A Rapid Adapting and Continual Learning Spiking Neural Network Path Planning Algorithm for Mobile Robots

Harrison Espino, Robert Bain, Jeffrey Krichmar

CARL Lab, University of California, Irvine

Introduction

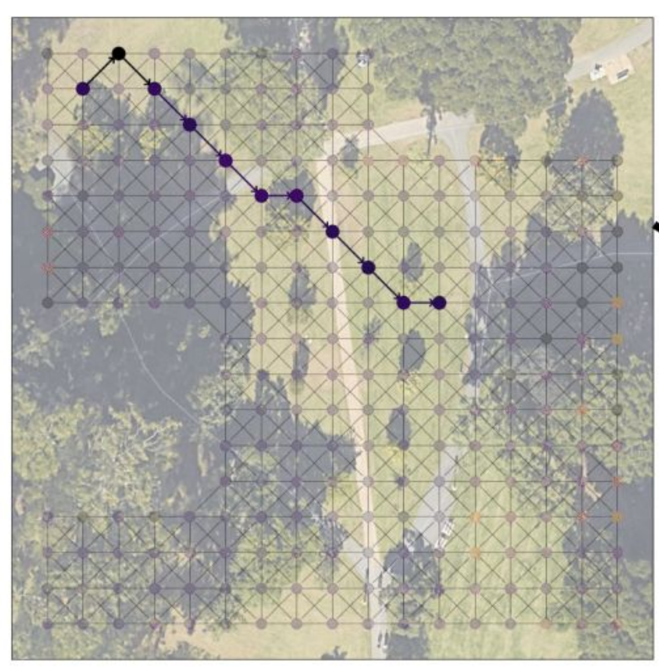
In this work, we present a neurorobotic navigation system inspired by hippocampal place cells and experience-dependent axonal plasticity. In just twelve hours of online training on a real robot, our model:

1. Simultaneously maps complex environments and plans paths over multiple measures of costs.
2. Adapts to changes in the environment without offline retraining.
3. Outperforms state-of-the-art path planning algorithms such as A*, RRT*, and D* Lite.

Our model is also compatible with neuromorphic hardware, which provide greater energy efficiency and hardware size advantages

Methodology Overview

1. Plan path using Spiking Wavefront Planner



Integrate and fire neurons defined by:

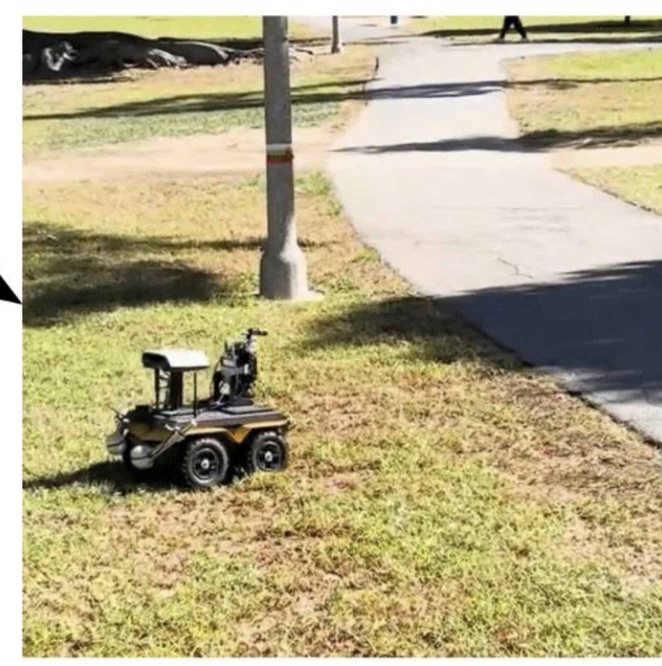
$$v_i(t+1) = u_i(t) + I_i(t+1)$$

$$u_i(t+1) = \begin{cases} -10 & if\ v_i(t) = 1 \\ min(u_i(t)+1, 0) & otherwise \end{cases}$$

$$I_i(t+1) = \sum_{j=1}^{N} \begin{cases} 1 & if\ d_{ij}(t) = 1 \\ 0 & otherwise \end{cases}$$

$$d_{ij}(t+1) = \begin{cases} D_{ij} & if\ v_j(t) \geq 1 \\ max(d_{ij}(t)-1, 0) & otherwise \end{cases}$$

