Eye tracking

Binus University Workshop Day 3

https://osdoc.cogsci.nl/binus2022

Sebastiaan Mathôt


cogsci.nl/smathot


cognitivescience


@smathot


@cogscinl

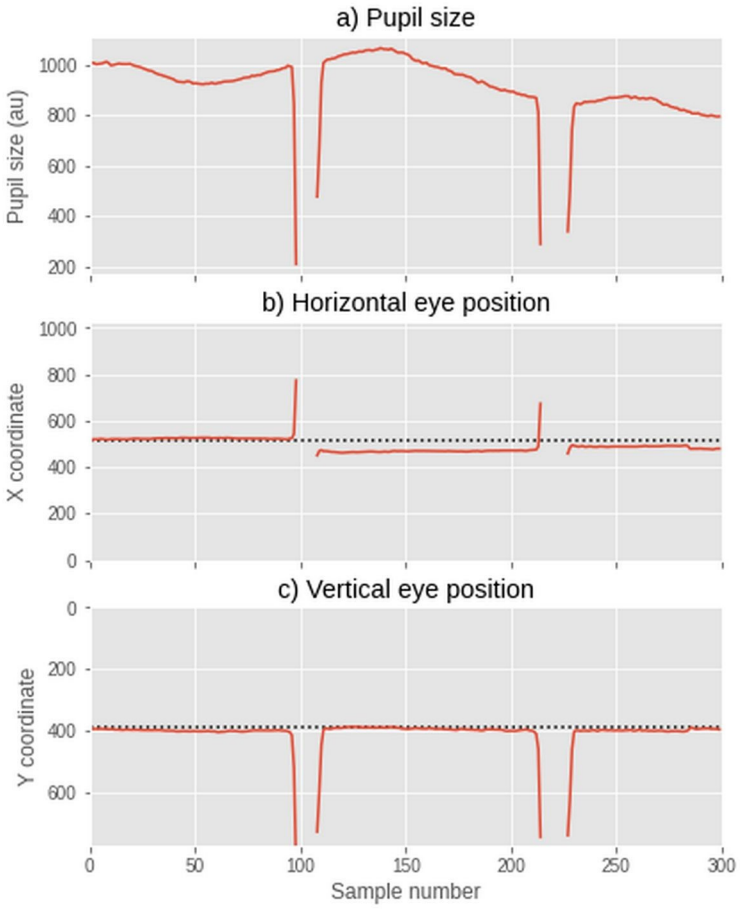

sebastiaanmathot

Today (day 3)

- Before the break
  - A general introduction to eye tracking
  - Working on a self-paced reading task
- After the break
  - Gaze-contingent self-paced reading
  - Continue working on your own experiment

