Sensory Updates to Combat Path-Integration Drift

Xiang Ji, Shrinu Kushagra, and Jeff Orchard*

Cheriton School of Computer Science
University of Waterloo
jorchard@uwaterloo.ca

Abstract. Even without sensory input, an animal can estimate how far it has moved by integrating its velocity, a process called path integration. The entorhinal cortex (EC) and hippocampus seem to be involved in path integration, and in an animal's perceived location in space. However, path integration is highly susceptible to accumulating errors. A real animal avoids this problem by incorporating sensory input (e.g. vision) and updating its perceived position. The best path integration models do not yet incorporate this sensory-updating feature. In this paper, we extend one such model to enable sensory updating, and demonstrate its effectiveness in a series of computer simulations of spiking neural-network models.

1 Introduction

Path integration is the process of tracking one's believed location by integrating velocity. Recent research has tried to explain how animals employ path integration when navigating through their environments. Robotic navigation is a broad topic, in which drift in "dead reckoning" is an issue, but we focus solely on solutions consistent with biological neuroscience.

Biological findings show that certain neural activities are involved in path integration. Some neurons in the hippocampus fire action potentials (spikes) only when the animal occupies a specific location in its environment, suggesting that these so-called *place cells* encode location. Some neurons in the entorhinal cortex (EC) fire bursts of activity at locations that form a hexagonal grid in the environment [1]; these neurons are called *grid cells*. It is now thought that place cells and grid cells are involved in path integration. Also, many neurons in the EC and hippocampus exhibit membrane potential oscillations at a frequency between 4 and 12 Hz, which varies with the animal's velocity and have thus been dubbed *velocity-controlled oscillators*, or *VCO*s [2,3]. It is becoming evident that path integration results from the interaction between the animal's velocity and these oscillators [2,4].

Many biologically-based path integration models use interference between oscillators to explain how neurons generate such spatial activity patterns. As the animal moves around, the relative phases of the oscillators change and result in

* Corresponding author.

O. Zaïane and S. Zilles (Eds.): Canadian AI 2013, LNAI 7884, pp. 263–270, 2013.

a spatial pattern of constructive and destructive interference. Different combinations of oscillators can be used to create different firing patterns [5,6,7]. For example, [4] proposes a theory of path integration that ties together grid cells, place cells, and theta-phase precession (not discussed here) into a coherent and expandable framework using VCOs.

However, path integration tends to be effective only for short paths, and accumulates error quickly [8]. Scientists have discussed how sensory input might update or correct the path integration system [6,9]. Generally speaking, they propose a theory that place cells are activated by recognition of a location or landmark, and that these place cells feed back to the grid cells to induce the corresponding phase differences.

A spiking-neuron model of path integration incorporating sensory updates was proposed in [10]. However, their model is not consistent with the oscillator-based models and so does not exhibit grid-cells or phase precession. But it demonstrates that sensory update is feasible.

In this paper, we propose an extension to the interference-based model in [4]. Our extension enables updating of the internal state with sensory input, but automatically reverts back to path integration when sensory input is not available.

2 Background

The oscillating frequency θ of a VCO is a linear function of the animal's velocity $\mathbf{v}$, given by $\theta = \mathbf{c} \cdot \mathbf{v} + \theta_0$, where $\mathbf{c}$ is a constant vector specific to the VCO, and θ_0 is a constant [2]. The phase difference at time t between two VCOs with frequencies $\theta_1(\tau) = \mathbf{c}_1 \cdot \mathbf{v}(\tau) + \theta_0$ and $\theta_2(\tau) = \mathbf{c}_2 \cdot \mathbf{v}(\tau) + \theta_0$, can be written as

$$\phi(t) = \int_0^t \left((\mathbf{c}_1 \cdot \mathbf{v}(\tau) + \theta_0) - (\mathbf{c}_2 \cdot \mathbf{v}(\tau) + \theta_0)\right) d\tau = \mathbf{c}_\delta \cdot \int_0^t \mathbf{v}(\tau) d\tau = \mathbf{c}_\delta \cdot \mathbf{x}(t),$$

where $\mathbf{x}(t)$ is the animal's position at time t, and $\mathbf{c}_\delta = \mathbf{c}_1 - \mathbf{c}_2$. In this way, the phase difference between VCOs encode position.

