Cosc 341 A2 Report
Ryan Pybus

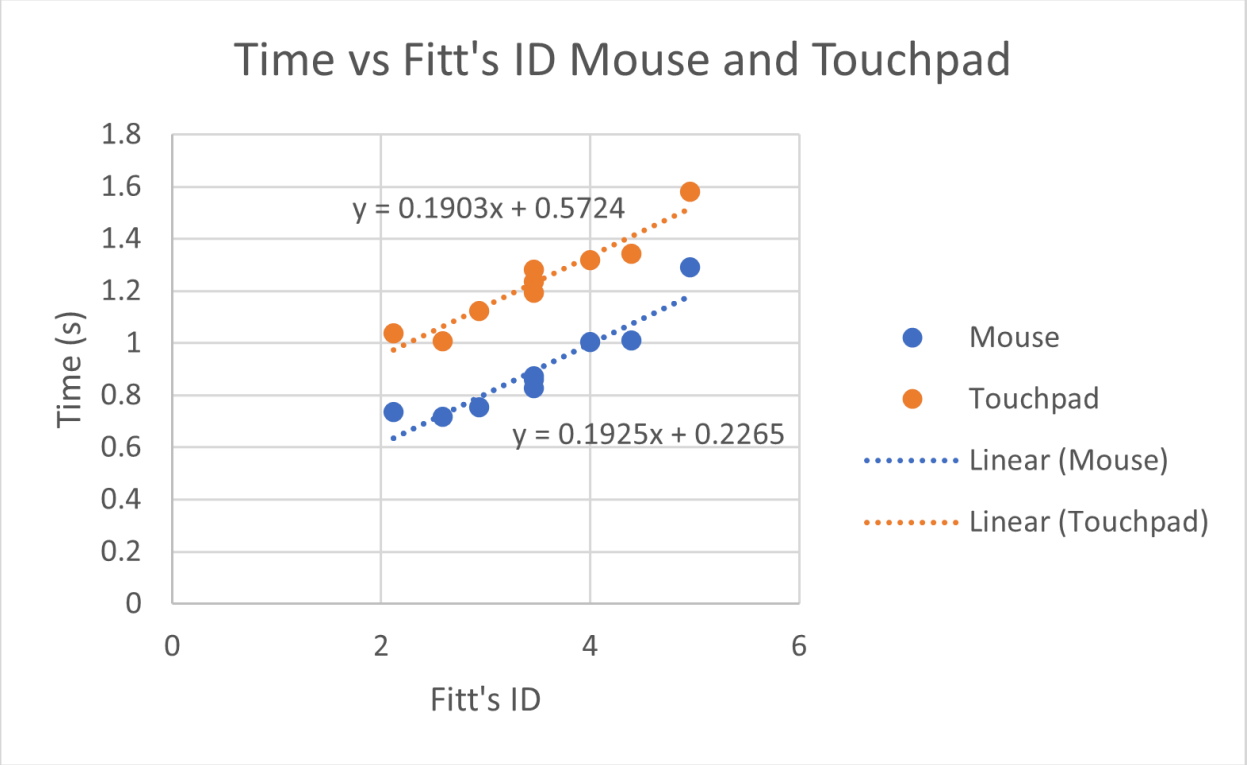
Results:

The Fitts's Law equation for the mouse was found to have a = 0.2265 and b = 0.0.1925, While the touchpad had a = 0.5724 and b = 0.1903. So the touchpad had a greater delay than the mouse, but both had very similar accelerations. Therefore, the touchpad is an overall slower method of pointing.

The total number of errors with the mouse was 6, with only 4 for the touchpad. The extra time taken by the touchpad likely allows users to more easily correct their trajectory mid-movement. So the touchpad compensates for its slowness by being a bit more accurate.

All three participants said that they prefer to use a mouse rather than a touchpad whenever possible.

Data was processed in python using the code found in FittsAnalysis.ipynb.



Overview of the program:

Before playing, the subject enters an ID and their device type on the Writer object. On play the test will start with the easiest configuration (largest size and smallest amplitude). After the completion of each round, the targets are destroyed and new ones created in different places with appropriate sizes. Once all nine settings are completed, it terminates. The subject can then repeat with the other device.

The radius and size values were selected by just running the program and using a ruler to measure the size of objects on the screen. It was found that 1.5 was approximately equal to 5 cm for the radius, and that a scale of 0.3 made the targets about 0.5cm in diameter.

The targets are placed exactly 40 degrees away from each other at the same distance from the center as their positions are calculated programmatically using
x = radius*cos(i *PI*2/9) and y = radius*sin(i *PI*2/9), for i = 0..8 to make nine circles.

Errors are detected by using Master, which has the list of all target game objects, to check if the object clicked by the mouse is the next in the list.

Data is logged in a csv file called <DeviceType><ID>.csv, located in the Assets folder.

Challenges:

- Scene Transition

My original intent was to make 9 different scenes, each with its own version of the Master script which would specify where to place the targets and how large. The tenth scene would be a ‘level select’ so that at runtime, the test subject would be on the level select screen and would need to select a configuration, do it, and select the next. This seemed wasteful though, as it would involve 9 ‘Master’ scripts with only small variations. I looked into trying to send data on scene transitions, so that on button press it would load the test scene, but with specific position/size values for that button. Eventually though I realized that it would be much simpler to have only one scene, and have a script (SceneSaver) track which configuration we were on and which had been done (ordering them one to nine). Then Master could just reload after each round completion, taking its values from a switch statement based off of the count in SceneSaver.