Neuromorphic Swarms: Taking Milling from Sim-to-Real

Spiking Neural Networks for Emergent Swarms

Kevin Zhu, Shay Snyder, Ricardo Vega,
Maryam Parsa, Cameron Nowzari

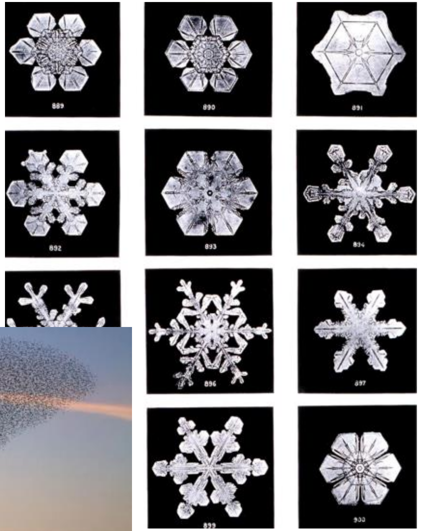
GEORGE MASON UNIVERSITY

Neuro-Inspired Computational Elements (NICE) 2025

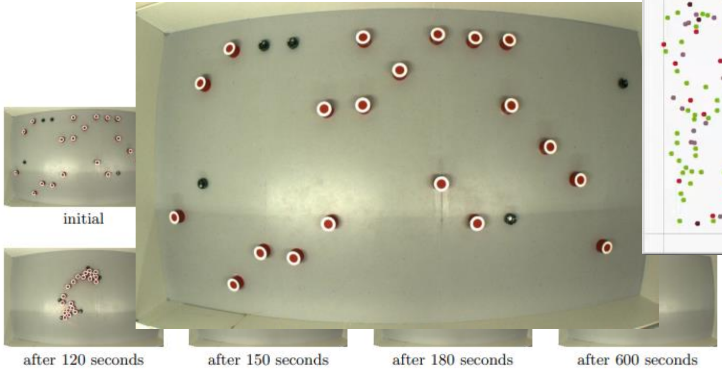

00:00:00:00

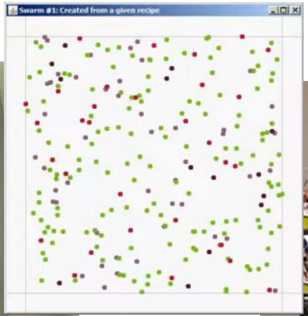
Emergent behaviors are typically “discovered” in two ways:

- Naturally (via evolution)
- Using intuition and experimentation

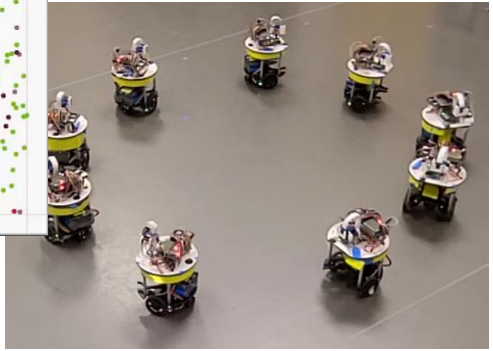


M. Gauci, 2014

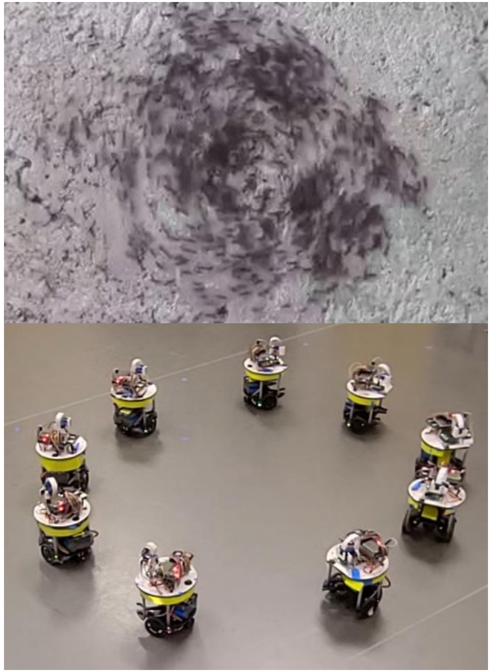


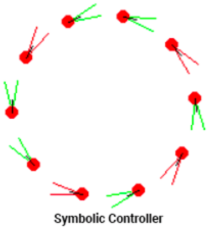


H. Sayama, 2011



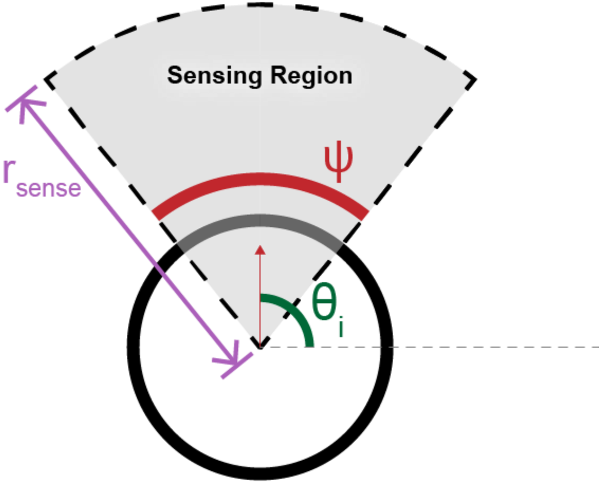
in 2014, Gauci devised a super-simple algorithm that results in milling





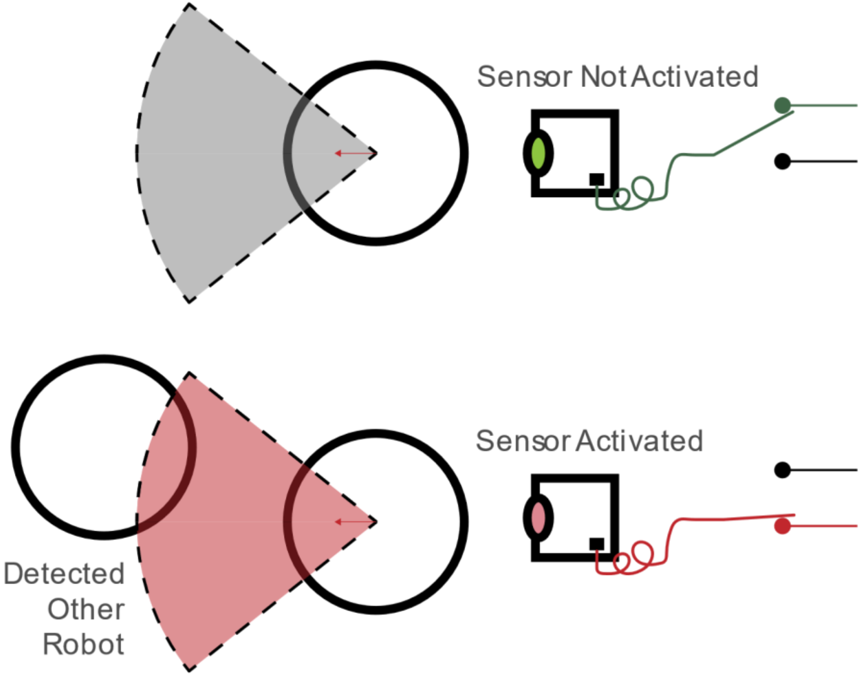
It requires:

