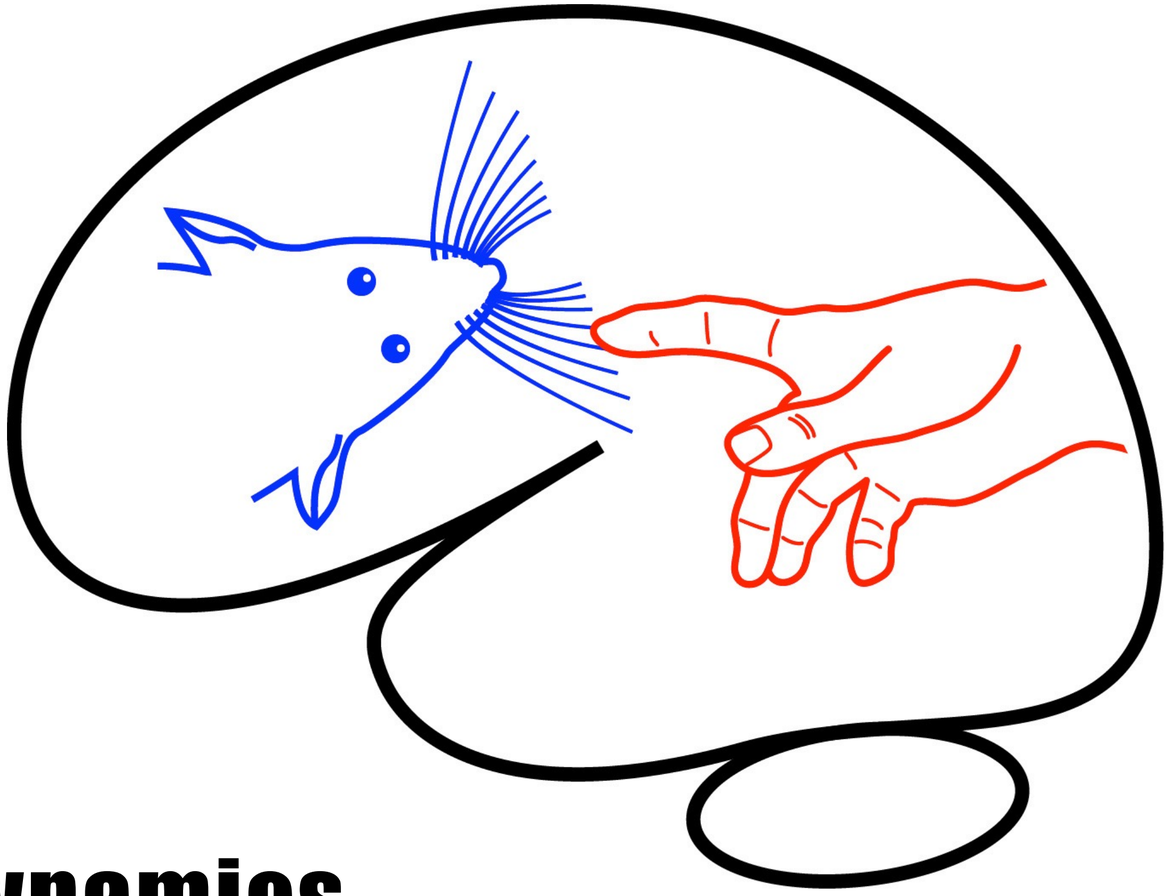
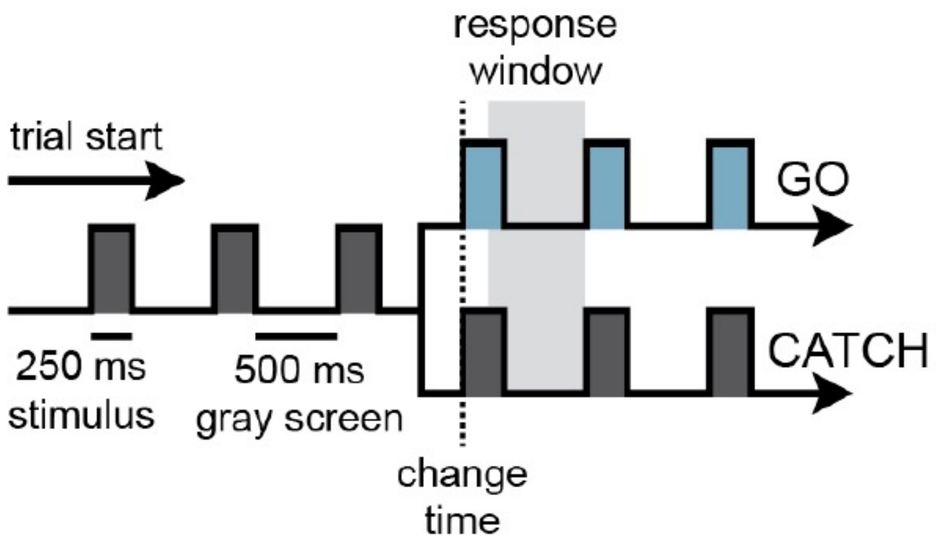
EPFL

