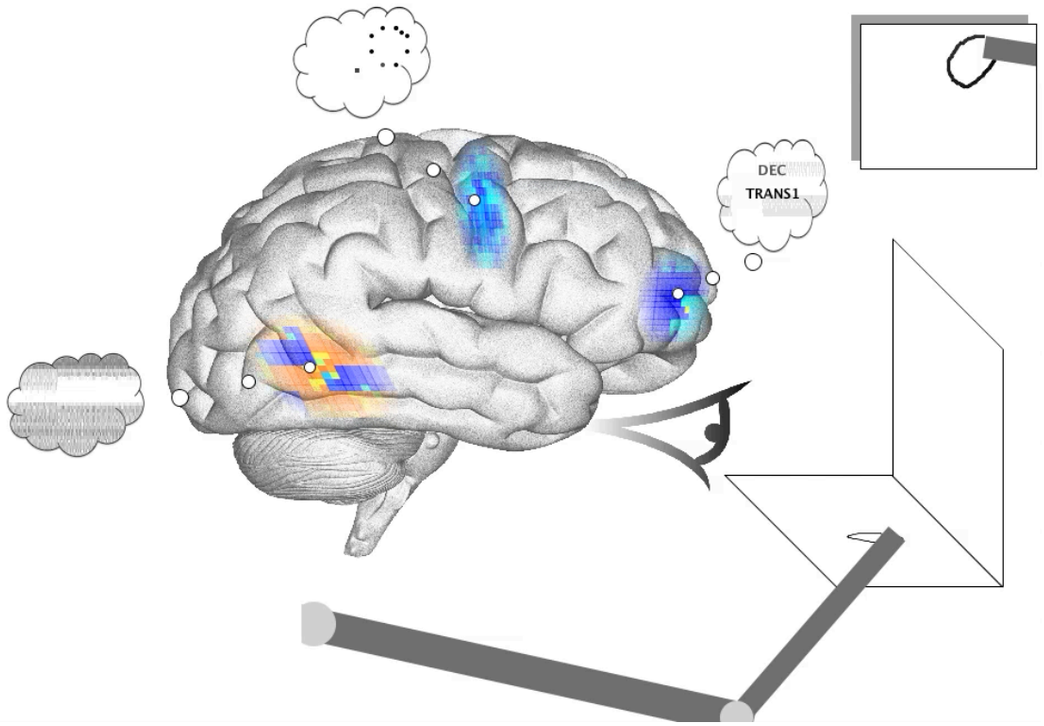
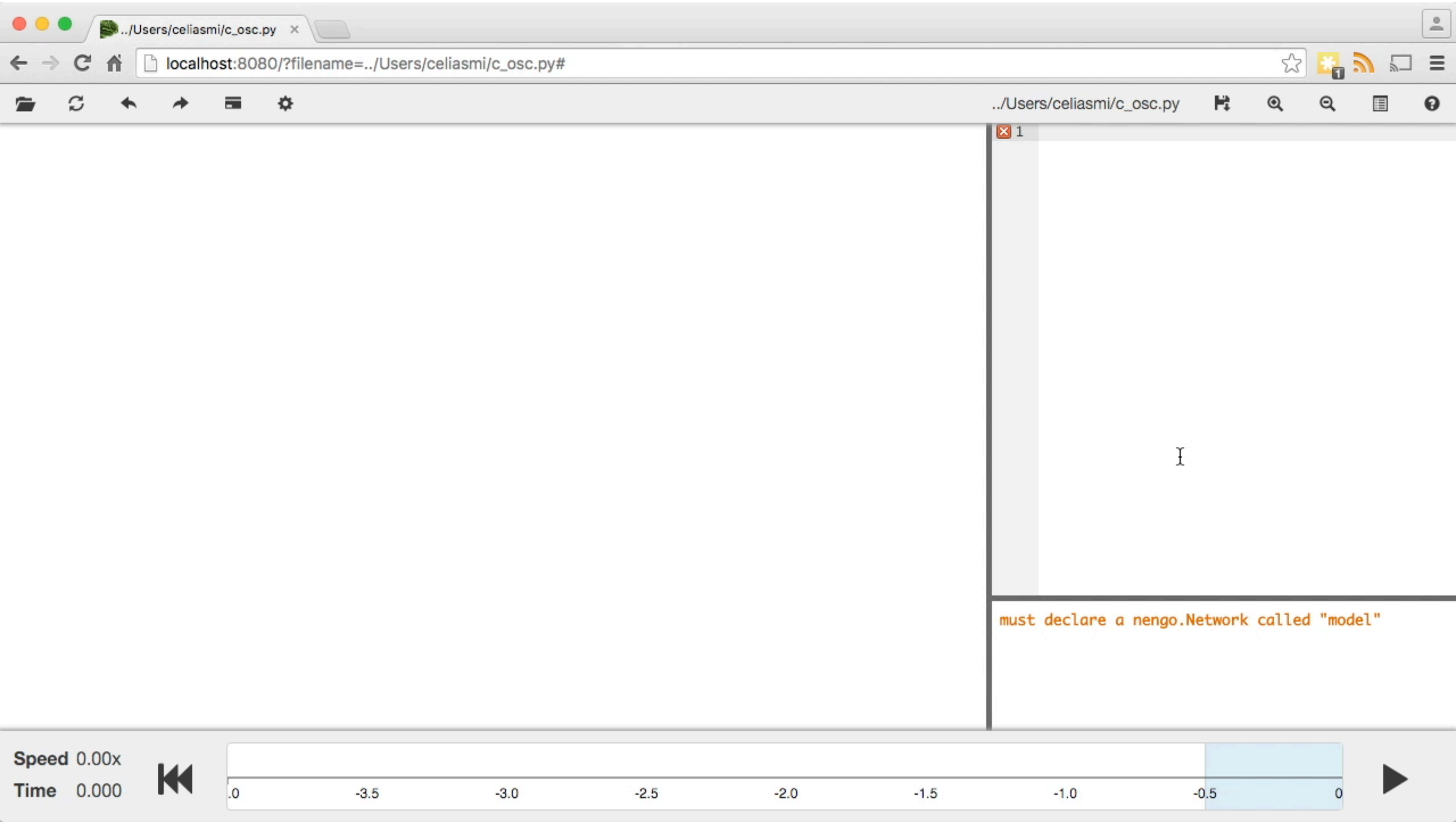


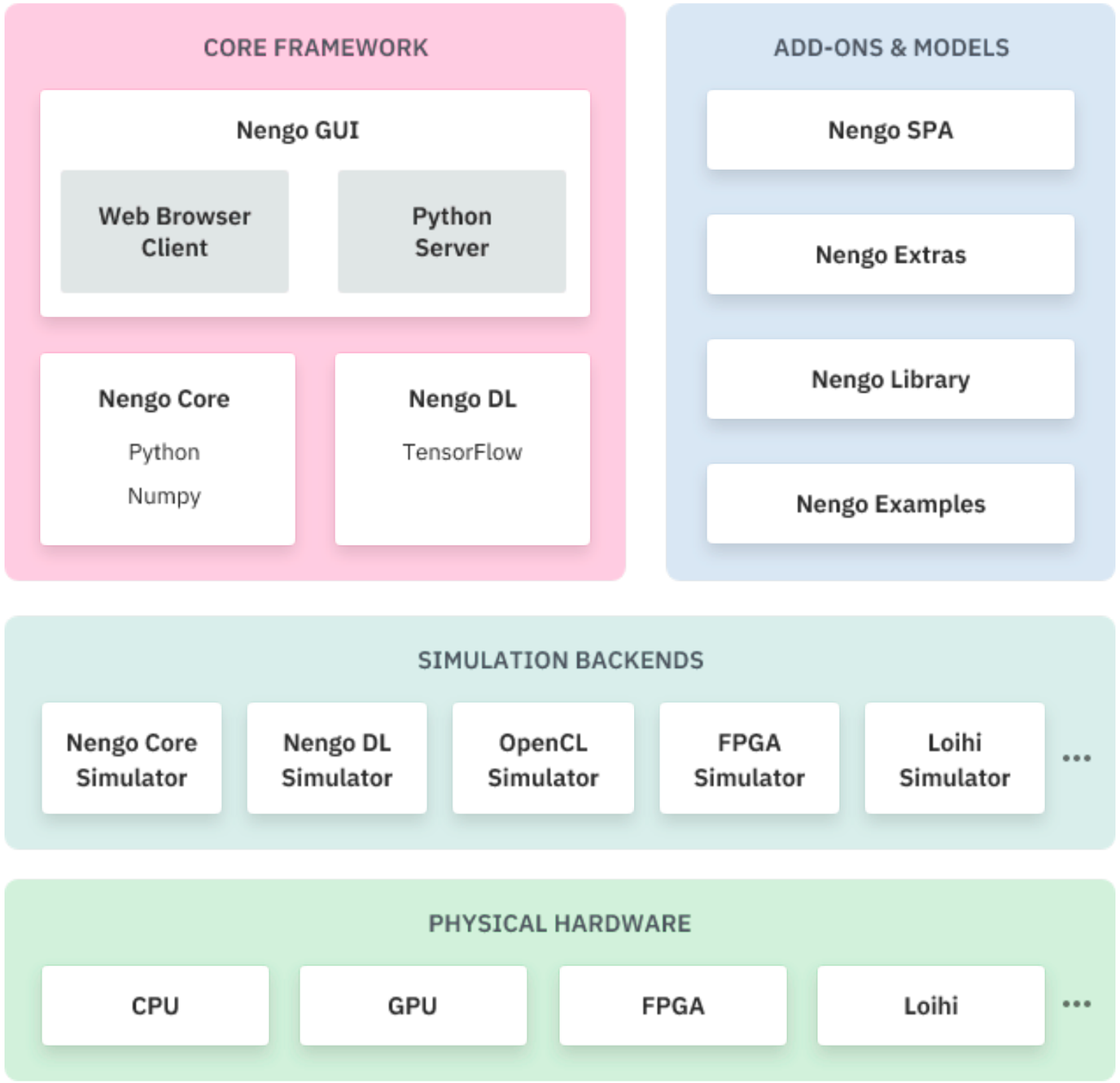
Things you can do with Nengo

Centre for Theoretical Neuroscience, U of Waterloo
Chris & Terry

Nengo

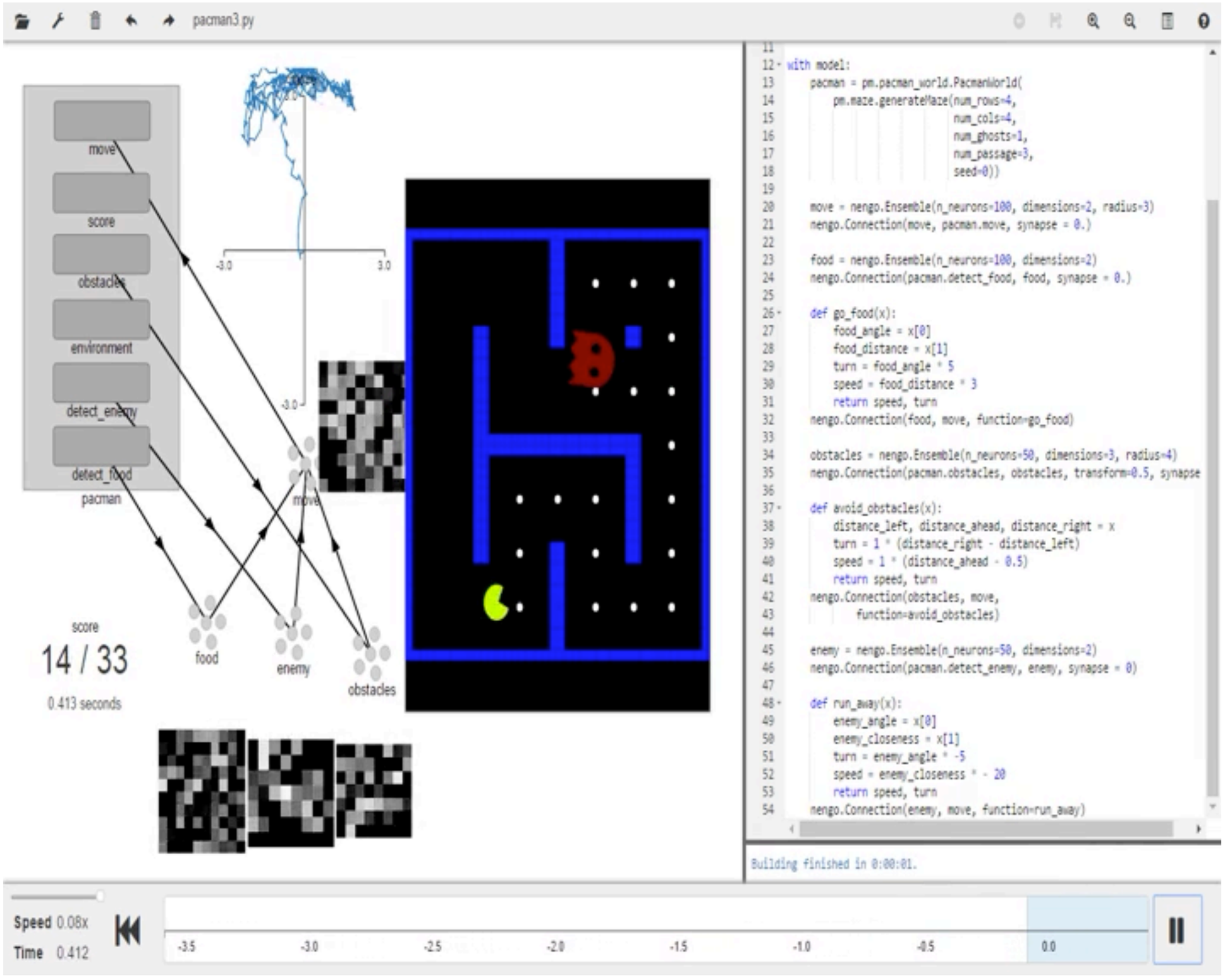

../Users/celiasmi/c_osc.py
localhost:8080/?filename=../Users/celiasmi/c_osc.py#
../Users/celiasmi/c_osc.py
1
must declare a nengo.Network called "model"
Speed 0.00x
Time 0.000
.0
-3.5
-3.0
-2.5
-2.0
-1.5
-1.0
-0.5
0