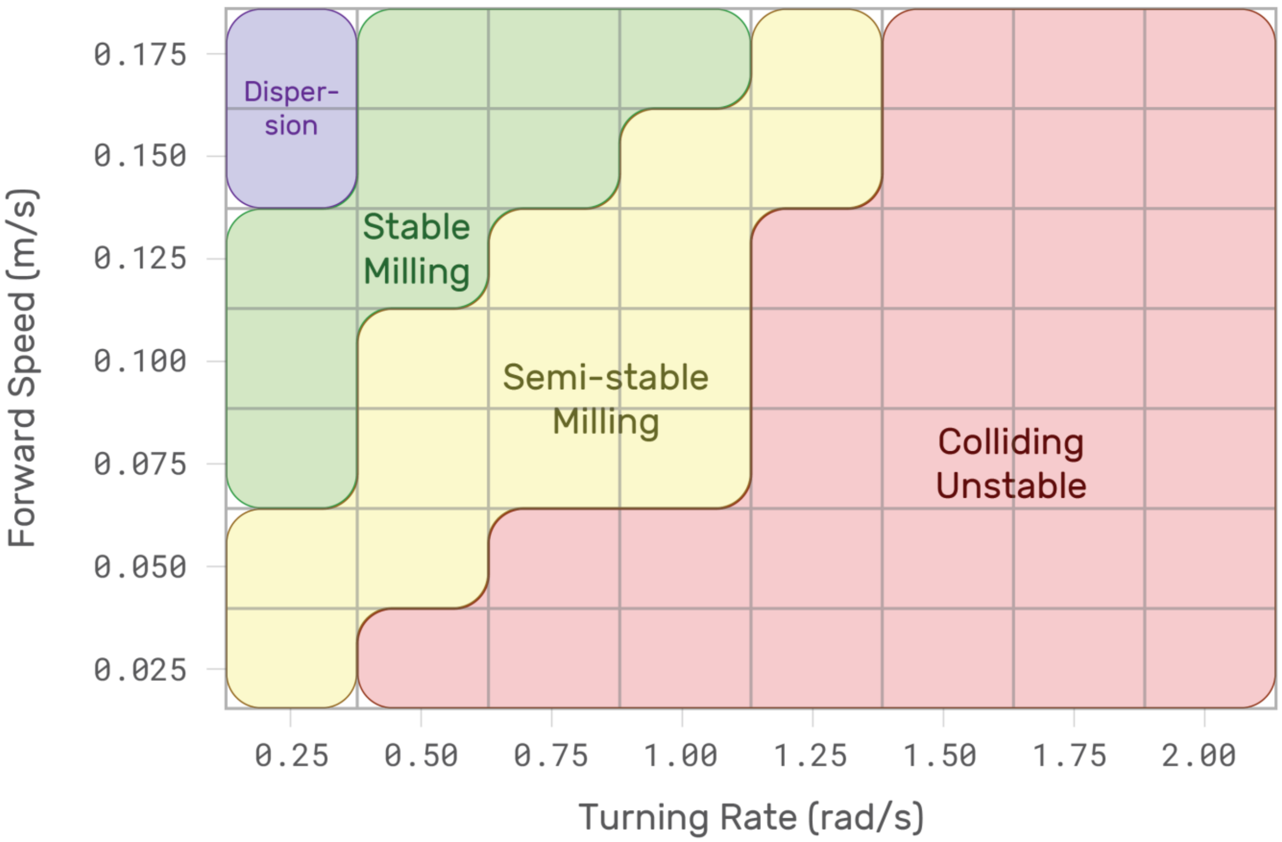
a binary sensor
and two lines of code

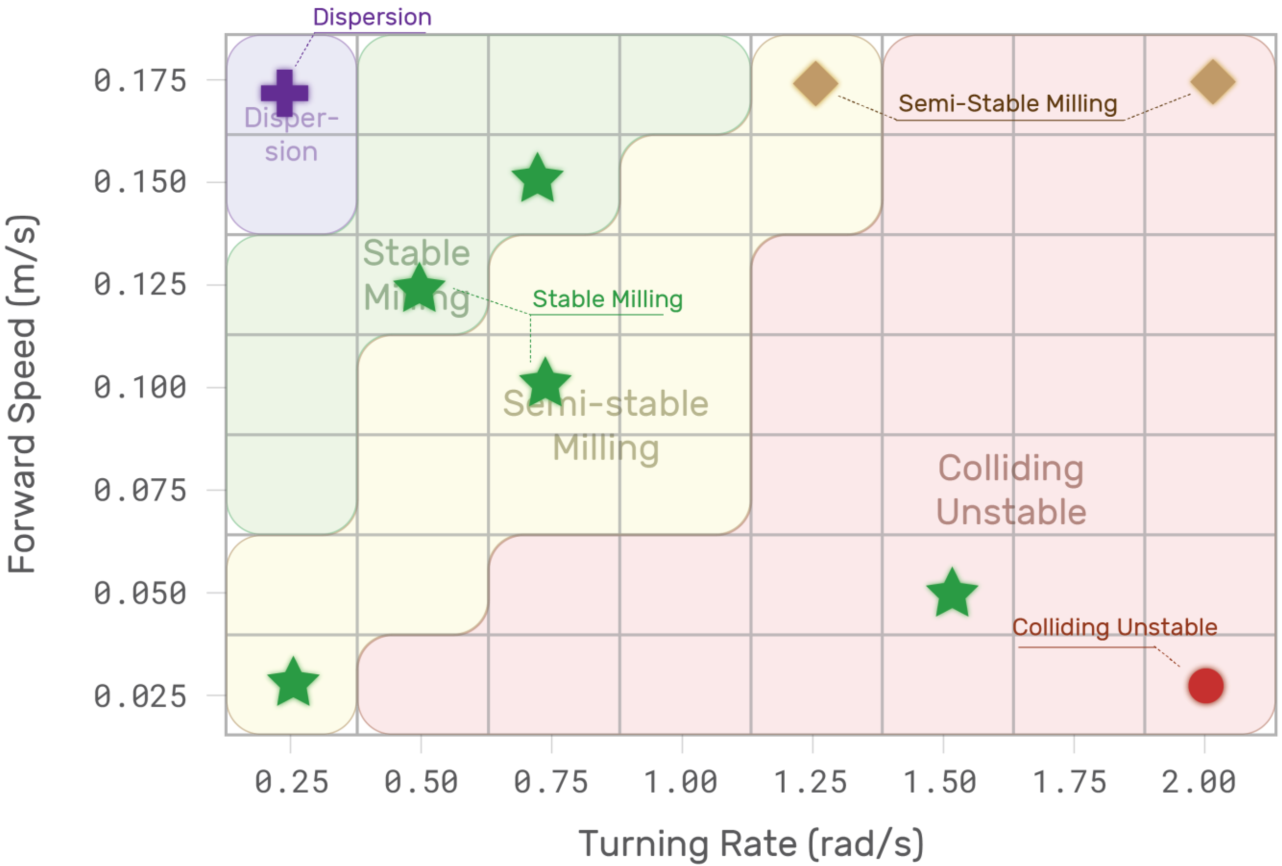


An infrared sensor detects if there's another robot within its field of view or not.

```
if (see):
  go forward while turning left
else:
  go forward while turning right
```