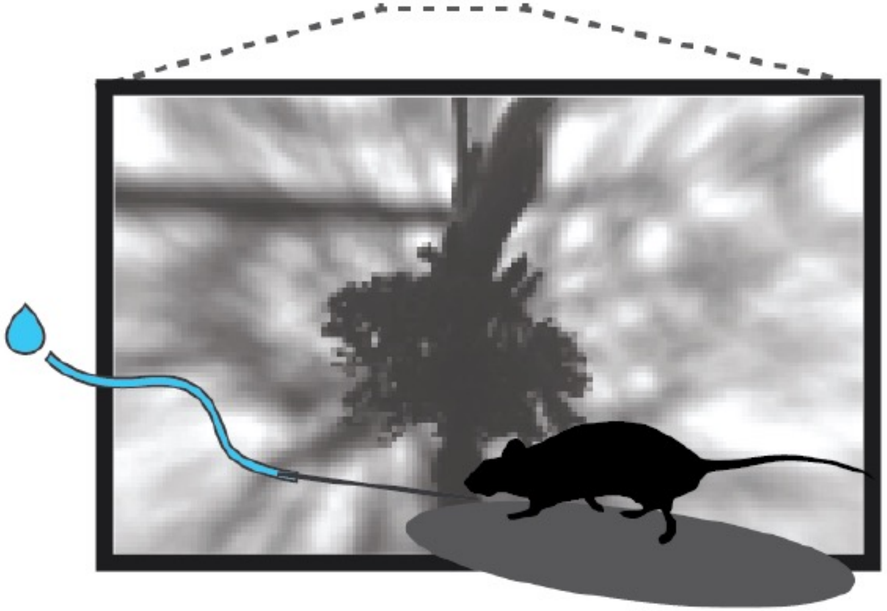


Allen Institute for Neural Dynamics
Visual Behavior Neuropixels dataset

Carl Petersen

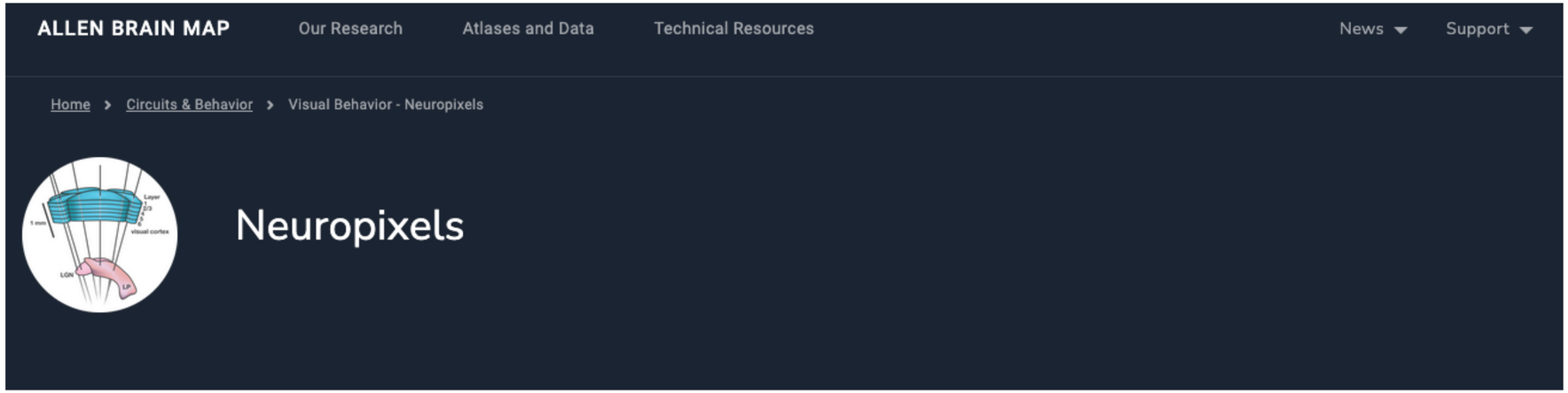
Visual change detection task





Allen Institute - internet pages

https://portal.brain-map.org/circuits-behavior/visual-behavior-neuropixels



Overview

Our ability to perceive the sensory environment and flexibly interact with the world requires the coordinated action of neuronal populations distributed throughout the brain. Yet, the detailed patterns of spiking activity that underlie perception and behavior are not well understood. To further our understanding of the neural basis of behavior, the Visual Behavior project used the Allen Brain Observatory (diagrammed below) to collect a large-scale, highly standardized dataset consisting of recordings of neural activity in mice that have learned to perform a visually guided task. This dataset can be used to investigate how patterns of spiking activity across the visual cortex and thalamus are related to behavior and also how these activity dynamics are influenced by task-engagement and prior visual experience.

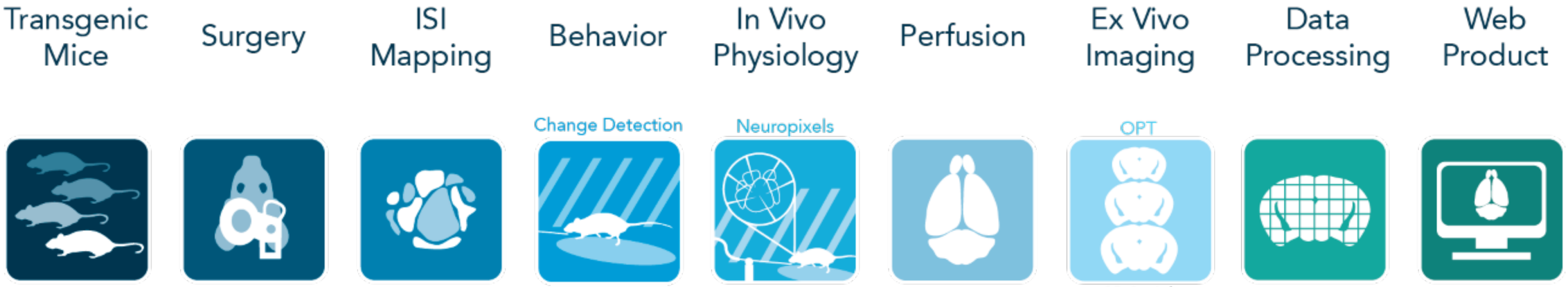
The Visual Behavior Neuropixels dataset includes 153 sessions from 81 mice. These data are made openly accessible, with all recorded timeseries, behavioral events, and experimental metadata conveniently packaged in Neurodata Without Borders (NWB) files that can be accessed and analyzed using our open Python software package, the AllenSDK.