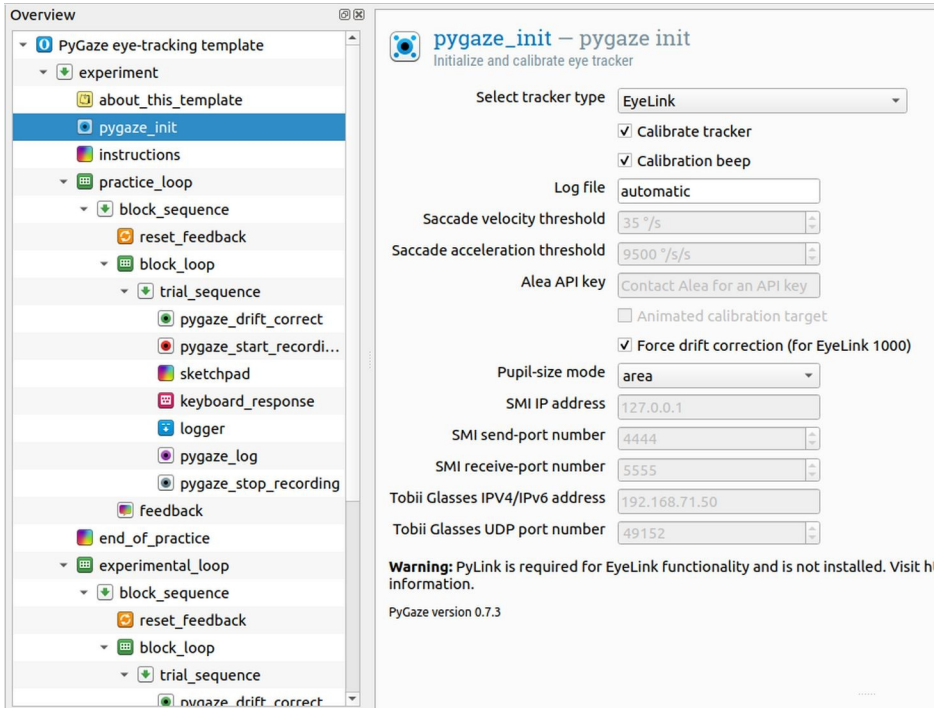
Eye tracking

- What do eye trackers record?
- Always: gaze position (time series)
  - x, y coordinates
  - Relative to a display
  - Or relative to a world camera
  - Or uncalibrated
- Usually: pupil size (time series)
  - In arbitrary units
  - Or in millimeters of diameter
- Usually: an event log
  - Timestamped messages



Eye tracking

- Eye tracking is not standardized
  - Different log-file formats
  - Different sampling rates
  - Different software-development kits (SDKs)
  - Technical idiosyncrasies
  - No standard analysis pipeline
- Attempts at standardization
  - Building experiments
    - PyGaze/ OpenSesame
    - iohub/ PsychoPy
  - Analyzing data (script-based)
    - gazeR
    - python-eyelinkparser



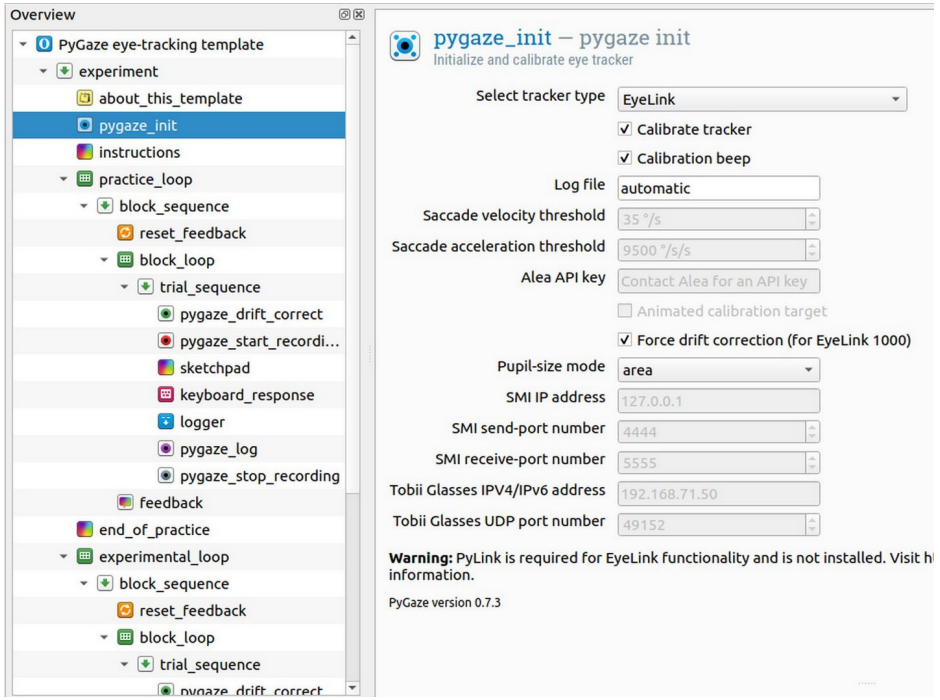
Eye tracking

- An eye-tracker setup consists of
  - Experiment PC: runs the experiment
  - Recording PC: connected to eye tracker
  - Bi-directional communication
- For some eye trackers the experiment and recording computers are the same