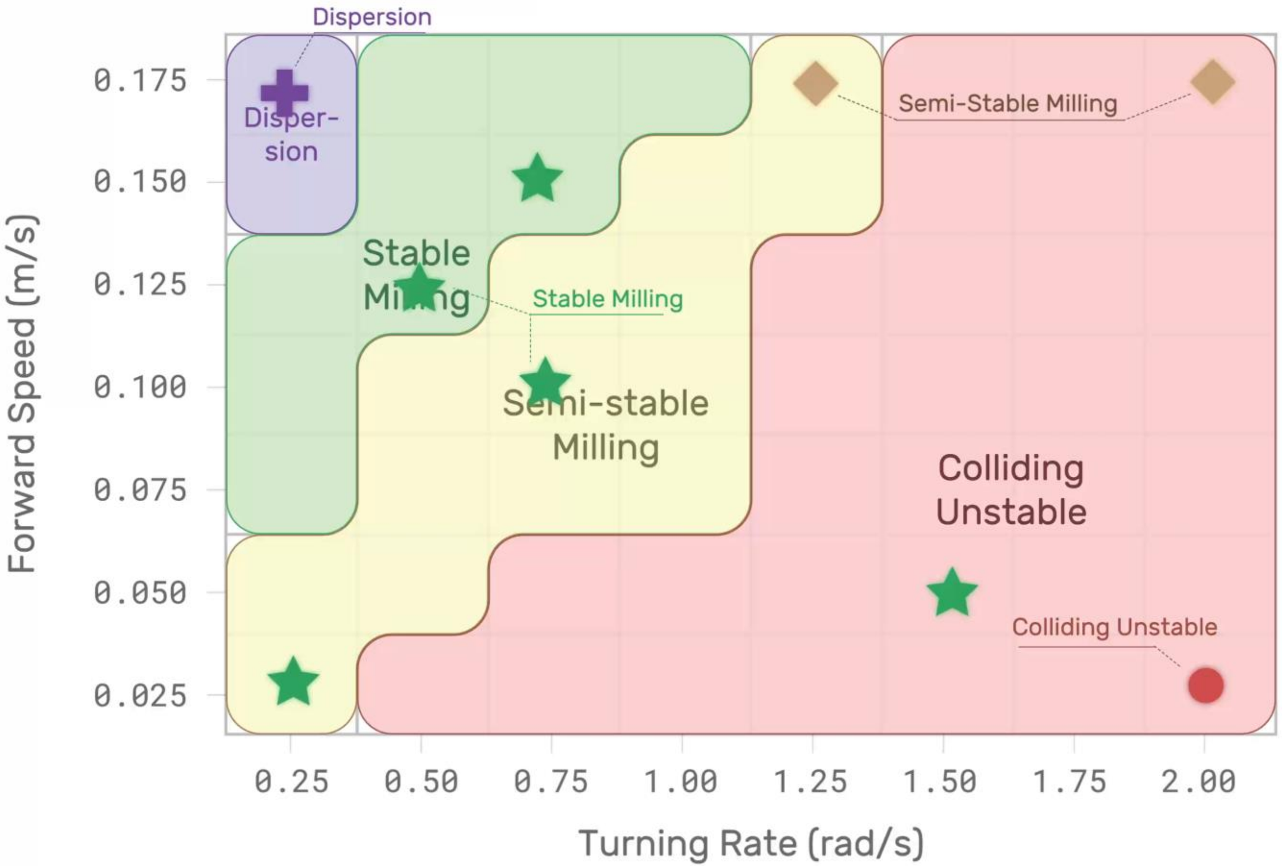


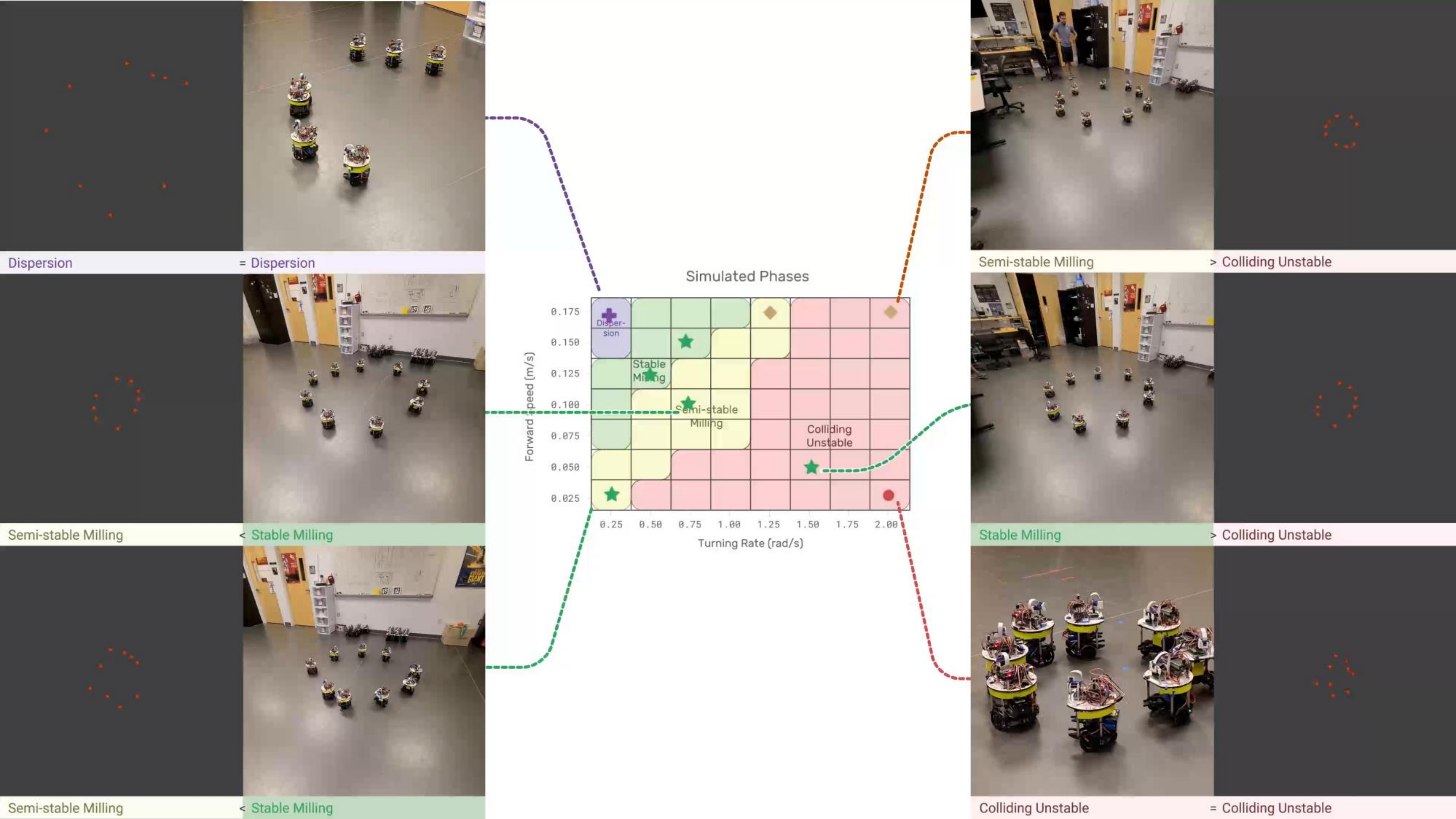

Disper-
sion
Stable
Milling
Semi-stable
Milling
Colliding
Unstable
0.175
0.150
0.125
0.100
0.075
0.050
0.025
Forward Speed (m/s)
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Turning Rate (rad/s)



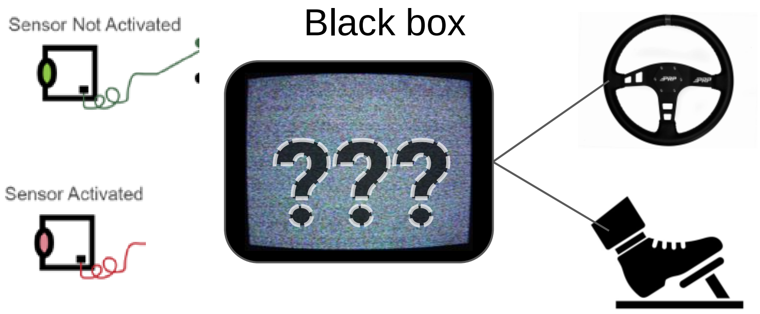
Dispersion
Disper-
sion
Stable
Milling
Stable Milling
Semi-stable
Milling
Semi-Stable Milling
Colliding
Unstable
Colliding Unstable
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Forward Speed (m/s)
Turning Rate (rad/s)

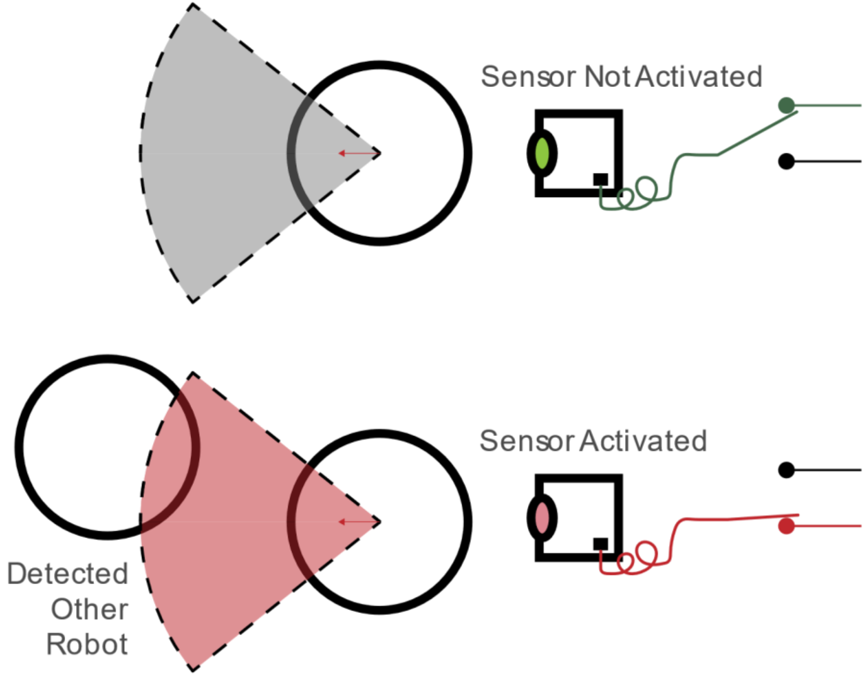
Simulated Phases

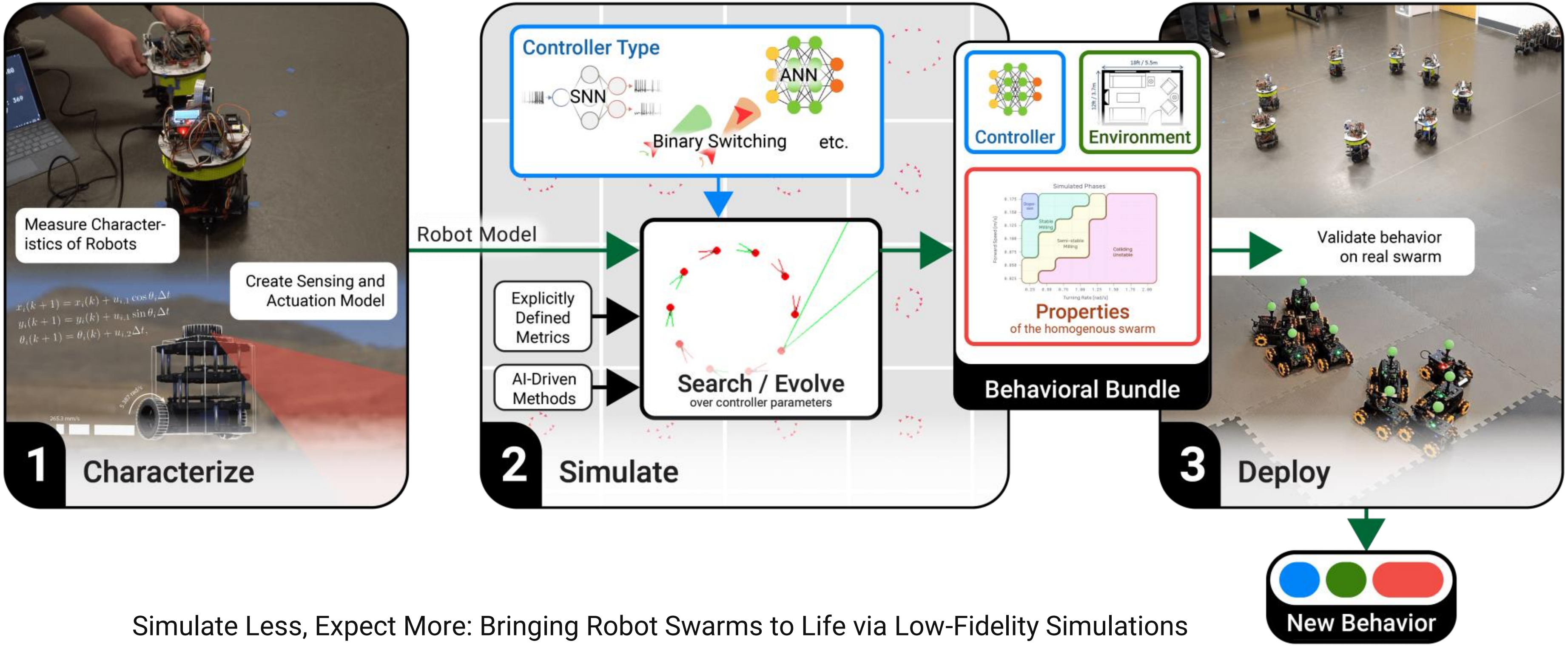



Dispersion
= Dispersion
Semi-stable Milling
< Stable Milling
Semi-stable Milling
< Stable Milling
Simulated Phases
Forward Speed (m/s)
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Turning Rate (rad/s)
Disper-sion
Stable Milling
Semi-stable Milling
Colliding Unstable
Semi-stable Milling
> Colliding Unstable
Stable Milling
> Colliding Unstable
Colliding Unstable
= Colliding Unstable

An infrared sensor detects if there's another robot within its field of view or not.







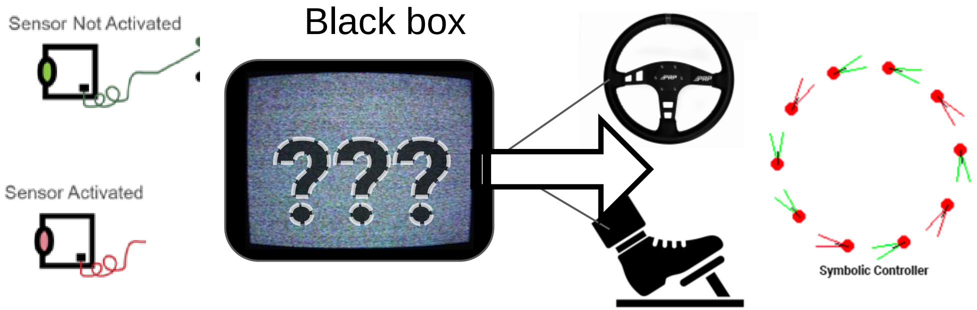
Simulate Less, Expect More: Bringing Robot Swarms to Life via Low-Fidelity Simulations
https://arxiv.org/abs/**2301.09018**

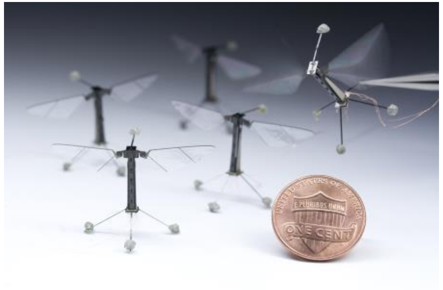
Emergence is hard to control.