For simplicity, let us consider the case when the animal is restricted to move only in 1-D (e.g. along the x-axis). In this case, both $\mathbf{v}$ and $\mathbf{c}$ would be 1-D vectors or scalars. Let us consider 3 VCOs with the $\mathbf{c}$ values $c_1 = 5$, $c_2 = 7$ and $c_3 = 9$, respectively. The phase difference between the first and the second VCO is then $2 \cdot x(t)$. Thus we can see that it encodes the actual position. Similarly, the phase difference between the second and third VCO is also $2 \cdot x(t)$. One more point to note is that the phase difference between equally spaced VCOs is the same. Thus, if we have an array of uniformly spaced VCOs, then we expect a constant phase increment from one VCO to the next.

At any instant of time, the phases of the VCOs form a straight line (considering 1-D motion) or a plane (considering 2-D). We call this a *phase-ramp*; its slope encodes the animal's position. As the animal moves from one place to another, the phase ramp tilts to track its location. As discussed in [4], these phase ramps can be used to generate various spatial activity patterns like place cells, grid cells, and other spatial maps.

3 Architecture and Methods

3.1 Phase Coupling

If we had perfect VCOs, we could expect the phase to keep its linear trend, even as the rat's movements altered the slope. In reality, however, neural oscillators are not perfect and tend to drift out of phase. With time, the phase along the array loses its ramp-like structure and no longer encodes the animal's position.

To overcome this problem, additional nodes are used to couple adjacent VCOs. These so-called *phase-step* nodes store the slope of the phase-ramp and supply feedback between the VCOs so that they maintain the correct phase difference [4]. Each pair of adjacent VCOs is coupled using a phase-step node, so for an array of D VCOs we have $D-1$ phase-step nodes. The phase-step nodes work in three steps as follows:

1. Each phase-step node computes the phase difference between its two connected VCOs. The phase-step node gets the unit-length phase vector (x, y) from each of the two VCOs and computes their phase difference using,

$$(c', s') = (x_1, y_1)\ \overline{(x_2, y_2)} = \big(x_1x_2 + y_1y_2, -x_1y_2 + y_1x_2\big)$$

   where (x_1, y_1) and (x_2, y_2) are the oscillator states and (c', s') the computed phase difference (expressed in the form of a unit-length phase vector).
2. Each phase-step node broadcasts its computed phase difference to the other phase-step nodes, and they arrive at a consensus and store the result.
3. Each phase-step node sends a correction to its afferent VCOs to bring their phase difference closer to the consensus. Given the consensus phase difference (c, s), we rotate (x_1, y_1) clockwise to get (x_2', y_2'), an approximation of (x_2, y_2). Likewise, we rotate (x_2, y_2) counter-clockwise to get (x_1', y_1'). This is done using,

$$(x_1', y_1') = (x_2, y_2)\ \overline{(c, s)} \quad \textit{and} \quad (x_2', y_2') = (x_1, y_1)\ (c, s)\,.$$

   We then compute the error using $(\Delta x, \Delta y) = (x', y') - (x, y)$. We only need to compensate for a fraction of that error. Hence the phase of the 1st VCO (x_1, y_1) is adjusted by $0.2 \cdot (\Delta x_1, \Delta y_1)$ and similarly for the other VCO. This process is depicted in Fig. 1.

This design has the effect of maintaining a constant phase difference along the VCOs. The design is shown in Fig. 2 (left). This phase difference, along the array of VCOs, encodes the animal's perceived position.

3.2 Sensory Input

As mentioned, the VCOs are not perfect oscillators. Even though there is phase coupling, the slope of the phase ramp drifts as it accumulates error over time. Hence, the animal's perceived position drifts from its actual position. An actual animal avoids this problem by using sensory feedback. In terms of our model,