Nengo ecosystem



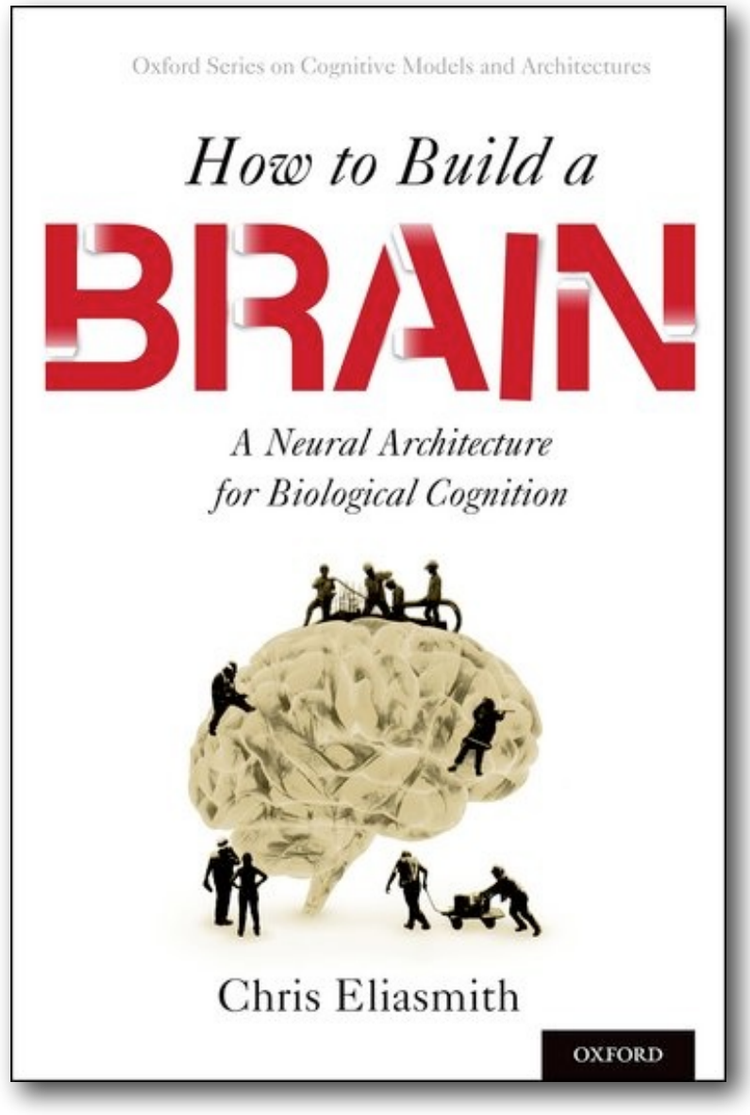
Examples

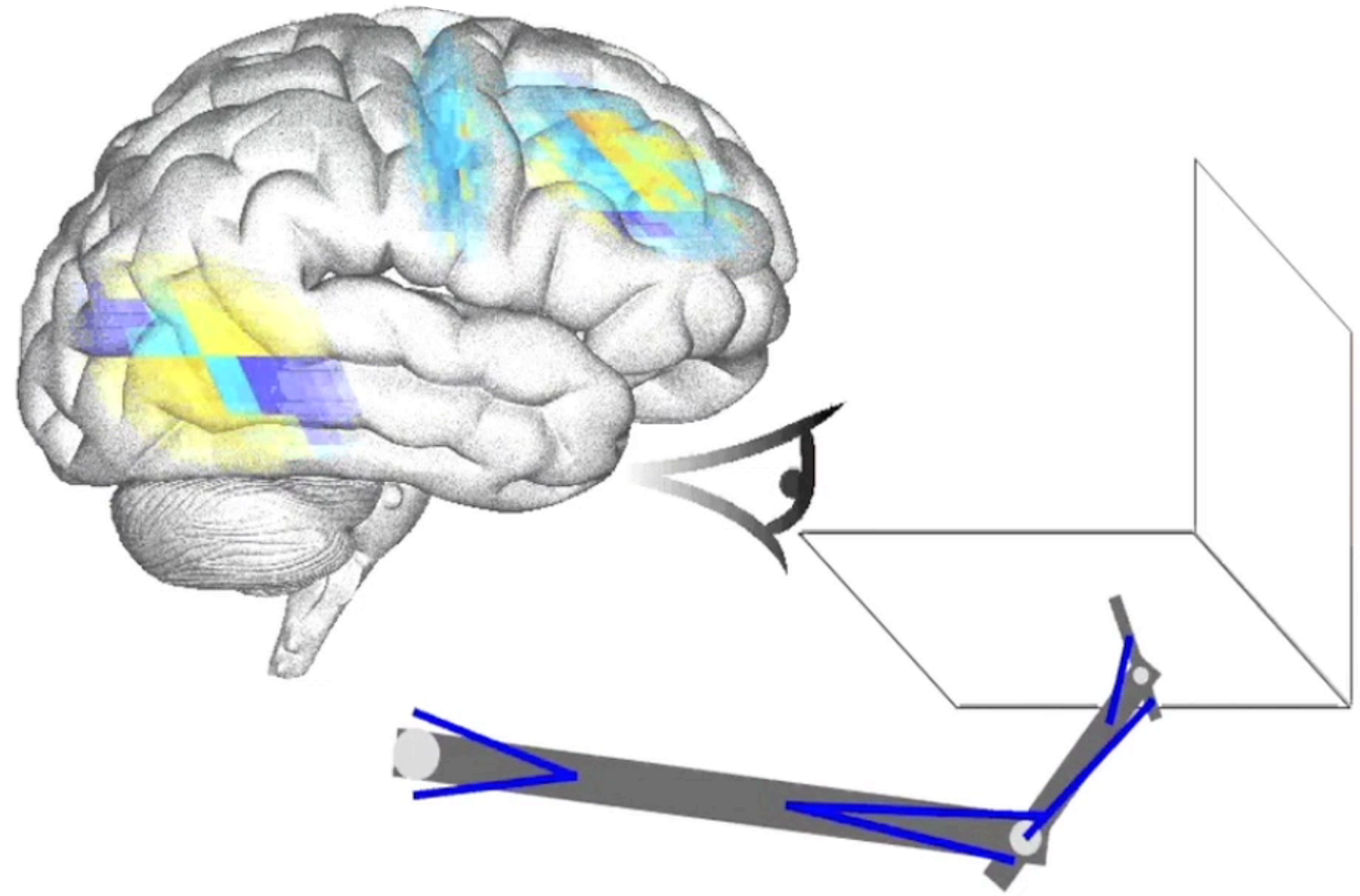
Pacman


pacman3.py
move
score
obstacles
environment
detect_enemy
detect_food
pacman
3.0
-3.0
3.0
-3.0
move
food
enemy
obstacles
score
14 / 33
0.413 seconds
with model:
pacman = pm.pacman_world.PacmanWorld(
pm.maze.generateMaze(num_rows=4,
num_cols=4,
num_ghosts=1,
num_passage=3,
seed=0))
move = nengo.Ensemble(n_neurons=100, dimensions=2, radius=3)
nengo.Connection(move, pacman.move, synapse = 0.)
food = nengo.Ensemble(n_neurons=100, dimensions=2)
nengo.Connection(pacman.detect_food, food, synapse = 0.)
def go_food(x):
food_angle = x[0]
food_distance = x[1]
turn = food_angle * 5
speed = food_distance * 3
return speed, turn
nengo.Connection(food, move, function=go_food)
obstacles = nengo.Ensemble(n_neurons=50, dimensions=3, radius=4)
nengo.Connection(pacman.obstacles, obstacles, transform=0.5, synapse
def avoid_obstacles(x):
distance_left, distance_ahead, distance_right = x
turn = 1 * (distance_right - distance_left)
speed = 1 * (distance_ahead - 0.5)
return speed, turn
nengo.Connection(obstacles, move,
function=avoid_obstacles)
enemy = nengo.Ensemble(n_neurons=50, dimensions=2)
nengo.Connection(pacman.detect_enemy, enemy, synapse = 0)
def run_away(x):
enemy_angle = x[0]
enemy_closeness = x[1]
turn = enemy_angle * -5
speed = enemy_closeness * - 20
return speed, turn
nengo.Connection(enemy, move, function=run_away)
Building finished in 0:00:01.
Speed 0.08x
Time 0.412
-3.5
-3.0
-2.5
-2.0
-1.5
-1.0
-0.5
0.0

Spaun 2.0

- Spaun (Semantic Pointer Architecture Unified Network)
- Simulated in Nengo
- 6.3 million neurons (was 2.5)
- 20 billion connections (was 7.5)
- 12 different tasks (was 8)
- Including basic perceptual tasks...





And more cognitive tasks...

1	11	111
4	44	444
5	55	?

7

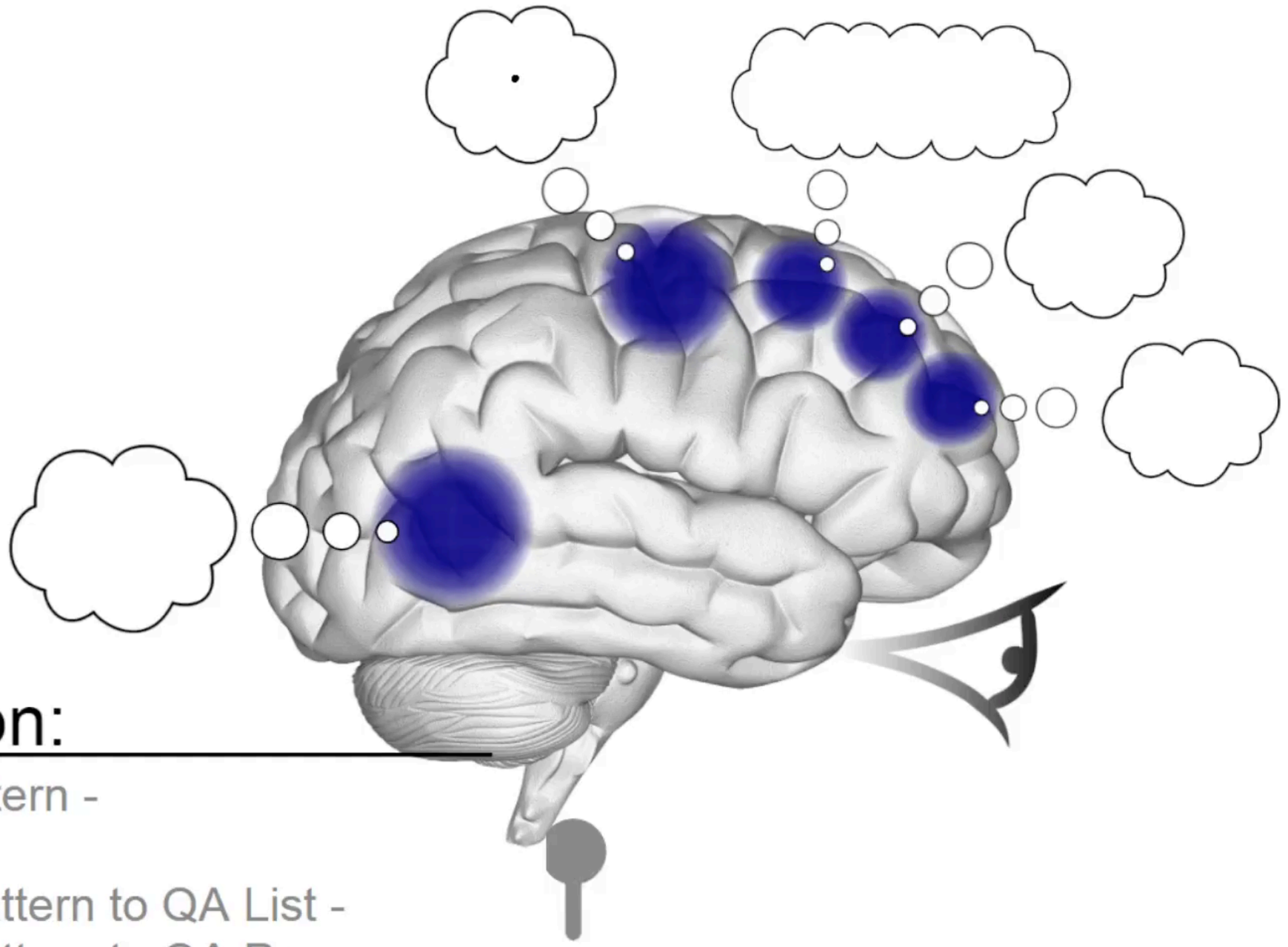
7

?