With emergence, **dumb robots** can make **clever swarms**

Simulations help you find emergent behaviors faster, but **mind the sim2real gap**

Let’s see if Spiking Neural Networks are suitable.

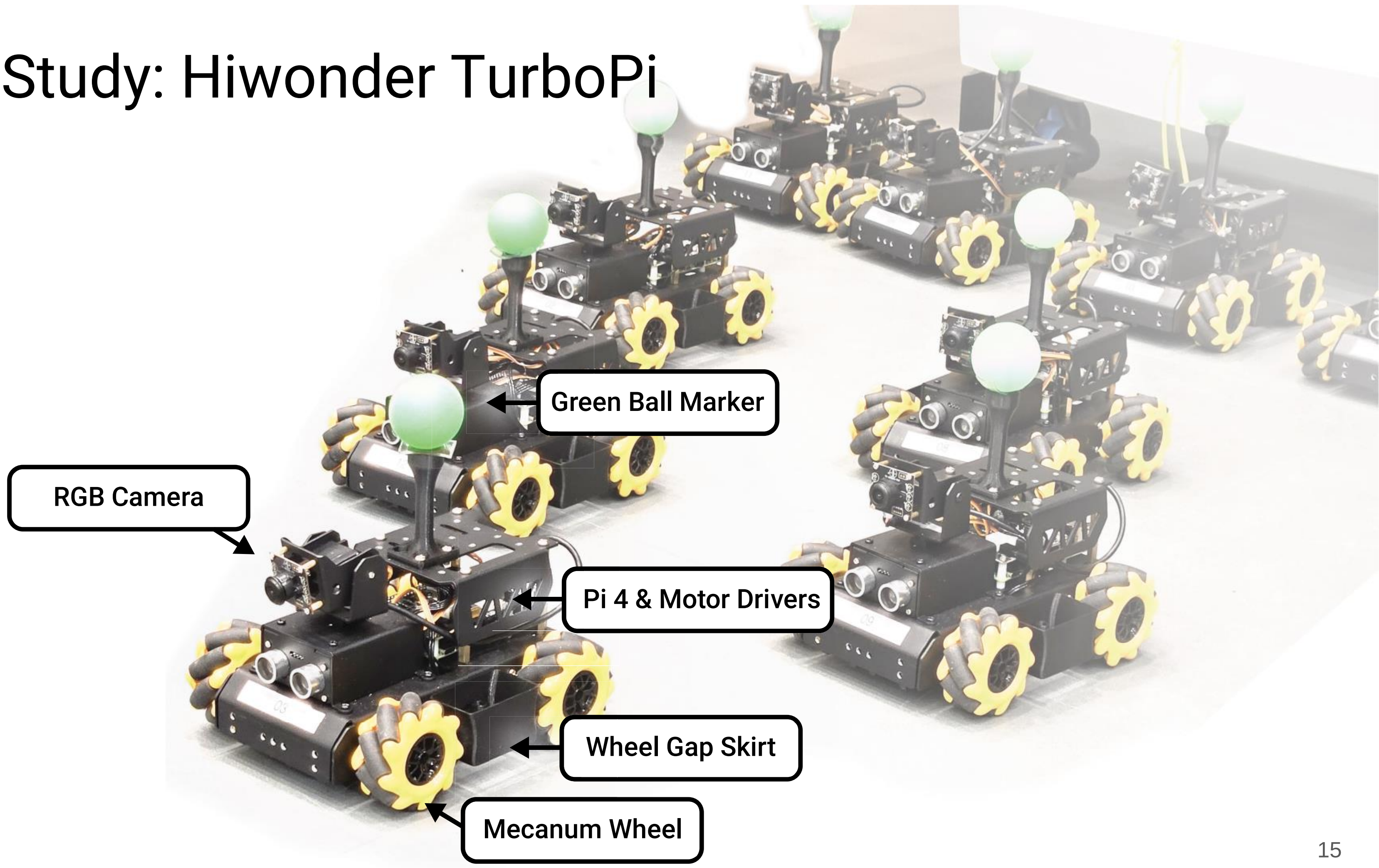


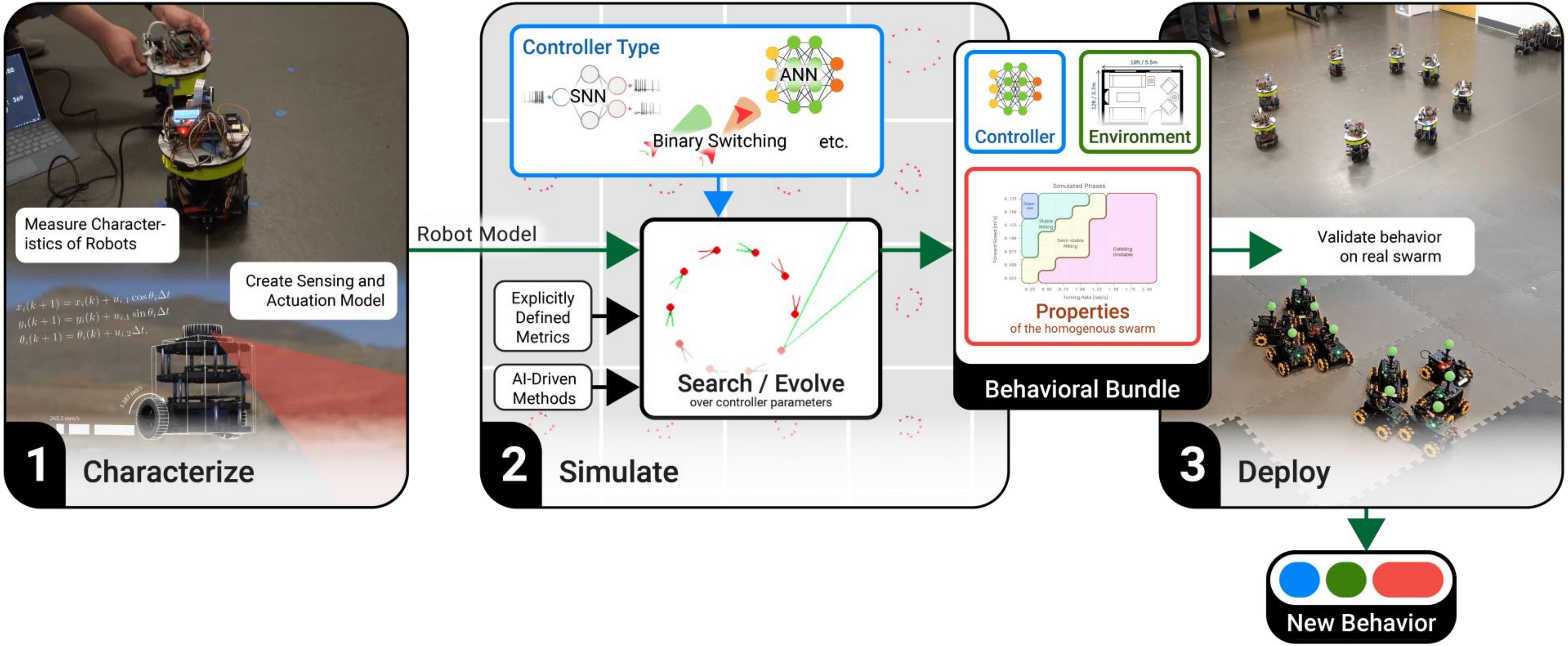


Better **S**ize, **W**eight, **a**nd **P**ower

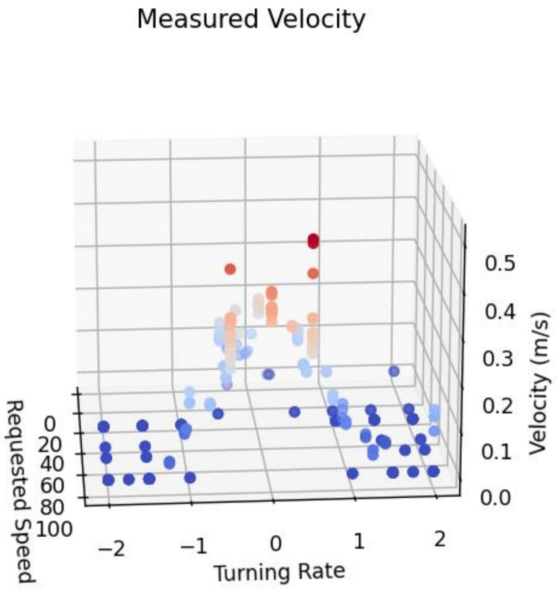
Natively operates over sequences of data