Download Data **Download Technical Whitepaper**

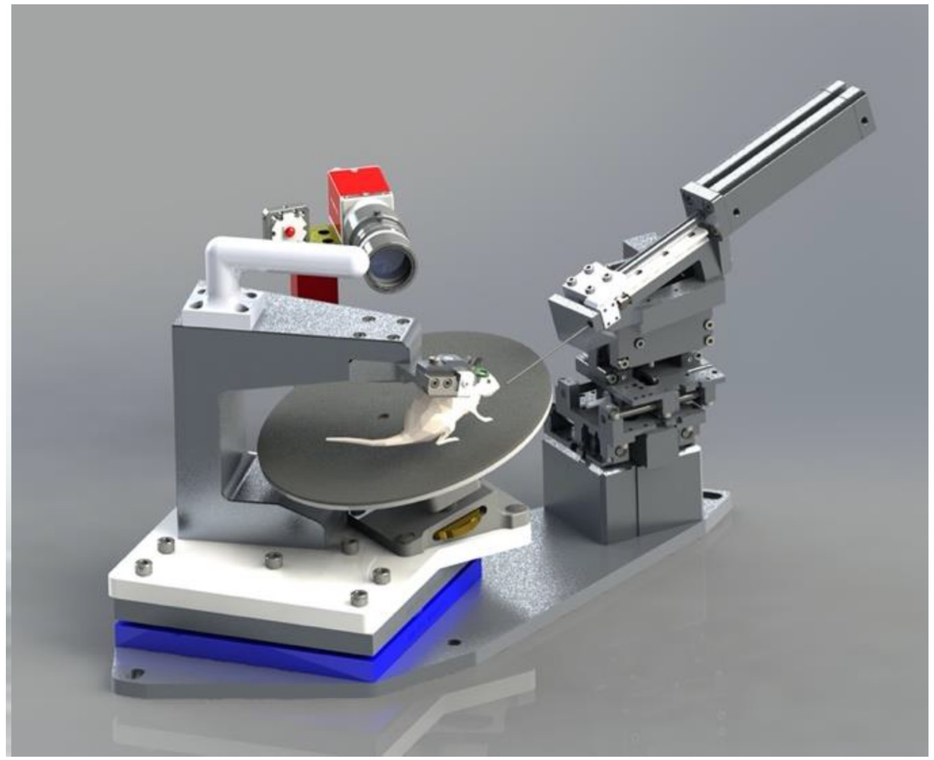
Allen Institute - white paper

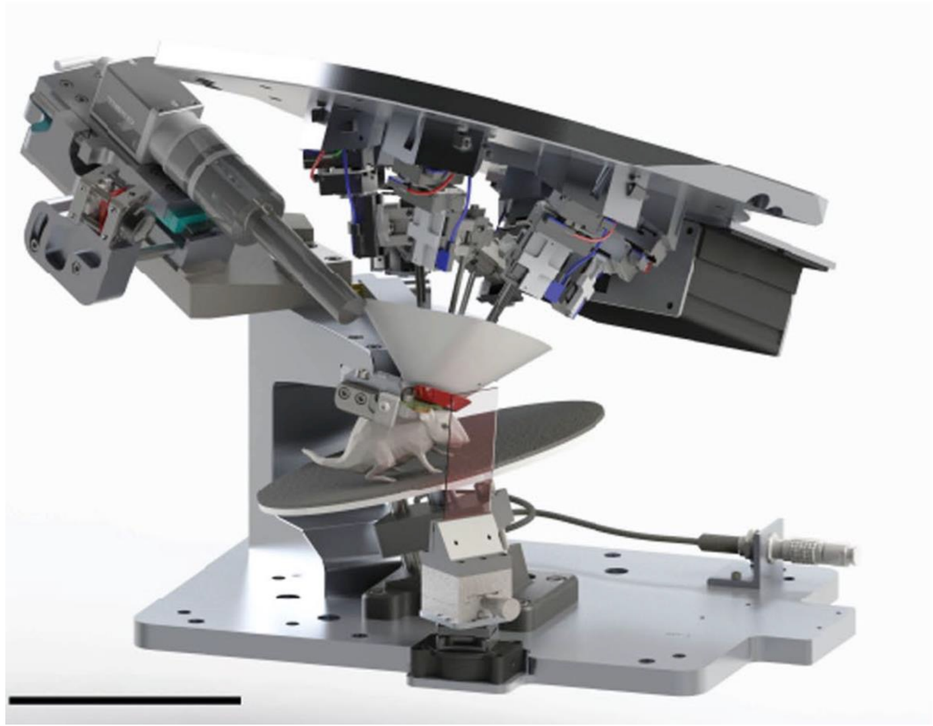
EPFL

Experiment overview

