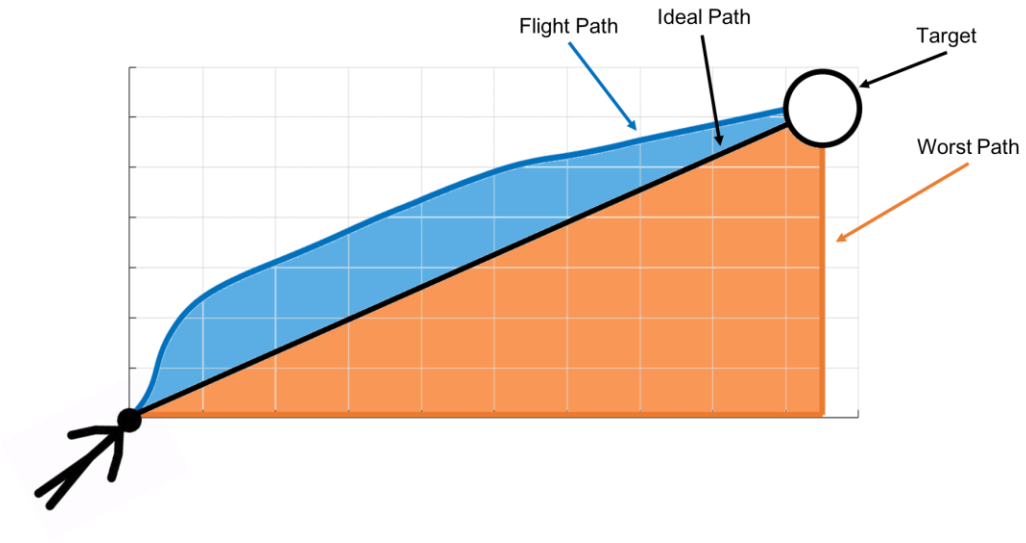

Figure 5: A visualization of the error definition. The error is defined as the ratio of the blue area to the orange area. This normalization allows the comparison of errors of both the steep and shallow target distances.

4. Results and Discussion

There Comparisons were made between isotropic and anisotropic trust models, as well as two different magnitudes of thrust values in an attempt to determine which would yield the lower error. Both comparisons had the same result: no statistically significant difference.

4.1 Isotropic vs Anisotropic Thrust

The first comparison will be between isotropic and anisotropic thrusts. The expected result is that isotropic thrusts would be easier to control, because their magnitudes are constant with direction. That is also the most intuitive functionality when imagining an engine which exerts thrust. The error for isotropic and anisotropic thrusts were 0.257 ± 0.077 and 0.256 ± 0.10 respectively for the shallow target. For the steep target, the errors were 0.207 ± 0.067 and 0.201 ± 0.052. There is no statistically significant difference between the average error of the thrust values at either the steep or shallow distance.