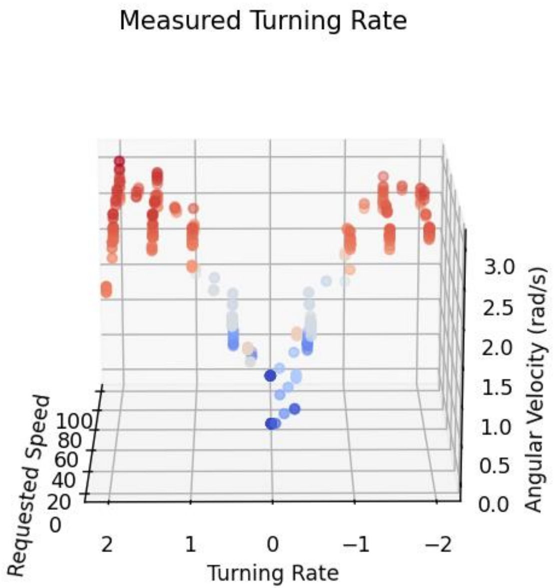
Case Study: Hiwonder TurboPi

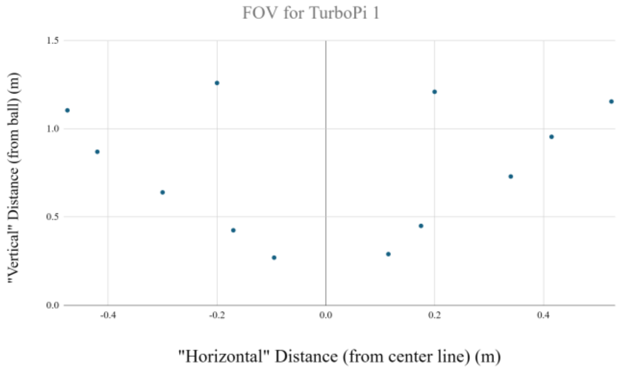



Measure Character-istics of Robots
Create Sensing and Actuation Model
1 Characterize
Robot Model
Controller Type
SNN
ANN
Binary Switching
etc.
Explicitly Defined Metrics
AI-Driven Methods
Search / Evolve
over controller parameters
2 Simulate
Controller
Environment
Simulated Phases
Properties
of the homogenous swarm
Behavioral Bundle
Validate behavior on real swarm
3 Deploy
New Behavior

Characterization



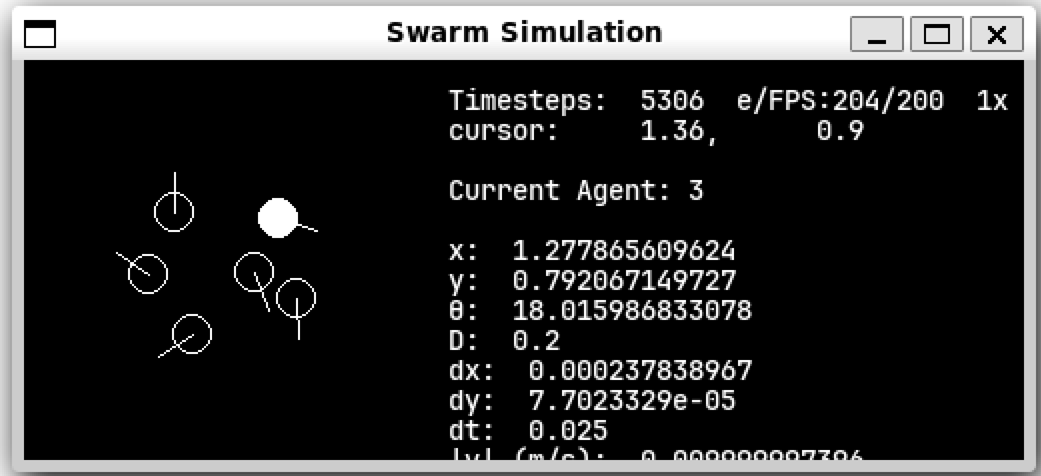




Constant Wheelspeed Outputs

Detection region

RobotSwarmSimulator



https://github.com/kenblu24/RobotSwarmSimulator

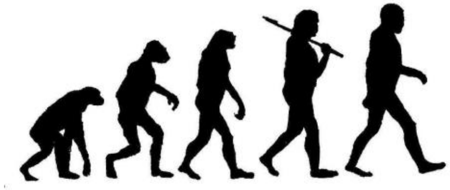
Simulator originally developed by **Connor Mattson,** Daniel Brown @ University of Utah

Objective Function

$$\lambda = 1 - \max(\overline{\Phi}, \overline{\tau}),$$

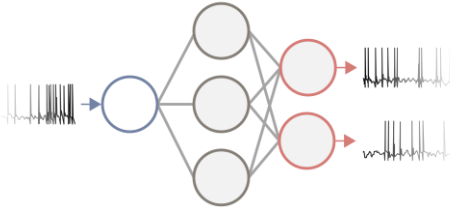
"Circliness"

Search Method



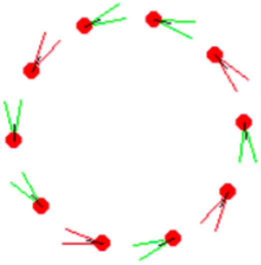
Evolutionary Learning

Class of Controllers



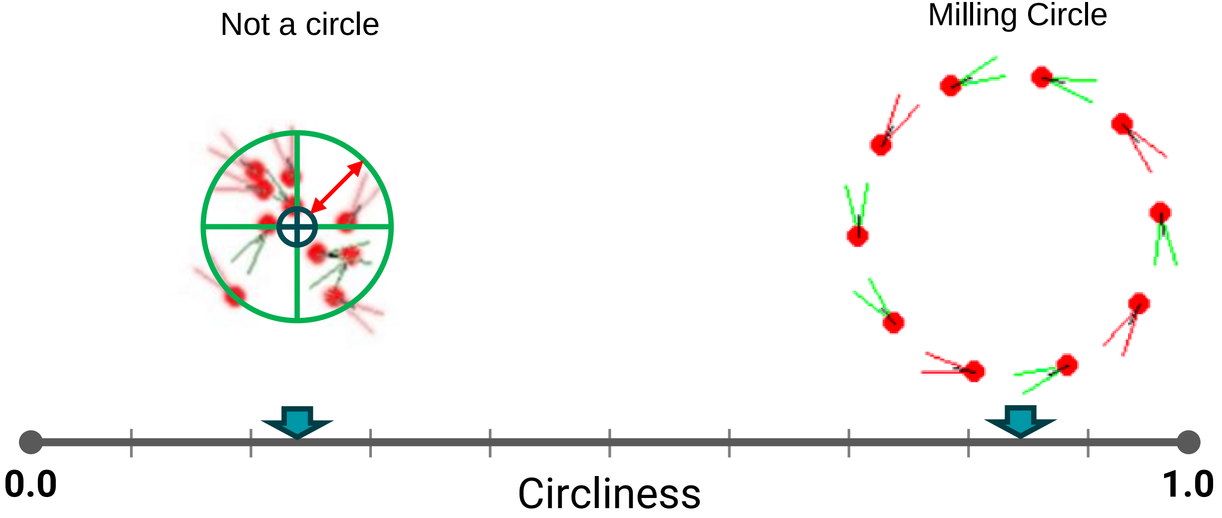
Simulated SNN: Caspian

Emergent Behavior



"Milling"

Objective Function: Circliness



$$\lambda(k) = 1 - \max\{\bar{\phi}(k), \bar{\tau}(k)\}$$

Fatness

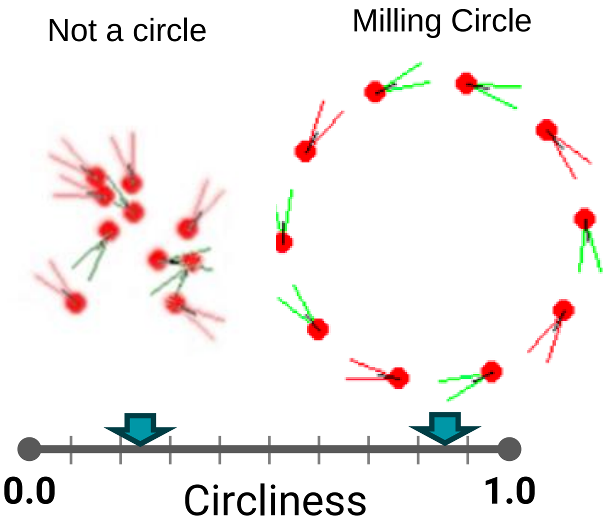
Tangentness

Averaged over the last T_{window}=450 timesteps

$$[k - T_{\text{window}},\ k]$$

See: Taylor, Luzzi, Nowzari, ACC 2020

Objective Function: Circliness



Problem Definition

Maximize this

$$\lambda(k) = 1 - \max\{\bar{\phi}(k), \bar{\tau}(k)\}$$

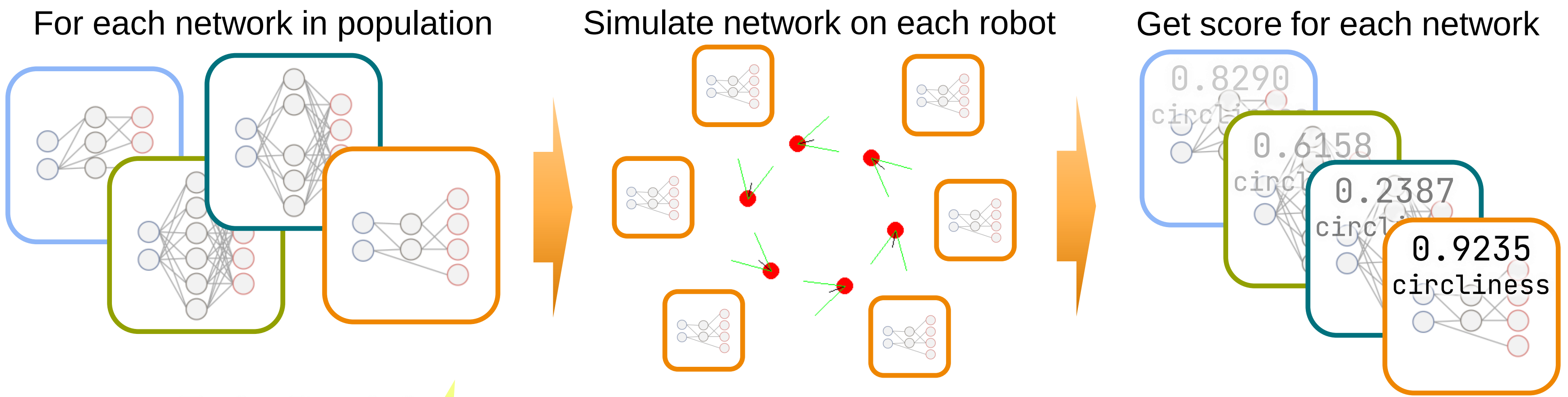
Averaged over the last T_{window}=450 timesteps $[k - T_{\text{window}},\ k]$