With no changes to Spaun between tasks...

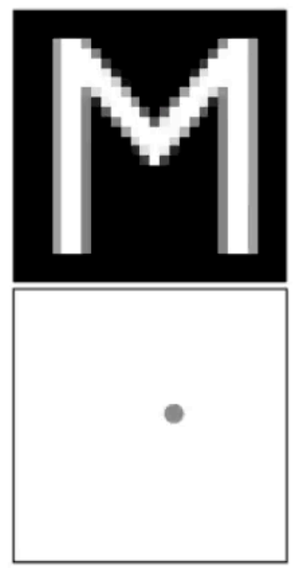
(12 tasks include: recognition, copy drawing, reinforcement learning, counting, serial working memory, question answering, RVC, RPM, stimulus matching, motor adaptation, stimulus/response, instruction following)

Instruction Following

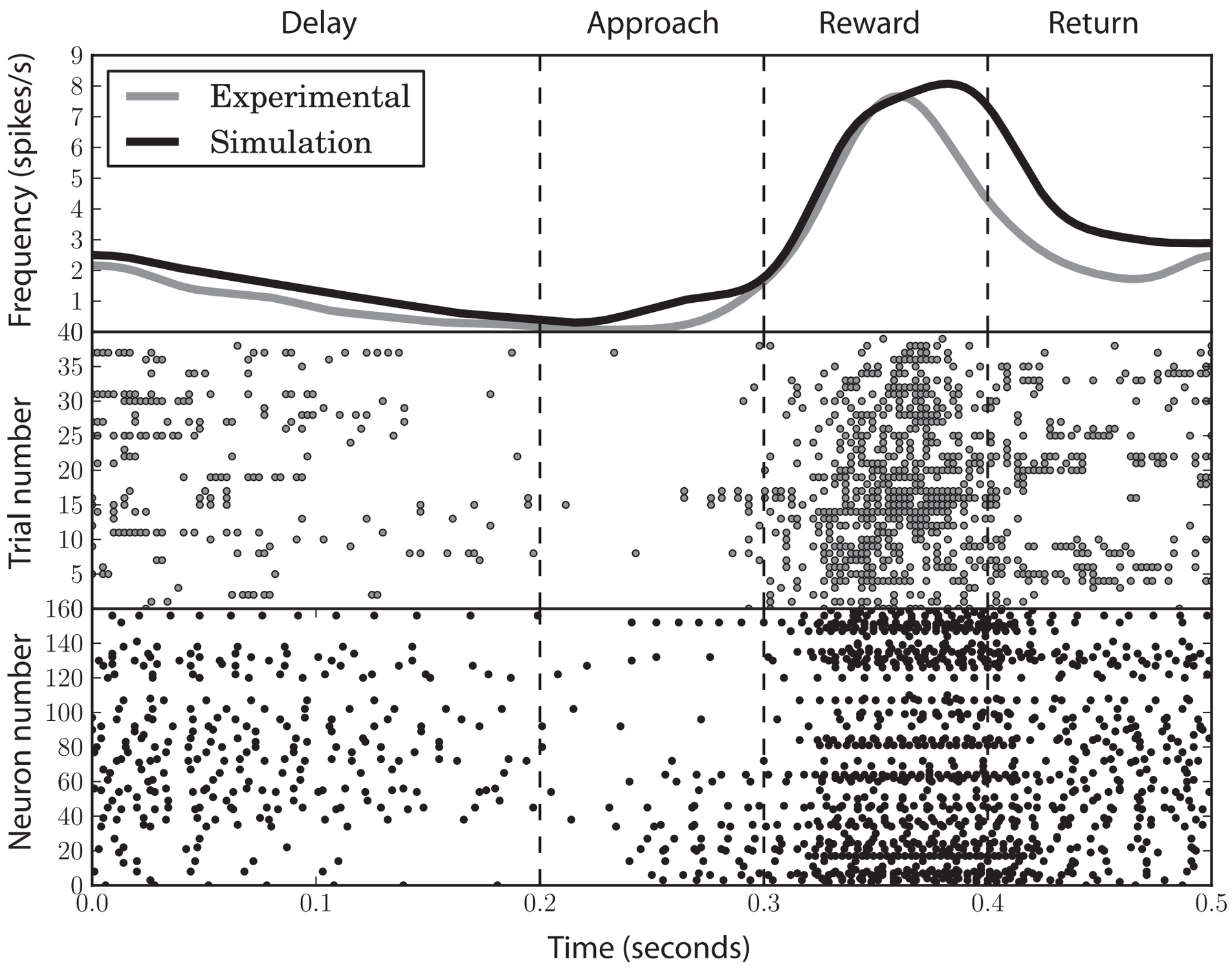


Instruction:

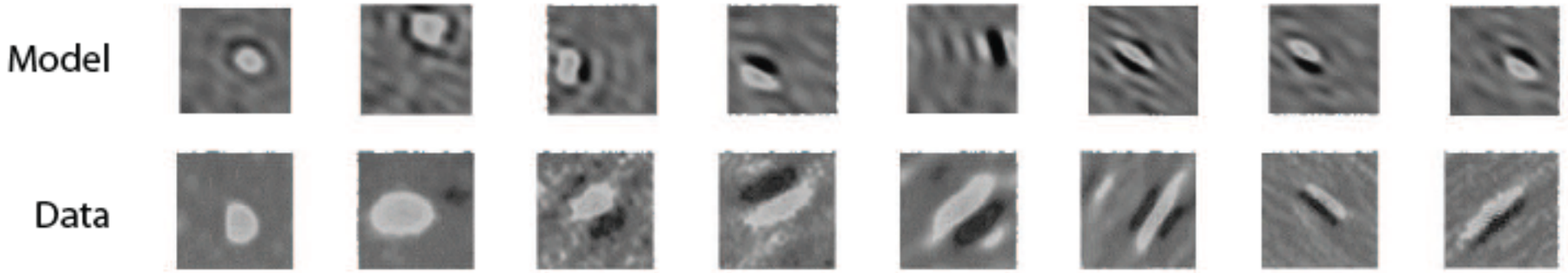
1. Find Pattern -
2. Do QA -
3. Apply Pattern to QA List -
4. Apply Pattern to QA Pos -



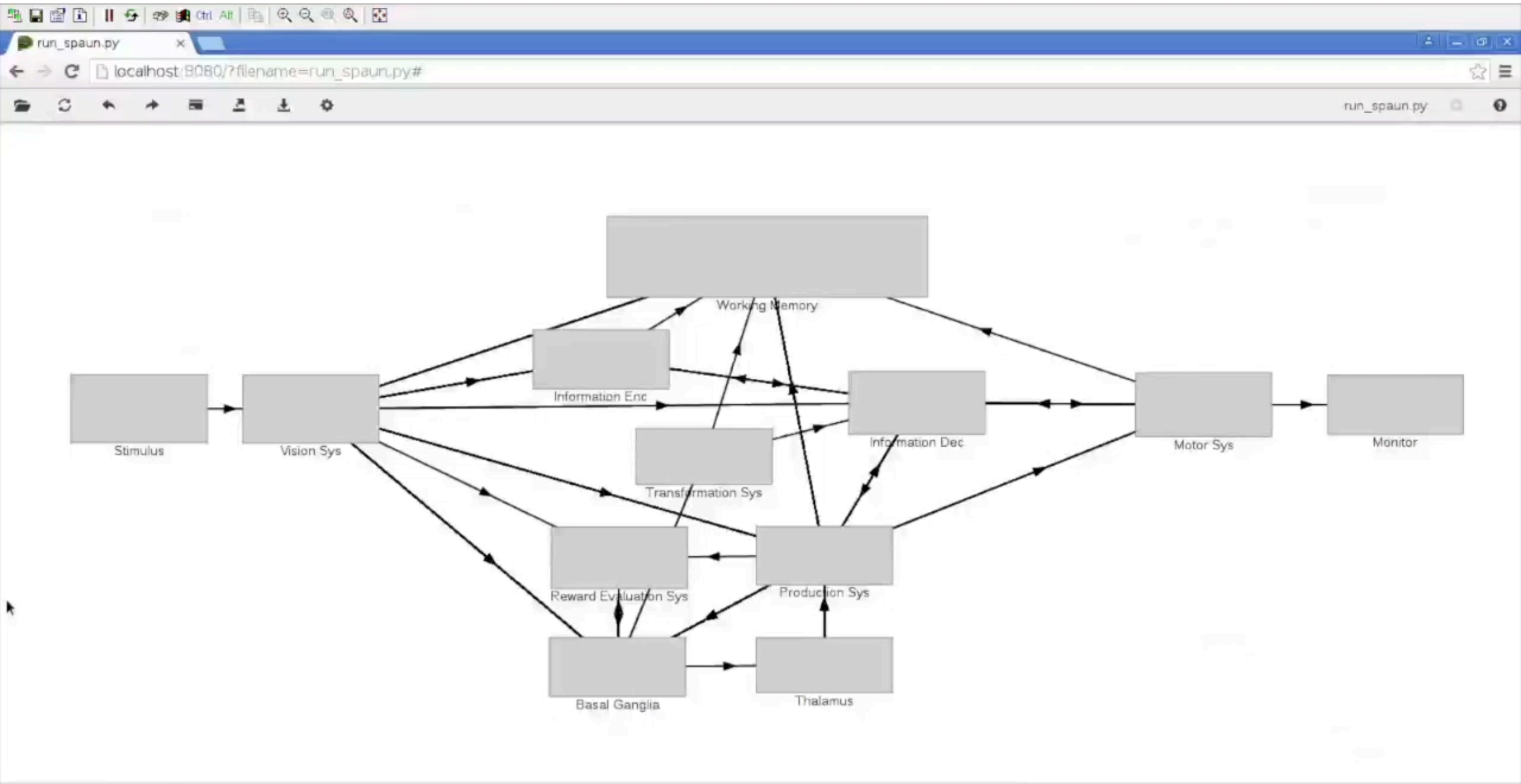
Bandit Task


Delay
Approach
Reward
Return
Experimental
Simulation
Frequency (spikes/s)
Trial number
Neuron number
Time (seconds)