EyeLink 1000 setup
Source: SR Research

PyGaze

- PyGaze is a Python library for eye tracking
  - Supports different brands
  - Integrates with OpenSesame
- We will use PyGaze to implement a fancy gaze-contingent, self-paced reading experiment
  - But we'll start slow



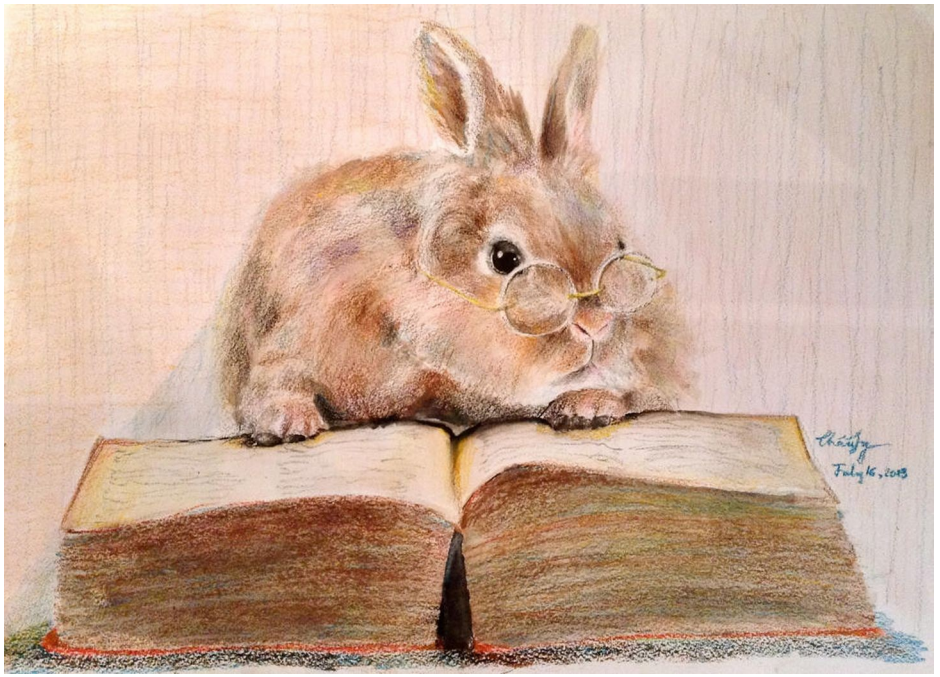
Eye tracking in a browser

- High-quality eye tracking requires a laboratory set-up
- But poor quality eye tracking can be done in a browser
  - Using the webcam
  - WebGazer.js is the best-known implementation
  - Works with OSWeb [1]