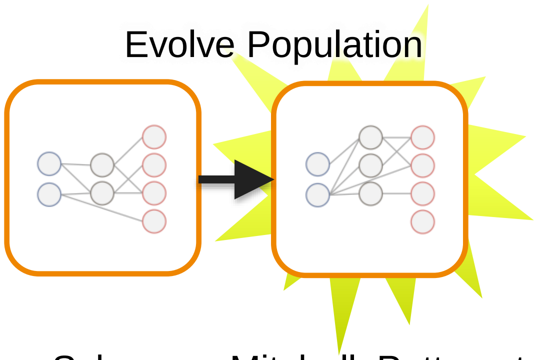
subject to

- Same **initial starting condition** & environment
- Same agents and sensing/actuation characteristics
- **Same controller** for all agents (indep. memory)
- Run simulation for 1000 ticks and get circliness

See: Taylor, Luzzi, Nowzari, ACC 2020

Evolutionary Optimization of Neuromorphic Systems (EONS)

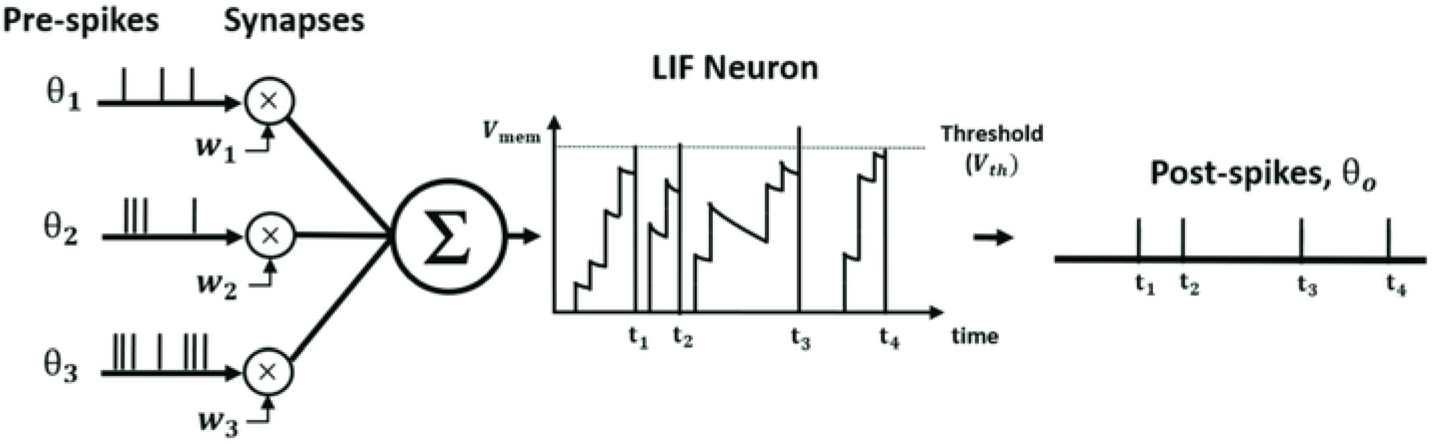




- Crossover w/ Tournament Selection
- Mutate (add/delete nodes & edges, perturb parameters)
- Keep 4 best networks
- Replace some networks with new ones

See: Schuman, Mitchell, Patton et al., NICE 2020

Caspian



- Discrete Spike times and amplitudes
- Neuron Charge Threshold
- Neuron Leak
- Axon Weight (edge weight)
- Axon Delay

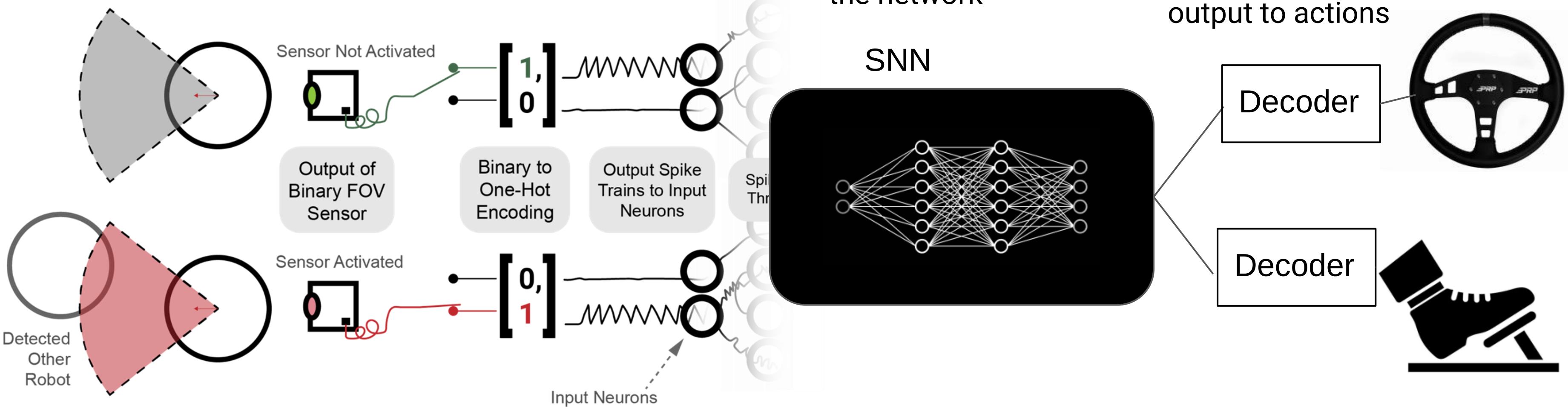
See: “Caspian…” J. Mitchell et al., NICE 2020

For each robot,

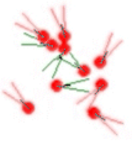
We encode the sensor output to spikes

Let the spikes propagate through the network

And decode the output to actions



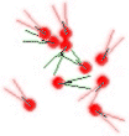
Simulation of the best network from select generations



