Integrating Spiking Neural Networks and Deep Learning Algorithms on the Neurorobotics Platform

Rachael Stentiford[1], Thomas C. Knowles[1], Fabrice O. Morin[2], Benedikt Feldoto[2], Deniz Ergene[2], and Martin J. Pearson[1]

[1] Bristol Robotics Laboratory, University of the West of England, Bristol, UK
{rachael.stentiford,tom.knowles,martin.pearson}@uwe.ac.uk

[2] Department of Informatics, Technical University of Munich, 85748 Munich, Germany
{morinf, feldottob, ergened}@in.tum.de

Abstract. We present a neurorobotic model that can associate self motion (odometry) with vision to correct for drift in a spiking neural network model of head direction based closely on known rodent neurophysiology. We use a deep predictive coding network to learn the generative model of representations of head direction from the spiking neural network to views of naturalistic scenery from a simulated mobile robot. This model has been deployed onto the Neurorobotics Platform of the Human Brain Project which allows full closed loop experiments with spiking neural network models simulated using NEST, a biomimetic robot platform called WhiskEye in Gazebo robot simulator, and a Deep Predictive Coding network implemented in Tensorflow.

Keywords: Neurorobotics Platform · Predictive coding · Spiking Neural Network · pyNEST · NRP · WhiskEye · Head Direction

1 Introduction

Neurorobotics is a discipline which works towards understanding the brain using a collaborative approach of robotics, biologically plausible machine learning approaches and spiking brain models. It undertakes to explore new neuroscience questions, particularly questions for which conventional approaches are not viable and opening up new experimental avenues in neuroscience. Neurorobotics works towards one of the pillars of the NC3Rs: Replacement, by using advanced tools to address neuroscience questions without the use of animals [9].

The Neurorobotics Platform is a tool developed as part of the Human Brain Project for conducting embodied robotics experiments with embedded bioinspired brain and control systems [3]. It provides synchronisation between spiking neural networks (SNNs) such as NEST or pyNN; and robot control tools such as ROS and Gazebo. Robot behaviours can be specified using the ROS framework and data from published topics and the spiking network can be captured and exported after the experiment.

In this work, we describe using Tensorflow to perform inference with pre-trained deep neural networks and interface those with spiking neural networks on this platform. We model the rodent head direction cell system as an SNN which estimates the current head angle of a simulated robot based on self motion (ideothetic cues), and provide allothetic (environmental) information to the network using a predictive coding network implemented with Tensorflow. We use the head direction cell system as a candidate model for showing the potential of simulated embodied experiments combining SNNs and bio-inspired machine learning, reflecting our previous work [13].

1.1 Background

Although the rodent head direction system works in the absence of vision, relying on self-motion (ideothetic) information to track head direction, the signal is subject to accumulated error (drift) [5, 6, 11, 14, 16, 17]. Rodents use allothetic information such as vision, to counter this drift in head direction estimate [14]. This requires forming learned associations between visual scenes and the current head angle, so that the estimated head angle can be corrected when this visual scene is experienced again. This visual control of head direction begins at the Lateral Mammillary Nuclei (LMN) [15], stabilising the head direction signal at its origin. The head direction is thought to originate in the reciprocal connections between the LMN and dorsal tegmental nuclei (DTN) [1, 2]. We model the head direction system as a continuous attractor network, exploiting these excitatory and inhibitory connections between these two regions.

To learn associations between heading and the visual scene we employ a generative paradigm that aims to learn to generate appropriate data samples like the training data in the appropriate contexts. The Predictive Coding Network (PCN) based on MultiPredNet [10], was re-purposed to receive head direction and vision information rather than vision and tactile. The PCN works by outputting a prediction from its latent layer that passes through the nodes of the hidden layers (if any) to the input later of each modality. At each layer, the prediction from the layer above is compared to the activity at the current layer and the difference (error) calculated. Weights between layers are then updated locally according to their prediction errors. For a full description of both the PCN and the SNN see [13].