[1] https://osdoc.cogsci.nl/3.3/manual/eyetracking/webgazer/

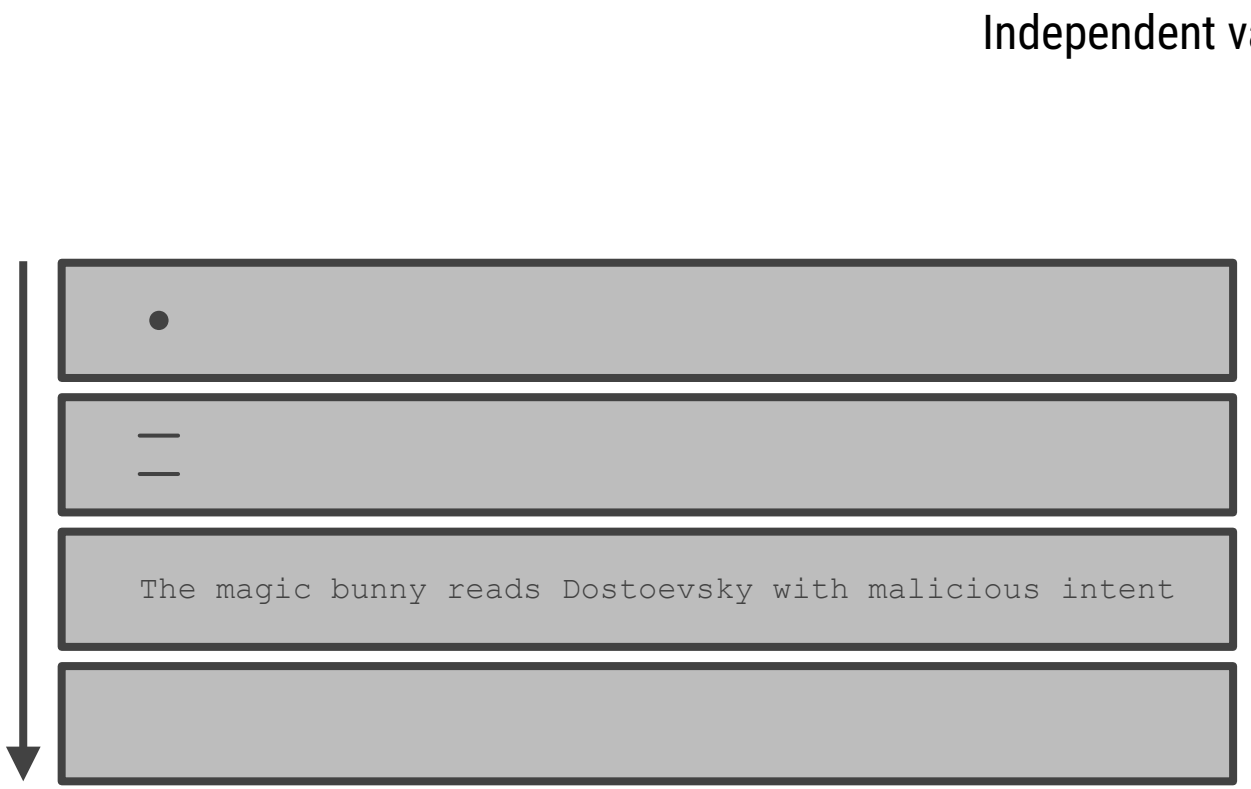
Hands-on workshop

- As said: We'll implement a self-paced reading task
- I'll suggest some pointers
- But of course the details are up to you!



Source: MagicFlyingBunnies at deviantart

Hands-on workshop



Independent variable: sentence complexity (?)