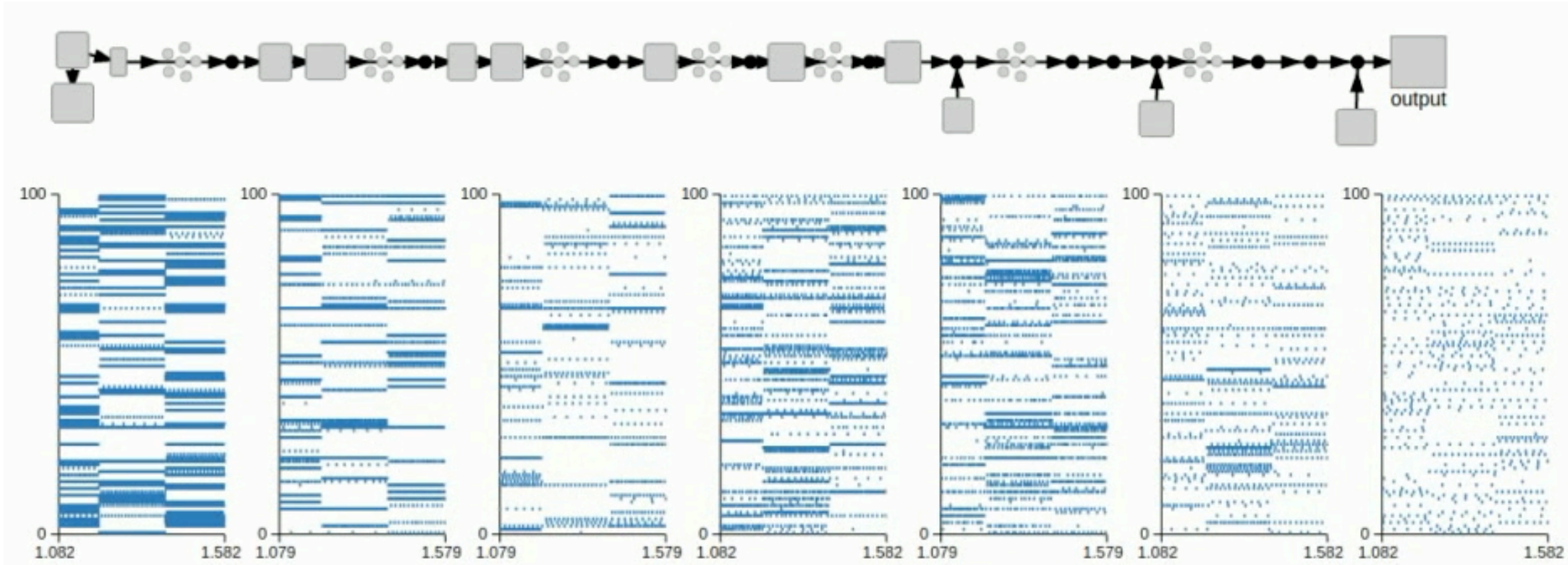
V1 Tuning



Spaun 2.0 fly through



Spiking Deep Network



PAPILLON

CARDIGAN0

TOY_TERRIER

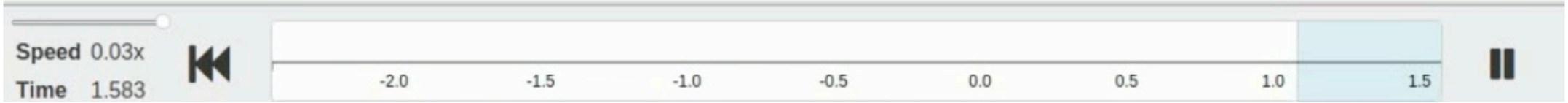
SIAMESE_CAT

SHETLAND_SHEEPDOG

CHIHUAHUA

PEMBROKE

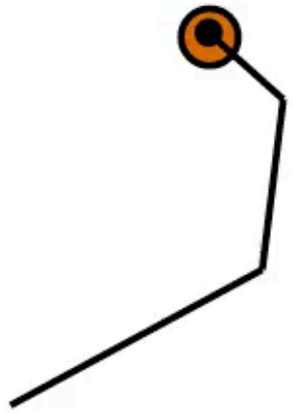
Eric Hunsberger



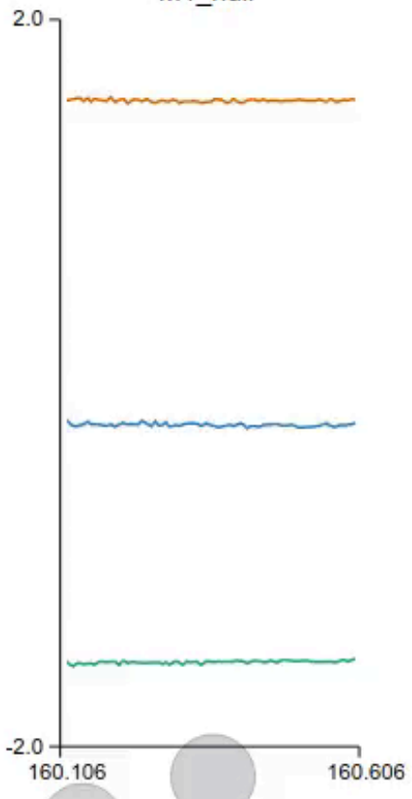
target_node

1.39

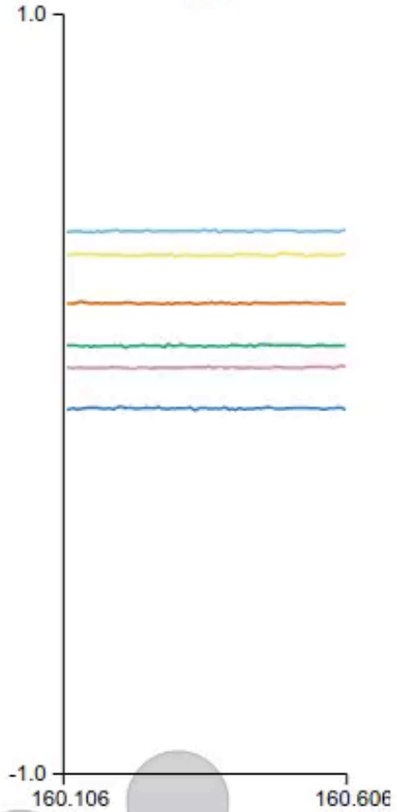
2.58



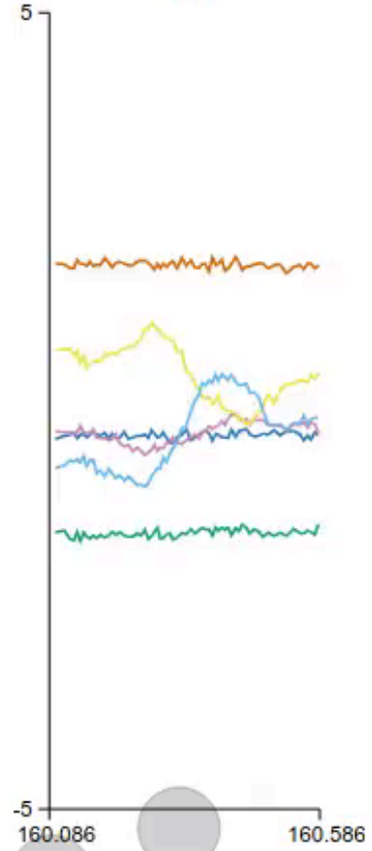
M1_null