2 Method

2.1 Neurorobotics Platform

We use a local copy of the NRP version 3.2.0. The key feature of the NRP is the Closed Loop Engine (CLE), that ties together many *de facto* standard open source software for robotics and physics simulation. These include, but are not limited to, ROS (Noetic)[12], Gazebo (11.3)[8] and NEST (2.18)[4], unified into a complete simulation platform. The individual components are orchestrated

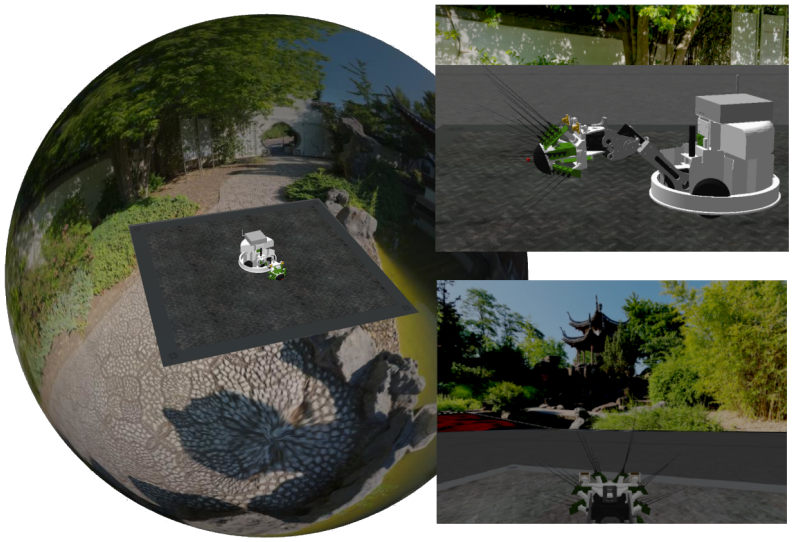

Fig. 1. Illustration showing the Chinese garden environment projected onto a sphere, with WhiskEye robot on a floating platform (left). Image of the WhiskEye robot in the environment (top right). Image from a camera mounted on the WhiskEye robot (bottom right).

by the CLE so that data can be passed between them and reacted to within a consistent timescale. This interaction is governed by Transfer Functions that describe communication between each system. For example, NEST's tick rate is typically much higher than Gazebo's, and so is repeatedly paused to output spike data via source neurons. This spike data can be read by Transfer Functions and processed into a form suitable for publishing as a conventional ROS Message on a ROS Topic.

The simulation platform builds upon Gazebo and its integrated ODE physics engine to allow for objects to be imported as DAE files with associated textures. These objects are grouped into SDF model files that describe the linkage of these objects into robots and environments, as well as sensors and kinematic chains where appropriate. Agents within the simulation can be controlled in many ways, including publishing to control topics directly, executing SMACH state machines or with spiking networks running in NEST. Gazebo resolves interactions with the simulation environment, including reading sensory data that itself can be translated into spike data and fed into the NEST model. In this way, each component is able to receive data from disparate systems without concern for asynchronous behaviour. This is all assisted by a web portal GUI that allows for the environment to be altered, robots to be added on the fly and transfer functions to be created, enabled, disabled and deleted as needed.

