Rogues Gallery – RatSLAM Team (Spring 2023)

Hiren Kumawat, Colten Webb, Suzan Manasreh

Introduction & Goals

Introduction

The Neuromorphic Team specializes in neuromorphic, also known as brain-inspired, computing applications. This semester, most of our efforts have been centered on the RatSLAM algorithm and our proposed variation of it, as well as investigating existing neuromorphic frameworks and hardware.

Semester Goals

- Introduce new members to critical neuromorphic concepts, architectures, and literature, with an emphasis on RatSLAM functionality.
- Use Lava packages to experiment with SNN implementation, primarily as it pertains to RatSLAM.

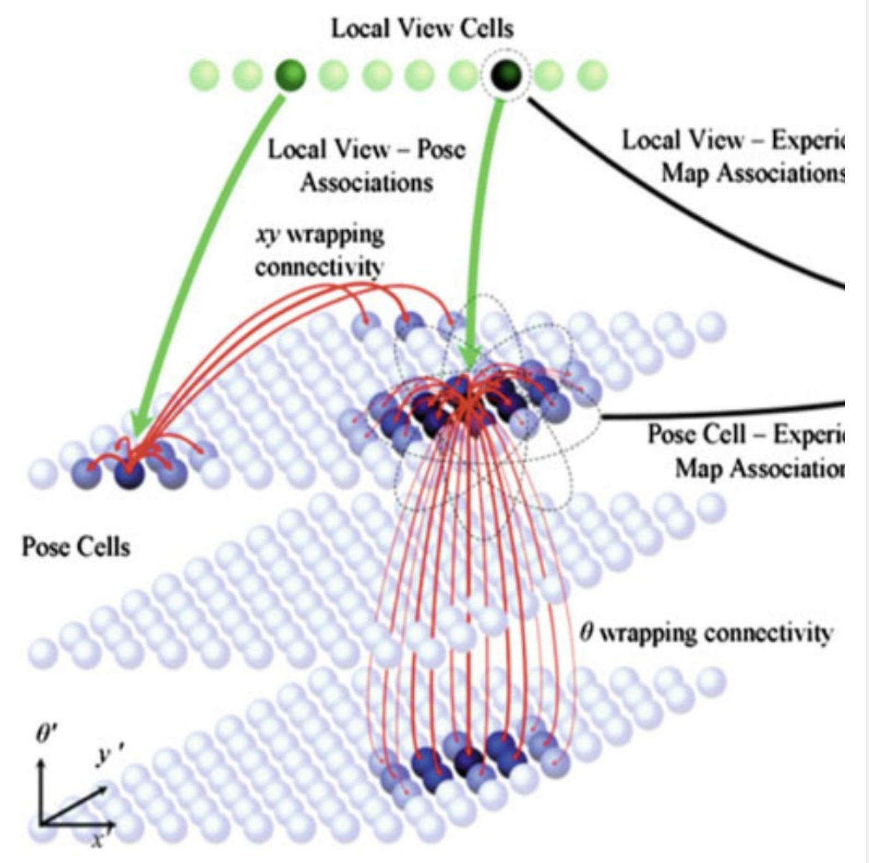
Key Concepts & Results

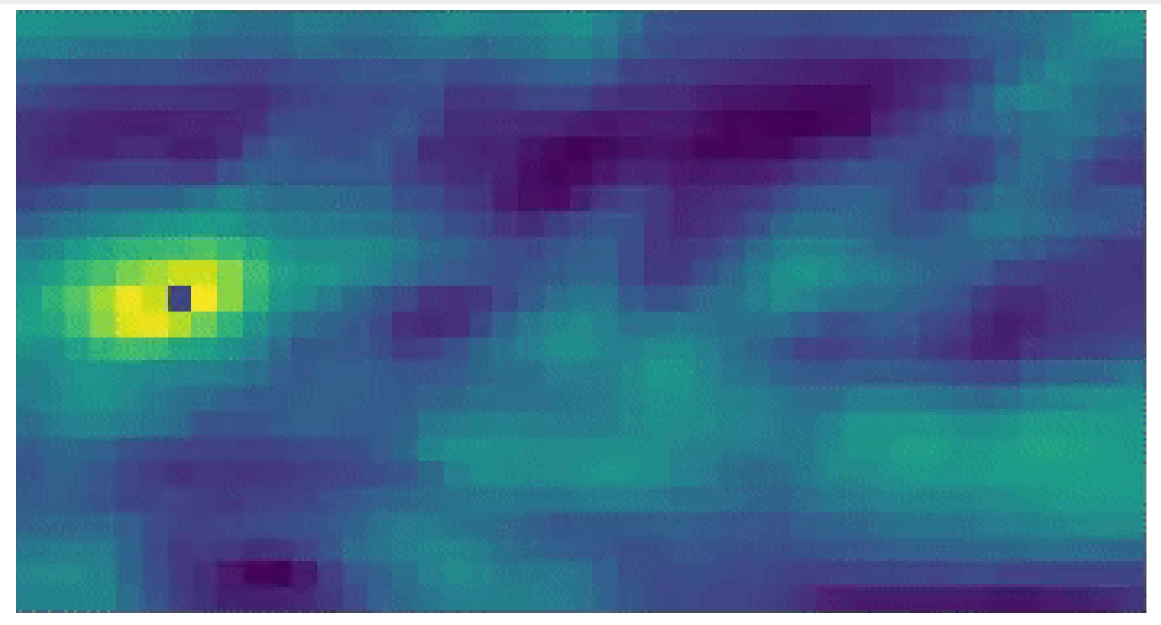
Overview

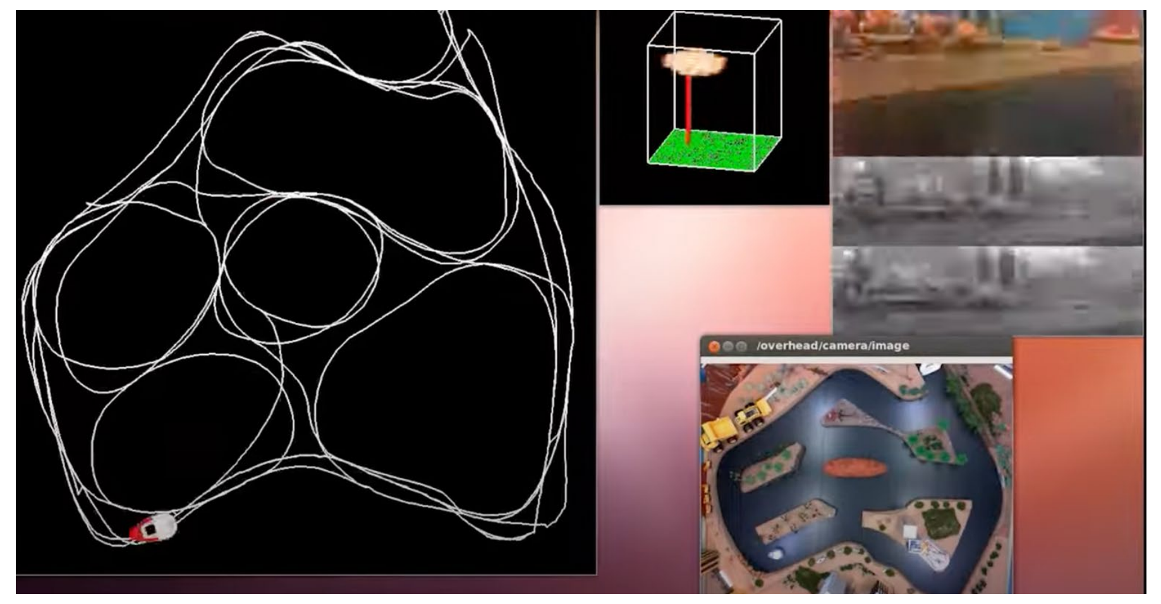
- To meet the resource constraints of the NeuroCar, we are seeking to develop an autonomous navigation system based on RatSLAM and implemented with Lava, Lava-Dnf, and Lava-DL.
- The goal is to have an end-to-end system for obtaining vision and odometric input, processing it to create a map of the environment and track the agent's orientation and location within it, and to produce a stream of acceleration and steering values to efficiently navigate between two map landmarks.