Gen 1

Timesteps: 3

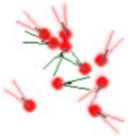
Circliness : 0.072



Gen 2

Timesteps: 3

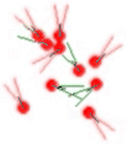
Circliness : 0.072



Gen 5

Timesteps: 3

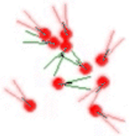
Circliness : 0.072



Gen 10

Timesteps: 3

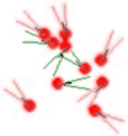
Circliness : 0.072



Gen 20

Timesteps: 3

Circliness : 0.072



Gen 100

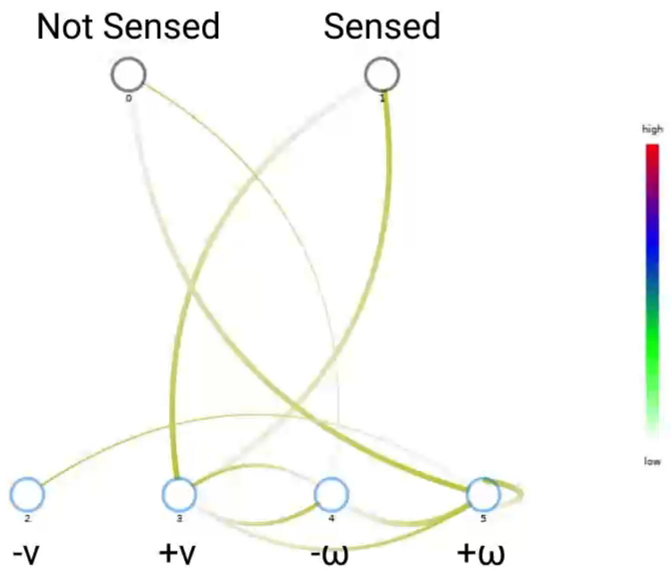
Timesteps: 3

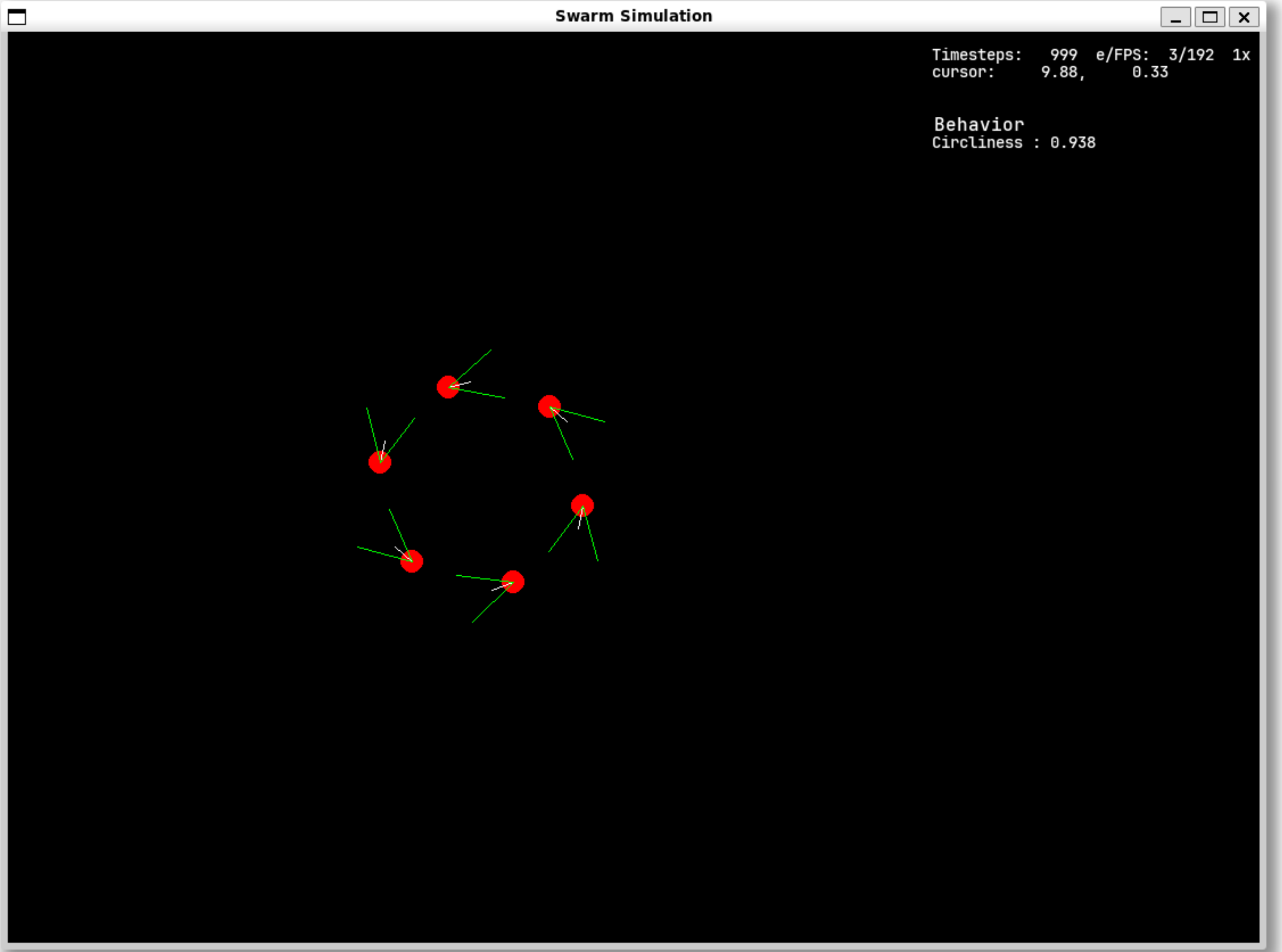
Circliness : 0.072

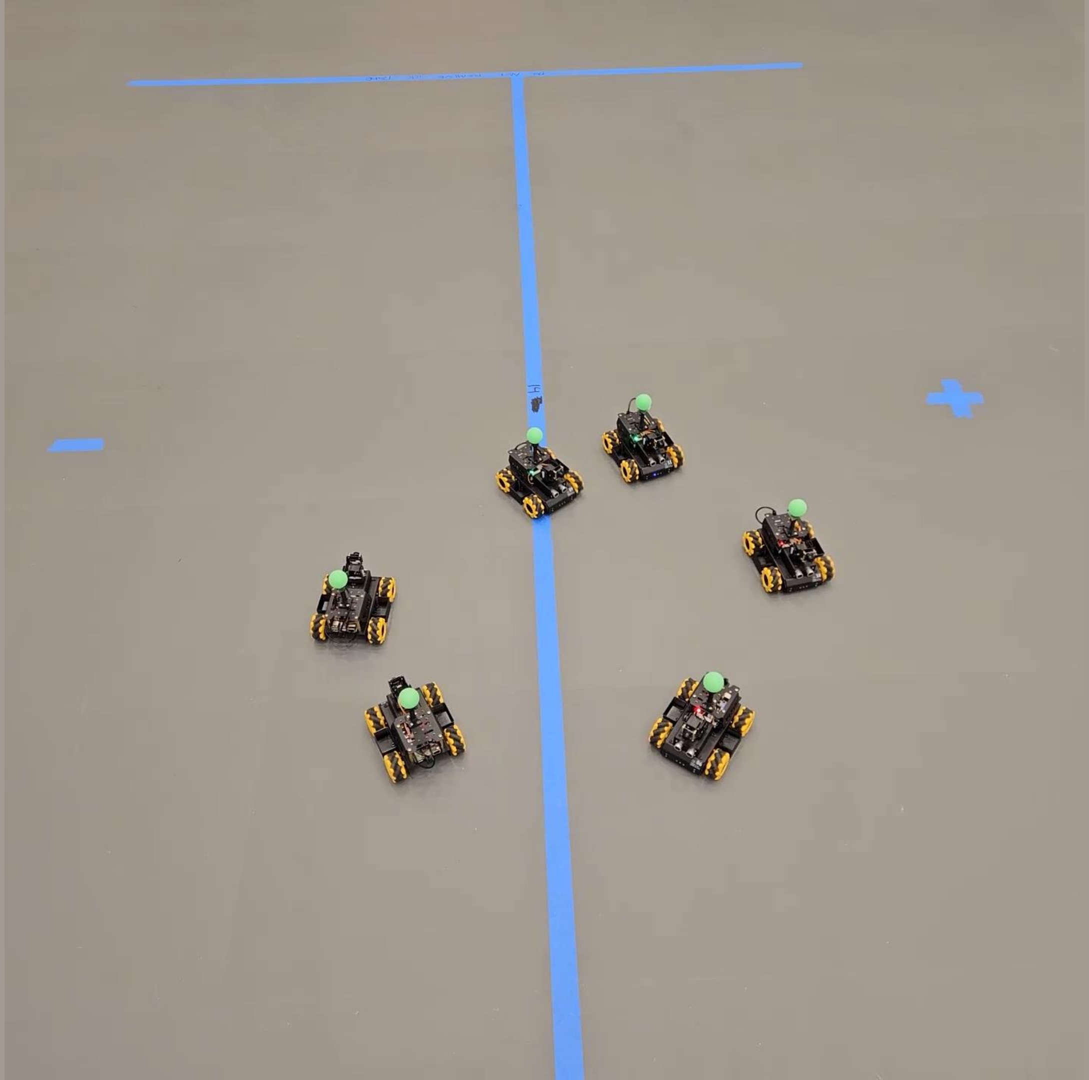
2x

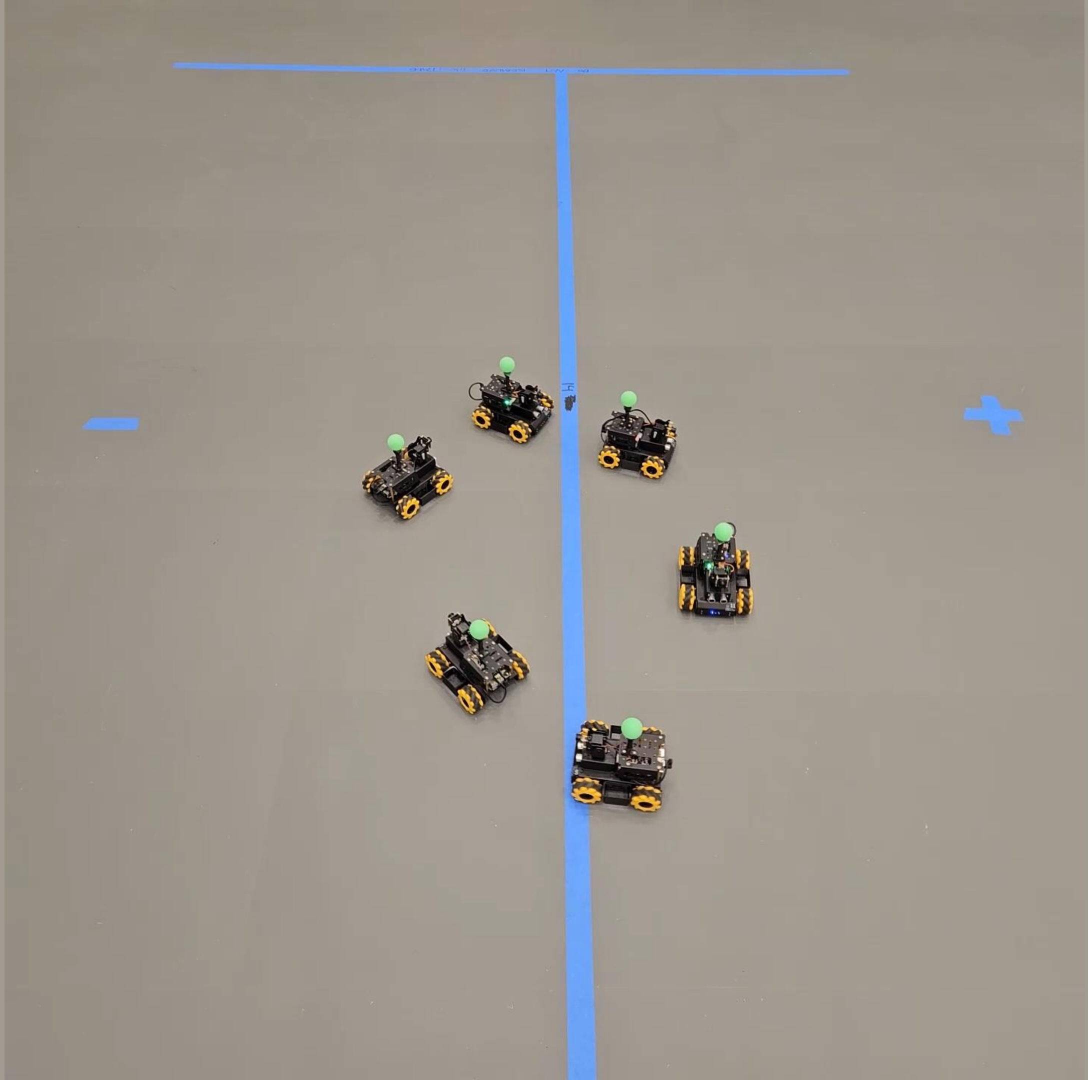
Timesteps: 4

Here's a peek inside the spiking neural network

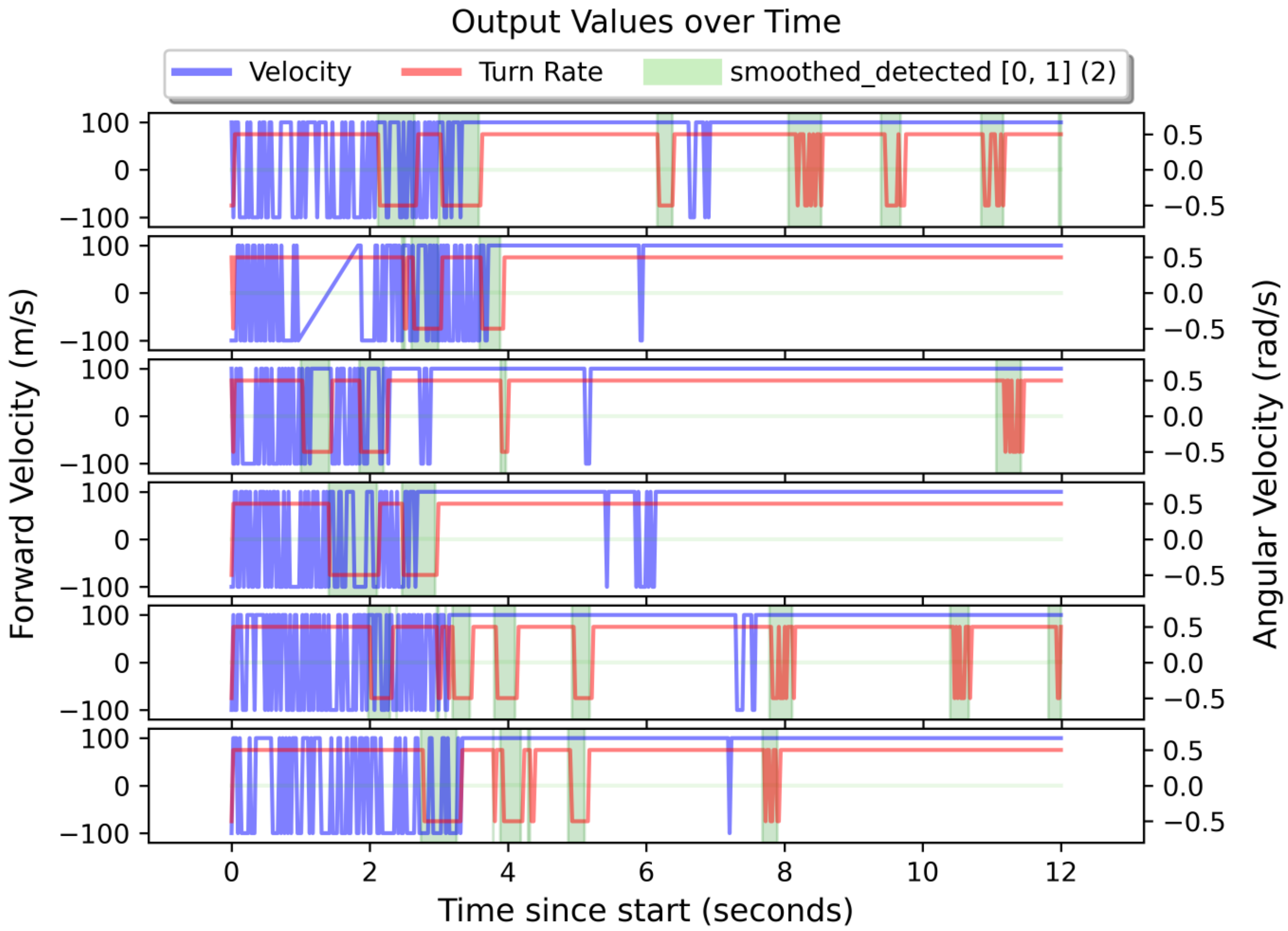



Swarm Simulation
Timesteps: 999 e/FPS: 3/192 1x
cursor: 9.88, 0.33
Behavior
Circliness : 0.938

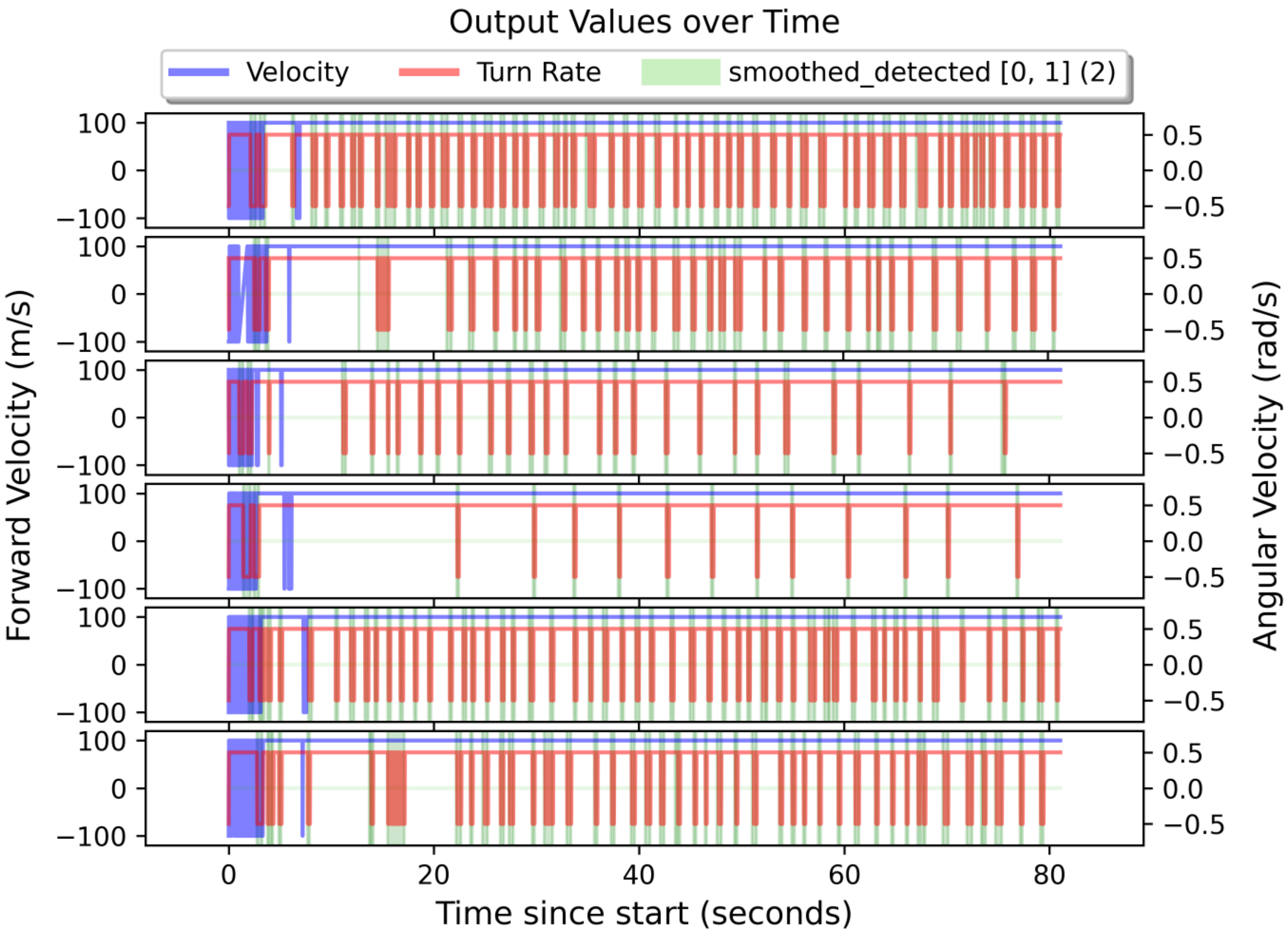





00:00:59:24

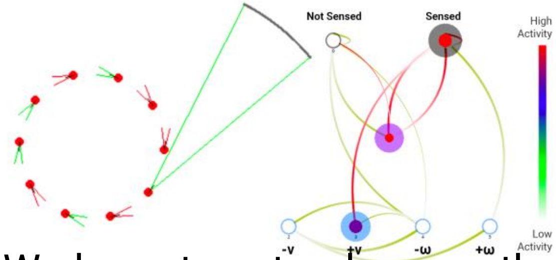


Output Values over Time
