User Guide:

This folder contains the PyNN code simulating a two layer topology on SpiNNaker platform.

DVS data recording setting & processing:

Approximately 30cm away from laptop screen. Keep the ball trajectory at the centre of the camera field of view.

1. Download jAER from initLAB. (Current version might fail to recognize the issue number of the DVS128 camera. Need to liaising with IniLab to get current version)
2. Record from DVS camera. After recording, open the AEDAT file. Click “More” and find the right range for event number by dragging the cursor on timeline. (Spike event should cover the ball trajectory move from one side of the FOV to the opposite)
3. Import the range to the all\_four\_directions\_rev05.py in folder raw AEDAT file processing
4. Jaer library code \_\_init\_\_.py in paer folder has been modified from python library index paer 0.1.3
5. Normalise the ball trajectory to 88 by input\_normalizer.py (name and number of file need to be changed for this python code)

SpiNNaker Simulation:

Training:

PyNN code for training two layer topology is in the two layers topology Training -> Training folder.

* Wrapper.py:
  + Import and configure relevant parameters of the neural network topology
  + Use I\_based\_model.py

|  |  |
| --- | --- |
| Neural Network Parameter Name | Variable Name |
| Number of input recording to train one direction | num\_training |
| STDP parameter | stdp\_param |
| Leaky integrate & fire neuron model | cell\_params\_lif |
| Synapse delay | delay |
| Trained weight matrix | weights\_import |
| Ball Trajectory Period | animation\_time |
| Simulation Mode | excitatory\_mode, inhibitory\_mode, training\_mode |
| Input layer neuron population | pre\_pop\_size |
| Output layer neuron population | post\_pop\_size |
| Number of noise spikes introduced | num\_noise |
| Inhibitory synapse weights | I\_syn\_weight |
| Simulation Timing Setup | setup\_cond |

* I\_based\_model.py:
  + Code to synthesis neural network
* Hexfunc\_test.py:
  + Display trained synapse weight in a 2D coloured bar figure
* Post-training.py:
  + PyNN code to run after training simulation test