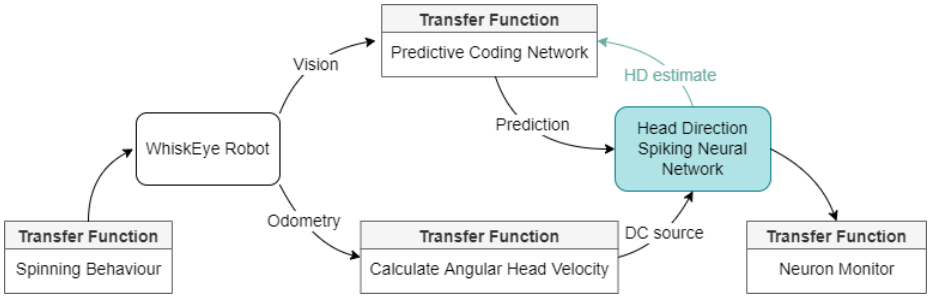

Fig. 2. Diagram showing the structure of the experiment in the NRP. The simulated WhiskEye robot turns on the spot as described by a transfer function. The pose of the robot is sent to a transfer function which calculates angular head velocity and converts it to current inputs to the conjunctive cells of the spiking neural network (SNN). The camera feed from the robot is passed to the trained Predictive Coding Network (PCN) in a transfer function and used to produce predictions which are converted to current input to each of the Head Direction cells in the SNN. The head direction estimate from the SNN, taken as the average of the active cells since the last prediction, is also passed to the PCN to measure prediction error.

Integrating Tensorflow To expand the capabilities of the NRP for our purposes we have installed Tensorflow as a library to build and run the Predictive Coding Network. Compatibility issues necessitated installing an older version of Tensorflow (2.3) with Protobuf 3.9.2 as a dependency. As Tensorflow 2.3 requires different versions of key libraries (most notably numpy) than NRP 3.2, Tensorflow and its dependencies were installed in a separate Virtual Environment within the relevant Experiment folder. To access Tensorflow within the Brain File, the virtual environment's libraries are appended to the current namespace using the inbuilt **site** library. This allows Tensorflow to execute within its own isolated environment and allows both versions of it and the NRP's dependencies to coexist.

The Tensorflow code was appended to the Brain File. Though intended to set up NEST models, the Brain File also blocks simulation setup whilst it is running, making ideal for any heavy computations that need to be run once. Tensorflow code included defining the Predictive Coding model itself, creating a new session, and loading trained weights from file. To avoid garbage collection, the model had required refactoring from its 'desktop' version, appending all variables to a Network object that would persist as an NRP Brain File variable. These Brain File variables are accessible across Transfer Functions, unlike conventional Python globals which are not usable within Transfer Functions. Once the model has been built and weights loaded, the inference function was also exposed as an NRP Brain File variable. This makes it visible to Transfer Functions running during the simulation to pass data to the PCN and receive predictions back from it.

Term	Meaning
Experiment	A collection of robot, brain and asset files for a particular simulation. This is described by an EXC file, which holds the settings of the simulation.
CLE	The Closed Loop Engine. This orchestrates the various components of the NRP with a unifying simulation clock, and enables communication between them via transfer functions.
Transfer Function	A Python script that is run as part of the CLE's loop. They are decorated with @Robot2Neuron or @Neuron2Robot accordingly.
Brain File	A Python file that describes the NEST model to be loaded.

Table 1. List of terms used within the NRP

.

This allows us to close the loop, producing head direction predictions from camera images *in situ* as the simulation runs, and pass these predictions as current input to the brain model to track and correct drift in the estimated head angle in real time. The PCN can also receive live input from the NEST model, with each tick of the CLE retrieving the activity of the NEST devices that form its own ideothetic prediction of head direction. This provides both a target for causal inference for the PCN and allows error between the PCN and NEST estimates to be calculated, both which would be impossible without the synchronised execution of Tensorflow and NEST that the NRP supports. The flow diagram shown in figure 2 describes the interplay between major components of the experiment.

2.2 WhiskEye

WhiskEye in the NRP is a simulated version of a rat-inspired omnidrive robot with RGB cameras in place of eyes and active whisker-like tactile sensors reported previously (see figure 1). WhiskEye was integrated into the Human Brain Project's Neurorobotics Platform (NRP) as part of prior work [7, 10].

ROS topics including body angle, neck position, camera feeds published related to the robot are subscribed to in transfer functions and these data passed to the predictive coding network (PCN) or spiking neural network (SNN), see below. Camera feeds must be deserialised and converted into the flattened RGB format that the PCN expects; this is accomplished by the PCN Inference transfer function.

2.3 Environment

Within the NRP we situate a sphere mesh onto which a background image is projected, enclosing a central suspended platform with grey stone texture for

WhiskEye to move on. The sphere is large enough that translation within the bounds of the platform lead to no perceptible changes to the visual scene. This ensures the robot nominally observes the same scene when it returns to that head angle, analogous to rodents observing distal environmental cues.