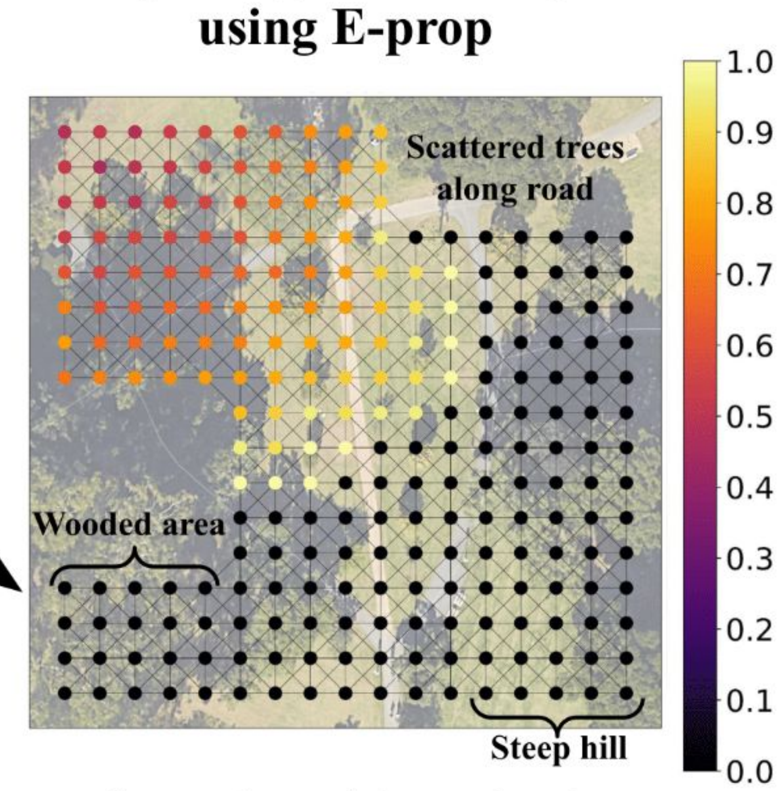
2. Execute path on robot, collecting cost data



Current cost: Energy expended by wheels
Obstacle cost: Time spent around obstacles
Slope Cost: Steepness of terrain

$$map_{xy} = [(c_{1,1}, o_{1,1}, s_{1,1}), (c_{0,2}, o_{0,2}, s_{0,2}), \ldots, (c_{7,11}, o_{7,11}, s_{7,11})]$$

3. Update planner weights using E-prop



Connection weights updated:

$$D_{ij}(t+1) = D_{ij}(t) + \delta(e_i(t)(map_{xy} - D_{ij}(t))$$

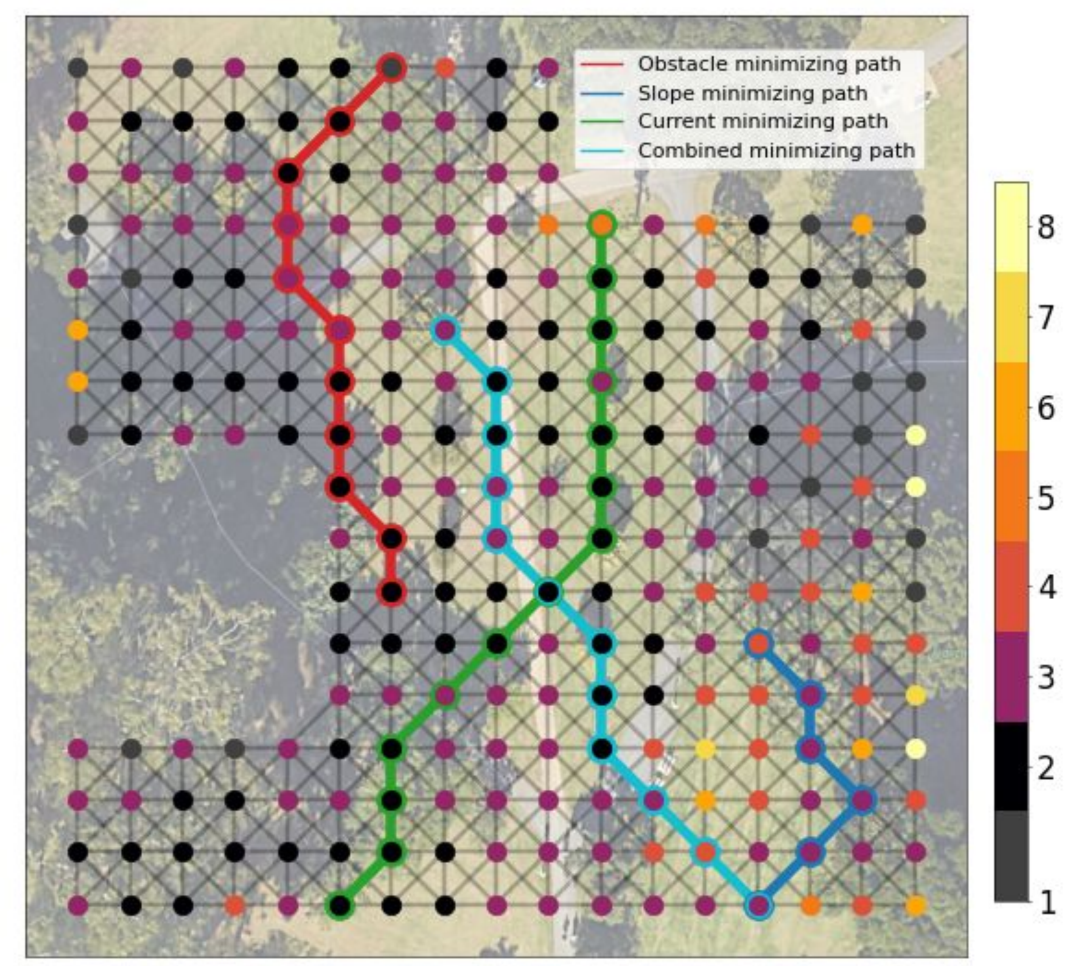
$$e_i(t+1) = \begin{cases} 1 & if\ v_j(t) \geq 1 \\ e_i(t) - \frac{e_i(t)}{\tau} & otherwise \end{cases}$$