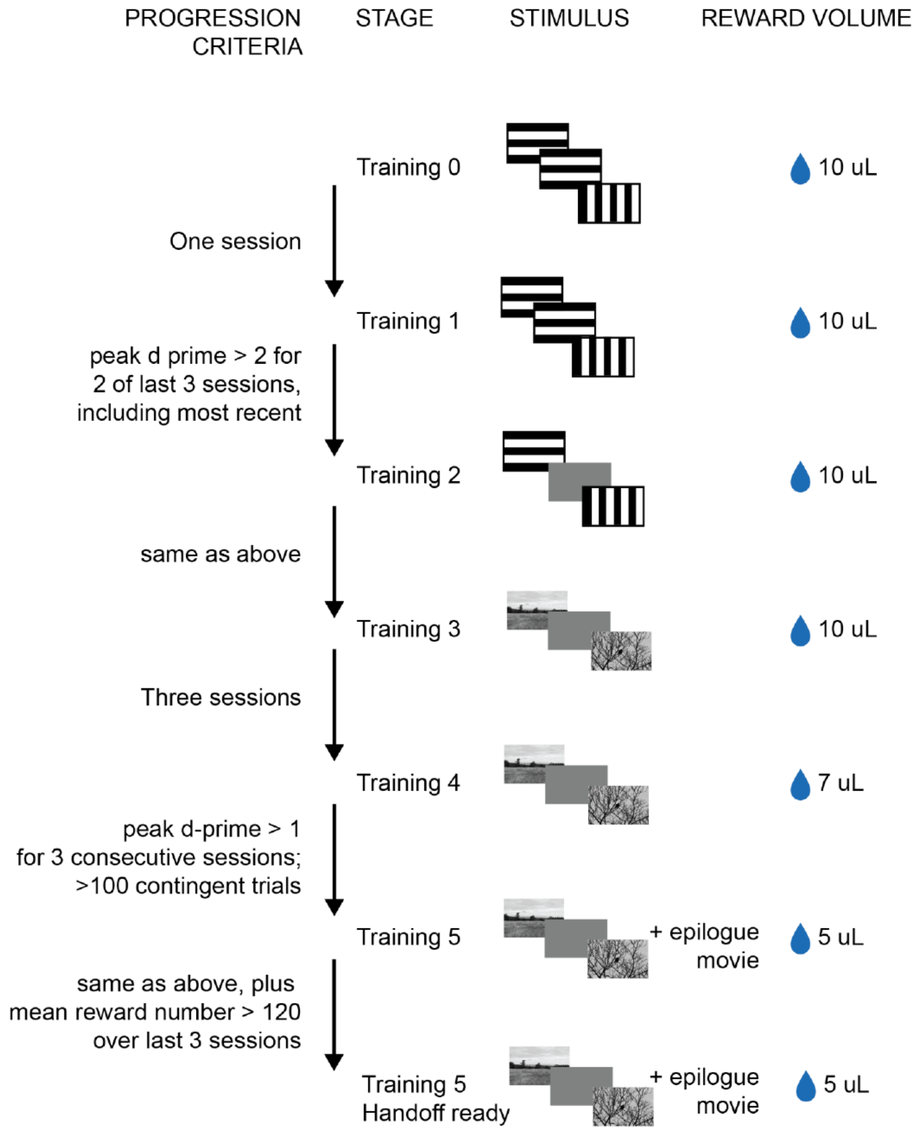


Behavioral setup





Behavioral training



Images

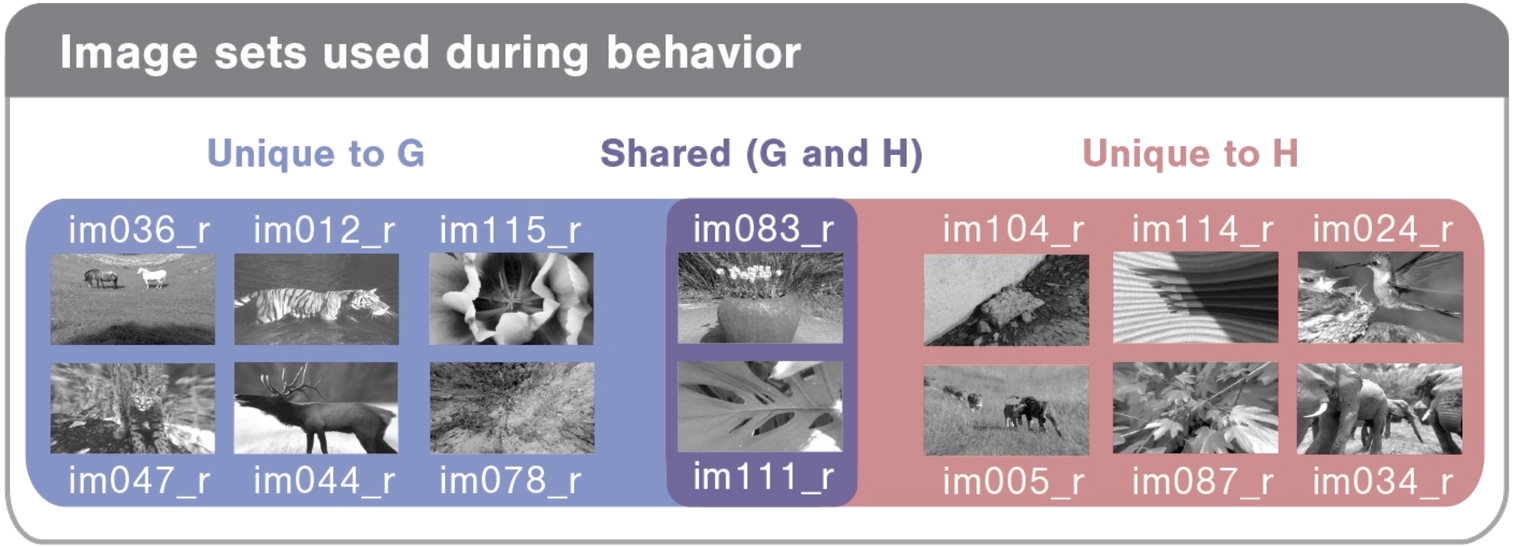
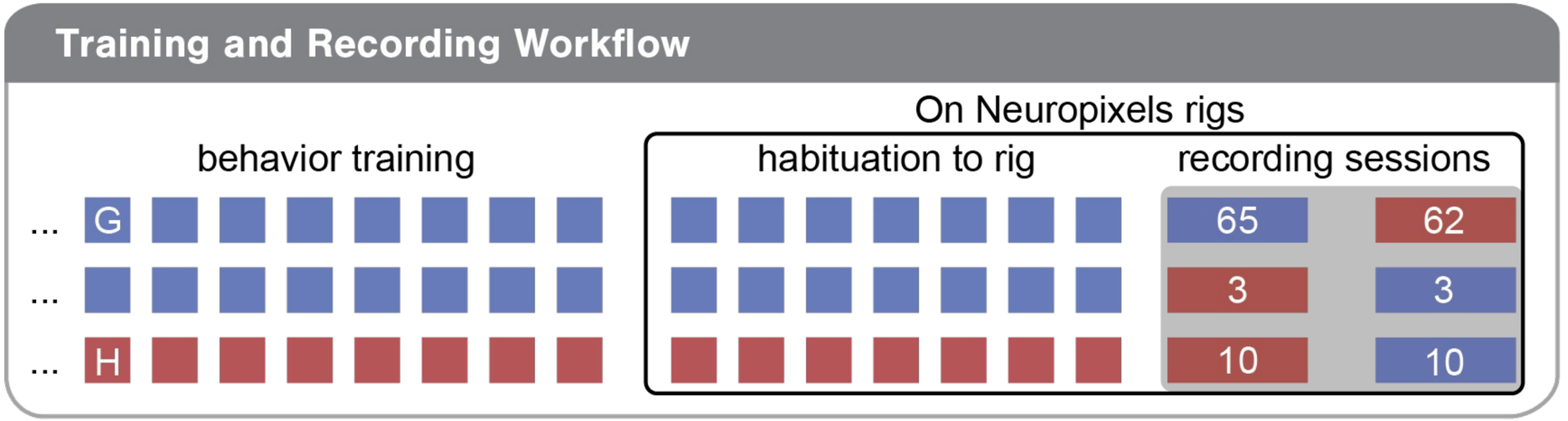
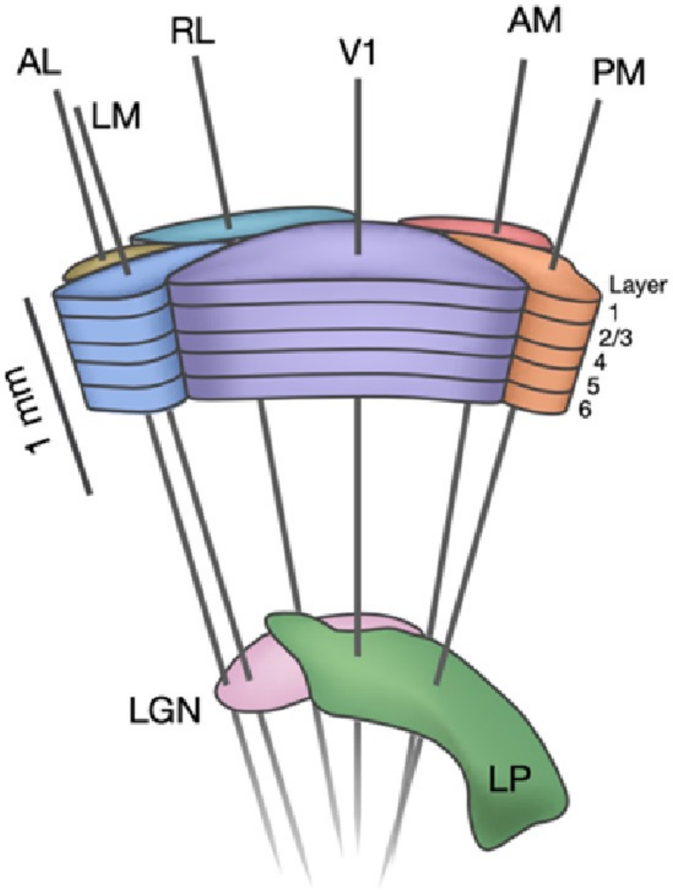