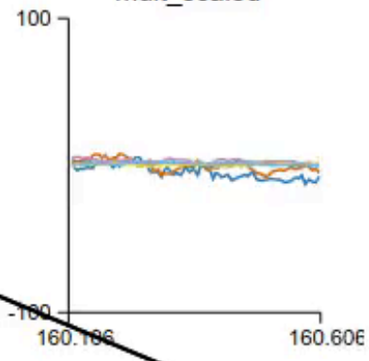
M1



CB

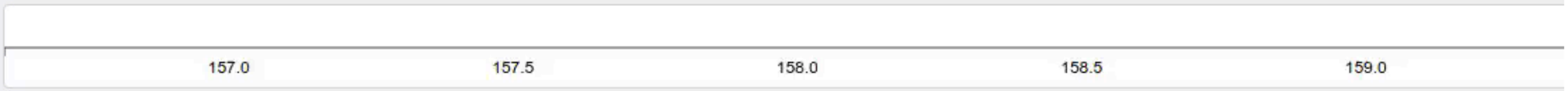


mult_scaled

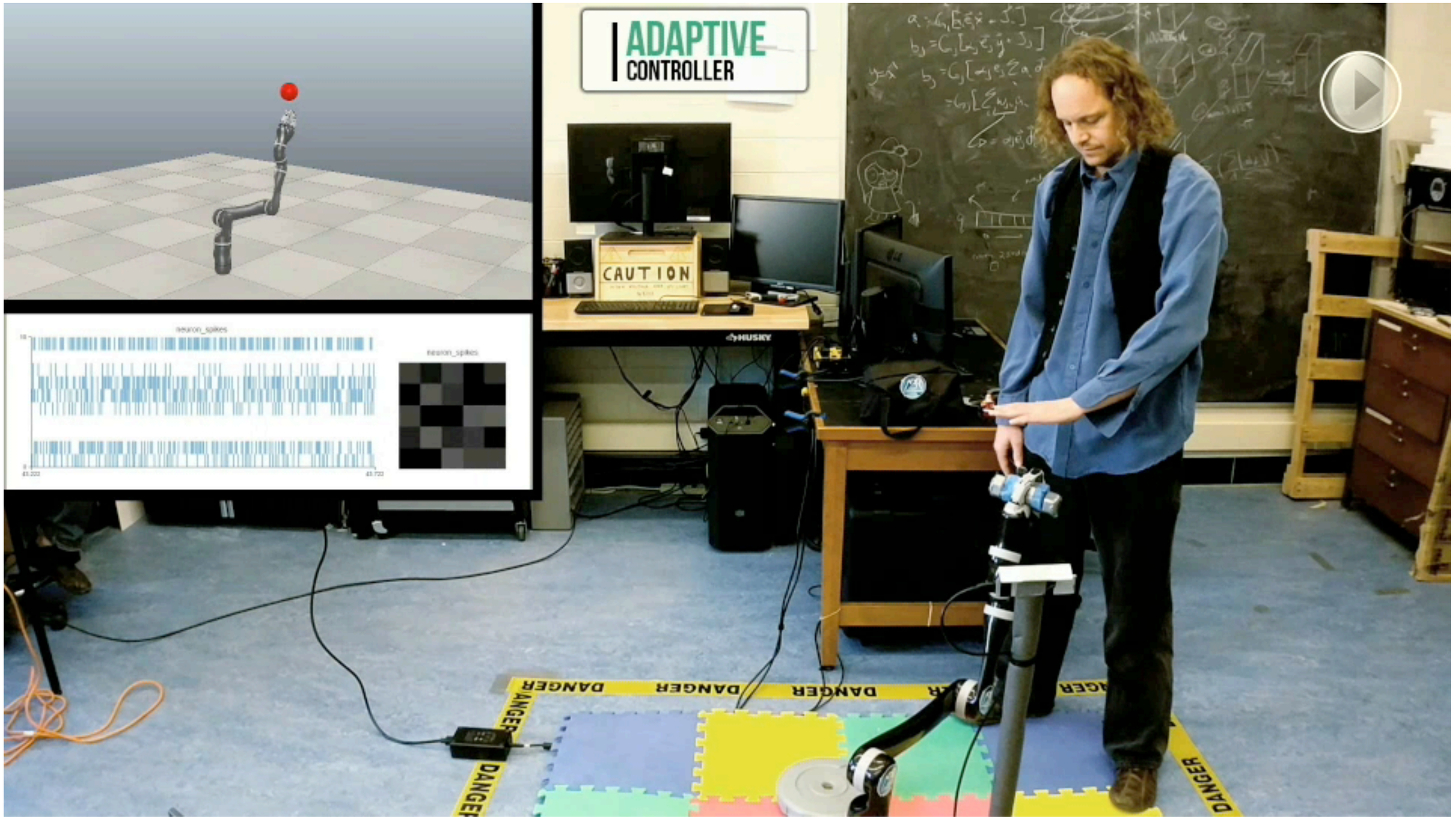


Speed 0.13x

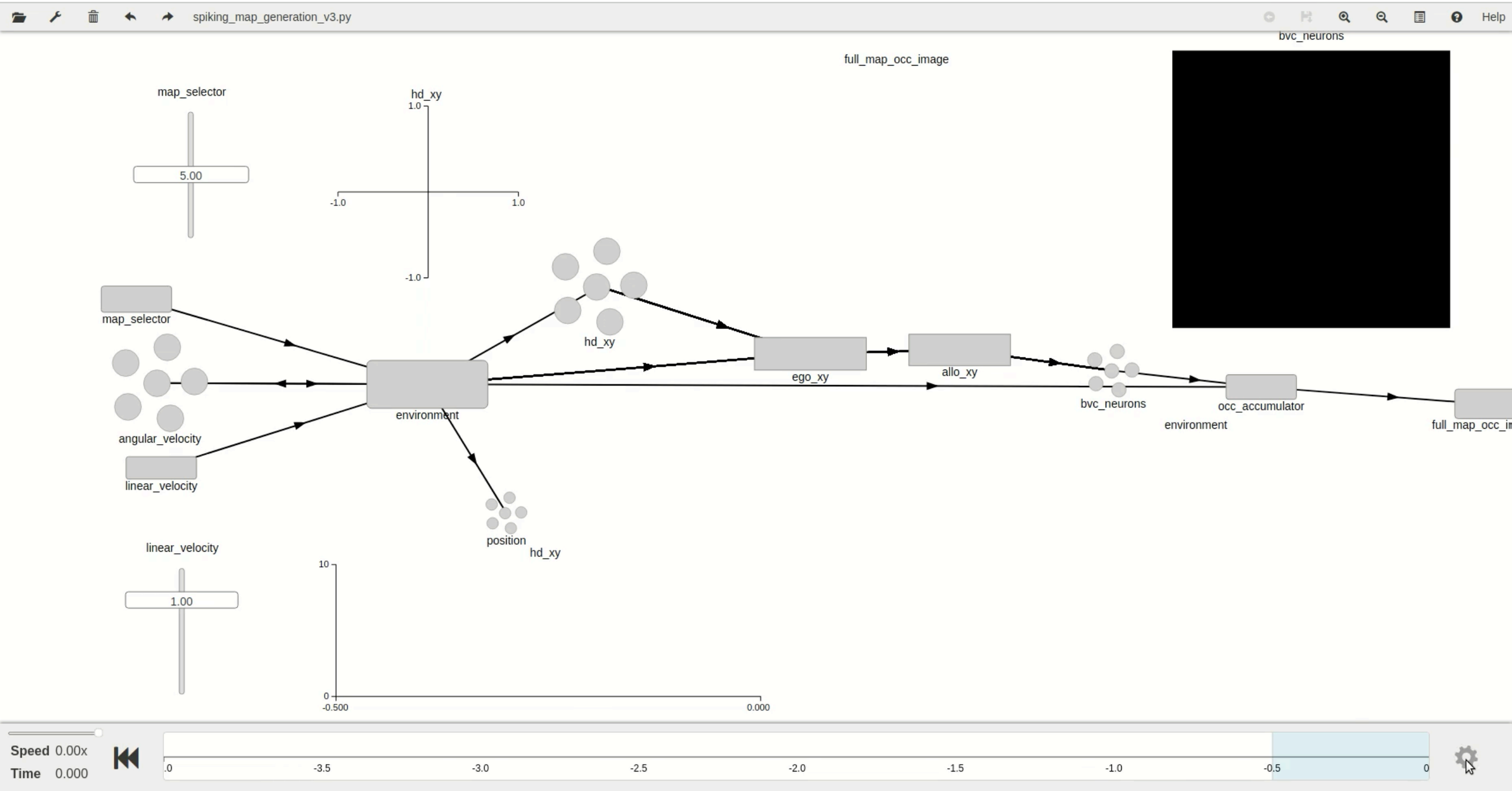
Time 160.606



Spiking Adaptive Control



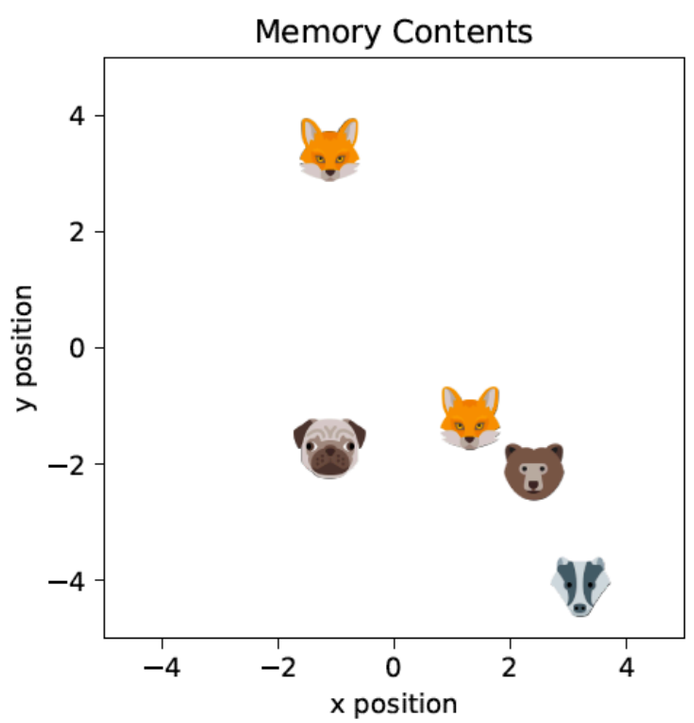
Travis DeWolf/ABR

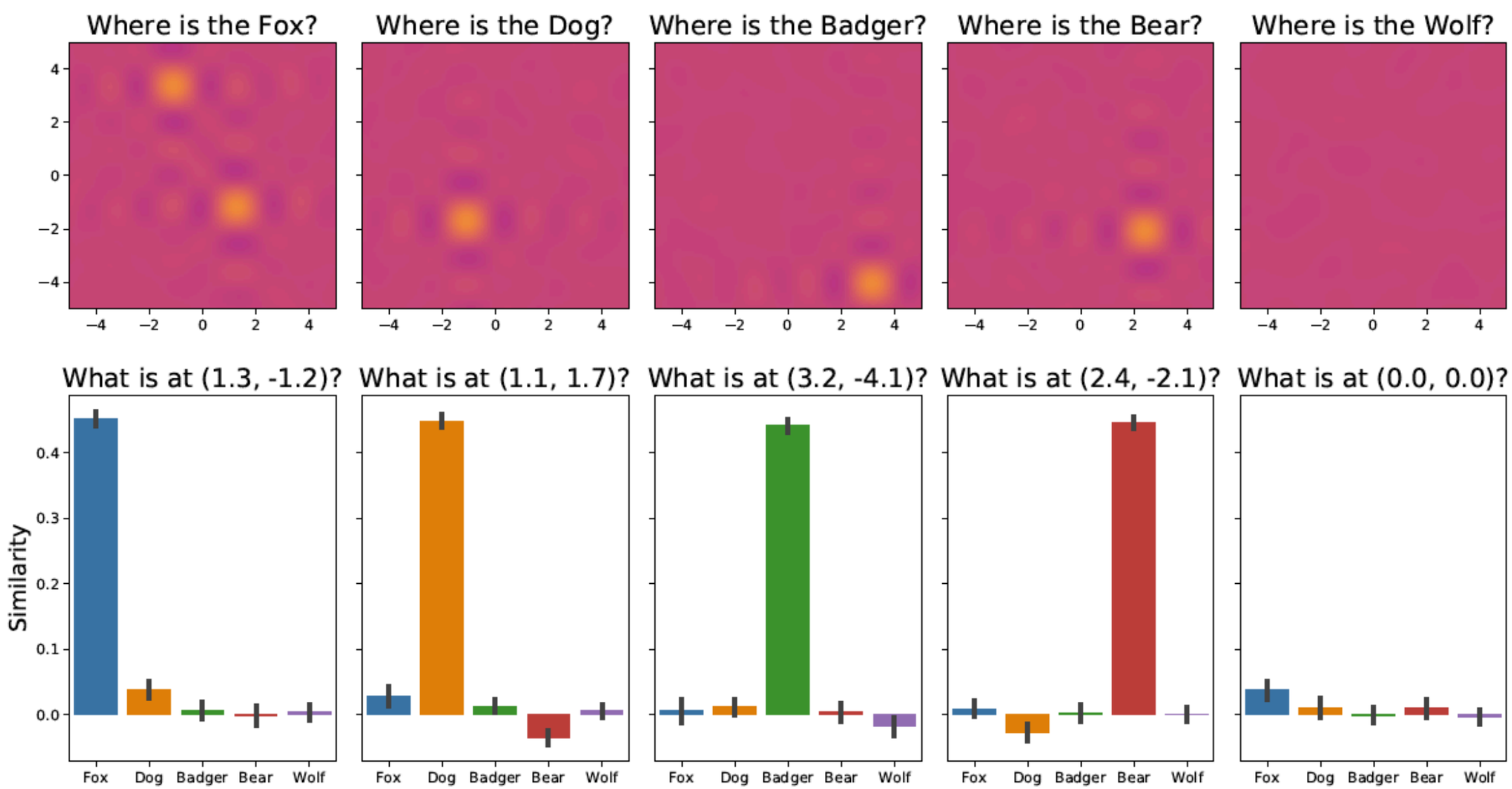


Spatial Exploration

Brent Komer

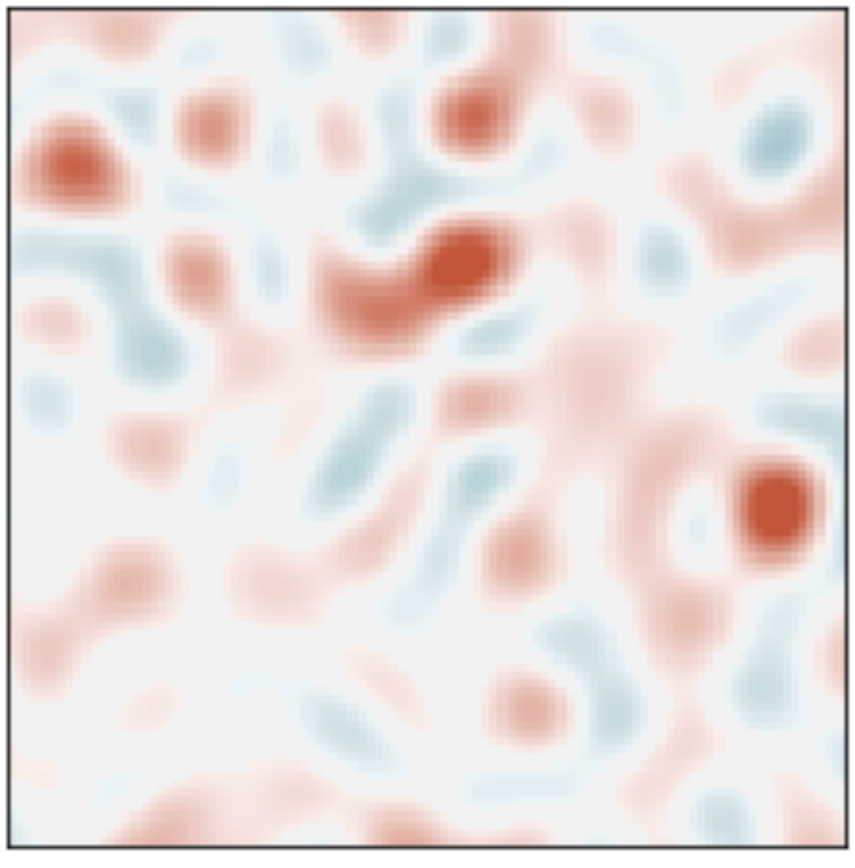
Spatial Semantic Pointers (SSPs)