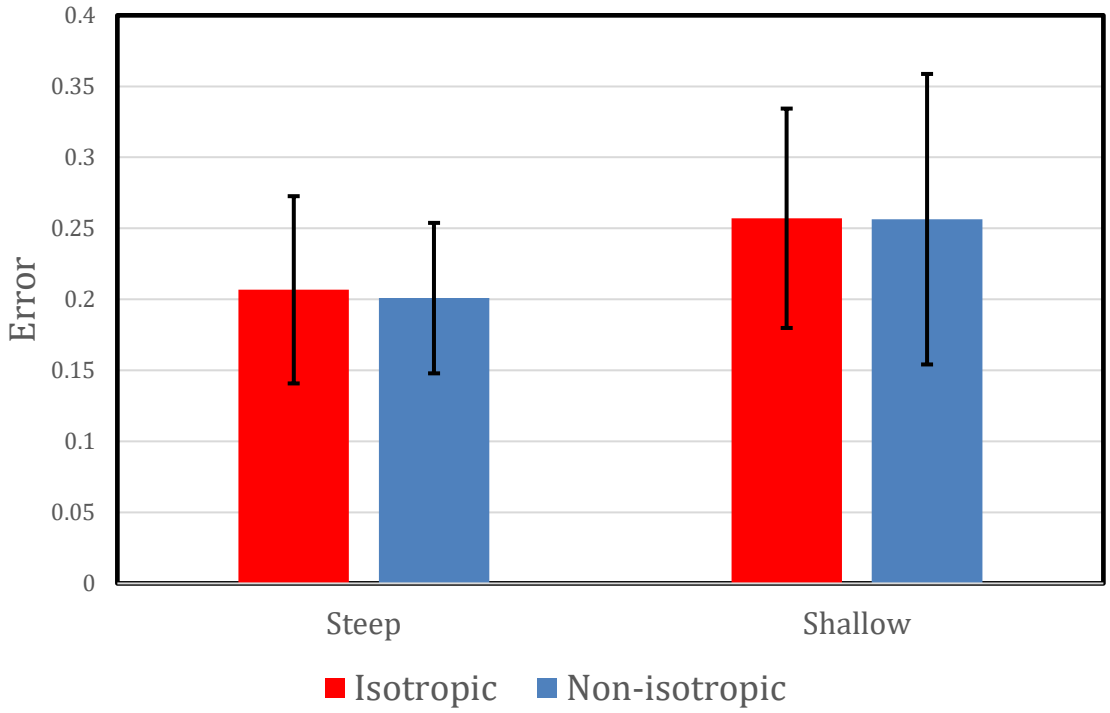

Figure 6: The average error for isotropic and anisotropic thrust values. The two data sets are not statistically different.

To visualize this, the errors are plotted in Fig 7. Plotting one thrust value against the other in this manner easily allows the visualization of the ratio of those errors. Each x or y value represents a set of 3 runs to the same target with the respective thrust value for 8 test subjects. A reference line with slope 1 is plotted to compare the ratios. If isotropic thrusts were significantly better, results the points on this plot would be concentrated above the reference line. Instead, most of the points are clustered near the reference line, while balanced on either side of it.

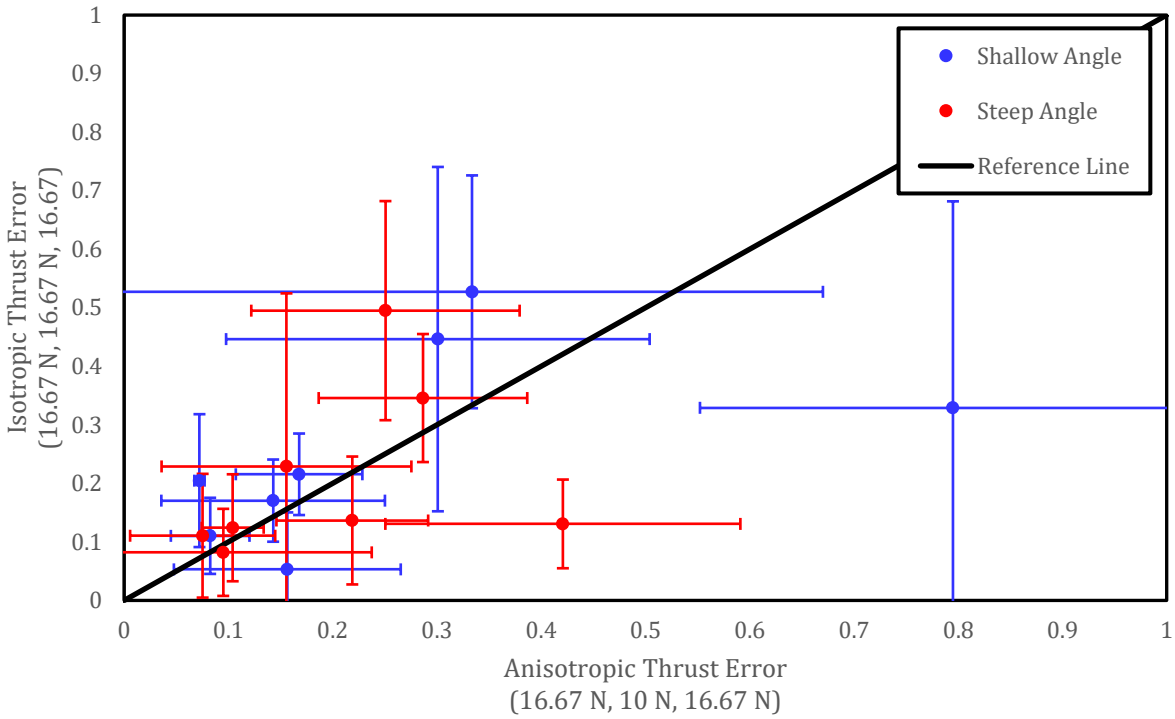

Figure 7: A comparison of isotropic and anisotropic thrust errors. Each point represents the average error of a test player at each thrust value and target distance. The reference line is plotted to compare the ratio of the errors (i.e. points above the line have a higher error with isotropic thrust than anisotropic thrust.