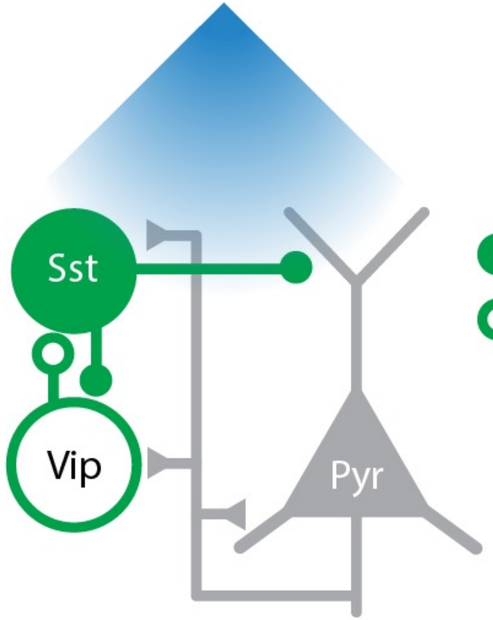
Image sets used during behavior
Unique to G
Shared (G and H)
Unique to H
im036_r
im012_r
im115_r
im083_r
im104_r
im114_r
im024_r
im047_r
im044_r
im078_r
im111_r
im005_r
im087_r
im034_r


Training and Recording Workflow
On Neuropixels rigs
behavior training
habituation to rig
recording sessions
...
G
...
...
H
65
62
3
3
10
10

