Supporting Text: Bayesian dynamics of image stabilization in the brain

Yoram Burak,[1] Uri Rokni,[1] Markus Meister,[1,2] and Haim Sompolinsky[3,4]

[1] *Center for Brain Science, Harvard University, Cambridge, MA 02138, USA*

[2] *Department of Molecular and Cellular Biology, Harvard University, Cambridge, Massachusetts 02138, USA*

[3] *Interdisciplinary Center for Neural Computation, Hebrew University, Jerusalem, 91904, Israel*

[4] *Center for Brain Science, Harvard University, Cambridge, MA 02138*

Contents

I. THE FACTORIZED BAYESIAN DECODER

In this section we consider the case where RGC response is instantaneous. We first define the problem mathematically (part A). We then derive the ideal Bayesian decoder (part B) and the factorized Bayesian decoder (part C). The implementation of the factorized decoder to the case of binary pixels yields equations (2)–(5) of the main text (part D, Eqs. (S21)–(S23) and (S25)). In simulations we used a version of the decoder in which time is discretized, as described in part E. Finally, we present a comparison between the factorized decoder and the ideal decoder for small 1-dimensional images (part F).

A. The problem

We assume a 2D image of n x n pixels with fixed intensities $\{s_i\}$. The image shifts with time with a trajectory $x\left(t\right)$. For simplicity, we assume that changes in $x(t)$ occur only at discrete times, separated by fixed intervals of duration Δt. Eventually we will take the limit of $\Delta t \to 0$. The trajectory is drawn by a Markov process with a transition matrix T:

$$P\left[x\left(t+\Delta t\right) \mid x\left(t\right)\right] = \delta_{x(t+\Delta t),x(t)} + T\left[x\left(t+\Delta t\right) \mid x\left(t\right)\right]\Delta t \tag{S1}$$

where

$$\sum_x T\left(x \mid x'\right) = 0 \tag{S2}$$

and the summation is over all N possible values of x. The derivations below are valid for any choice of the transition matrix. To describe two-dimensional diffusion we choose

$$T(x \mid x') = \begin{cases} D & , \quad |x-x'| = 1 \\ -4D & , \quad x = x' \\ 0 & , \quad |x-x'| > 1 \end{cases} \tag{S3}$$

We define the initial shift to be $x\left(0\right) = 0$. There is a set of n x n neurons, each observing a single pixel. The response $r_i\left(t\right)$ of neuron i is an inhomogeneous Poisson process, whose instantaneous

rate depends on the incident image pixel $s_{i-x(t)}$

$$P\left[r_i\left(t\right) \mid s_{i-x}\right] = \delta_{r_i(t),0}\left[1-\lambda\left(s_{i-x(t)}\right)\Delta t\right] + \delta_{r_i(t),1}\lambda\left(s_{i-x(t)}\right)\Delta t \tag{S4}$$

where we assumed that Δt is sufficiently small that in each interval the neuron emits at most one spike. The function λ describes the relation between instantaneous firing rate and incident pixel intensity. We assume that given the stimulus, the neurons are uncorrelated. In the following, we denote the vectors of pixel intensities and neural responses by omitting the index i, i.e. by s and $r\left(t\right)$, respectively. The problem is to infer, at any time t, the fixed vector of pixel intensities s, given the past spike trains $r\left(0\right),...,r\left(t\right)$. Specifically, we are interested in the continuous time limit, i.e. $\Delta t \to 0$.

B. The Bayesian filter

The optimal Bayesian estimation requires a computation of the posterior distribution $P\left[s, x\left(t\right) \mid r\left(0\right),...,r\left(t\right)\right]$, which we denote more shortly by $P\left(s,x;t\right)$. This distribution can be marginalized over $x\left(t\right)$ to obtain the the posterior of s. $P\left(s,x;t\right)$ can be computed iteratively by

$$P\left(s,x;t\right) = \frac{1}{Z}\prod_{i=1}^{N} P\left[r_i\left(t\right) \mid s_{i-x}\right] \sum_{x'=1}^{N} P\left(x \mid x'\right) P\left(s,x';t-\Delta t\right) \tag{S5}$$

where Z is a normalization constant. We substitute Eqs. (S4) and (S1) in Eq. (S5), and take $\Delta t \to 0$. At times when there are no spikes $P\left(s,x;t\right)$ evolves smoothly according to

$$\frac{\partial P\left(s,x;t\right)}{\partial t} = \left[R_{\text{tot}}\left(t\right) - \sum_{i=1}^{N}\lambda\left(s_i\right)\right] P\left(s,x;t\right) + \sum_{x'=1}^{N} T\left(x \mid x'\right) P\left(s,x';t\right) \tag{S6}$$

where $R_{\text{tot}}\left(t\right)$ is the total expected firing rate,

$$R_{\text{tot}}\left(t\right) = \sum_{s,x} P\left(s,x;t\right) \sum_{i=1}^{N}\lambda\left(s_i\right) \tag{S7}$$