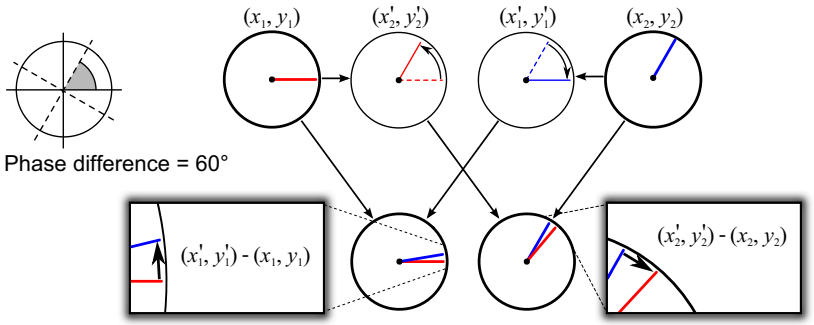

Fig. 1. The operations performed by a phase-step node. The VCOs have phase vectors (x_1, y_1) and (x_2, y_2). In this example, the consensus phase difference is 60°.

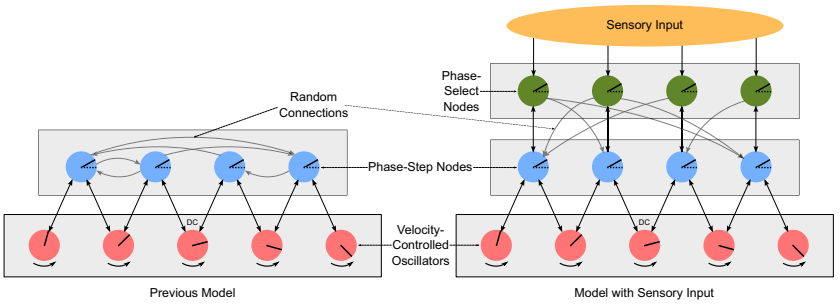

Fig. 2. Architecture of the VCOs, phase-step nodes, and phase-select nodes. The left side shows the architecture from [4], before sensory input was incorporated into the model. The right figure shows the modified architecture to incorporate sensory input.

the sensory input needs to update the slope of the phase ramp (which is stored in the phase-step nodes) to reflect the correct location. This updating then propagates to the VCOs.

This correction is achieved by introducing *phase-select* nodes in the architecture. This is the main contribution of this paper.

From the sensory system, we assume the phase-select nodes receive the correct phase difference that should be observed between adjacent VCOs. There are $D-1$ phase-select nodes, in 1-to-1 correspondence with the phase-step nodes, as shown in Fig. 2 (right).

In the previous model, the phase-step nodes arrived at a consensus phase difference among themselves. However, in our extended model, each phase-step node simply sends its computed phase difference to its corresponding phase-select node, which stores it. These phase-select nodes also receive and store the true phase from the sensory input (if it is available).

The job of each phase-select node is to select one of those two options and send it back into the path-integration system. It sends back the true phase difference

if the sensory input is available, and sends back the estimated phase difference otherwise. In our implementation, this selection is controlled by a variable that takes the value of 0 or 1.

The phase differences sent back from each phase-select node is broadcast to a random selection of phase-step nodes, and the phase-step nodes store the average. This round-trip (phase-step → phase-select → phase-step) has the same consensus-building effect as in step 2 of the previous architecture [4].

Here we have described the architecture for motion in 1-D. This method extends trivially to 2-D. In our simulations, we used three such 1-D arrays, called *propellers*, oriented at 0°, 120° and 240°. Each propellor has 9 VCO nodes (300 neurons each), 8 phase-step nodes (500 neurons each), and 8 phase-select nodes (500 neurons each). The model uses the Neural Engineering Framework (NEF) [11], and is implemented using Nengo (nengo.ca). The NEF is a framework for encoding and decoding data using populations of spiking leaky integrate-and-fire neurons. More details about the model's implementation can be found in [4].

4 Experiments and Results

We conducted several experiments in which a virtual rat runs around in a circular pen. Our aim is to examine our model's performance in path integration, both with and without sensory input, and compare that performance to behaviours we would expect from a real animal. Throughout this section, we will use the term "lights on" or "bright environment" to indicate the situation when the sensory input is available, and "lights off" or "dark environment" for the case when the sensory input is not available.

Each propeller encodes the rat's displacement along its length, while all three collectively encode the rat's location in 2-D. The environment is a round platform with a radius of 1. The velocity profile of the virtual rat is generated randomly. Also, the rat changes direction when it reaches the boundary of the platform. The actual position is computed by numerically integrating the velocity, whereas the path integration system sets its position from the phase ramp of the VCOs.