2.4 Brain Model

The brain model is a pyNEST (2.18) spiking neural network (SNN). Using a network structure supported by experimental observations, we define four populations each of 180 standard leaky integrate-and-fire neurons (iaf_psc_alpha) which use alpha-function shaped synaptic currents. These populations represent head direction cells in the Lateral Mammillary nuclei (LMN) and Dorsal Tegmental nuclei (DTN); and two conjunctive head direction by asymmetric angular head velocity populations [1]. LNM cells have constant current input of 450 pA that maintains spontaneous firing at a rate of 50 spikes per second prior to inhibitory input from the DTN.

Reciprocal connections between the LMN and DTN have been shown to be essential for generating the head direction signal [2]. A single stationary bump of activity is produced by providing inhibitory input to all cells surrounding the most active LMN cell. Excitatory connections from LMN neurons to the DTN with declining synaptic strength as a function of distance, inhibition is returned from DTN cells with synaptic strength decreasing as a function of distance offset by a constant (μ).

Conjunctive cells are connected one to one with LMN cells offset by one cell either clockwise or anticlockwise. Conjunctive cells require both AHV and HD input to fire, and shift the bump around the ring. AHV input is provided by a transfer function. For further description of the spiking neural network structure see [13].

2.5 Transfer Functions

AHV input Subscribing to */whiskeye/body/pose* and */whiskeye/head/bridge_u*, which publishes neck angles, the body pose (x, y, θ) is transformed into head pose (x, y, θ) through the kinematic chain of the neck. An NRP Brain File variable keeps track of the previous head angle, so the angular head velocity (AHV) can be calculated. If there is a change in AHV, this is then transformed into current which is injected into the conjunctive cell populations via two *dc_source* devices. A *CSVRecorder* device records the computed head angle.

Spinning Head angle is altered by continuously writing to the */whiskeye/body/cmd_vel* topic z variable, producing a continuous spinning motion of the WhiskEye robot.

Neuron Monitor LMN head direction cell population spikes are collected using a *spike_recorder device*, and written to a CSV using a *CSVRecorder*. Recorded spikes are also displayed in the spike train window.

Predictive Coding Network A multimodal predictive coding network based on the previously proposed MultiPredNet [10], attempts to reconstruct each pair of inputs - images of visual scenes and head direction - from a multimodal latent space. Prior work [10, 13] has shown this network's effectiveness at predicting head direction from natural scenes, the bio-plausiblity of its learning rule and connectivity making it an ideal tool for neurorobotic experiments. Having been trained in an offline setting on at least one full rotation of views from the same environment, these weights are loaded into the network during setup. This enables it to produce accurate predictions of head direction based on the allothetic cues of the visual scene. These predictions are 180 elements in width at each update step, matching the number of HD cells. The prediction values are injected as current into the network via 180 *dc_source* devices connected one to one with the LMN population. To prevent negative current injections, any negative values in the prediction are set to 0.

For efficiency, the Predictive Coding Network is loaded at the same time as the Brain Model via the Experiment's Brain File. This ensures the Tensorflow libraries, network structure and saved weights are only loaded once. As discussed in 2.1, the model is declared in the Brain File and the Inference function made available by an NRP Brain File variable accessible across Transfer Functions.

2.6 Spike Analysis

The most active cell in 40 ms bins is identified and assumed to be the current most active cell and the peak of the bump. Converting the cell number to a value between π and $-\pi$, these values are then compared to the ground truth head angle. The difference between the estimated and ground truth head angle indicates accumulation of drift over time, with total error measured as Root Mean Squared Error (RMSE).

3 Results

3.1 Head Angle Estimated by the SNN Follows Ground Truth

The bump of activity in the SNN is centred on the current estimate of head angle. Movement of the bump is driven by current input to the conjunctive layers which are connected one cell clockwise or anticlockwise around the ring. In the absence of corrective input from the PCN, the bump driven by ideothetic input only is subject to drift (Figure 3A blue), as the model for transferring AHV to current is not optimal. The difference between the ground truth head angle and the estimates head angle increases over time (Figure 3B blue), resulting in a total error of 2.42 radians (1.263 RMSE), after 3 minutes of simulation.