Neuropixels recordings



Optotagging to identify two cortical inhibitory cell classes

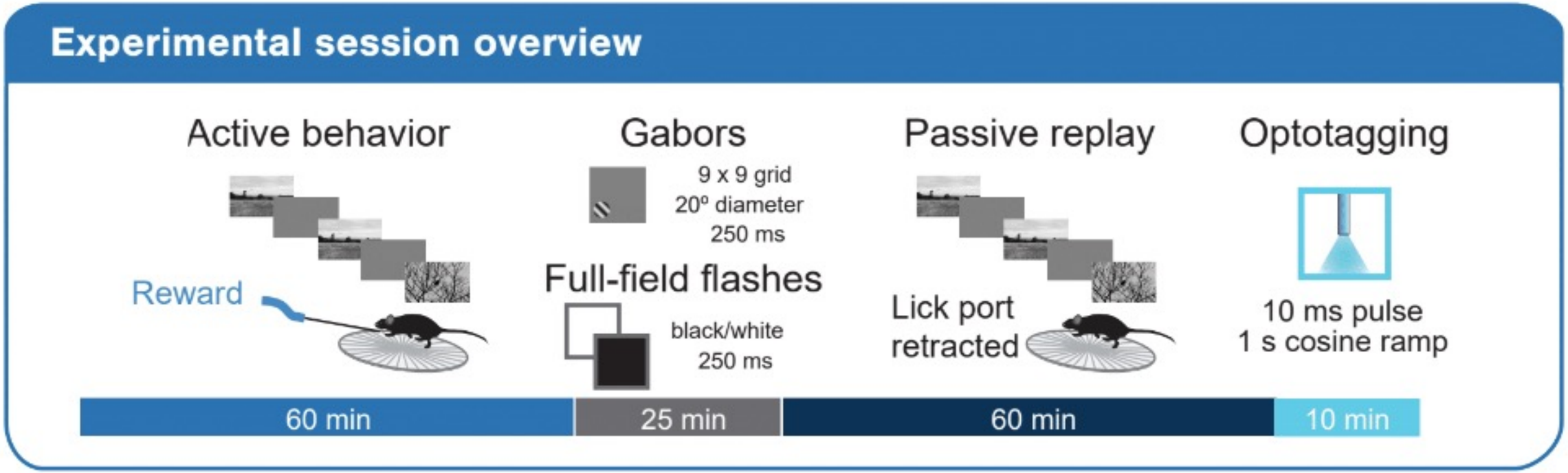


- Sst-IRES-Cre x Ai32
- Vip-IRES-Cre x Ai32

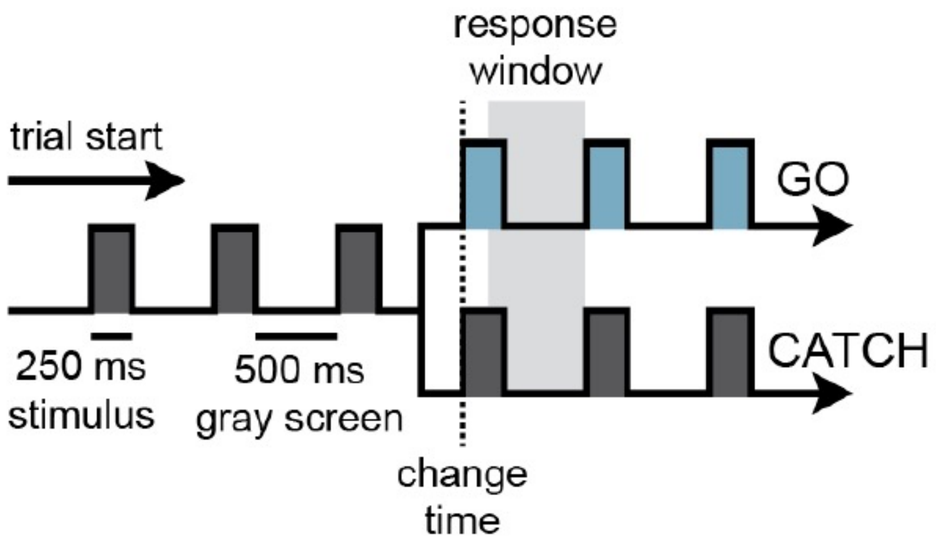
+

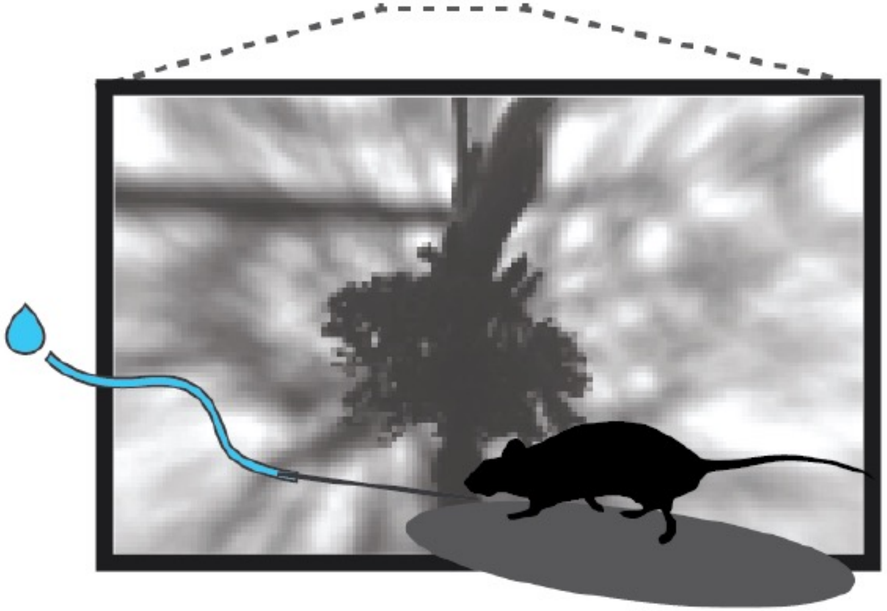
C57BL6J (control)