Experiment 1. We start by testing the model's performance in both bright and dark environments. When lights are turned on, the rat should be able to sense the actual position and correct the state of the VCOs. Conversely, in dark environments the rat can only determine its position by path integration.

We ran the model both with and without sensory input, and examined the Euclidean-distance error between the actual and the perceived position. Five trials were run for each of the two conditions, each lasting 30 seconds of simulation time. We computed the average error over the five runs, and filtered the error with an averaging window of 100 ms. The filter removes much of the spike noise, similar to how a synapse would filter the incoming spikes.

The results are shown in Fig. 3. The figure shows that in bright environments, the mean Euclidean error maintains a low and somewhat-constant value of around 0.08 (about 10% of the radius). However, when the lights are off,

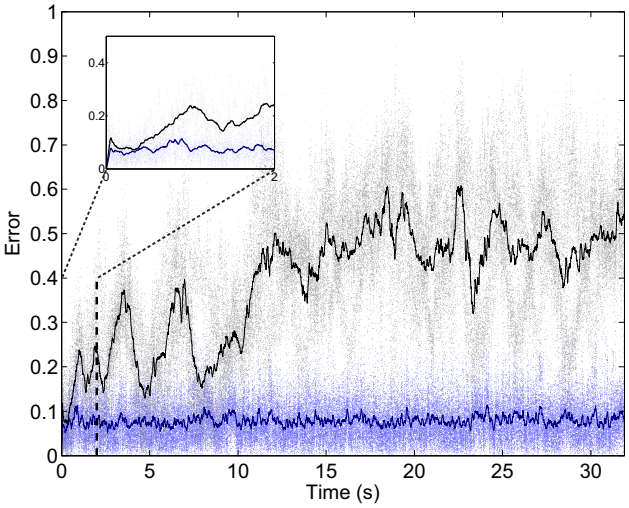

Fig. 3. Euclidean distance error of the perceived location over time. Each dot in the figure is a decoded location, while lines are the filtered average of errors. Black indicates the dark condition, and blue indicates bright.

the error grows to 0.5 over several seconds, indicating that the rat's perception of its position has diverged substantially from its true location. The inset in the figure shows that the error grows a lot even in the first 2 seconds in the dark. The results illustrate that our model effectively uses the actual position information to guide the perceived position.

Experiment 2. In this experiment, we displace the virtual rat from the centre of the platform, but set its path integration system so it thinks it is at the centre. Then we turn the lights on and measure how quickly the perceived position is corrected. We expect a real animal to update its percept quickly.

We set the initial perceived position at $(0, 0)$, and chose different actual positions for sensory input. Sixteen different positions were used: $(\pm 0.2, \pm 0.2)$, $(\pm 0.3, \pm 0.3)$, $(\pm 0.4, \pm 0.4)$, $(\pm 0.5, \pm 0.5)$. Each trial ran for 1 second of simulation time. We averaged the Euclidean distance error across all initial positions with the same displacement, then filtered the result with an averaging window of 10 ms.

We measured how long it took for the error to drop below 0.1. The error typically dropped below 0.1 within 200 ms. We also noted that convergence time increased as the displacement increased. This shows us that even when the perceived position is very different form the actual position, the animal is very quickly (approx. 200 ms) able to adjust its perception to match reality.

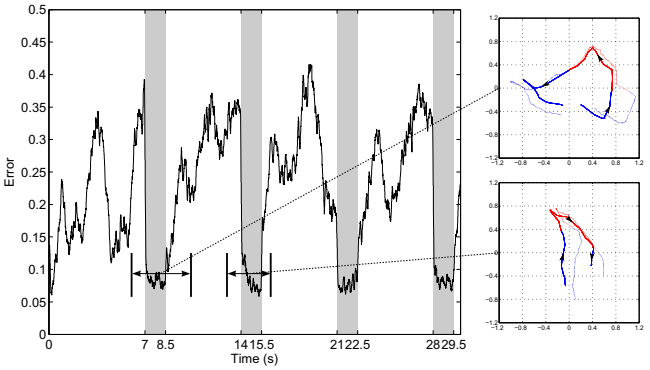

Fig. 4. Position error as the lights are turned on and off. The grey bands indicate when the lights are on. In the sample trajectories on the right, blue indicates when the lights are off, and red indicates when the lights are on. The solid line shows the actual trajectory, while the dotted line shows the perceived trajectory.