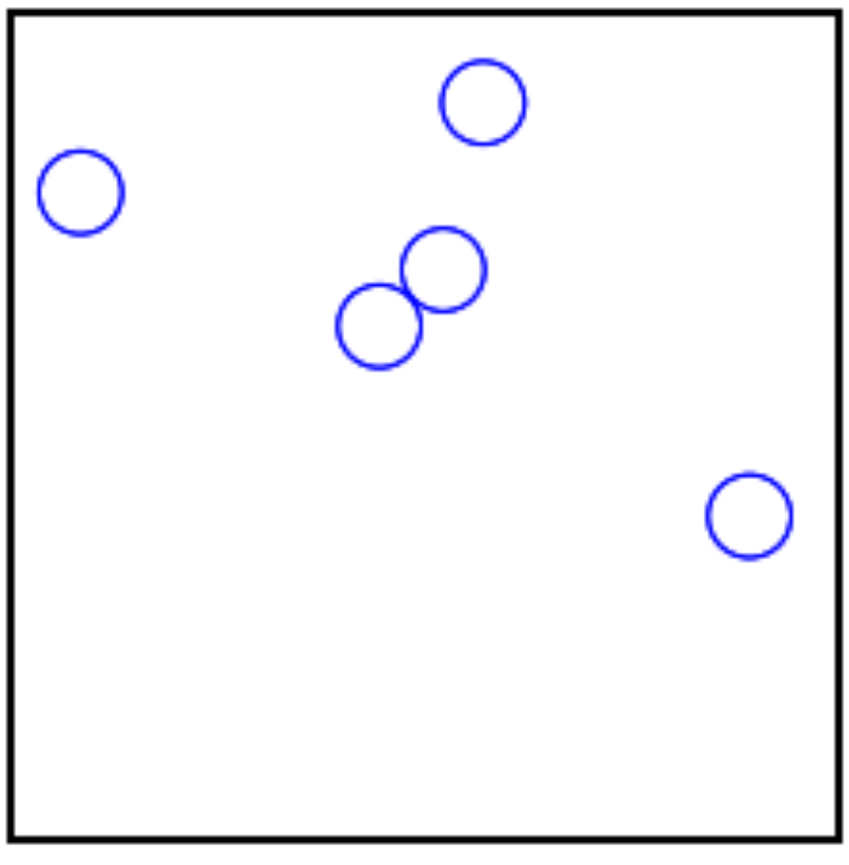




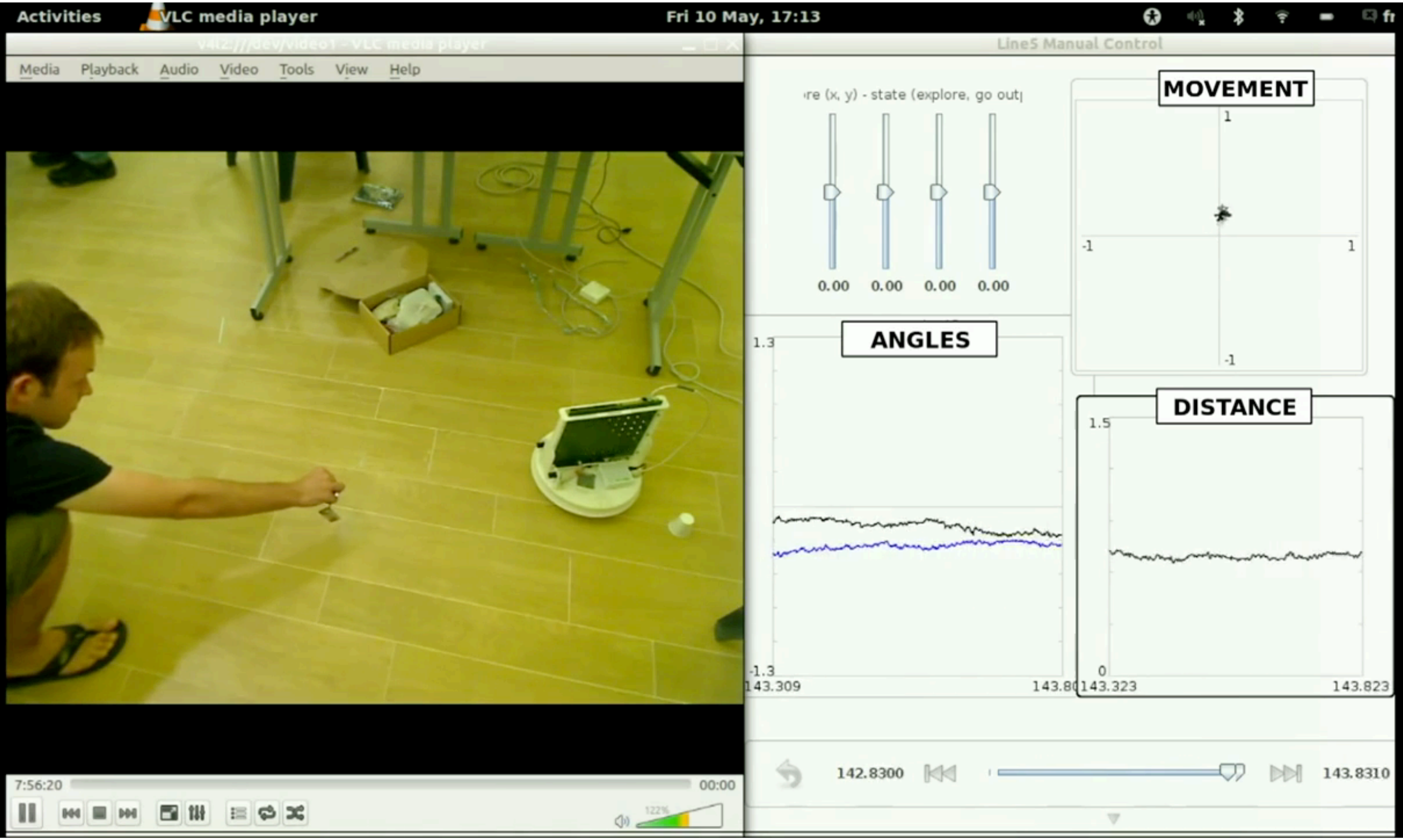
Brent Komer

Dynamic SSPs

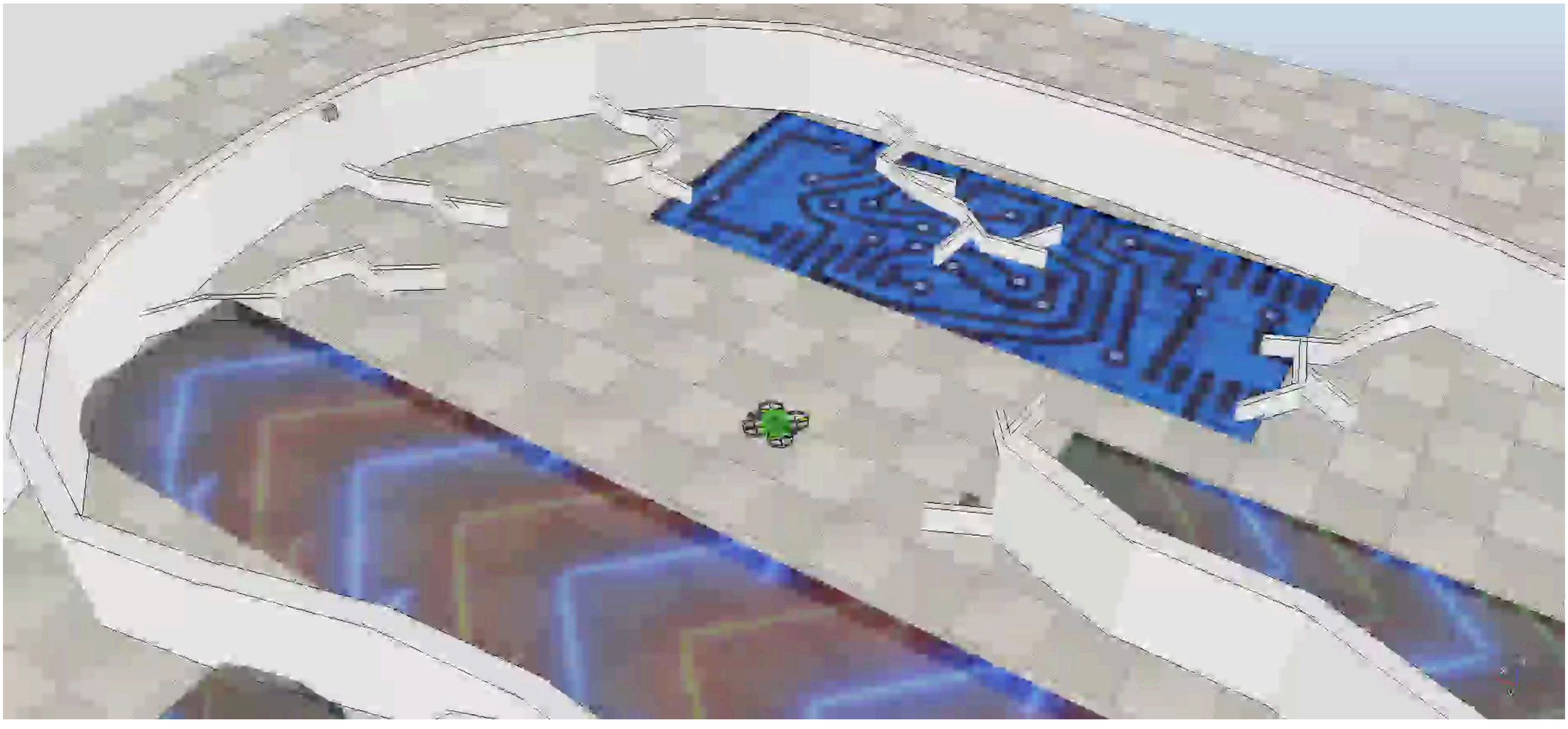




Nicole Dumont/ABR



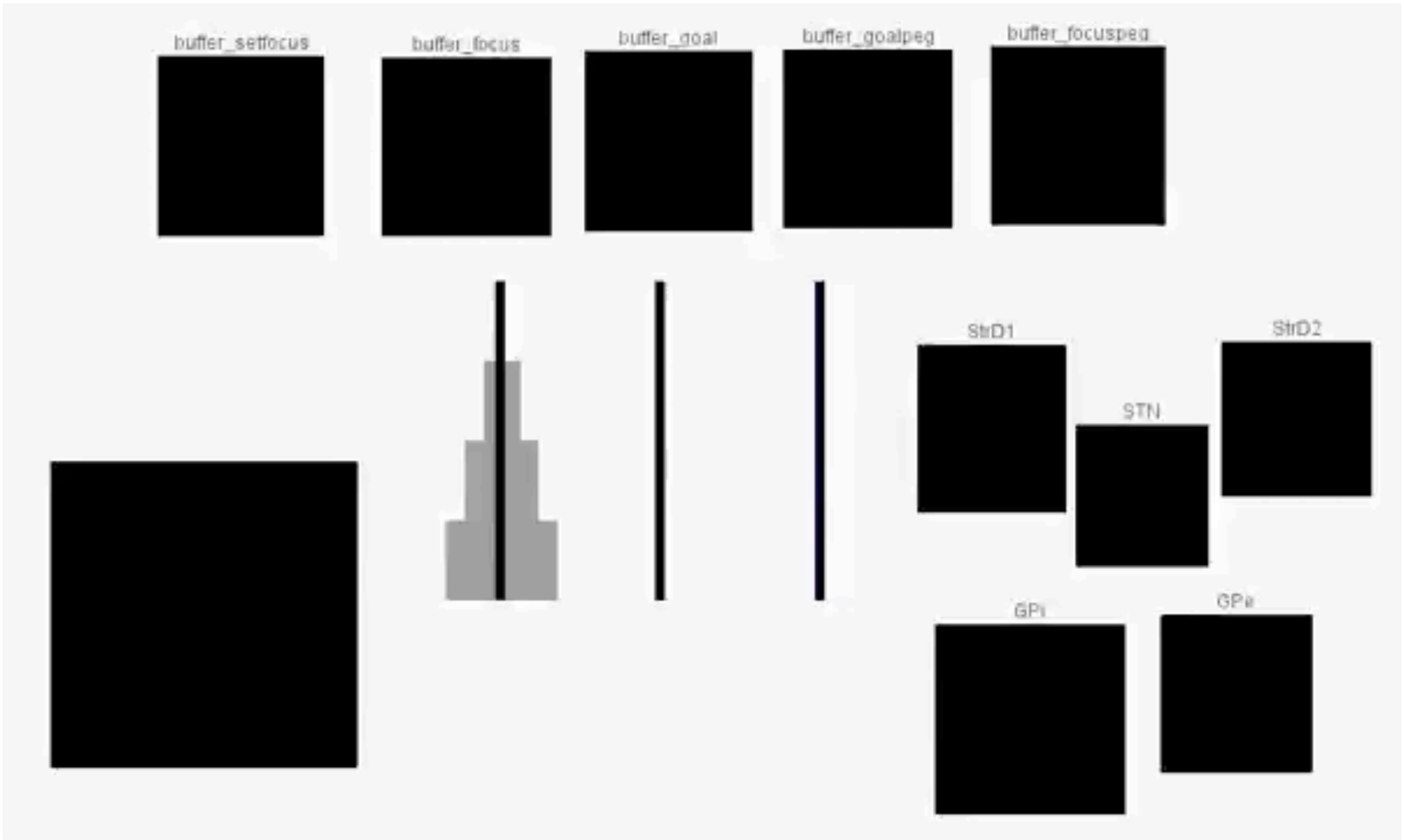
SpiNNaker Team



Brent Komer

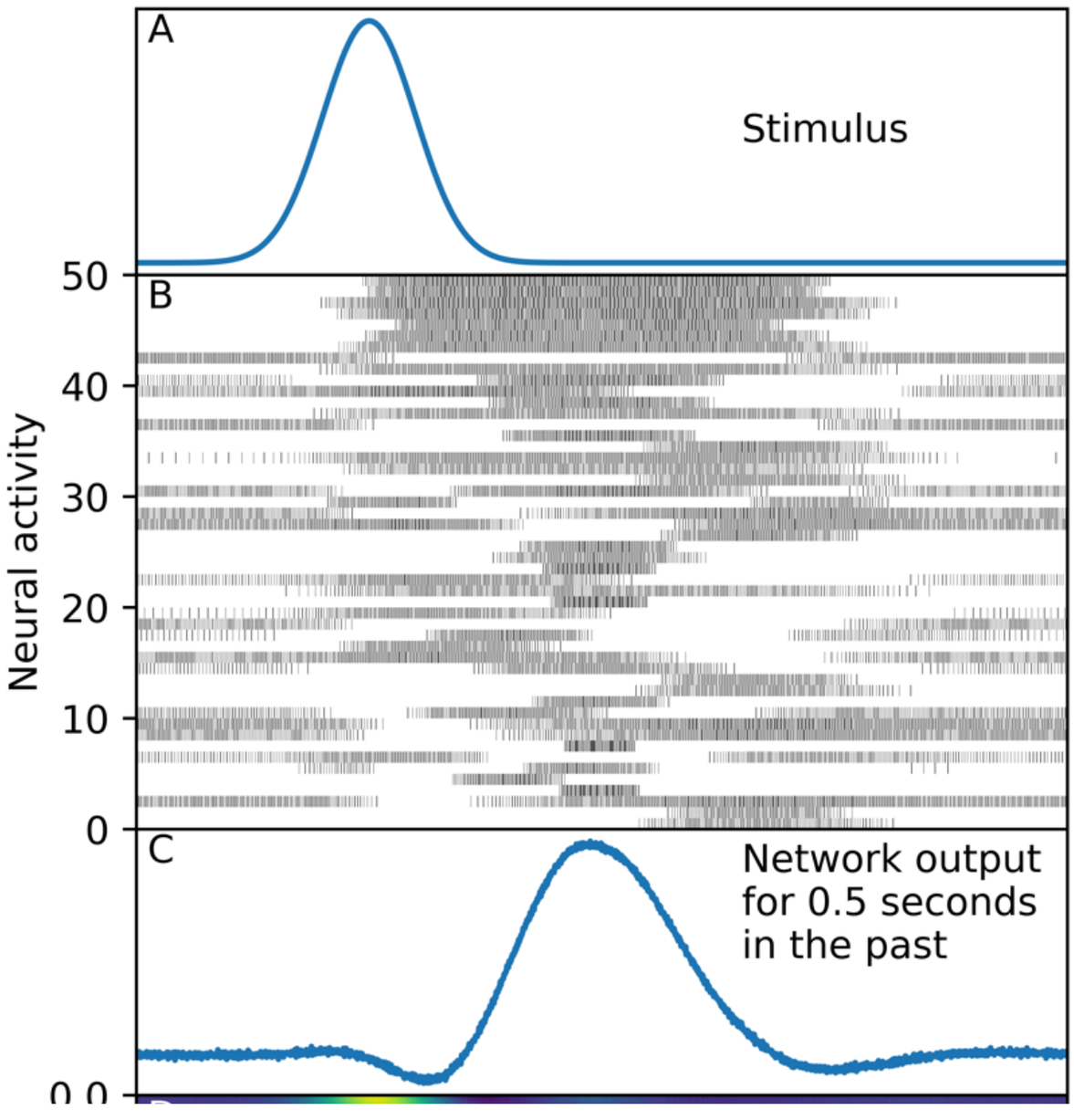
Integration

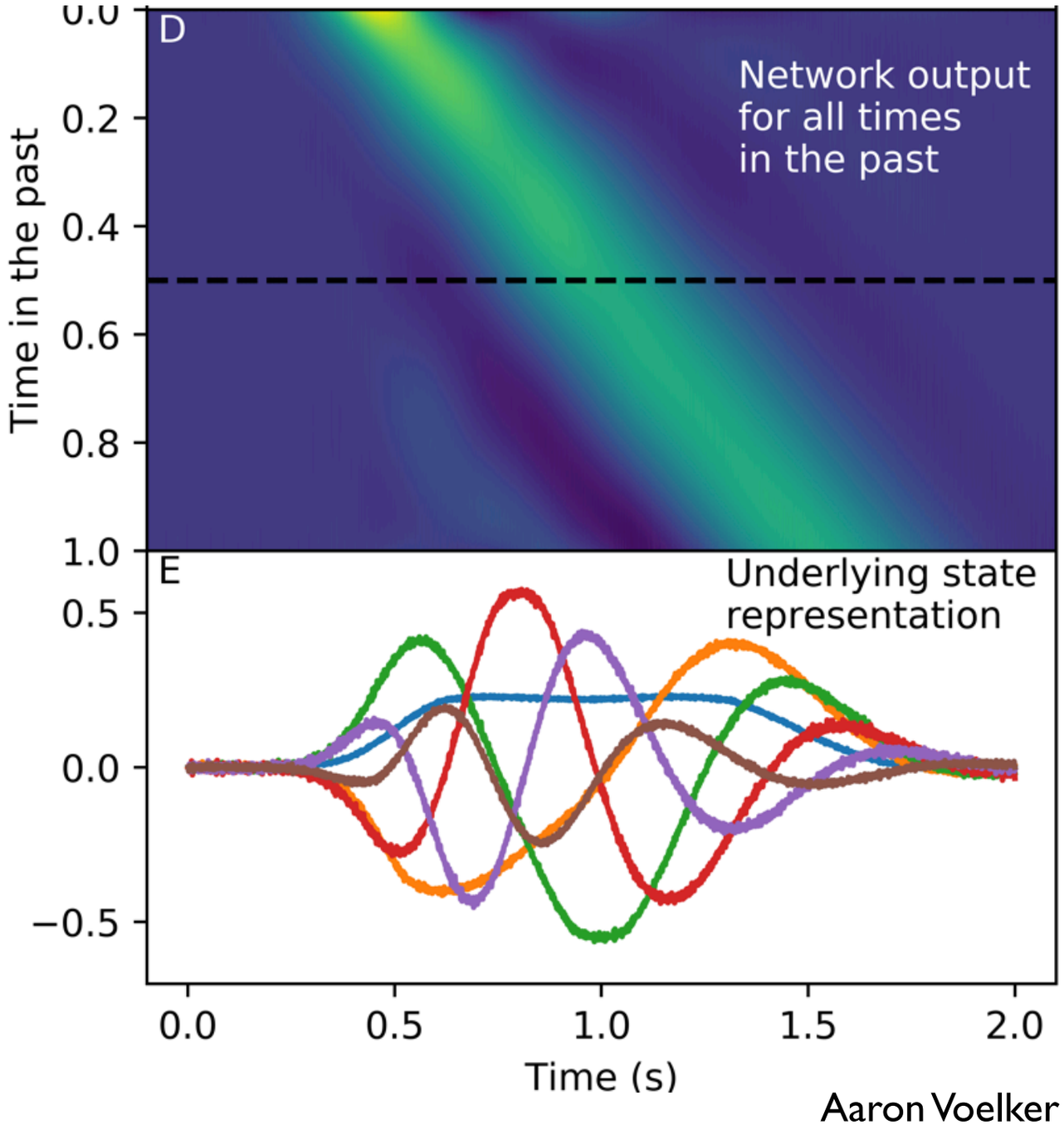
Pawel Jaworski/ABR



Terry Stewart