When the PCN transfer function is active, producing predictions of the current head angle based on the current visual scene observed by the robots cameras. These predictions are converted to current and injected into the head direction cells, as a corrective input. As seen in our previous work [13], this corrective signal reduced the drift resulting in total error of 0.43 radians (82.23% reduction; 0.370 RMSE).

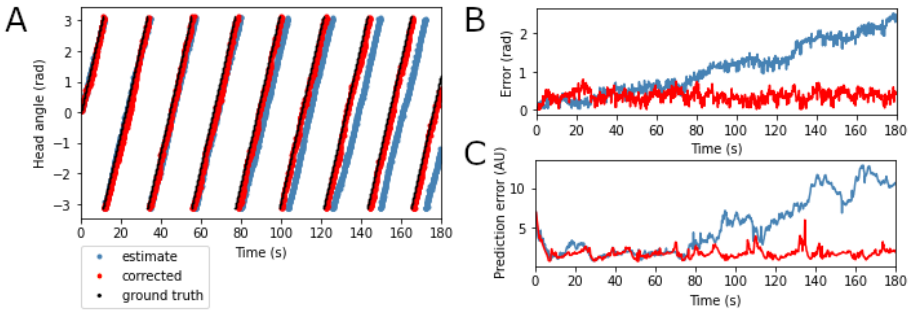

Fig. 3. Estimated head angle from the spiking neural network (SNN). (A) Plot showing estimated head angle with ideothetic drive only (blue), the corrected estimated head angle (red) which is updated using predictions from the predictive coding network (PCN), and ground truth head angle (black). (B) Error measured as the difference between the estimate and the ground truth is shown. Allothetic input from the PCN results in minimised drift and the corrected estimate and ground truth are almost indistinguishable. (C) Plot showing the prediction error at each time point when the head direction estimate is passed back to the PCN, when the predictions are used to correct for drift (red) or not (blue).

3.2 PCN Prediction Error Increases with Drift

Until this point we have measured the drift in the SNN by directly comparing the HD estimate to the ground truth. However rodents do not have access to this ground truth information in order to evaluate the confidence of the HD estimate. As part of its inference process, the PCN produces a prediction error between the expected head angle and its reconstruction, which may be a suitable alternative.

The NRP allows both the SNN and the PCN to run synchronously as the robot moves. This makes it possible to send continuous feedback between the two models. The predictions from the PCN can be used to update the SNN estimate, and in return the current HD estimate supplied to the PCN to be compared to the prediction generation based on the visual data.

By passing the current head angle to the PCN, the prediction error between the presented head angle and the reconstruction is calculated. When the predictions are used to correct for error in the HD estimate, the prediction error remains low (Figure 3C red) with small peaks representing small inconsistencies that remain in the corrected head direction estimate. If we generate the predictions but do not feed these to the SNN, the head direction estimate drifts, resulting in an overall gradual increase in error (Figure 3C blue). The prediction error in the drifting case correlates strongly with the calculated error (Figure 3B,C blue; 0.929 Pearson correlation coefficient). When the predictions are used to correct for error in the HD estimate, the prediction error reflects the reconstruction quality for each of the head angles (Figure 3C red). The oscillation in

the prediction error indicates the prediction quality is not equal across all head angles.

3.3 Ideothetic information drives network in periods of darkness

During periods of darkness or when distal visual information is not available because it is obscured by proximal objects, animals must rely upon ideothetic information to keep track of their head direction. To observe the response of the network to ambiguous visual information we obscured a 90 degree portion of the visual scene (Figure 4A blue shaded region). During this period the predictions produced by the PCN became close to a flat line, and very little current was injected into the network. Figure 4A shows ideothetic information drives the bump during the dark period with minimal drift. Figure 4B shows while the calculated error between the ground truth and the estimated head angle remains low (red), the prediction error (green) from the PCN shows peaks not visible in the calculated error which match up the the periods of darkness. This is a strong signal that the predictions are inaccurate, and in future experiments could be used to trigger learning after extended periods of poor prediction.

