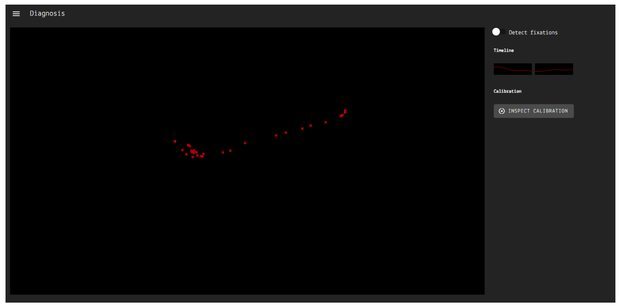
Fixation: A universal framework for experimental eye movement research

Mostafa Elshamy, University of Southern California
Peter Khooshabeh, DEVCOM Army Research Laboratory

ETRA 2021

USC Institute for Creative Technologies

DEVCOM ARMY RESEARCH LABORATORY



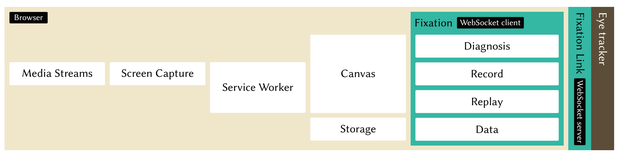
INTRODUCTION

- Eye tracking technology has advanced remarkably and is being used to study computational models of vision, social interaction, and human–computer interaction.
- Despite efforts to democratize eye tracking, commercial software coupled to specialized hardware remains dominant. Cost, licensing restrictions, and sparse documentation impede research progress.
- User-friendly software for conducting experimental research using neural network / video-based gaze estimation models inspired by advances in machine learning is not available yet.
- Our methods address universal compatibility, scalable data processing, and data privacy and are generally applicable to high-frequency recording and real-time visualization of neural or physiological signals.

Why we are open-sourcing Fixation and releasing it for free

Commercial eye tracking software may process raw gaze data in partially specified pipelines to protect intellectual property. We hope more researchers will have access to production-grade tools needed to obtain reliable, and verifiable datasets. This promotes openness, decreases community fragmentation, and accelerates eye movement research.

ARCHITECTURE / MODULES



1. **Diagnosis** Processing a raw gaze stream and fixation detection.
2. **Recording** Capture and storage of stimulus and user camera frames during detected fixations.
3. **Replay** Playback of stored frames overlaid with a visualization.

Idea of a Web Application

- Application modules use Web APIs to provide their functionality.
- Performance concerns when processing data at **60–300 Hz** in real time can be addressed by *running scripts concurrently via web workers*.
- Data privacy concerns in cloud environments can be addressed by *storing sensitive data locally in the browser cache*.

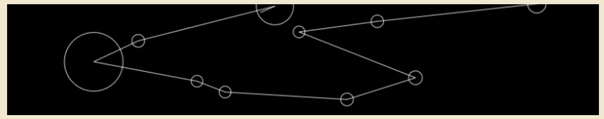
Fixation Link

```
{
  x: 0.305161, // (0,0): top-left, (1,1): bottom-right
  y: 0.560296,
  timestamp: 159391614.476313 // us
}
```

We reduce the problem of universal compatibility to mapping data to a JSON format before it is streamed over the network using *fixation links (WebSocket server)* to a client app (*WebSocket client*) running in-browser.

TX300	EyeX	Simulator
Tobii	Tobii	
Python	Node.js, C#	Python

DIAGNOSIS

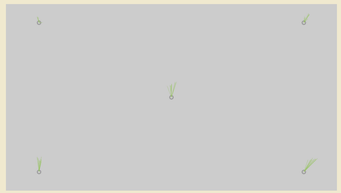


1 Is the client connected and can it detect fixations?

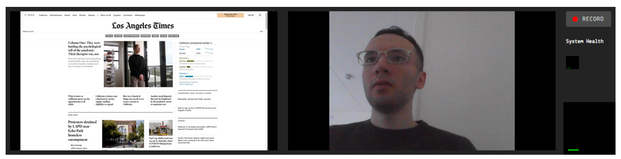
Fixation detection significantly reduces data size without compromising high-level analysis. We apply **dispersion-threshold identification (I-DT)** with dispersion threshold D_t and frequency f determined by a configurable duration threshold (often **100–200 ms**).

2 Are there large calibration errors?

Five-point calibration is adequate to determine if a re-calibration is needed before recording. We consider points at (0.1, 0.1), (0.9, 0.1), (0.5, 0.5), (0.1, 0.9), (0.9, 0.9) and plot horizontal and vertical mean absolute error E for n raw data points.



RECORDING



- Synchronization of independent data streams is a major problem in multimodal systems often resolved by NTP based synchronization protocols or manual alignment.
- **How to reduce problems of experimental reproducibility and dataset reliability?** Recognizing that little to no visual processing is achieved during a saccade, we capture stimulus and user camera frames at each identified fixation rather than record independent data streams.

Stimulus / User Camera Recording Procedure

1. The `getDisplayMedia()` / `getUserMedia()` method from the Screen Capture API / Media Streams API returns a `MediaStream` object which can then be stored into a video element's `srcObject`.
2. When we identify a fixation, an `OffscreenCanvas` can use a video element as an image source and draw the image onto the canvas.
3. A prespawned web worker encodes the stimulus image, adds it to a JSON document representing the fixation and stores the augmented document in cache storage.