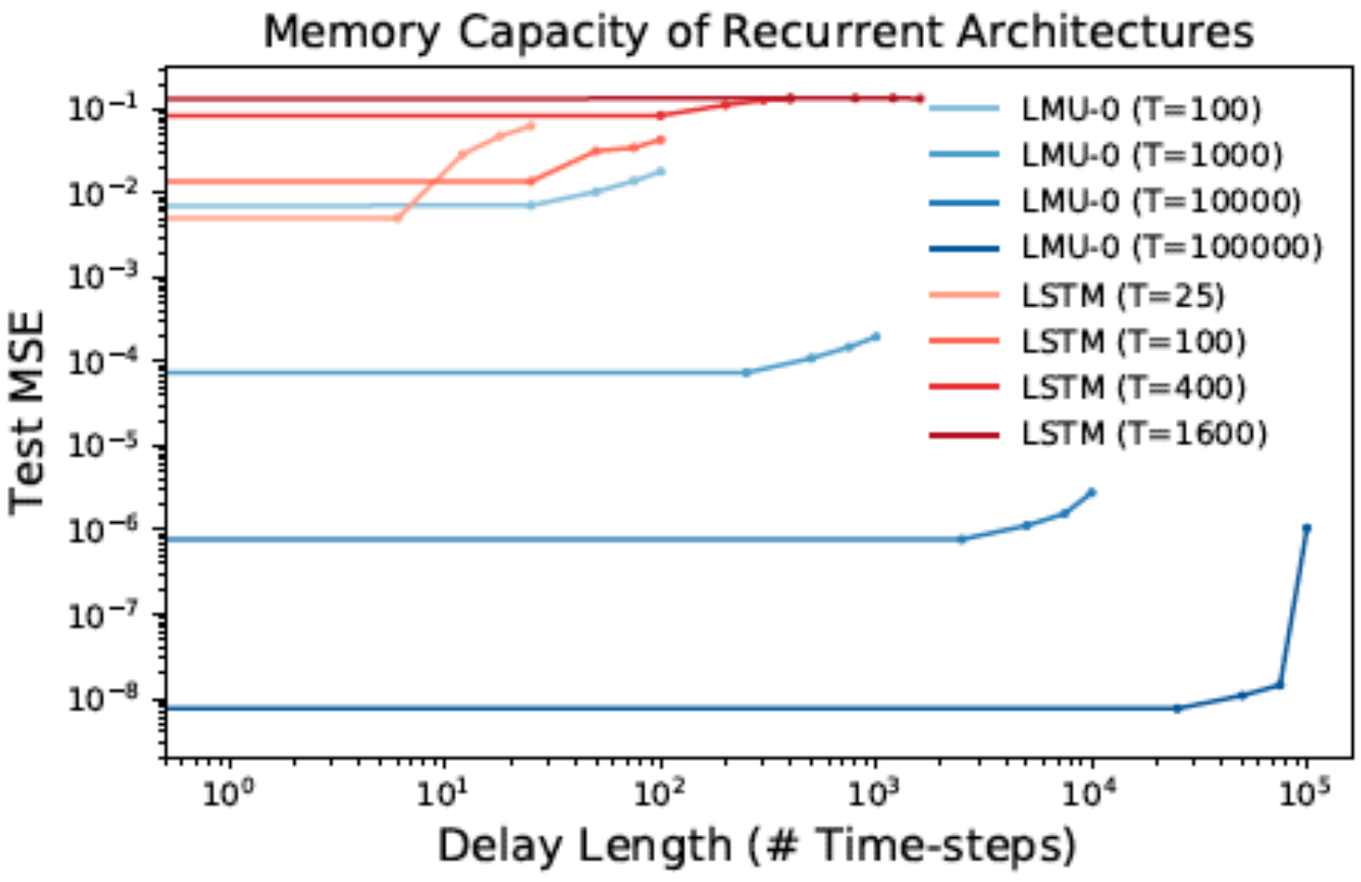
Delay Network





Aaron Voelker

LMU vs LSTM



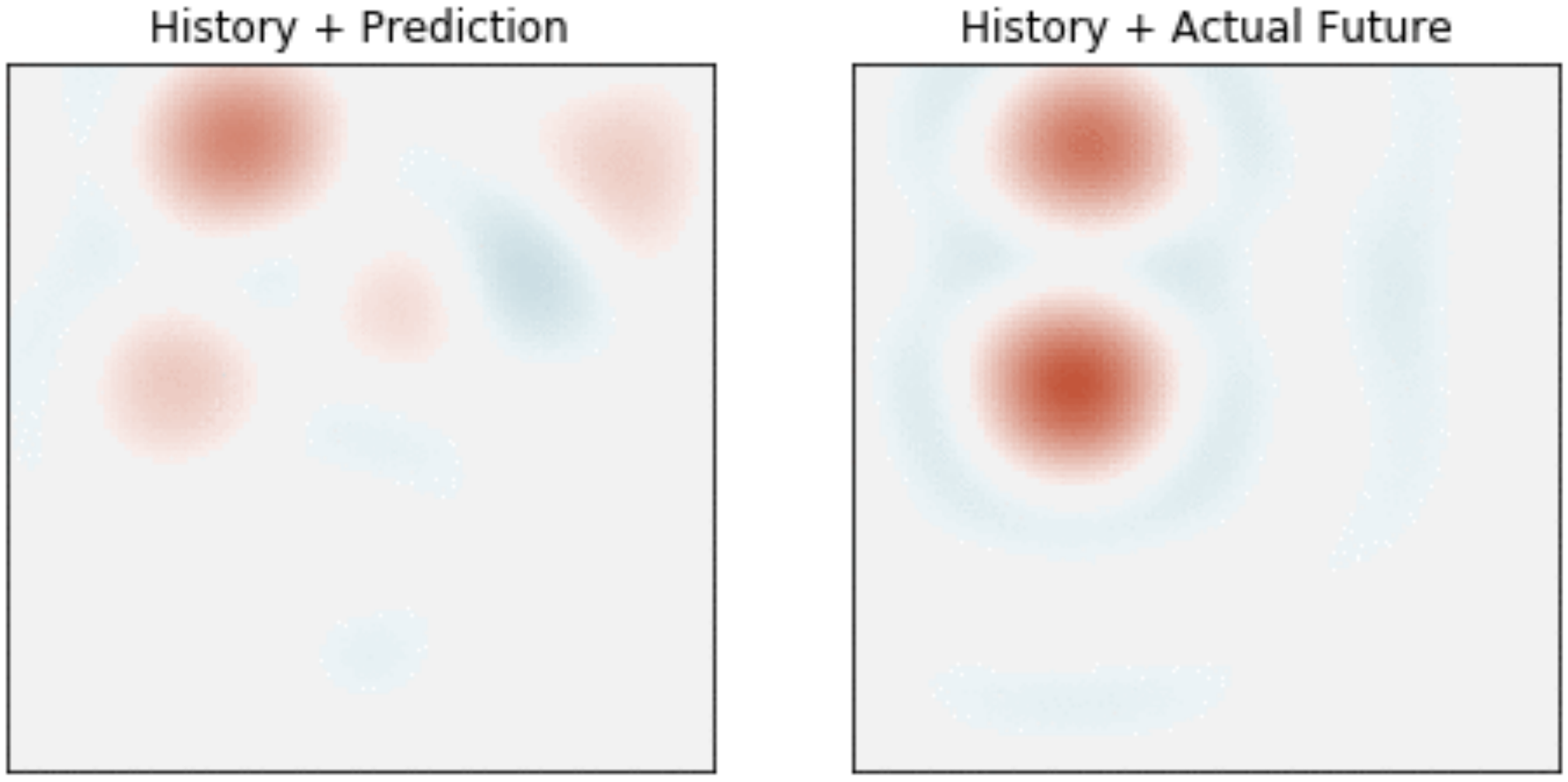
psMNIST

Model	Validation	Test
RNN-orth	88.70	89.26
RNN-id	85.98	86.13
LSTM	90.01	89.86
LSTM-chrono	88.10	88.43
GRU	92.16	92.39
JANET	92.50	91.94
SRU	92.79	92.49
GORU	86.90	87.00
NRU	95.46	95.38
LMU	**96.97**	**97.15**
FF-baseline	92.37	92.65