Pose Cells and SNNs

- RatSLAM relies on a pose cell network to determine the location of a landmark. This network is an attractor network, which can be modeled by an SNN. Asynchronous computation of the input spikes by SNNs reduces the algorithm's power consumption.







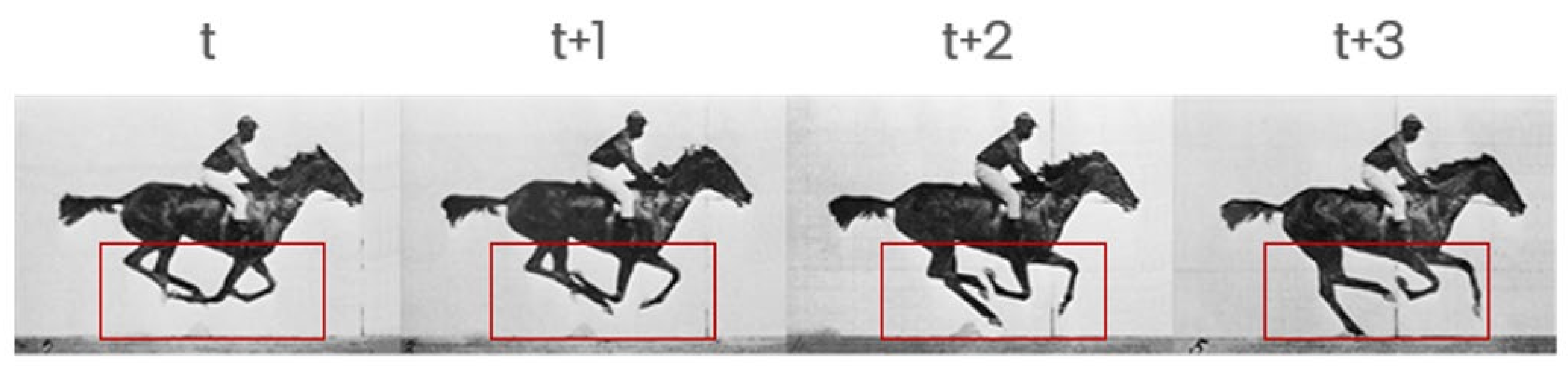
rg-slam

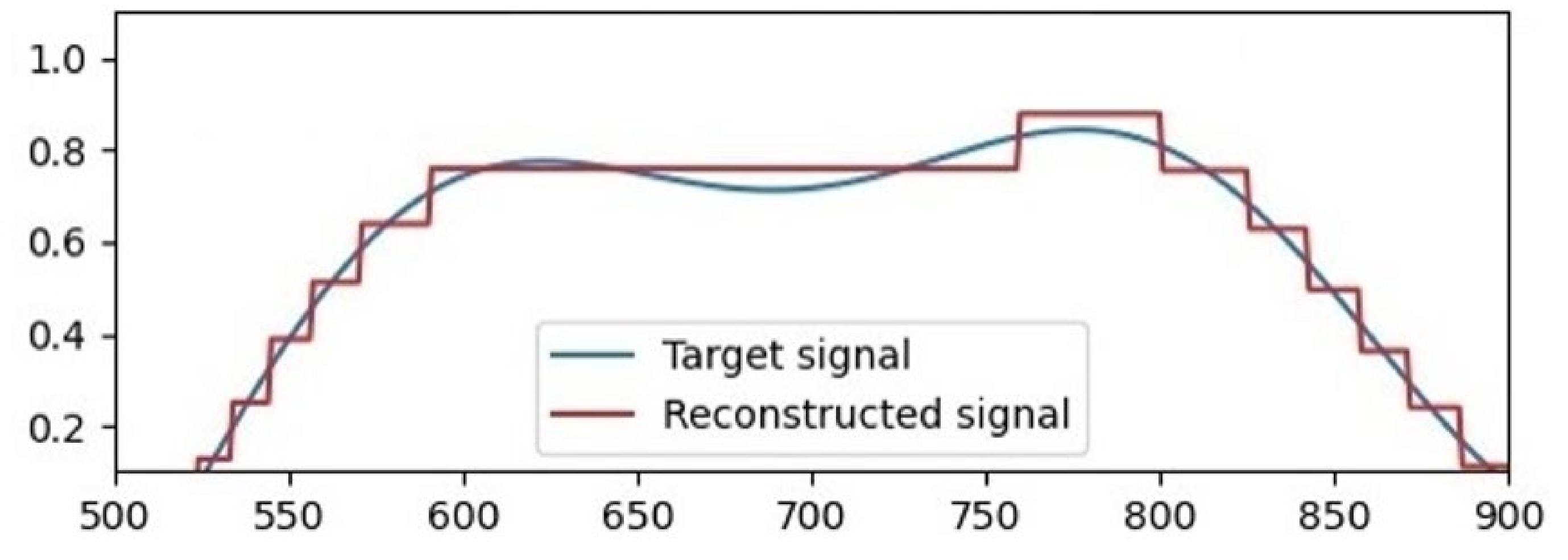
- The repo containing our implementation of RatSLAM. Starting from a CPU-based implementation, we're working on incrementally replacing components (like the pose cell and view cell network) with algorithms that can be run on Loihi with the lava framework.

