# Rogues Gallery – NeuroCar & OpenAiGym Teams (Fall 2022)

James Wood, Jinchu Li, Austin Chemelli, Anthony Wong, Reetesh Sudhakar, Keke Li

## Introduction & Goals

### Introduction

The Neuromorphic Team specializes in neuromorphic, also known as brain-inspired, computing applications.

### Semester Goals

- Develop an architecture for an autonomous navigation system, with communication between components.
- Develop and compare neuromorphic and non-neuromorphic approaches to problems.
- Introduce new members to neuromorphic tools and concepts.

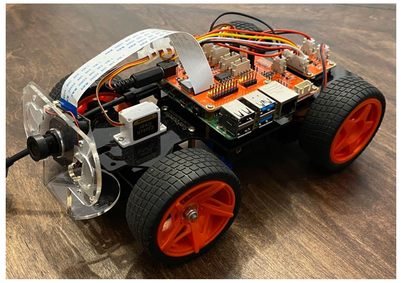
## Key Concepts & Results

### The Neuro-Car

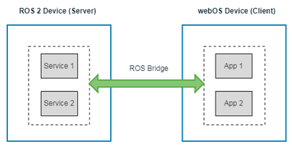
A small-scale, low-power autonomous vehicle that utilizes spiking neural networks to power onboard decision-making.

**Software / Hardware**

- Raspberry Pi 4
- Raspian / Ubuntu (Docker)
- Python servo and throttle bindings from SunFounder.
- Raspberry Pi Camera Module



ROS




### ROS

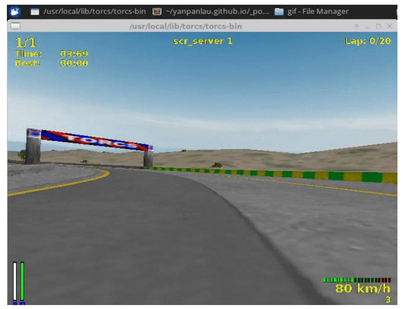
(Robotic Operating System)

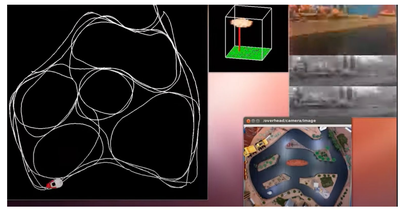
- Middleware to interact between various sensors, devices, and control code.
- OpenRatSLAM is implemented with ROS 1 (melodic) but modern ROS 2 is not compatible.
- The rosbridge can be used to interopt between ROS 1 and 2.

### DDPG

(Deep Deterministic Policy Gradient)

- Takes in SLAM data and makes throttle / steering decisions.
- Training method involves the use of two pairs of networks, in order to stabilize learning.



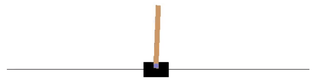
### RatSLAM

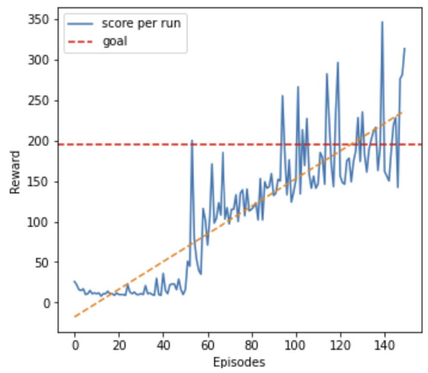
- Simultaneous Localization and Mapping (SLAM) algorithm inspired by rat brains; a neuromorphic method for mapping and tracking an environment.

### OpenAIGym

Exploring scenarios/problems in which neuromorphic algorithms are preferable solutions over traditional AI/ML techniques

- Selected CartPole problem
- Tested agents using traditional Q learning and DQL
- Developing neuromorphic implementation using Nengo






### Nengo

- Python package that supports the creation of neuromorphic algorithms
- Allows for the construction of brain-mimicking neural networks

### OpenAIGym

- Open source Python API containing a collection of environments to run learning agents on


## Challenges & Next Steps

### Challenges

- Constructing a Nengo network for the Cartpole scenario.
- Establishing stable connection to Neuro-Car (for wireless controller, clients).
- Develop a simulation environment for rapid NeuroCar development.

### Next Steps

- Train a DDPG model to work on the NeuroCar based on OpenRatSLAM output.
- Generate comparison data between neuromorphic and non-neuromorphic Cartpole solutions.
- Cleaning up the Docker environment for better containerized development..