
The motion planning neural circuit in goal-directed navigation as Lie group operator search: Supplementary Information

Junfeng Zuo^{1,3}
zuojunfeng@pku.edu.cn

Ying Nian Wu²
ywu@stat.ucla.edu

Si Wu¹
siwu@pku.edu.cn

Wen-Hao Zhang^{3,4*}
wenhao.zhang@utsouthwestern.edu

¹Peking-Tsinghua Center for Life Sciences, Academy for Advanced Interdisciplinary Studies, School of Psychology and Cognitive Sciences, IDG/McGovern Institute for Brain Research, Center of Quantitative Biology, Peking University.

²Department of Statistics, University of California, Los Angeles.

³Lyda Hill Department of Bioinformatics, UT Southwestern Medical Center.

⁴O'Donnell Brain Institute, UT Southwestern Medical Center.

Contents

1	The one-dimensional rotation Lie group	2
1.1	The structure of $\mathbb{U}(1)$	2
1.2	The representation space of $\mathbb{U}(1)$	3
2	Computational complexity	4
3	The full circuit model	5
4	Theoretical analysis of the full circuit dynamics	6
4.1	Stationary network responses	6
4.2	Rotation of sensory representation	6
4.3	Calculation of speed	7
5	Network simulation details	8
6	Extension to 2D scenarios	9

*Corresponding author.

1 The one-dimensional rotation Lie group

1.1 The structure of $\mathbb{U}(1)$

We consider a 1D rotation Lie group $\mathbb{U}(1)$ which rotates the 1D stimulus s . A group element $R(\theta) \in \mathbb{U}(1)$ rotates the stimulus s to $s + \theta$, and mathematically it can be denoted as $R(\theta) \circ s = s + \theta$ where \circ denotes a group action. Based on the requirement of equivariant representation (Eq. (1)), a neural representation $u(x - s)$ which is equivariant to the 1D rotation group should satisfy (Fig. 1B),

$$u[x - R(\theta) \circ s] = u[x - (s + \theta)] = \hat{R}(\theta) \circ u(x - s), \quad (\text{S1})$$

where $\hat{R}(\theta)$ is the 1D rotation operator inducing the rotation of neural representation (Fig. 1B). It is worthy to distinguish that $\hat{R}(\theta)$ and $R(\theta)$ act on different spaces: $R(\theta)$ induces rotation on the original stimulus s space, while $\hat{R}(\theta)$ is the rotation operator acting on the space of neural representation $u(s)$. From the definition (Eq. (S1)) we could directly derive the properties of 1D rotation operators $\hat{R}(\theta)$ [1],

$$\hat{R}(0) = 1, \quad (\text{S2a})$$

$$\hat{R}(\theta)\hat{R}(\phi) = \hat{R}(\theta + \phi) = \hat{R}(\phi)\hat{R}(\theta), \quad (\text{S2b})$$

$$\hat{R}(\theta)^{-1} = \hat{R}(-\theta). \quad (\text{S2c})$$

Since the rotation is continuous, the amount of rotation can be made infinitesimally small. Consider an infinitesimal rotation $\delta\theta \rightarrow 0$ on the stimulus s , then the corresponding change of neural representation is,

$$\hat{R}(\delta\theta) \circ u(x - s) = u(x - s - \delta\theta),$$

Taking a first-order Taylor expansion of the above equation,

$$\begin{aligned} \hat{R}(\delta\theta) \circ u(x - s) &\approx u(x - s) - \delta\theta \partial_x u(x - s), \\ &= (1 - \delta\theta \partial_x) \circ u(x - s), \\ &= (1 + \delta\theta \hat{J}) \circ u(x - s), \end{aligned} \quad (\text{S3})$$

where $\partial_s = \partial/\partial s$ denotes the derivative over s . Also in Eq. (S3) we define

$$\hat{J} \equiv -\partial_x, \quad (\text{S4})$$

as the rotation *generator*. The generator characterizes the tangential direction of rotation near the identity element and forms a basis of the Lie *algebra*.

Eq. (S2b) suggests a large rotation can be decomposed as a composition of many infinitesimal rotations,

$$\hat{R}(\theta) = \underbrace{\hat{R}\left(\frac{\theta}{N}\right) \cdots \hat{R}\left(\frac{\theta}{N}\right)}_N \equiv \left[\hat{R}\left(\frac{\theta}{N}\right) \right]^N. \quad (\text{S5})$$

Considering the large limit of N and utilizing Eq. (S3), the above equation can be converted into

$$\begin{aligned} \hat{R}(\theta) &= \lim_{N \rightarrow \infty} \left[\hat{R}\left(\frac{\theta}{N}\right) \right]^N, \\ &\approx \lim_{N \rightarrow \infty} \left(1 + \frac{\theta}{N} \hat{J} \right)^N, \\ &= \exp(\theta \hat{J}). \end{aligned} \quad (\text{S6})$$

It is clear to see the rotation operator \hat{R} is an exponential map of the rotation generator \hat{J} . Also, the exponential map can be derived by differentiating the rotation operator:

$$\frac{d\hat{R}(\theta)}{d\theta} = \hat{J} \cdot \hat{R}(\theta),$$

then we can further derive $\hat{R}(\theta) = \exp(\theta \hat{J})$.