Velocity
Turn Rate
smoothed_detected [0, 1] (2)
Forward Velocity (m/s)
Angular Velocity (rad/s)
100
0
−100
0.5
0.0
−0.5
0
2
4
6
8
10
12
Time since start (seconds)

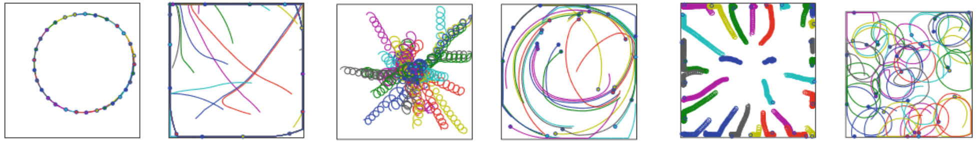


Output Values over Time
Velocity
Turn Rate
smoothed_detected [0, 1] (2)
100
0
−100
0.5
0.0
−0.5
Forward Velocity (m/s)
Angular Velocity (rad/s)
0
20
40
60
80
Time since start (seconds)

Milling is just one behavior.

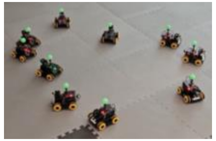


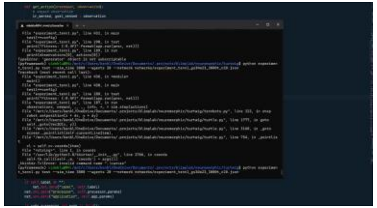
We hope to extend our methodology to the remaining behaviors we know are possible with the binary controller…



(a) Cyclic pursuit (b) Wall following (c) Aggregation (d) Milling (e) Dispersal (f) Random

And beyond...





Contact:
kzhu4@gmu.edu

Also see our prior work:
https://ieeexplore.ieee.org/document/10766566

Or check out our simulator:
https://kenblu24.github.io/RobotSwarmSimulator/