- 10^4 greater difficulty with 10^{-6} better error
- 500 vs 41k parameters

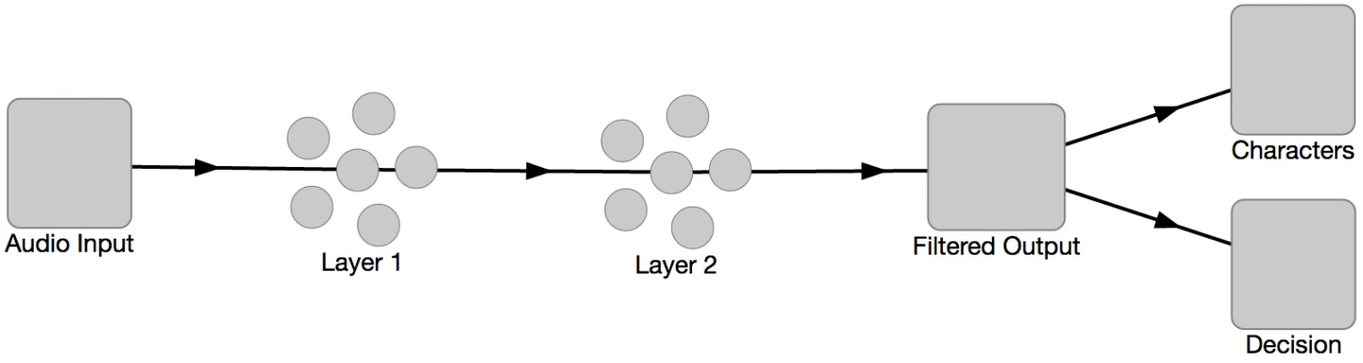
Aaron Voelker

LMUs and SSPs



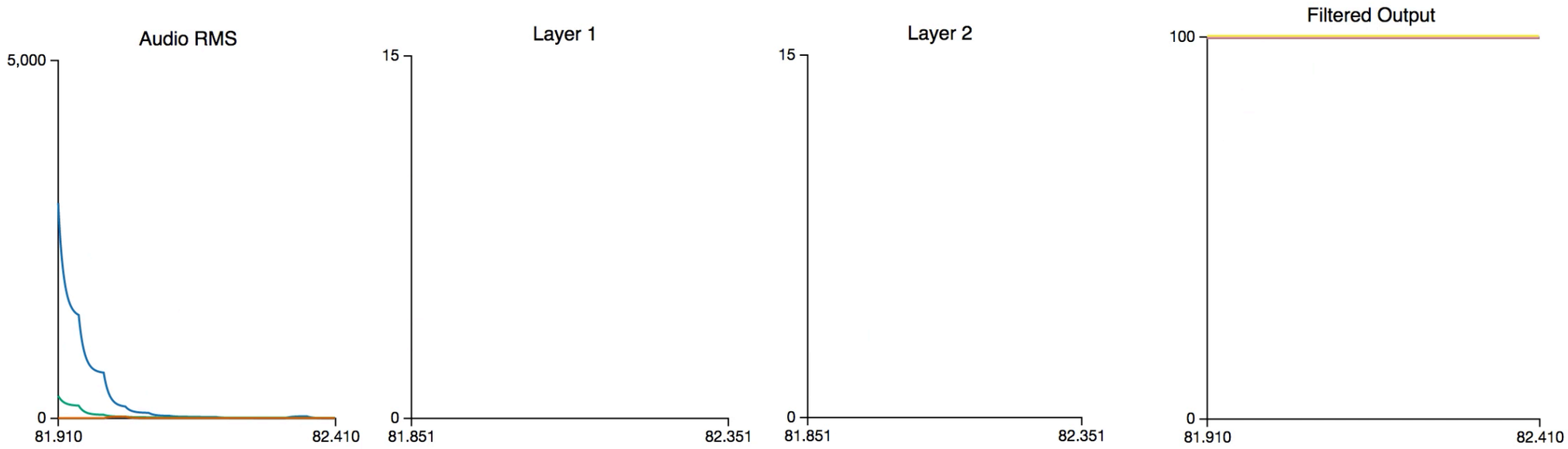
Aaron Voelker/ABR

reference_demo_model.py



Prediction: o

Decision: Not accepted!

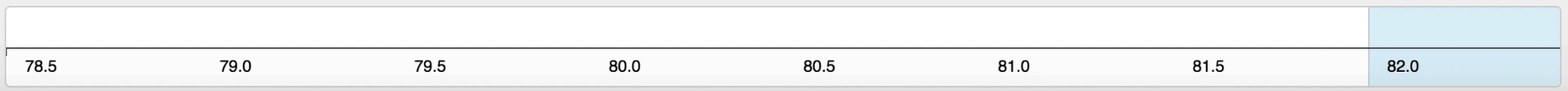


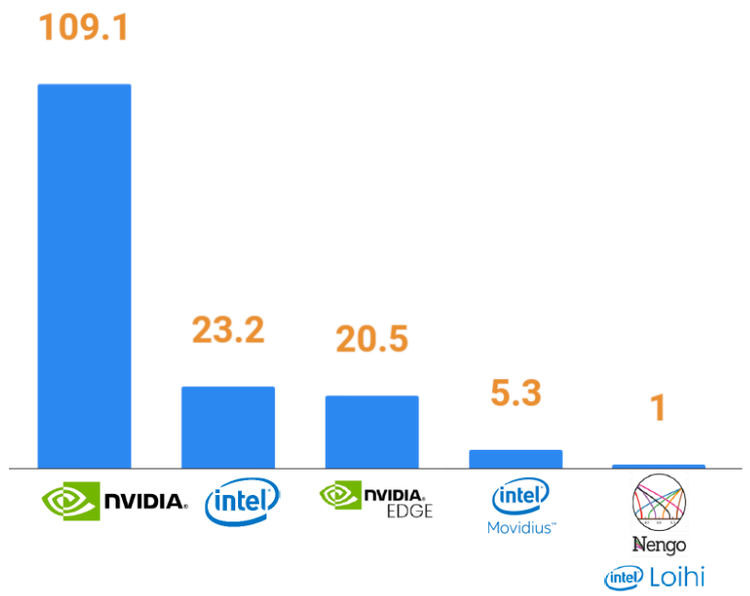

```
import pickle
import nengo
import string
import numpy as np

import audio_stream

# set this to use your chosen
if not audio_stream.started():
    audio_stream.start('Plantr

n_inputs = 390
n_outputs = 29
n_neurons = 256

audio_stream.min_thresh = 100
audio_stream.voc_thresh = 800

allowed_text = ['go', 'right',
id_to_char = np.array([x for x

def default_value(t, x):
    '''Node function to predic
    y = x + 100  # just for vi
    y[-1] += 0.5 # boost the b
    return y


with open('speech_commands_par
    params = pickle.load(pfile

# core speech model for keywor
with nengo.Network() as model:
    model.config[nengo.Connect
```

```
Build finished in 0:00:01.
```

Speed 0.63x

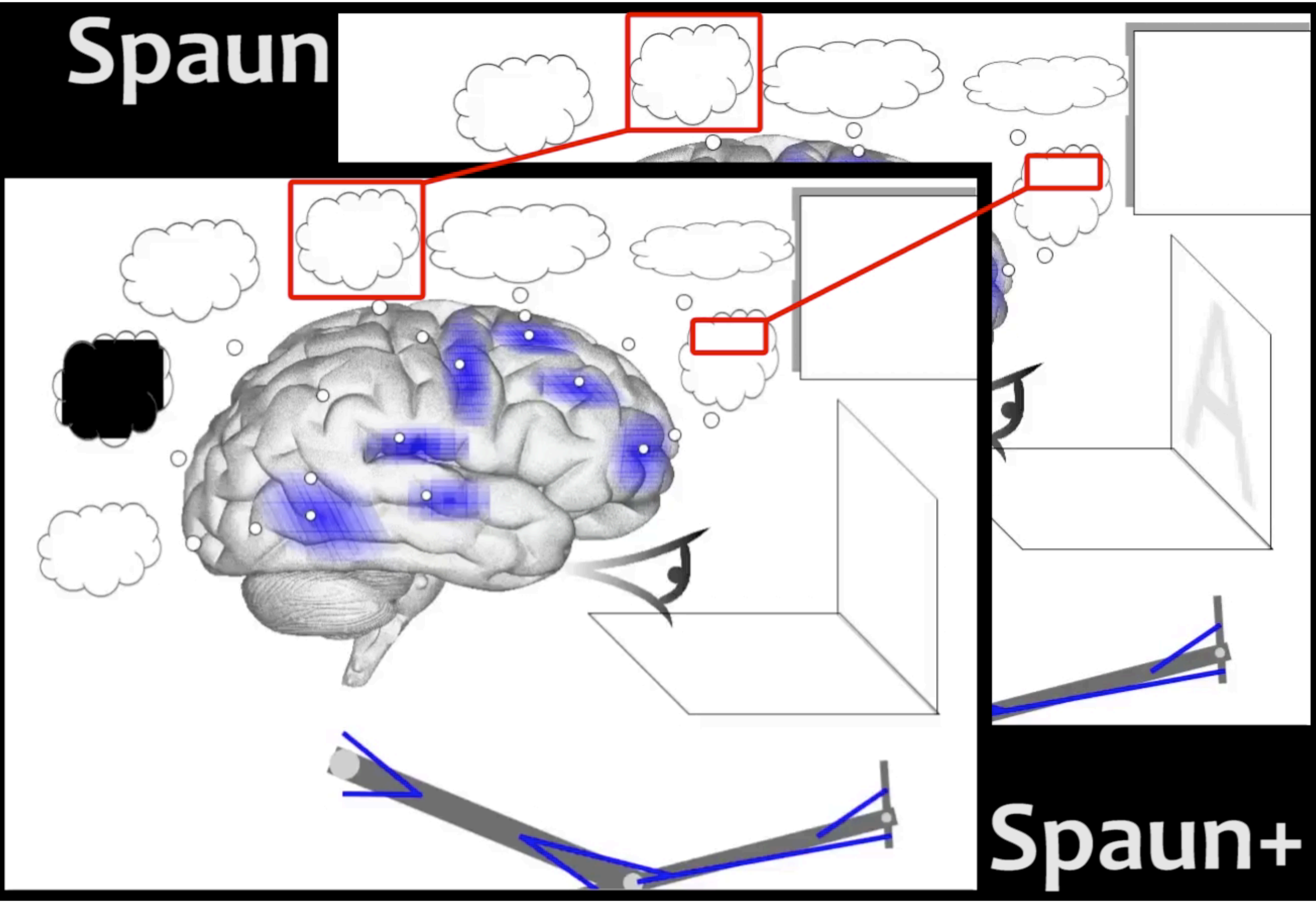
Time 82.410



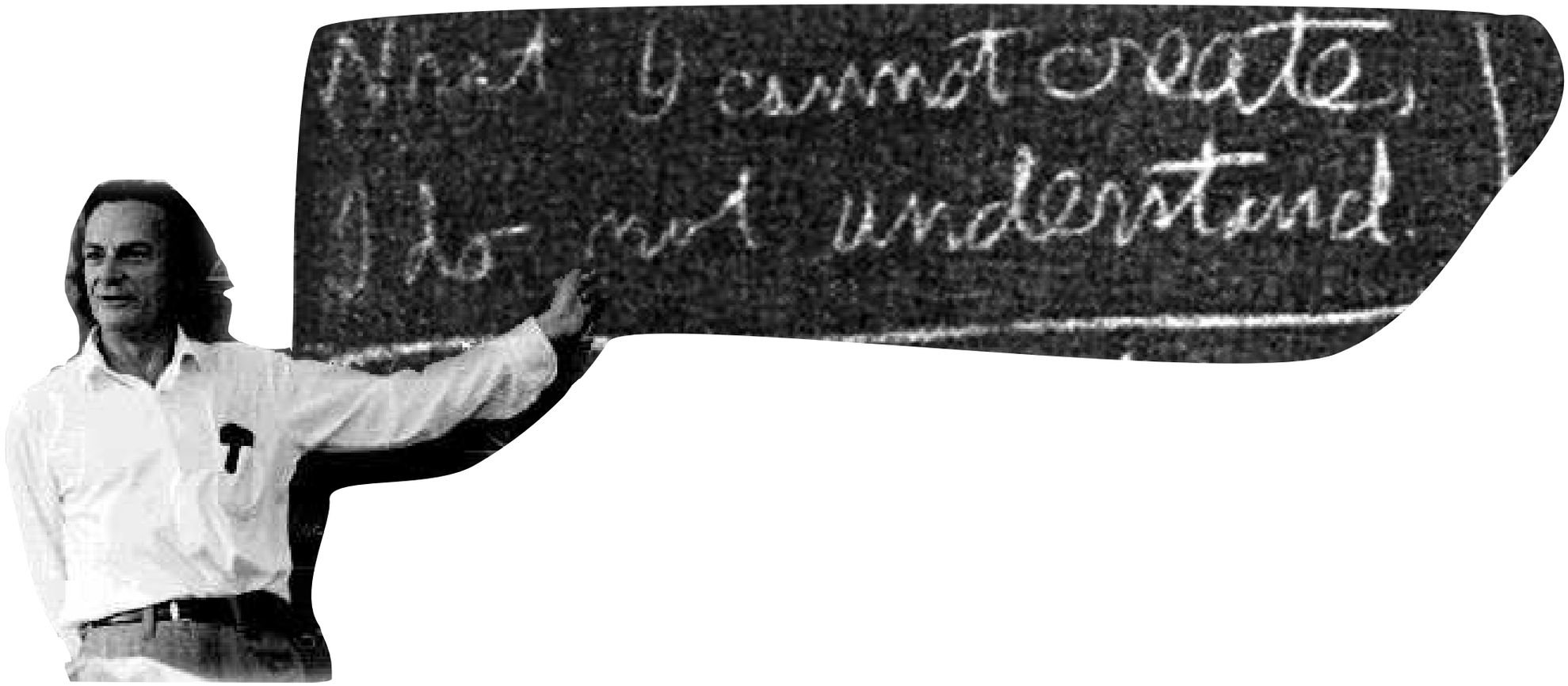


Peter Blouw/ABR

Spaun



Jan Gosmann and Xuan Choo



Nengo

If you want more…

- 2019: https://www.youtube.com/watch?v=5w0BzvNOypc
- 2018: https://www.youtube.com/watch?v=NwtYgBB2N6I
- 2017: https://www.youtube.com/watch?v=lLn9pTfzsAE
- 2016: https://www.youtube.com/watch?v=K-o-MJJY7ss