1.2 The representation space of $\mathbb{U}(1)$

According to Eq. (S2b), rotation group is an abelian group (group operators are commutative). It can be proved that such commutative operators share a set of common eigenfunctions that expand a representation space. To illustrate this concept, we explore the eigenfunction of the generator \hat{J} :

$$\hat{J} \cdot e^{-i\omega x} = -\partial_x e^{-i\omega x} = i\omega e^{-i\omega x}, \quad i = \sqrt{-1}. \quad (\text{S7})$$

Eq. (S7) reveals that $\exp(-i\omega x)$ performs as the eigenfunctions of rotation group generator \hat{J} , associated with the corresponding eigenvalue $i\omega$. With these eigenvalues, we concrete the abstract group actions into specific elements in the representation space spanned by the eigenfunctions. $\exp(-i\omega x)$ also happens to be the basis of *Fourier* transformation.

Through the exponential map, we can extend this representation into each group operator:

$$\hat{R}(\theta) = e^{\theta \hat{J}} = e^{i\omega \theta}, \quad (\text{S8})$$

which shows how the group operator \hat{R}_θ rotates the eigenfunction:

$$\hat{R}(\theta) \cdot e^{-i\omega x} = e^{i\omega \theta} \cdot e^{-i\omega x} = e^{-i\omega(x-\theta)}. \quad (\text{S9})$$

If we denote $f_\omega = e^{-i\omega x}/\sqrt{2\pi}$, its easy to check that the set of functions $\mathbf{f}_\omega = \{f_\omega(x)\}$ constitutes a representation space of the rotation group. Consider an arbitrary function $g(x)$ and its representation $\mathbf{g}_\omega = [g_\omega]$ in this space, then $g(x) = \mathbf{g}_\omega \cdot \mathbf{f}_\omega = \sum_\omega g_\omega f_\omega(x)$. The rotation of $g(x)$ can be applied by a dot product:

$$\hat{R}(\theta) \circ g(x) = \sum_\omega g_\omega [\rho_\omega(\theta) f_\omega(x)] = \sum_\omega g_\omega f_\omega(x - \theta) = g(x - \theta), \quad (\text{S10})$$

where $\rho_\omega(\theta)$ is the component of $\hat{R}(\theta)$ in the representation space.

Naturally, the objective function defined in Eq. (2) can be formulated as:

$$\begin{aligned} L(\theta) &= \int_x [\hat{R}(\theta) \circ u_s(x-s)] u(x-h) dx \\ &= \langle \rho(\theta) U(s), U(h) \rangle, \\ &= \sum_\omega \rho_\omega(\theta) U_\omega(s) U_\omega^\dagger(h), \end{aligned} \quad (\text{S11})$$

where subscript ω denotes the vector components.

2 Computational complexity

We argue that our model performs operator searching in a computation-saving manner. In the main text, we have compared our model with two canonical algorithms. Here we provide a more detailed comparison in Tab. S1S2S3. We consider the stimulus and goal directions are represented by N neurons respectively (corresponding to discretizing the continuous function $u(s)$ and $u(h)$), each of which is a N dimensional vector. The group convolution takes N iterations within which performs N multiplications, so the overall complexity is in $O(N^2)$ (ignoring the complexity of sorting). The complexity of Fourier transformation is mainly determined by fast Fourier transform (FFT), which is $O(N \log N)$. Each iteration of our model performs two inner products containing N multiplications. It's worth noting that our model does not need to traverse all possible operators, but go along the gradient direction. If we take an assumption that the gradient (as well as speed) decreases linearly when approaching the optimal, the time (iterations) it takes will be in $O(\log |h - s|)$, and the overall complexity will be $O(N \ln |h - s|)$.

