Mouse input lab: design and evaluation

1 Introduction

1.1 Goal of the lab

In this lab, we will design and implement a transfer function for mice from scratch. A transfer function g maps the speed of movement of the cursor $\mathrm{V}_{\mathrm{disp}}$ to the speed of the physical device V_{dev}:

$$V_{\mathrm{disp}} = g(V_{dev}).$$

When $g(u) = 1$, we have a case of absolute pointing, otherwise we say we have a case of relative pointing.

We consider the function $g(u) = cu^r$, where c and r are two parameters. You will first run some preliminary calibration tests to determine useful parameters. Then, you will implement a few transfer functions and decide on a preferred one. You will finally design and conduct a controlled experiment, and analyze the resulting data.

Command line instructions given should work on Linux and MacOs, but should be adapted for Windows.

1.2 What is given to you

- `window.py` the main code that allows you to run an experiment with a cursor inside a controlled environment with targets.
- `cursor.py` where an absolute position cursor is implemented, and where you will implement other cursors
- `test_window.py` which runs the window in an event loop
- `design_fitts2D.csv` an example file that can be loaded by window to create experiment conditions

2 Preliminaries

Task 1. *Download the zip* `TP_IHM.zip` *from Moodle. This zip contains the code that you will build upon in this lab. Then, launch a virtual environment and install pygame, numpy and the latest version of pyglet lower than 2.0. Make sure you can run the file* `test_window.py`*.*

Help. *On Unix, run* `python3 -m venv .env` *to create a virtual environment, then* `source .env/bin/activate` *to activate it. Inside the virtual environment, run* `python3 -m pip install "pyglet<2"` *to install a pyglet version smaller than 2.0.* `test_window.py` *will launch a full screen window, you can press the escape key to change focus. You can find documentation for pyglet 1.5.X at* `https://pyglet.readthedocs.io/en/pyglet-1.5-maintenance/`

Task 2. *Deactivate the transfer function on your computer. How can you verify that the transfer function is deactivated?*

Help. *On linux with Gnome, you can do* `gsettings set org.gnome.desktop.peripherals.mouse accel-profile "flat"`*. On MacOs you can do* `defaults write .GlobalPreferences com.apple.mouse.scaling -1`*. On Windows, you can do it manually via the settings.*

Task 3. *Consider the function $g(u) = cu^r$. Plot the function for several values of $(c, r) \in \mathbb{R}^2$. Optional: Compute $g'(u)$ and $g''(u)$ and consider the concavity or convexity of g for different values of c and r. What would you consider good parameters?*

Help. *If $g''(x) \geq 0$ for $x \in \mathcal{I}$, then g is concave on $\mathcal{I}$. Conversely, if $g''(x) \leq 0$ for $x \in \mathcal{I}$, then g is convex on $\mathcal{I}$. Consider $r = 0$; $c = 0$, $r = 1$, $r > 1$ and $c > 0$, $r < 1$ and $c > 0$, $0 < r < 1$ and $c > 0$, $0 < r < 1$ and $c < 0$, $r < 0$ and $c > 0$, $r < 0$ and $c < 0$*

Solution: constant, constant, linear, convex, concave, concave, convex, convex, concave

3 Calibration

Task 4. *Determine the minimum and maximum speed that you can attain with a mouse, as well as the minimum and maximum speed that you attain comfortably. To store cursor data, implement the* `log` *function.*

Task 5. *Write a* `ConstantCDGainCursor` *class that implements a constant gain function $g(u) = k$. Experiment different values of k; what value of cursor gain do you like best?*

4 Multiple Cursors

Task 6. *Write a* `CumstomCursor` *class that implements a gain function $g(u) = cu^r$. Find suitable values for c and r.*

Task 7. *Look at the* `PolyCursor` *class and explain how it works.*

Task 8. *Use the* `PolyCursor` *class to qualitatively compare $g(u) = 1$, $g(u) = k$ and $g(u) = cu^r$. Which do you prefer?*

5 Evaluations

Task 9. *Run a pilot to estimate the difference in movement time between the* `CustomCursor` *and the* `AbsoluteCursor`.

Task 10. *Based on this pilot data, use a sample size calculator (*e.g., ***https://clincalc.com/stats/samplesize.aspx****) to determine the sample size that you will need to reliably discriminate between the two techniques in a controlled experiment.*

Task 11. *Some transfer functions will feel "unnatural" at first, but we can adapt to them, sometimes fairly quickly. For example, a transfer function with $c < 0$ will make the cursor go up and left when the device is going right and down. Evaluate your performance with an unnatural transfer function over several identical blocks. Suggest a simple model of adaptation/learning that could explain your data, and fit it.*

Task 12. *What is the implication of the previous result on the evaluation between two different interaction techniques, and how to address it in practice?*

Task 13. *Run a controlled study with few participants to compare both techniques. Report on the results (about two pages including graphs).*

6 Modeling

The distribution of pointing times is often not Gaussian. A useful model is the exponentially modified Gaussian (EMG) distribution:

$$p(x) = \frac{1}{2\beta} \exp\left(\frac{1}{2\beta}(2\mu + \frac{\sigma^2}{\beta} - 2x)\right) \operatorname{erfc}\left(\frac{\mu + \sigma^2/\beta - x}{\sqrt{2}x}\right), \tag{1}$$

where erfc is the Gaussian error function. The Gaussian distribution is

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(\mu - x)^2}{2\sigma^2}\right) \tag{2}$$

Task 14. *Set $\mu = 0$ and $\beta = 1$, and plot the EMG distribution for various values of σ, compare to the Gaussian.*

Task 15. *Load the file* `pointing_data_sample.txt` *and plot the data.*

Task 16. *What is the log-likelihood of a sequence of data $x = (x_1, x_2, \ldots, x_n)$ under the two models? Implement two functions that compute this likelihood as a function of x and the model parameters.*

Help. *You may use* `scipy.stats` *to get the distributions of the Gaussian and EMG distributions directly.*

Task 17. *Maximize the log-likelihood to determine the most likely parameters of the model, for both models. Which model is the most appropriate?*

Help. *You can minimize a function with* `scipy.optimize.minimize`*. The most likely model is the one with highest likelihood.*

Task 18. *You can achieve the same with* `scipy.stats.continuous_rv.fit`*. Verify you get the same estimates.*

7 Conclusion

Task 19. *Read the paper No more bricolage by Casiez* et al. *(on Moodle), and reflect on this lab (write 10 to 15 lines).*