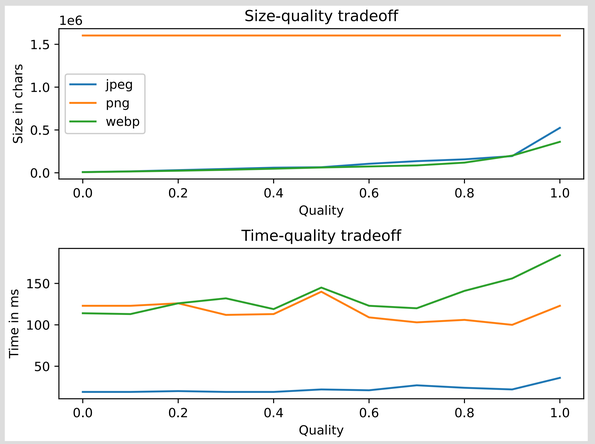
```
{
  id: 0,
  ...
  duration: 150.485869,
  stimulus: "..."
  user_cam: "..."
}
```

- For each new task, select the worker with minimum load (i.e., number of frames it is processing) out of **a pool of *n* worker threads** and communicate load back to the main thread after performing it async.
- **Transferable objects** are used to transfer ownership of a byte array representing the image between main thread and worker to further reduce processing latency.

Tuning Stimulus Image Encoding Parameters

Girl with a Pearl Earring [Johannes Vermeer 1665] / Public Domain

- We can optimize image encoder options using a simple procedure: plot execution time and encoded image size as a function of image quality (using a 0.1 or 0.01 step) for every compression format.
- JPEG is faster than both PNG and WebP, but a WebP encoder may produce fewer artifacts for graphical stimuli. PNG is lossless and results in significantly larger base64-encoded images.



REPLAY

- Verification of data integrity.
- Exploration of potential pockets where further analysis may be useful.
- Running a retrospective think-aloud study.

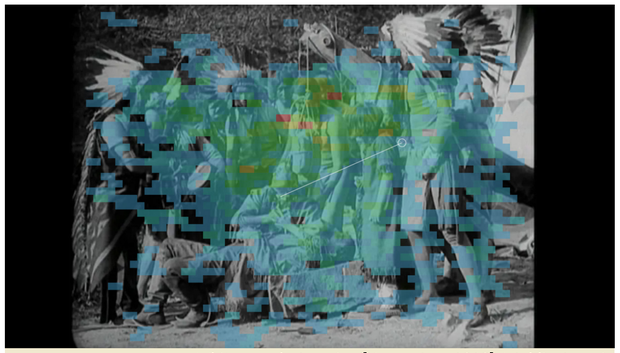
Replay Procedure

1. Use a buffer of n frames and request the next n frames from a single prespawned worker when we start consuming the current buffer.
2. Bypass decoding of base64-encoded images and instantly render frames on the canvas by creating n hidden elements in the document and setting their background image to the encoded image.

Gaze visualization methods

1. **Scanpaths** depict temporal viewing order.
2. **Heatmaps** depict aggregate gaze more intuitively.

Both depict areas of relative cognitive importance.



White Fawn's Devotion [Young Deer 1910] / Public Domain

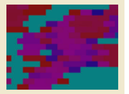
Rendering Real-time Dynamic Heatmaps

1. Consider a grid G of n tiles where each tile has size $w_t \times h_t$ approximately equal to expected eye tracking error.
2. On a new gaze point $p = (x,y)$ we increment i_t, intensity of tile t if t contains p. A prespawned worker accumulates intensity.
3. Map accumulated intensity count normalized relative to maximum intensity i_{max} to a tile's alpha channel or RGB color space via a gradient.

GPU vs CPU: Finding the maximum intensity

Open question Does smooth rendering via a Gaussian point-spread function have advantages beyond visual familiarity given eye tracking measurement error?

CPU	CPU, tiled	GPU
O(wh)	O(√*wh*)	O($\log_2$ wh)
	$w_t \approx \sqrt{w}$ and $h_t \approx \sqrt{h}$	



DATA MANAGEMENT

- Data can be exported as a JSON file and imported later by dragging-and-dropping it on a designated drop target.
- Chrome and Firefox allow the browser to use up to 80% and 50% of total disk space respectively for cache storage.

Duration	Frames	Usage
04:05	1305	50Mb (< 1%)
15:51	4171	184Mb (< 1%)

DISCUSSION

- Existing open-source eye tracking software supports recording from hardware of a particular form factor (i.e., head-mounted), few popular models, or focuses on post-experimental analysis of prerecorded data.
- Fixation is differentiated by its hardware-agnostic interface, inherently synchronized datasets, and accessibility on the web without compromising data privacy.
- We were able to experimentally verify our methods at **60–300 Hz**, but they can in theory scale to higher sampling rates (e.g., **1000 Hz**).

Future work

- Benchmark performance and grow the feature set to support use cases found to be problematic.
- Add more fixation links to support popular models and new models.
- Additional event detection algorithms and gaze visualization methods.
- Explore using the platform to generate large datasets for training gaze estimation models.

CODE / CONTACT

- Try Fixation at **polar-ocean-09884.herokuapp.com**.
- Code is available at **github.com/melhosseiny/fixation** (MIT license).
- Corresponding author can be contacted at elshamy@alumni.usc.edu.