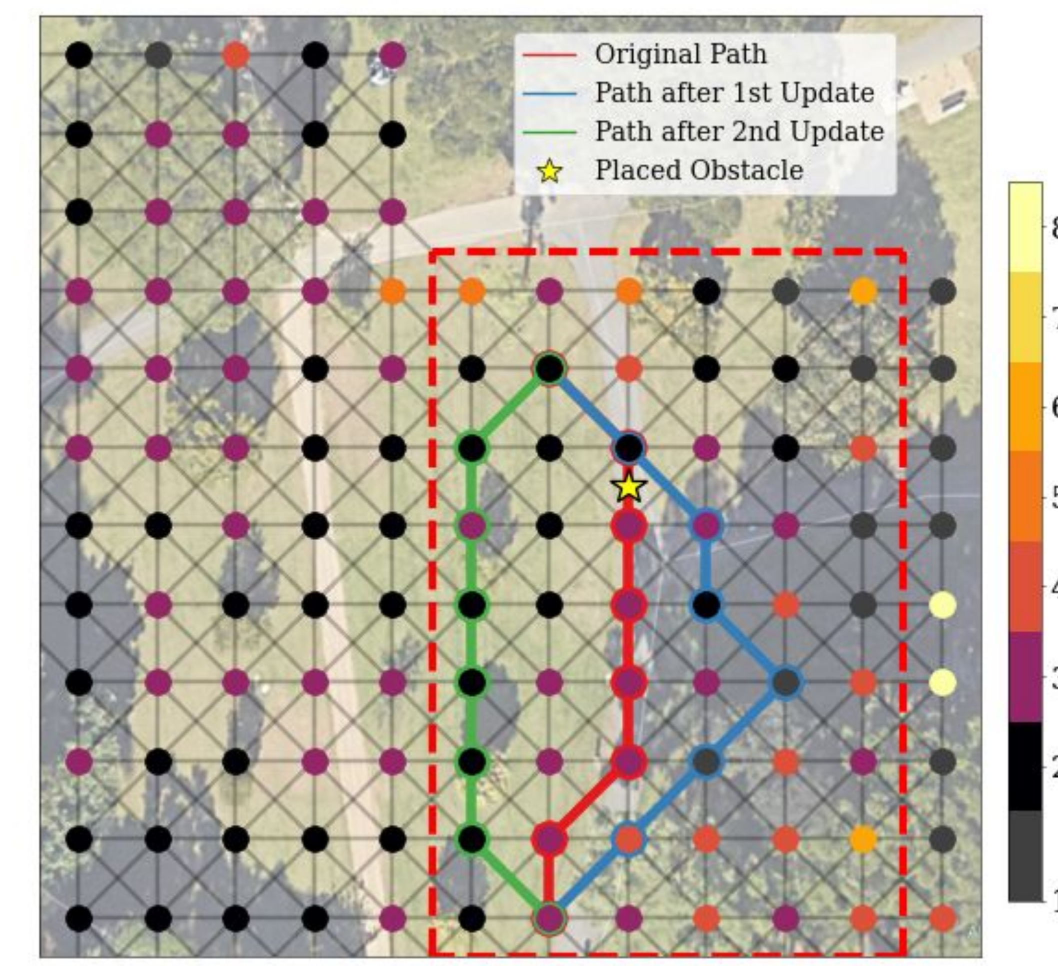
Results

Left: Nodes are colored according to the mean delays *D* of the learned model.

- When minimizing the current cost, the robot took a straight path, avoiding changes in terrain.
- When minimizing obstacle cost, the robot avoided trees and benches
- When minimizing slope cost, the robot avoided the steep hill

Right: We placed an obstacle in front of the learned robot's initial route.
After a single model update, the new route avoided the obstacle by traversing up the hill.
After a second update, the final route traveled across the grass field.





After exhaustively simulating all possible paths through the learned costmap, we found that the Spiking Wavefront Planner planned **significantly lower cost** and **shorter** paths than A*, RRT*, and D* Lite.

TABLE II: Comparison between RRT*, A*, a Naive planner, D* Lite, and SWP on simulated paths.

Planner	RRT*	A*	Naive	D* Lite	SWP
Path Length	59.01*	51.73*	47.73*	53.60*	50.91
Current Cost	22.94*	20.17	22.13*	21.08*	20.08
Obstacle Cost	14.21*	13.21*	15.44*	13.00*	12.80
Slope Cost	25.63*	22.19	23.79*	22.78*	22.17
Normalized Cost	22.07*	19.55*	21.85*	20.01*	19.36

* *denotes p<0.05; t-test with Bonferonni correction*

Please scan this QR code for videos and a link to the paper for this research.

Conclusions and Future Work

Our brain inspired model learned a complex environment autonomously through experience, integrating multiple measures of cost across different sensor modalities into its map. When faced with environmental changes, the model adapted to avoid the new obstacle after a single model update, and continued to find more cost-effective paths after subsequent updates.

In the future, we plan to use vision to estimate the cost of nearby locations, incorporate memory replay, and develop a fully neuromorphic implementation.