Drift correction
Self-paced

Fixation bars
500 ms

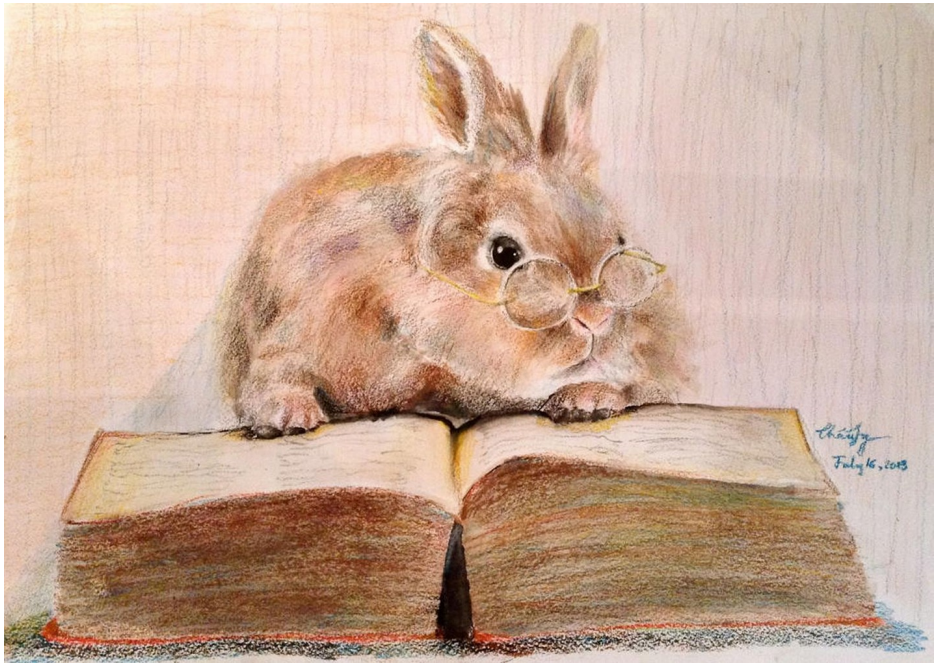
Sentence
Until response

Blank
500 ms

Dependent variable: response time

Hands-on workshop

- Start from the *Default template*
- At the start of the experiment
  - **pygaze_init** → initialize the eye tracker in advanced dummy mode
- At the start of each trial
  - **pygaze_drift_correct**
  - **pygaze_start_recording**
- At the end of each trial
  - **pygaze_log**
  - **pygaze_stop_recording**

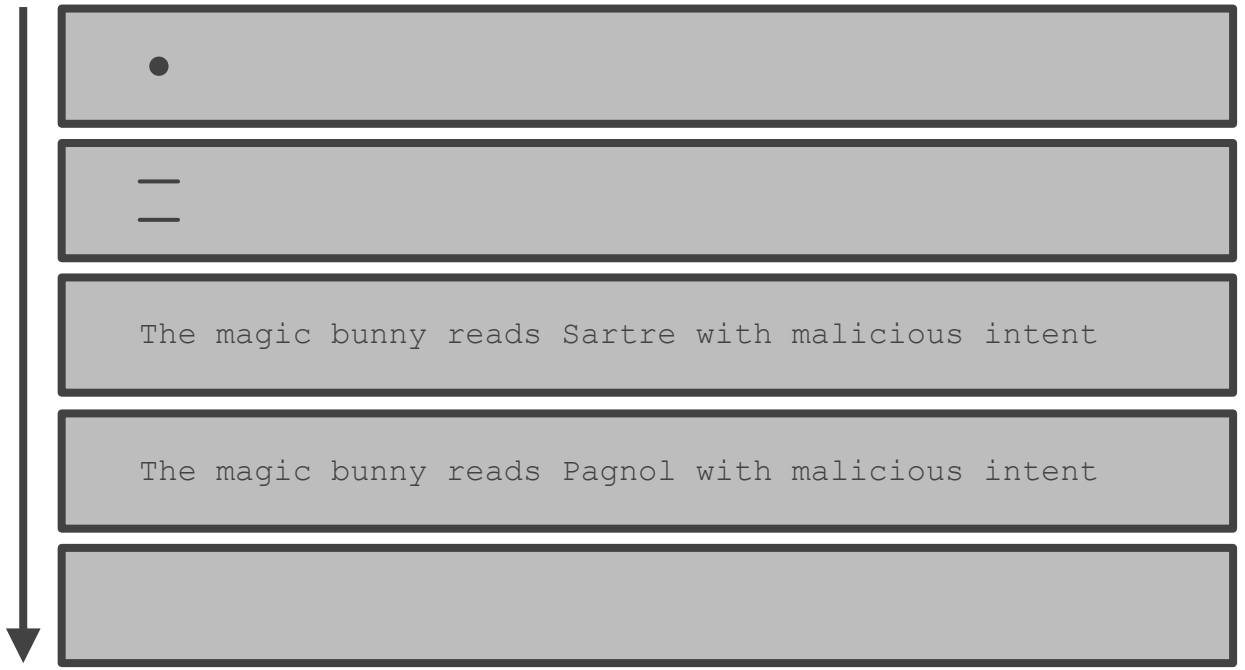


Source: MagicFlyingBunnies at deviantart

Let’s get to work!

Slides: https://osdoc.cogsci.nl/binus2022

Adding gaze contingency



Drift correction
Self-paced

Fixation bars
500 ms

Sentence
Hard: Until fixation on fourth word
Easy: Until first saccade

Sentence
Until key press

Blank
500 ms