Recording session

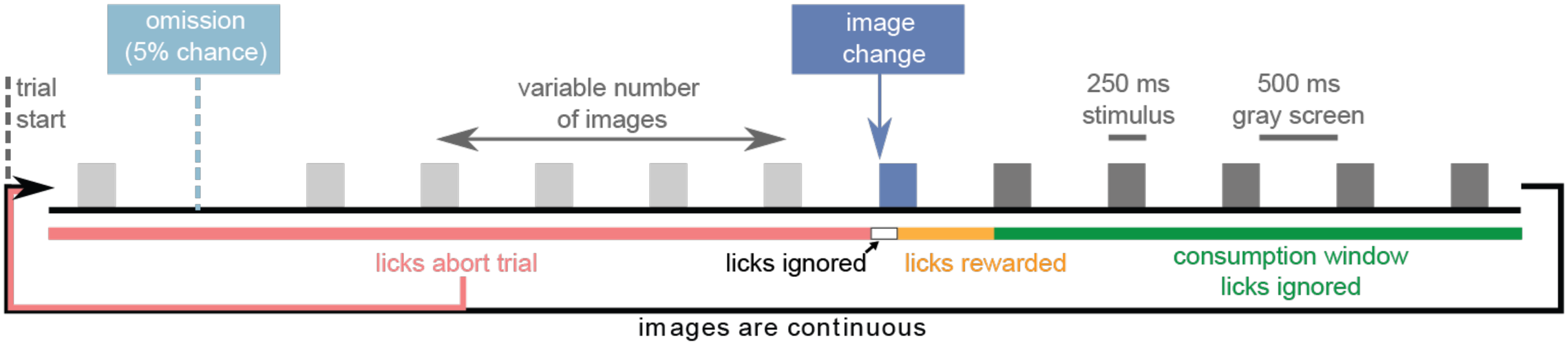


Visual change detection task

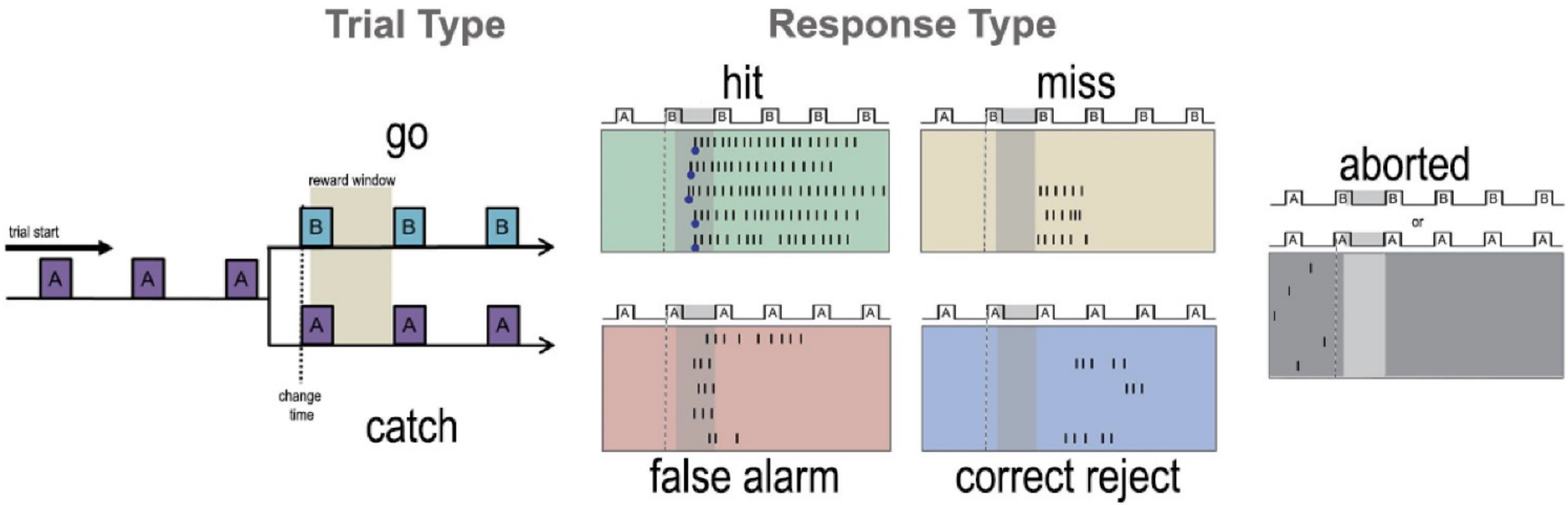




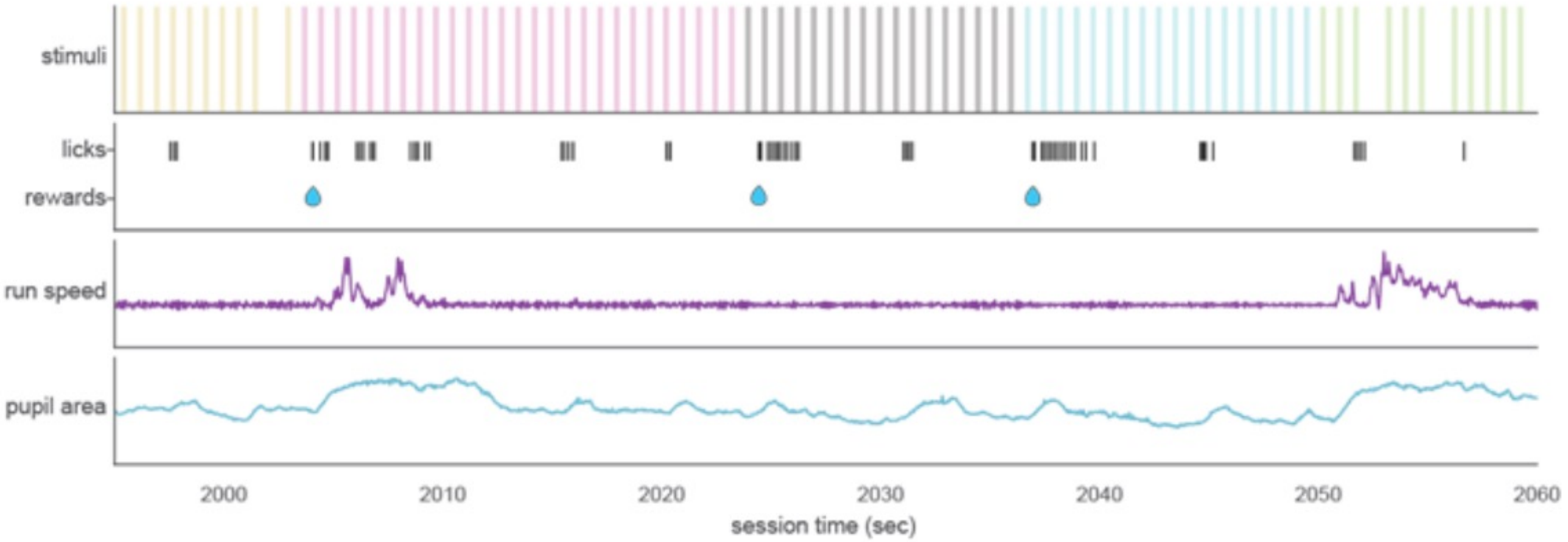
Trial structure



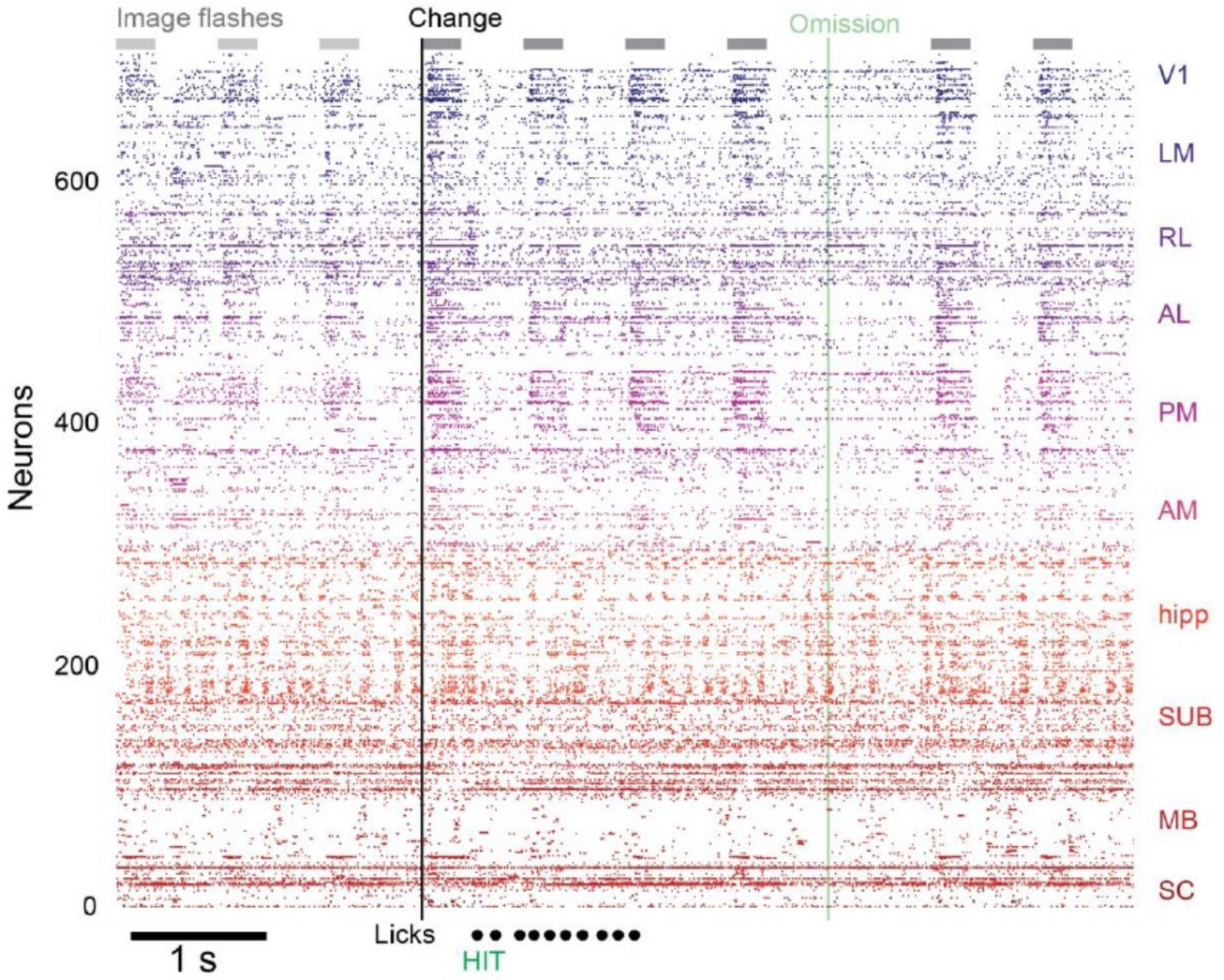
Trial types


Trial Type
Response Type
go
reward window
trial start
A
B
change time
catch
hit
miss
aborted
or
false alarm
correct reject

Example behavior



Example spiking activity



Dataset

- 103 sessions
- >30 brain regions sampled
- >200K recorded units

Analysis can address:

- Perception
- Stimulus novelty
- Task-engagement
- Multi-regional interactions
- …

NWB files

AIND_VBN	21 July 2025 at 19:01	424.49 GB
code	13 August 2025 at 15:57	875 KB
pdfs	23 June 2025 at 13:34	8 MB
visual-behavior-neuropixels	23 June 2025 at 13:15	424.48 GB
behavior_ecephys_sessions	23 June 2025 at 13:15	424.31 GB
1043752325	23 June 2025 at 12:24	2.76 GB
ecephys_session_1043752325.nwb	15 August 2024 at 20:12	2.76 GB
1044016459	23 June 2025 at 12:24	2.93 GB
ecephys_session_1044016459.nwb	15 August 2024 at 20:12	2.93 GB
1044385384	23 June 2025 at 12:25	2.65 GB
ecephys_session_1044385384.nwb	15 August 2024 at 20:13	2.65 GB
1044389060	23 June 2025 at 12:25	2.15 GB
1044594870	23 June 2025 at 12:25	2.66 GB
1044597824	23 June 2025 at 12:26	2.72 GB
1046166369	23 June 2025 at 12:26	4.35 GB
1046581736	23 June 2025 at 12:26	2.73 GB
1047969464	23 June 2025 at 12:27	3.2 GB
1047977240	23 June 2025 at 12:27	2.61 GB
1048189115	23 June 2025 at 12:28	2.76 GB
1048196054	23 June 2025 at 12:28	3.1 GB
1049273528	23 June 2025 at 12:28	2.98 GB
1049514117	23 June 2025 at 12:29	2.8 GB

Python code – Myriam Hamon

```python
# plotting all neurons all trials

time_before_change = 1
duration = 2.5
area_change_responses =[]
for iu, unit in area_units.iterrows():
    unit_spike_times = spike_times[iu]
    unit_change_response, bins = makePSTH(unit_spike_times,
                                          change_times-time_before_change,
                                          duration, binSize=0.01)