It is worth noting the large variation in the measurements makes it hard interpret the data. The inconsistency in flight path from player to player is unavoidable, because of inherent difference in ability of each player. However, discrepancies within each players measurements were caused by the learning curve associated with each thrust value, and the mechanic itself. As a player learns, his error trends to zero, and the variation in measurement also trends to zero. But then the difference in error is no longer significant to the game designer/developer, because they are both so close to zero.

The two sets of data are not statistically different, which means ultimately it is up to the game designer/developer to decide what set of thrusts work best in the given application. For example, in a game where low to the ground flying is encouraged, and the designer wants the players to have high horizontal mobility, it would make sense to have a anisotropic thrust value like the one tested in this experiment.

4.1 Comparing Thrust Magnitudes

The second comparison will be between two anisotropic thrusts, whose scaling vectors have magnitudes (16.67 N, 10 N, 16.67 N) and (25 N, 15 N, 25 N), which will be referred to as low and high thrust respectively. The low thrust data is the same as the anisotropic thrust as above, with an average error of 0.256 ± 0.10 for the shallow target, and 0.20 ± 0.052 for the steep target. The

high thrust data had an average error of 0.197 ± 0.050 for the shallow target, and 0.219 ± 0.091 for the steep target. The difference of these mean values is not significant.

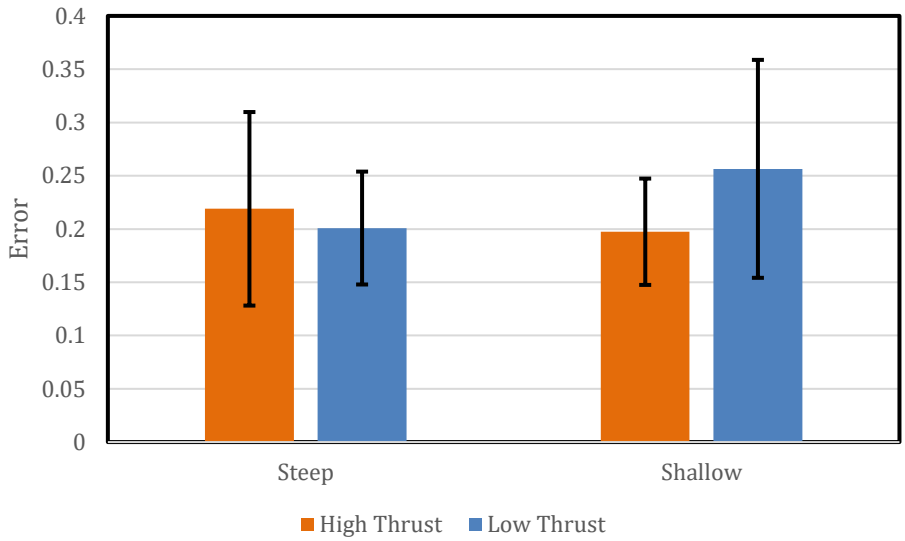

Figure 8: The average error for high and low anisotropic thrust values. The two data sets are not statistically different.

Plotting the error by person shows the same trend as the isotropic to anisotropic comparison (Fig 9). The points are balanced around the line with slope 1, which makes it easy to visualize that not a single thrust has less error than another across the entire population. Because there is no difference in the data, it suggests that developers are free to choose the thrust values which are more appropriate for their game without costing their players the ability to control their movement. It is worth noting that there is likely an upper limit of thrust values which are controllable, but that was not found in this comparison.