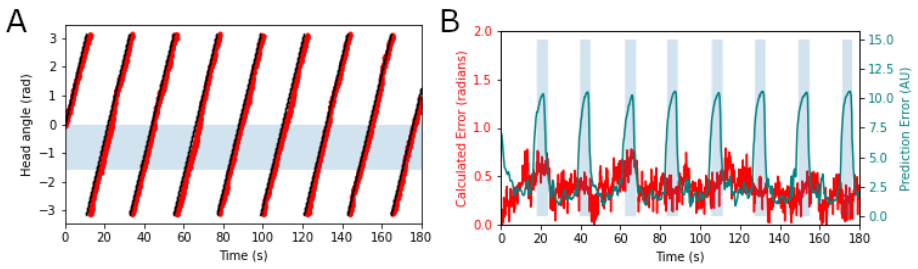

Fig. 4. Network response during periods of low visual information. (A) Plot showing the corrected estimated head angle (red) which is updated using predictions from the predictive coding network (PCN), and ground truth head angle (black). Visual information is obscured in the region shaded blue. (B) Plot showing the calculated error (red) between the ground truth and the estimate, and the prediction error from the PCN (red). Visual information is obscured in the regions shaded blue.

4 Discussion

This work is a successful example of using machine learning and spiking neural networks in a closed loop, embodied, situated model on the Neurorobotics platform (NRP). We have replicated the results of our previous work reducing drift in a spiking neural network (SNN) model of the head direction system using

head angle predictions made by a predictive coding network (PCN). The NRP allowed us to close the loop, with the SNN and the PCN running synchronously as the robot moves, controlled by the CLE of the neurorobotics platform (NRP).

Closing the loop opens up new avenues for reciprocal information transfer between the PCN and the SNN, previously not possible before NRP integration. As the prediction error produced by the PCN reflects drift in the network, it could serve as an indicator of poor predictions. In novel environments or in the dark, where visual information is unreliable, we would expect this error to increase and the SNN to drift, as the PCN would be unable to reconstruct the head angle associated with a novel visual scene. This opens up new potential for online learning of new pairs of head angle and visual scene, triggered by increases in the prediction error, which again would not be possible outside of the NRP.

Currently the simulation inside the NRP runs slower than real time. The NRP allows this by blocking processes, such as transfer functions (including the PCN) and the brain model, until the current process is complete. As the online NRP currently utilises CPUs only, we have restricted our work to CPUs, however Tensorflow is suited to using GPUs and as NEST is CPU intensive, a combination of the two may lead to faster simulation. Integration with SpiNNaker and Loihi on the NRP opens doors to transferring this type of experiment into real time on a robot with neuromorphic hardware.

Using these methods, neuroscience questions can be explored, combining SNN and machine learning in the NRP, and in turn new questions for experimental studies generated. We anticipate similar experiments will encourage collaboration between experimental and computational neuroscientists or roboticists, working towards the principles of Replacement and Reduction of animal research set out by the NC3Rs.

Acknowledgment

This research has received funding from the European Union's Horizon 2020 Framework Programme for Research and Innovation under the Specific Grant Agreement No. 945539 (Human Brain Project SGA3).

Chinese Garden in Stuttgart panorama taken and released under CC0 license by Andreas Mitchok.

References

1. Bassett, J.P., Taube, J.S.: Neural correlates for angular head velocity in the rat dorsal tegmental nucleus. The Journal of Neuroscience **21**, 5740–51 (8 2001)
2. Blair, H.T., Cho, J., Sharp, P.E.: The anterior thalamic head-direction signal is abolished by bilateral but not unilateral lesions of the lateral mammillary nucleus. The Journal of neuroscience **19**, 6673–83 (8 1999)
3. Falotico, E., Vannucci, L., Ambrosano, A., Albanese, U., Ulbrich, S., Vasquez Tieck, J.C., Hinkel, G., Kaiser, J., Peric, I., Denninger, O., Cauli, N., Kirtay, M., Roennau, A., Klinker, G., Von Arnim, A., Guyot, L., Peppicelli, D., Martínez-Cañada,