Lava, Lava-dnf, Lava-DL

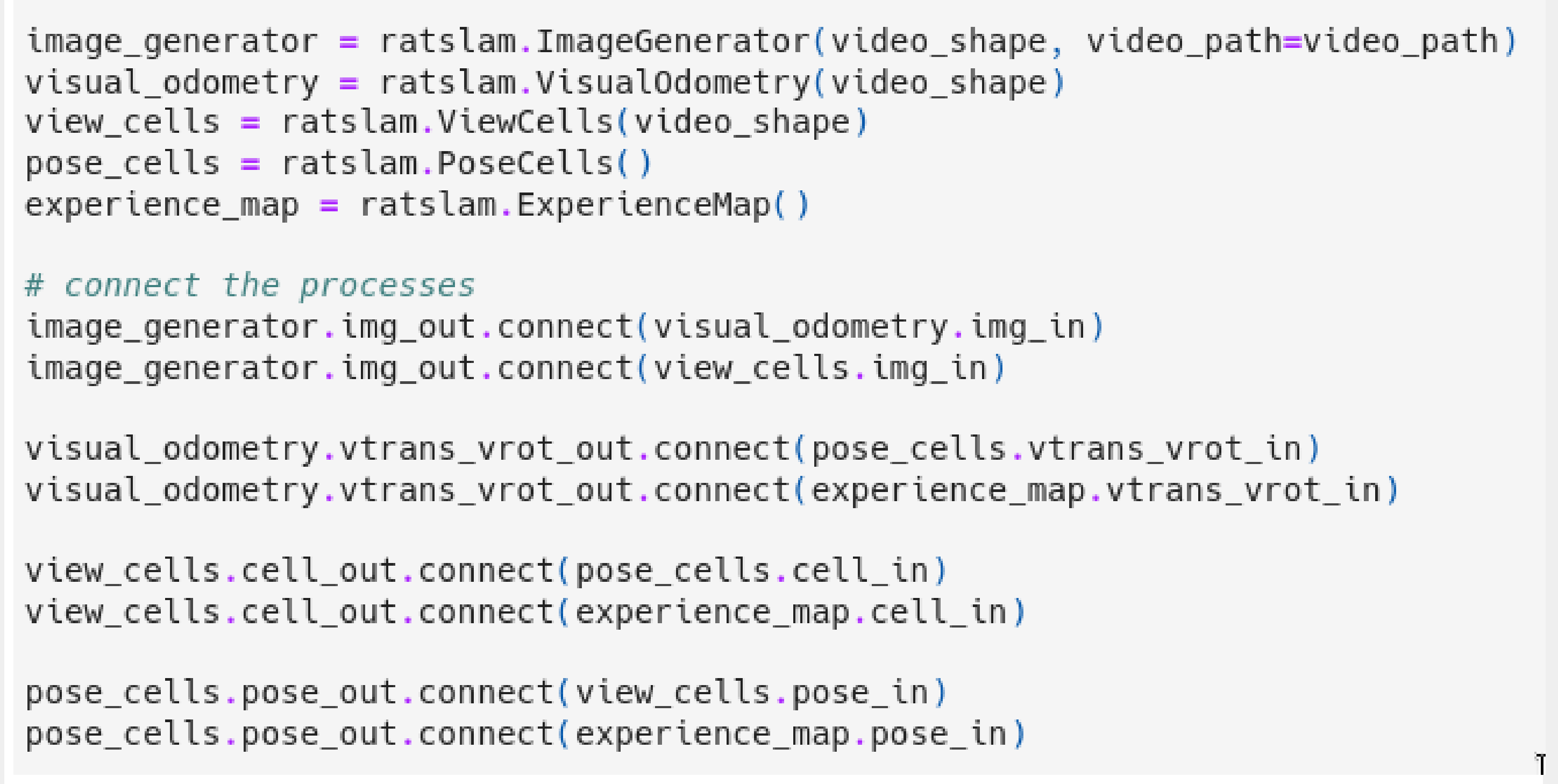
- Lava is a Python framework for simulating SNNs and building applications that have interconnected CPU and Loihi (a neuromorphic chip) components.
- Lava-DNF lets us define a dynamic neural field and run it on Loihi as an SNN, as an improvement to pose cells.
- Lava-DL lets us train an SNN using ANN techniques like gradient descent. This allows us to generate better embeddings to represent view cells, for example.

Example: Training SNN with Lava-DL to predict odometry





Challenges & Next Steps



```
image_generator = ratslam.ImageGenerator(video_shape, video_path=video_path)
visual_odometry = ratslam.VisualOdometry(video_shape)
view_cells = ratslam.ViewCells(video_shape)
pose_cells = ratslam.PoseCells()
experience_map = ratslam.ExperienceMap()

# connect the processes
image_generator.img_out.connect(visual_odometry.img_in)
image_generator.img_out.connect(view_cells.img_in)

visual_odometry.vtrans_vrot_out.connect(pose_cells.vtrans_vrot_in)
visual_odometry.vtrans_vrot_out.connect(experience_map.vtrans_vrot_in)

view_cells.cell_out.connect(pose_cells.cell_in)
view_cells.cell_out.connect(experience_map.cell_in)

pose_cells.pose_out.connect(view_cells.pose_in)
pose_cells.pose_out.connect(experience_map.pose_in)
```

Challenges

- The dataset provided with the RatSLAM paper has dozens of parameters tuned to its algorithm, we need to change them so it performs well with the changes we have made to the components.

Next Steps

- Generate comparison data between neuromorphic and non-neuromorphic SLAM algorithms, particularly on the basis of latency, efficiency, and accuracy– benchmarking.
- Explore integrating Lava-DL for view cells, and Lava-DNF for pose cells