Table S1: Group convolution: algorithm and complexity of finding and applying operators

Group convolution	
1: procedure FINDING OPERATOR	
2: for $i = 1$ to N do	▷ Iteration over all rotation operators
3: $\theta_i \leftarrow \theta_{i-1} + 2\pi/N$	
4: $L(i) \leftarrow \sum_{j=1}^N u(x_j - h)u(x_j - s + \theta_i)$	▷ $O(N)$ addition, $O(N)$ multiplication
5: end for	
6: $k \leftarrow \arg \max_i L(i)$	▷ The most efficient sorting algorithm: $O(N \log N)$
7: $\theta^* \leftarrow \theta_k$	
8: return $\hat{R}(\theta^*)$	
9: end procedure	
10: procedure APPLYING OPERATOR	
11: $u(h) \leftarrow \hat{R}(\theta^*) \circ u(s)$	▷ Rotate sensory responses: $O(N)$
12: end procedure	
13:	▷ Total complexity: $O(N^2)$

Table S2: Fourier transform: algorithm and complexity of finding and applying operators

Fourier transform	
1: procedure FINDING OPERATOR	
2: $U(s) \leftarrow \mathcal{F}[u(s)]$	▷ Fast Fourier transform: $O(N \log N)$
3: $U(h) \leftarrow \mathcal{F}[u(h)]$	▷ Fast Fourier transform: $O(N \log N)$
4: $\rho(\theta^*) \leftarrow U(h)/U(s)$	▷ $O(N)$ divisions
5: end procedure	
6: procedure APPLYING OPERATOR	
7: $U(s' = h) \leftarrow \rho(\theta^*)U(S)$	▷ $O(N)$ multiplications
8: $u(s') \leftarrow \mathcal{F}^{-1}[u(s')]$	▷ Inverse fast Fourier transform: $O(N \log N)$
9: return $R(\theta^*)$	
10: end procedure	▷ Total complexity: $O(N \log N)$

Table S3: The feedforward circuit model: computation and complexity in sequential rotation

Feedforward circuit model	
1: while $ h - s \geq \epsilon$ do	▷ Iteration over time $\sim \mathcal{O}(h - s)$
2: for $j = \text{to } N$ do	▷ Iteration over N neuron
3: $r_{\theta_+}(x_j) \leftarrow F[u(x_j - s + \Delta\theta) + u(x_j - h)]$	▷ 1 addition, 1 multiplication
4: $r_{\theta_-}(x_j) \leftarrow F[u(x_j - s - \Delta\theta) + u(x_j - h)]$	▷ 1 addition, 1 multiplication
5: end for	
6: $r_{v_+} = \sum_j r_{\theta_+}(x_j)$	▷ $\mathcal{O}(N)$ addition
7: $r_{v_-} = \sum_j r_{\theta_-}(x_j)$	▷ $\mathcal{O}(N)$ addition
8: $v_t \leftarrow r_{v_+} - r_{v_-}$	
9: return $R(v_t \Delta t)$	
10: end while	▷ Total complexity: $\mathcal{O}(N \log h - s)$

3 The full circuit model

We construct a full circuit model of the sensory-action loop, which contains three neural circuit modules with complementary functions. It has a sensory circuit model generating neural responses representing stimulus direction, $u(s)$, a motor circuit module implementing the rotation generator \hat{J} (Eq. (17)), and a sensorimotor transformation circuit module as we derived to do motion planning by computing the rotation speed v_t (Eq. (28)).

A recent study built a recurrent circuit model of *Drosophila*'s internal compass circuit corresponding to the sensory and the motor circuit module, and interpreted each module from Lie group equivariance perspective [2]. Therefore, we will construct the full circuit by combining the derived feedforward sensorimotor transformation circuit above, with the sensory and motor circuit from the recent study [2]. In below, we will use $u_m(x)$ and $r_m(x)$ to denote respectively the synaptic input and firing rate of a neuron with index (preferred stimulus/goal) x at neuronal population m at time t , while t is suppressed for concise.

Sensory circuit module. The sensory circuit module $u_s(x)$ represents the stimulus direction, corresponding to the internal compass circuit composed of E-PG neurons in *Drosophila*'s brain. The sensory module is modeled as a canonical ring attractor network dynamics ([3]),

$$\tau \dot{u}_s(x) = -u_s(x) + \rho W_{s,s} \star r_s(x) + \rho \sum_{m=s_{\pm}} W_{s,m} \star r_m(x), \quad (\text{S12})$$

where the firing rate $r_s(x) = [u_s(x, t)]_+^2 / (1 + k\rho \int [u_s(x, t)]_+^2 dx)$ via divisive normalization ([4]), with k determining global inhibition strength. The recurrent input $\rho W_{s,s} \star r_s$ is obtained by convolving the sensory neurons' activity with a Gaussian-profile kernel, $W_{s,s}(x) = w_0 \exp(-x^2/2a^2)$. It was suggested that the recurrent convolution kernel is the key to form the (rotation) group equivariant representation in a ring attractor dynamics [2]. τ and ρ are time constant and neuron density along the rotation group manifold respectively. $r_{s_{\pm}}(x)$ are two populations of P-EN neurons that will rotate the stimulus direction in $u_s(x)$ whose dynamics will be described below.