P., Ros, E., Maier, P., Weber, S., Huber, M., Plecher, D., Röhrbein, F., Deser, S., Roitberg, A., van der Smagt, P., Dillman, R., Levi, P., Laschi, C., Knoll, A.C., Gewaltig, M.O.: Connecting artificial brains to robots in a comprehensive simulation framework: The neurorobotics platform. Frontiers in Neurorobotics **11**, 2 (2017). https://doi.org/10.3389/fnbot.2017.00002

4. Gewaltig, M.O., Diesmann, M.: Nest (neural simulation tool). Scholarpedia **2**(4), 1430 (2007)
5. Goodridge, J.P., Taube, J.S.: Preferential use of the landmark navigational system by head direction cells in rats. Behavioral neuroscience **109**, 49–61 (1995). https://doi.org/10.1037/0735-7044.109.1.49
6. Goodridge, J.P., Dudchenko, P.A., Worboys, K.A., Golob, E.J., Taube, J.S.: Cue control and head direction cells. Behavioral neuroscience **112**, 749–761 (1998). https://doi.org/10.1037/0735-7044.112.4.749
7. Knowles, T.C., Stentiford, R., Pearson, M.J.: Whiskeye: A biomimetic model of multisensory spatial memory based on sensory reconstruction. In: Lecture Notes in Artificial Intelligence 13054. pp. 408–418. Springer (9 2021). https://doi.org/10.1007/978-3-030-89177-0_43
8. Koenig, N., Howard, A.: Design and use paradigms for gazebo, an open-source multi-robot simulator. In: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566). vol. 3, pp. 2149–2154 vol.3 (2004). https://doi.org/10.1109/IROS.2004.1389727
9. NC3Rs: The 3rs., https://www.nc3rs.org.uk/who-we-are/3rs
10. Pearson, M.J., Dora, S., Struckmeier, O., Knowles, T.C., Mitchinson, B., Tiwari, K., Kyrki, V., Bohte, S., Pennartz, C.M.: Multimodal representation learning for place recognition using deep hebbian predictive coding. Frontiers in Robotics and AI **8** (12 2021). https://doi.org/10.3389/frobt.2021.732023
11. Stackman, R.W., Golob, E.J., Bassett, J.P., Taube, J.S.: Passive transport disrupts directional path integration by rat head direction cells. Journal of neurophysiology **90**, 2862–2874 (11 2003). https://doi.org/10.1152/JN.00346.2003
12. Stanford Artificial Intelligence Laboratory et al.: Robotic operating system, https://www.ros.org
13. Stentiford, R., Knowles, T.C., Pearson, M.J.: A spiking neural network model of rodent head direction calibrated with landmark free learning. bioRxiv p. 2022.02.01.478640 (1 2022). https://doi.org/10.1101/2022.02.01.478640
14. Taube, J.S., Burton, H.L.: Head direction cell activity monitored in a novel environment and during a cue conflict situation. Journal of neurophysiology **74**, 1953–1971 (1995). https://doi.org/10.1152/JN.1995.74.5.1953
15. Yoder, R.M., Peck, J.R., Taube, J.S.: Visual landmark information gains control of the head direction signal at the lateral mammillary nuclei. Journal of Neuroscience **35**, 1354–1367 (2015). https://doi.org/10.1523/JNEUROSCI.1418-14.2015
16. Yoder, R.M., Taube, J.S.: Head direction cell activity in mice: robust directional signal depends on intact otolith organs. The Journal of neuroscience **29**, 1061–1076 (1 2009). https://doi.org/10.1523/JNEUROSCI.1679-08.2009
17. Yoder, R.M., Taube, J.S.: The vestibular contribution to the head direction signal and navigation. Frontiers in Integrative Neuroscience **8**, 32 (4 2014). https://doi.org/10.3389/fnint.2014.00032