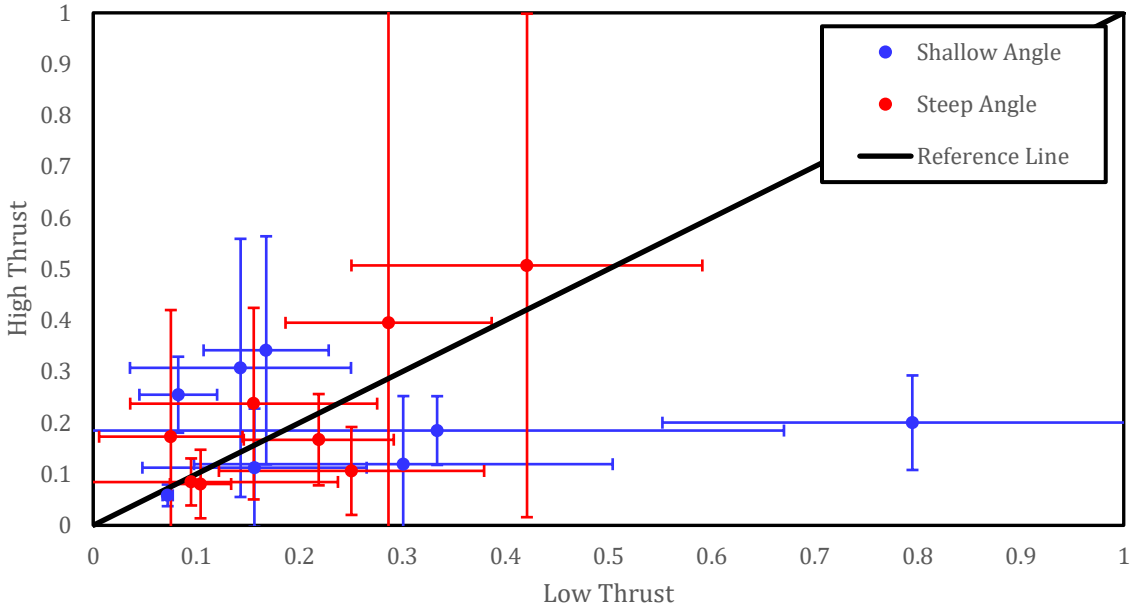

Figure 9: A comparison of the error for high and low anisotropic thrust values. Each point represents the average error of a test player at each thrust value and target distance. The reference line is plotted to compare the ratio of the errors (i.e. points above the line have a higher error with high thrust than low thrust.

5. Conclusions

This experiment seeks to if determine there is a difference in players ability to control their flight across different thrust models and values, in order to determine which of those models and values are most appropriate for a game. The results show no statistically significant difference in a player's error from an optimal path between any of the different thrust values. This suggests that the game designer/developer is free to choose the values and model as he wishes without compromising the usability of the mechanic.

One consideration that should be made is the type of movement the game designer wants to encourage. He can use the type of gameplay to drive the decision without costing the player anything in terms of ability. For example, as mentioned previously, a developer could choose to implement an anisotropic thrust model to encourage horizontal movement more than vertical movement, or vice versa.

The results of this experiment beg the following question: "Do players learn one set of thrust values faster than another?". This experiment does not suggest an answer that question, because efforts were made to remove knowledge and learning curve from this experiment, but similar methods as the ones applied here could answer that question.

Acknowledgments

The author would like to acknowledge Dr. Barbara Hughey for her advice in analyzing and presenting the results. The author would also like to thank his development team in helping create and refine the mechanic tested in this experiment.

References

[1] Buckley, S. "This Is How Valve's Amazing Lighthouse Tracking Technology Works". Gizmodo. Retrieved 2 July 2015.

[2] Distler, H., Pelah, A., Bell, A. and Thurrell, A. "The perception of absolute speed during self-motion" Poster presented at the annual meeting of the European Conference on Visual Perception, Oxford, England, 1998.

[3] Nilsson, N. C., Serafin, S. and Nordahl, R. "Establishing the Range of Perceptually Natural Visual Walking Speeds for Virtual Walking-In-Place Locomotion," in IEEE Transactions on Visualization and Computer Graphics, vol. 20, no. 4, 2014. pp. 569-578

[4] Slater, M., Usoh, M., and Steed A., 1995: "Taking Steps: The Influence of a Walking Technique on Presence in Virtual Reality," ACM Trans. on CHI, Special Issue on Virtual Reality Software and Technology pp. 201-219

[5] Usoh, M., Arthur K., Whitton M., Bastos, R., Steed, A., Slater, M. and Brooks Jr, F.

"Walking Walking > walking-in-place > flying, in virtual environments" pp. 359-364