    area_change_responses.append(unit_change_response)
area_change_responses = np.array(area_change_responses)
bin_centers_ephys = bins[:-1] + 0.01 / 2 -1

fig, ax = plt.subplots(5, 1, figsize=(10, 10), layout='constrained',  gridspec_kw={'height_ratios': [2, 1, 1, 1,1]})

xticks = np.arange(0, bins.size-1, 20)
xtick_labels = np.round(bins[:-1:20] - time_before_change, 2)
# Plot heatmap
clims = [np.percentile(area_change_responses, p) for p in (0.1, 99.9)]
im = ax[0].imshow(area_change_responses, aspect='auto', clim=clims, cmap='viridis')
cbar = fig.colorbar(im, ax=ax[0], orientation='vertical', label='Firing Rate (Hz)')

ax[0].set_title(f'Change-Aligned Responses in {area_of_interest}', fontsize=14)
ax[0].set_ylabel('Neuron (sorted by depth)', fontsize=12)
ax[0].set_xticks(xticks)
ax[0].set_xticklabels(xtick_labels)
ax[0].axvline(100, color='red', linestyle='--', linewidth=1, label='Change Time')

ax[1].plot( bin_centers_ephys, area_change_responses.mean(0))
ax[1].set_title(f'Change-Aligned Responses in {area_of_interest}', fontsize=14)
ax[1].set_ylabel('Average population activity in Hz', fontsize=12)
ax[1].axvline(0, color='red', linestyle='--', linewidth=1, label='Change Time')

ax[2].plot(bin_centers_speed, mean_speed, color='black', lw=2)
ax[2].set_title('Mean Running Speed', fontsize=14)
ax[2].set_xlabel('Time from change (s)', fontsize=12)
ax[2].set_ylabel('Speed (cm/s)', fontsize=12)
ax[2].axvline(0, color='red', linestyle='--', linewidth=1, label='Change Time')

ax[3].plot(bin_centers_licks, lick_rate, color='black', lw=2)
ax[3].set_title('Lick rate', fontsize=14)
ax[3].set_xlabel('Time from change (s)', fontsize=12)
ax[3].set_ylabel('Lick rate', fontsize=12)
ax[3].axvline(0, color='red', linestyle='--', linewidth=1)

ax[4].plot(bin_centers_eye, mean_eye_x, color='black', lw=2)
ax[4].set_title('Mean Pupil X coordinate', fontsize=14)
ax[4].set_xlabel('Time from change (s)', fontsize=12)
ax[4].set_ylabel('Speed (cm/s)', fontsize=12)
ax[4].axvline(0, color='red', linestyle='--', linewidth=1)

plt.show()
```