Sensorimotor transformation circuit module. Although theoretical derivations consider memoryless neurons in the sensorimotor transformation circuit (Eq. (19)), the full circuit model uses dynamic neurons, where the dynamics of $r_{\theta_{\pm}}$ and $r_{v_{\pm}}$ neurons are (converted from Eqs. (19) and (28)),

$$\tau \dot{u}_{\theta_{\pm}}(x) = -u_{\theta_{\pm}}(x) + \rho W_{\theta_{\pm},s} \star u_s(x) + \rho W_{\theta_{\pm},h} \star u_h(x), \quad r_{\theta_{\pm}}(x) = F[u_{\theta_{\pm}}(x)]. \quad (\text{S13a})$$

$$\tau \dot{u}_{v_{\pm}} = -u_{v_{\pm}} + \rho \int r_{\theta_{\pm}}(x) dx, \quad r_{v_{\pm}} = [u_{v_{\pm}}]_+, \quad (\text{S13b})$$

where $u_{\theta_{\pm}}$ (analogous to the PFL 3L and 3R neurons) compute the gradient of the objective function, i.e., $dL/d\theta$ (Eq. (19)), and the $u_{v_{\pm}}$ (analogous to the left and right DN neurons) determines the rotation speed representing the planned rotation operator parameter (Eq. (28)). In particular, the feedforward connection kernel $W_{\theta_{\pm},s}(x) = w_{\theta,s} \delta(x \mp \delta\theta)$ corresponds to the $w_s(x_v, x)$ in Eq. (24) with a connection phase shift of $\pm \Delta\theta$. Analogously, we have $W_{\theta_{\pm},h}(x) = w_{\theta,h} \delta(x)$ only that there is no phase shift of goal input. For the sake of simplicity, we set $w_{\theta,s} = w_{\theta,h}$.

Motor circuit module and motor-to-sensory feedback. The rotation generator \hat{J} for the inner neural motor-to-sensory feedback loop in *Drosophila*'s circuit can be implemented by two populations of P-EN neurons $u_{s_{\pm}}$ sending shifted feedback connections $W_{s,s_{\pm}}$ to sensory neurons u_s (E-PG

neurons, Eq. (S12)),

$$\tau \dot{u}_{s\pm}(x) = -u_{s\pm}(x) + w_{s\pm,s} r_s(x), \quad r_{s\pm}(x) = [(g_0 + r_{v\pm}) \cdot u_{s\pm}(x)]_+. \quad (\text{S14})$$

In particular, the gain of the neuron $r_{s\pm}$ is determined by the firing rate of rotation speed neurons $r_{v\pm}$ in the sensorimotor transformation circuit (Eq. (S13b)). The neurons $r_{s\pm}$ send shifted feedback connections to sensory neurons closing the sensory-action loop (Eq. (S12)),

$$W_{s,s\pm} = w_{s,s\pm} \exp[-(x \mp \Delta x)^2 / 2a^2], \quad (\text{S15})$$

where the phase of feedback connections from two populations of $r_{s\pm}$ shifts the same amount but towards opposite directions. It was found that the rotation generator \hat{J} emerges with non-zero weight phase shift $\mp \Delta x$. Moreover, the actual rotation speed v_t will be determined by the firing rate difference between $r_{s\pm}$, i.e., $\sum_x [r_{s+}(x) - r_{s-}(x)] \propto (r_{v+} - r_{v-}) \propto v_t$ which is proportional to the firing rate difference of rotation speed neurons.

4 Theoretical analysis of the full circuit dynamics

4.1 Stationary network responses

Sensory circuit module. Based on the previous literature[5], we propose the following Gaussian ansatz of network's stationary responses,

$$\begin{aligned} u_s(x-s) &= U_s e^{-(x-s)^2/4a^2}, \quad r_s(x-s) = R_s e^{-(x-s)^2/2a^2}, \\ u_{s\pm}(x-s) &= R_{s\pm} e^{-(x-s)^2/2a^2}, \quad r_{s\pm}(x-s) = R_{s\pm} e^{-(x-s)^2/4a^2}, \end{aligned} \quad (\text{S16})$$

where we have assumed that the P-EN neurons' responses $u_{s\pm}(x-s)$ and $r_{s\pm}(x-s)$ have the same position s on the stimulus manifold with the E-PG neurons' responses, which is equivalent to assume that the transmission delay from E-PG neurons to P-EN neurons and the time constant of P-EN neurons are small enough. Substituting the above Gaussian ansatz into the network dynamics of E-PG neurons (Eq. (S12)),

$$\begin{aligned} \frac{\tau U_s}{2a^2} \frac{ds}{dt} (x-s) e^{-(x-s)^2/4a^2} &= -U_s e^{-(x-s)^2/4a^2} + \frac{\rho w_0 R_s}{\sqrt{2}} e^{-(x-s)^2/4a^2}, \\ &+ \frac{\rho w_{s,s\pm}}{\sqrt{2}} \sum_{m=\pm} R_m e^{-(x-s-m\Delta x)^2/4a^2}. \end{aligned} \quad (\text{S17})$$