Experiment 3. For this experiment, the light is turned on and off repeatedly while we track the rat's perceived and actual position. We expect the perceived position to drift from the actual when the light is off, and then quickly converge back to the true position when the light is turned back on.

We set the lights to be on for 1.5 seconds out of every 7 seconds. Five trials were recorded, each lasting 30 seconds. We took the average error across all trials, and filtered the results with a 100 ms averaging window, as was done in experiment 1.

Figure 4 shows the error versus time. The intervals when the light is turned on is shaded in grey. The figure clearly shows that when the lights are turned on, the error falls almost instantaneously and holds its value. Notice the steep decline in error in Fig. 4 as soon as the lights are turned on (sensory feedback). However, when the lights are off, the error quickly climbs. Figure 4 also shows two examples of the actual trajectory of the rat superimposed on the perceived trajectory.

5 Conclusions and Future Work

Our model extends the EC model presented in [4] by including a mechanism for incorporating sensory corrections into the animal's perceived position. The path integration system, by itself, accumulates error and drifts from the correct position, consistent with the behaviour of animals [8]. However, supplying the correct position from sensory input alters the state of the system and brings it into harmony with the true position.

If our model is correct, then electrophysiological studies of the EC should uncover neurons whose activity is modulated heavily by the presence, or absence, of sensory positional information. When there is no sensory context for location, these neurons should behave differently than when there is.

The model that we extended included only three propellers. We expect that a full biological system would have far more propellers. Adding propellers would probably reduce the drift. We plan to build a more complete version of the model and investigate these questions.

References

1. Hafting, T., Fyhn, M., Molden, S., Moser, M., Moser, E.: Microstructure of a spatial map in the entorhinal cortex. Nature 436(7052), 801–806 (2005)
2. Welday, A.C., Shlifer, I.G., Bloom, M.L., Zhang, K., Blair, H.T.: Cosine Directional Tuning of Theta Cell Burst Frequencies: Evidence for Spatial Coding by Oscillatory Interference. Journal of Neuroscience 31(45), 16157–16176 (2011)
3. Zilli, E.A., Hasselmo, M.E.: Coupled Noisy Spiking Neurons as Velocity-Controlled Oscillators in a Model of Grid Cell Spatial Firing. Journal of Neuroscience 30(41), 13850–13860 (2010)
4. Orchard, J., Yang, H., Ji, X.: Navigation by path integration and the fourier transform: A spiking-neuron model. In: ZaÏane, O., Zilles, S. (eds.) Canadian AI 2013. LNCS (LNAI), vol. 7884, pp. 138–149. Springer, Heidelberg (2013)
5. Blair, H., Welday, A.C., Zhang, K.: Scale-Invariant Memory Representations Emerge from Moire Interference between Grid Fields That Produce Theta Oscillations: A Computational Model. Journal of Neuroscience 27(12), 3211–3229 (2007)
6. Burgess, N., Barry, C., O'Keefe, J.: An oscillatory interference model of grid cell firing. Hippocampus 17(9), 801–812 (2007)
7. Hasselmo, M.E., Brandon, M.P.: Linking Cellular Mechanisms to Behavior: Entorhinal Persistent Spiking and Membrane Potential Oscillations May Underlie Path Integration, Grid Cell Firing, and Episodic Memory. Neural Plasticity (2008)
8. Etienne, A.S., Maurer, R., Seguinot, V.: Path Integration in Mammals and its Interaction with Visual Landmarks. Journal of Experimental Biology 199, 201–209 (1996)
9. Williams, J.M., Givens, B.: Stimulation-induced reset of hippocampal theta in the freely performing rat. Hippocampus 13(1), 109–116 (2003)
10. Conklin, J., Eliasmith, C.: A Controlled Attractor Network Model of Path Integration in the Rat. Journal of Computational Neuroscience 18, 183–203 (2005)
11. Eliasmith, C., Anderson, C.H.: Neural engineering: Computation, representation, and dynamics in neurobiological systems. MIT Press, Cambridge (2003)