Similarly, the goal response will be:

$$u_h(x-h) = U_h e^{-(x-s)^2/4a^2}, \quad r_h(x-h) = R_h e^{-(x-s)^2/2a^2}, \quad (\text{S18})$$

Sensorimotor transformation circuit module Since there is no recurrent connection within the PFL3 neuron groups, the neural responses $u_{\theta\pm}$ and $r_{\theta\pm}$ are completely determined by the heading and goal inputs currents injected by E-PG and FC neurons. The stationary response will be:

$$u_{\theta\pm} = \rho w_{\theta,s} r_s(x-s) + \rho w_{\theta,h} r_h(x-h), \quad r_{\theta\pm} = \rho^2 w_{\theta,s}^2 [r_s(x-s) + r_h(x-h)]_+^2, \quad (\text{S19})$$

and the DN neurons integrate the activity of PFL3 neurons:

$$u_{v\pm} = \int r_{\theta\pm} dx, \quad r_{v\pm} = [u_{v\pm}]_+. \quad (\text{S20})$$

Ultimately, the DN neuron responses are feedback as a gain factor of the P-EN response to modulate the rotation of sensory representation.

4.2 Rotation of sensory representation

In order to study whether and how feedback inputs from P-EN neurons to E-PG neurons (the last term in Eq. (S12)) induce rotations on E-PG neurons' responses, we project Eq. (S17) onto the eigenfunction $f_1(x|s)$ corresponding to the rotation along the continuous stimulus manifold. The projection is computing the inner product between the E-PG neural dynamics (Eq. (S17)) and $f_1(x|s)$

[5]. Denoting the $f_1(x|s) = Z^{-1}(x-s)e^{-(x-s)^2/4a^2}$ with Z a normalization factor, we list the major calculations of the projection in the text below.

$$\begin{aligned} \text{LHS} &= \left\langle \frac{\tau \mathbf{U}_s}{2a^2} \frac{ds}{dt} (x-s)e^{-(x-s)^2/4a^2}, f_1(x|s) \right\rangle, \\ &= \frac{\tau \mathbf{U}_s}{2a^2} \frac{ds}{dt} \frac{1}{Z} \int (x-s)^2 e^{-(x-s)^2/2a^2} dx, \\ &= \frac{\tau \mathbf{U}_s}{2Z} \frac{ds}{dt} \sqrt{2\pi} a, \end{aligned}$$

The projection of the first two terms on the RHS of Eq. (S17) on $f_1(x|s)$ would be zero, because

$$\left\langle e^{-(x-s)^2/4a^2}, Z^{-1}(x-s)e^{-(x-s)^2/4a^2} \right\rangle \propto \int (x-s)e^{-(x-s)^2/2a^2} dx = 0.$$

At last, the projection of the last term of Eq. (S17) on $f_1(x|s)$ is

$$\begin{aligned} &\frac{\rho w_{s,s\pm}}{\sqrt{2}Z} \sum_{m=\pm} R_m \left\langle e^{-(x-s-m\Delta x)^2/4a^2}, (x-s)e^{-(x-s)^2/4a^2} \right\rangle, \\ &= \frac{\rho w_{s,s\pm}}{\sqrt{2}Z} \sum_{m=\pm} R_m \int [(x-s-m\Delta x/2) + m\Delta x/2] \exp \left[-\frac{(x-s-m\Delta x/2)^2}{2a^2} - \frac{\Delta x^2}{8a^2} \right] dx, \\ &= \frac{\rho w_{s,s\pm} \Delta x}{2\sqrt{2}Z} \sum_{m=\pm} m R_m \exp \left(-\frac{\Delta x^2}{8a^2} \right) \int \exp \left[-\frac{(x-s-m\Delta x/2)^2}{2a^2} \right] dx, \\ &= \frac{\rho w_{s,s\pm} \Delta x}{2\sqrt{2}Z} \sqrt{2\pi} a \exp \left(-\frac{\Delta x^2}{8a^2} \right) \sum_{m=\pm} m R_m. \end{aligned}$$

Combining all above calculations, the projection of Eq. (S17) onto the stimulus rotation direction can be eventually computed as

$$\begin{aligned} \frac{ds}{dt} &= \frac{\rho w_{s,s\pm} \Delta x}{\sqrt{2}\tau \mathbf{U}_s} \sum_{m=\pm} m R_m e^{-\Delta x^2/8a^2}, \\ &= \frac{\rho w_{s,s\pm} \Delta x}{\sqrt{2}\tau \mathbf{U}_s} (R_+ - R_-) e^{-\Delta x^2/8a^2}. \end{aligned} \tag{S21}$$

Suppose the amount of connection shift Δx is small enough compared with the connection width a , i.e., $(\Delta x)^2 \ll 8a^2$, the exponential terms in above equation can be ignored for simplicity. Reorganize the terms in above equation,

$$\frac{ds}{dt} \approx \frac{\rho w_{s,s\pm}}{\sqrt{2}\tau \mathbf{U}_s} (R_+ - R_-) \Delta x. \tag{S22}$$

In the stationary state, the network only rotates along the stimulus space and then we can derive the dynamics of sensory neuron response,

$$\begin{aligned} \frac{\partial u_s(x-s)}{\partial t} &= \frac{\partial u_s(x-s)}{\partial s} \frac{\partial s}{\partial t}, \\ &= \frac{\partial s}{\partial t} [\hat{J} \circ u_s(x-s)], \\ &= \frac{\rho w_{s,s\pm}}{\sqrt{2}\tau \mathbf{U}} (R_+ - R_-) \Delta x [\hat{J} \circ u_s(x-s)]. \end{aligned} \tag{S23}$$

4.3 Calculation of speed

Combining Eq. (S23) with the dynamical equation of P-EN neuron (Eq. (S14), we will find that the above dynamics is proportional to the difference between r_{v_+} and r_{v_-} :

$$\tau \dot{u}_s(x-s_t) \propto w_{s,s\pm} w_{s\pm,s} (r_{v_+} - r_{v_-}) \hat{J} \circ u_s(x-s_t). \tag{S24}$$

In order to obtain an analytical expression of the above equation, we can formally calculating the integral of $r_{v_{\pm}}$ (Eq. (S20)):

$$\begin{aligned}
r_{v_+} &= \rho^2 w_{\theta,s}^2 \int_x (r_s(x-s-\Delta\theta) + r_h(x-h))^2 dx \\
&= \rho^2 w_{\theta,s}^2 \int_x \left(\sum_{\omega} R_{\omega}(s+\Delta\theta) f_{\omega}(x) + \sum_{\omega} R_{\omega}(h) f_{\omega}(x) \right)^2 dx \\
&= \rho^2 w_{\theta,s}^2 \int_x \left(\sum_{\omega_1} R_{\omega_1}(s+\Delta\theta) f_{\omega_1}(x) \sum_{\omega_2} R_{\omega_2}(s+\Delta\theta) f_{\omega_2}(x) \right. \\
&\quad \left. + \sum_{\omega_1} R_{\omega_1}(h) f_{\omega_1}(h) \sum_{\omega_2} R_{\omega_2}(h) f_{\omega_2}(h) \right. \\
&\quad \left. + 2 \sum_{\omega_1} R_{\omega_1}(s+\Delta\theta) f_{\omega_1}(x) \sum_{\omega_2} R_{\omega_2}(h) f_{\omega_2}(h) \right) dx \tag{S25a} \\
&= \rho^2 w_{\theta,s}^2 \sum_{\omega_1, \omega_2} \left\{ ([R_{\omega_1}(0)\rho_{\omega_1}(s+\Delta\theta)][R_{\omega_2}(0)\rho_{\omega_2}(s+\Delta\theta)]^\dagger \right. \\
&\quad \left. + [R_{\omega_1}(0)\rho_{\omega_1}(h)][R_{\omega_2}(0)\rho_{\omega_2}(h)]^\dagger + 2[R_{\omega_1}(0)\rho_{\omega_1}(s+\Delta\theta)][R_{\omega_2}(0)\rho_{\omega_2}(h)]^\dagger \right\} \\
&\quad \cdot \left[\int_x f_{\omega_1}(x) f_{\omega_2}^\dagger(x) dx \right] \\
&= \rho^2 w_{\theta,s}^2 \sum_{\omega} (2R_{\omega}(0)R_{-\omega}(0) + R_{\omega}(0)R_{-\omega}(0)\rho_{\omega}(s+\Delta\theta-h)), \\
r_{v_-} &= \rho^2 w_{\theta,s}^2 \sum_{\omega} (2R_{\omega}(0)R_{-\omega}(0) + R_{\omega}(0)R_{-\omega}(0)\rho_{\omega}(s-\Delta\theta-h)), \tag{S25b}
\end{aligned}$$

where we have utilized $\rho_{\omega}(s)^\dagger = \rho_{-\omega}(s) = \rho_{\omega}(-s)$ and $R_{\omega}(0)^\dagger = R_{-\omega}(0)$. We used R_{ω} to denote the representation of r_s and r_h under the basis function f_{ω} . Then the firing rate difference can be obtained by:

$$\begin{aligned}
r_{v_+} - r_{v_-} &= \rho^2 w_{\theta,s}^2 \sum_{\omega} \|R_{\omega}(0)\|^2 (\rho_{\omega}(s+\Delta\theta-h) - \rho_{\omega}(s-\Delta\theta-h)) \\
&= \rho^2 w_{\theta,s}^2 \sum_{\omega} \|R_{\omega}(0)\|^2 \sin(\omega\Delta\theta) \sin[\omega(h-s)], \tag{S26}
\end{aligned}$$

where we used the expression of ρ_{ω} in Eq. (S10) and Euler equation $e^{i\theta} = \cos\theta + i\sin\theta$.

If we substitute the stationary states of the network in Eq. (S16)(S18)(S19)(S20), it will yield a specific expression:

$$\begin{aligned}
(r_{v_+} - r_{v_-}) &= \sqrt{\pi} a \rho^2 w_{\theta,s}^2 R_s^2 \cdot e^{-\frac{(s-h)^2}{4a^2}} \cdot e^{-\frac{\Delta\theta^2}{4a^2}} \cdot (e^{-\frac{\Delta\theta(s-h)}{2a^2}} - e^{-\frac{\Delta\theta(s-h)}{2a^2}}), \tag{S27} \\
&\approx^{\Delta\theta \rightarrow 0} \sqrt{\pi} a^{-1} \rho^2 w_{\theta,s}^2 R_s^2 \cdot \Delta\theta(h-s) e^{-\frac{(s-h)^2}{4a^2}} \propto v.
\end{aligned}$$

5 Network simulation details

The typical set of network parameters can be found in Table S4. In the text below we briefly explain the reasoning of network parameter setting. We simulated a continuous attractor network that consist of $N = 180$ neurons which were uniformly distributed in the space of s . To avoid boundary effect, we considered s to be a periodic variable in the network simulation and was in the range of $[-\pi, \pi)$. The periodic stimulus s doesn't affect substantially our theoretical results from the 1d rotation group which acts on an infinite region, as long as the connection width, i.e., a , is much smaller than the width of the direction space.

The code was written with *BrainPy* ([6]) and was simulated on MacBookPro laptop which has a 10-core M1 CPU and 32GB RAM.

To scale the connection strength in the network model, we set the connection strength relative to the critical strength, w_c , under which a CAN can hold a persistent (non-zero) population response by

itself. It can be calculated that

$$w_c = 2\sqrt{2}(2\pi)^{1/4}\sqrt{k\sigma/\rho} \approx 0.896. \quad (\text{S28})$$

In the network simulation, the instantaneous sensory representation s_t is linearly read out from the sensory neurons' response, $r_s(x, t)$ by using the population vector ([7]), i.e.,

$$s_t = \text{Angle} \left[\sum_j r(x_j, t)e^{ix_j} \right]. \quad (\text{S29})$$

where $i = \sqrt{-1}$ is the pure imaginary number.

Symbol	Description	Typical values (range)
N	Number of one neuron group	180
ρ	Neuronal density in the stimulus space	$N/2\pi$
a	Tuning width	$2\pi/9$
k	Global inhibition strength	5×10^{-4}
τ	Synaptic decaying time constant	1 (dimensionless)
dt	Time step in numerical simulation	0.1τ
w_0	The peak recurrent weight in the E-PG and FC2	$1.1w_c$
w_c	The critical recurrent weight of non-zero sustained response	0.896 (Eq. S28)
$w_{s\pm, s}$	The recurrent weight from E-PG to P-EN neurons	0.2
$w_{s, s\pm}$	The recurrent weight from P-EN to E-PG neurons	1
$w_{\theta, s}$	The feedforward weight from E-PG to PFL3 neurons	1
$w_{v, \theta}$	The feedforward weight from PFL3 to DN neurons	$0.2/N$
Δx	The connection shift from P-EN neurons to E-PG neurons	$\pi/18$
$\Delta \theta$	The connection shift from E-PG neurons to PFL3 neurons	$3\pi/8$
g_0	The baseline activity of speed neurons	0

Table S4: Typical parameters of the network model

6 Extension to 2D scenarios

We simulated our model under 2D scenario as an extension to our model. In an allocentric representation with an x-y coordinate, where the sensory representation will be shifted along four directions, i.e., $\pm x$ and $\pm y$, and then we expect to have four populations of neurons (the 2D counterparts of PFL3 left and PFL3 right). Then the pooling operation will be similar to the 1D case while we will have four output neurons (2D counterparts of DN left and DN right) which will generate actions along four directions in the 2D plane. The similar shifted connections were also considered in [8], while it didn't study the motion planning circuit. Specifically, we represented the location and goal using two separate 2-dimensional neuronal populations, each containing 128×128 neurons, referred to as EPG and FC2 neurons, as in the main text. The rest of the model remains largely unchanged, except for including four groups (north-, south-, east- and west-ward) of PENS, PFL3s, and DNs instead of two (left- and right-ward). Once we initiate the self-location and goal location, the EPG equivalent neurons generate an activity bump at the self-location, which then moves toward and ultimately terminates at the goal location. By plotting the DN neuron activities, we can verify that the movement velocity along the north-south axis is proportional to the firing rate difference between DN-North and DN-South neurons, and similarly for the east-west axis. The results are shown in Fig.S1

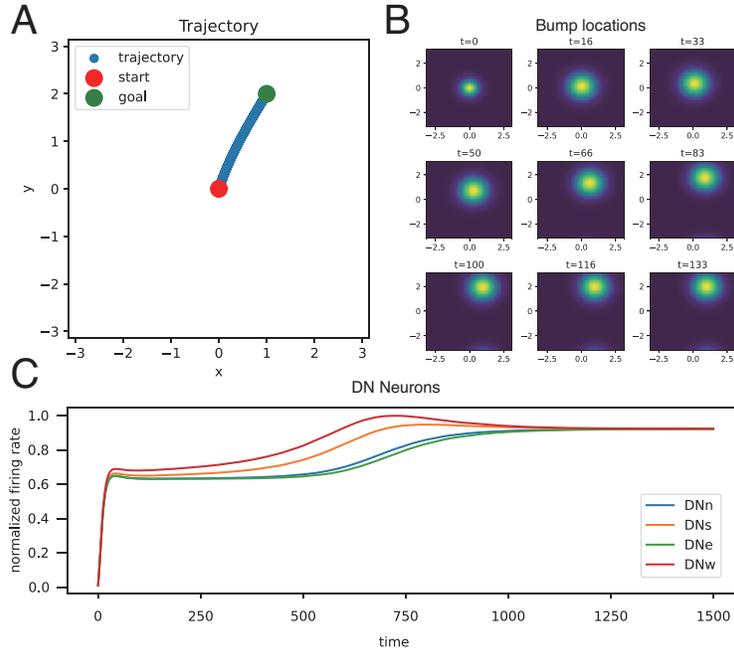


Figure S1: A. Encoded trajectory from start position to goal position. B. Neuron activity bumps at different time steps. C. DN neuron activities.

References

- [1] David J Griffiths and Darrell F Schroeter. *Introduction to quantum mechanics*. Cambridge university press, 2018.
- [2] Wenhao Zhang, Ying Nian Wu, and Si Wu. Translation-equivariant representation in recurrent networks with a continuous manifold of attractors. *Advances in Neural Information Processing Systems*, 35:15770–15783, 2022.
- [3] Sung Soo Kim, Hervé Rouault, Shaul Druckmann, and Vivek Jayaraman. Ring attractor dynamics in the drosophila central brain. *Science*, 356(6340):849–853, 2017.
- [4] Matteo Carandini and David J Heeger. Normalization as a canonical neural computation. *Nature Reviews Neuroscience*, 13(1):51–62, 2012.
- [5] C. C. Alan Fung, K. Y. Michael Wong, and Si Wu. A moving bump in a continuous manifold: A comprehensive study of the tracking dynamics of continuous attractor neural networks. *Neural Computation*, 22(3):752–792, 2010.
- [6] Chaoming Wang, Tianqiu Zhang, Xiaoyu Chen, Sichao He, Shangyang Li, and Si Wu. Brainpy, a flexible, integrative, efficient, and extensible framework for general-purpose brain dynamics programming. *Elife*, 12:e86365, 2023.
- [7] Apostolos P Georgopoulos, Andrew B Schwartz, and Ronald E Kettner. Neuronal population coding of movement direction. *Science*, 233(4771):1416–1419, 1986.
- [8] Yoram Burak and Ila R Fiete. Accurate path integration in continuous attractor network models of grid cells. *PLoS computational biology*, 5(2):e1000291, 2009.