When there is a spike at neuron i at time t_i, $P\left(s,x;t\right)$ evolves discontinuously according to

$$P\left(s,x;t_{i+}\right) = \frac{1}{R_i\left(t_{i-}\right)}\lambda\left(s_{i-x}\right) P\left(s,x;t_{i-}\right) \tag{S8}$$

where $R_i\left(t\right)$ is the firing rate expected at neuron i

$$R_i\left(t\right) = \sum_{s,x}\lambda\left(s_{i-x}\right) P\left(s,x;t\right) \tag{S9}$$

C. The factorized approximation

In the factorized approximation we approximate the posterior as a product of probabilities for position and for each pixel,

$$P\left(s,x;t\right)\cong P\left(x;t\right)\prod_{i=1}^{N}P_{i}\left(s_{i};t\right) \tag{S10}$$

To update $P(x;t)$ and $P_i(s_i;t)$ we first use Eq. (S5). We then recast the updated $p(s,x;t)$ into the factorized form by marginalizing it on x and s_i to obtain the updated $P(x;t)$ and $P_i(s_i;t)$, respectively. This procedure minimizes the Kullback-Leibler divergence between $p(s,x;t)$ and the updated factorized approximation. We insert Eq. (S10) into Eq. (S6), and obtain the dynamics of the factorized posterior when there are no spikes

$$\frac{\partial P\left(x;t\right)}{\partial t}=\sum_{x'=1}^{N}T\left(x\mid x'\right)P\left(x';t\right) \tag{S11}$$

$$\frac{\partial P_{i}\left(s_{i},t\right)}{\partial t}=\left[\rho_{i}\left(t\right)-\lambda\left(s_{i}\right)\right]P_{i}\left(s_{i},t\right) \tag{S12}$$

where $\rho_i(t)$ is the firing rate expected by the neuron which at time t observes s_i

$$\rho_{i}\left(t\right)=\sum_{s_{i}}\lambda\left(s_{i}\right)P_{i}\left(s_{i},t\right) \tag{S13}$$

Similarly, we insert Eq. (S10) into Eq. (S8) to obtain the discontinuous change in the factorized posterior when neuron i spikes

$$P\left(x;t_{i+}\right)=\frac{1}{R_{i}\left(t_{i-}\right)}\rho_{i-x}\left(t_{i-}\right)P\left(x;t_{i-}\right) \tag{S14}$$

where

$$R_{i}\left(t\right)=\sum_{x}\rho_{i-x}\left(t\right)P\left(x;t\right) \tag{S15}$$

and

$$\begin{aligned}P_{k}\left(s_{k},t_{i+}\right) &= P_{k}\left(s_{k},t_{i-}\right) \\ &\times \frac{1}{R_{i}\left(t_{i-}\right)}\left[\lambda\left(s_{k}\right)P\left(x=i-k;t_{i-}\right)+\sum_{x\neq i-k}\rho_{i-x}\left(t_{i-}\right)P\left(x;t_{i-}\right)\right]\end{aligned} \tag{S16}$$

which can be rewritten as

$$P_{k}\left(s_{k},t_{i+}\right)=\left\{1+\frac{\left[\lambda\left(s_{k}\right)-\rho_{k}\left(t\right)\right]P\left(x=i-k;t_{i-}\right)}{R_{i}\left(t_{i-}\right)}\right\}P_{k}\left(s_{k},t_{i-}\right) \tag{S17}$$

or, using Eq. (S14), as

$$P_{k}\left(s_{k},t_{i+}\right)=\left\{1+\frac{\left[\lambda\left(s_{k}\right)-\rho_{k}\left(t\right)\right]P\left(x=i-k;t_{i+}\right)}{\rho_{k}(t_{i-})}\right\}P_{k}\left(s_{k},t_{i-}\right) \tag{S18}$$

D. The binary s_i case

When s_i are binary variables, we can describe $P_i(s_i,t)$ by

$$m_i(t) = P_i(s_i = 1, t) \tag{S19}$$

In this case, we can write the firing rate function $\lambda(s_i)$ as

$$\lambda(s_i) = \lambda_0 + \Delta\lambda s_i \tag{S20}$$

where λ_0 is the firing rate for $s_i = 0$, λ_1 is the firing rate for $s_i = 1$, and $\Delta\lambda = \lambda_1 - \lambda_0$. The evolution of $P(x,t)$ in the absence of spikes is unchanged from Eq. (S11)

$$\frac{\partial P(x;t)}{\partial t} = \sum_{x'=1}^{N} T\left(x \mid x'\right) P\left(x';t\right). \tag{S21}$$

This is equation (2) in the main text, where the operator $D\nabla^2$ in discrete space represents the transition probabilities $T(x|x')$ of Eq. (S3). The evolution of $m_i(t)$ in the absence of spikes is derived from Eq. (S12) and yields Eq. (3),

$$\frac{\partial m_i(t)}{\partial t} = -\Delta\lambda\left[1 - m_i(t)\right] m_i(t) \tag{S22}$$

When neuron i fires a spike, $P(x,t)$ is updated by [see Eq. (S14)]

$$P(x;t_{i+}) = \frac{1}{R_i(t_{i-})}\lambda\left(m_{i-x}(t_{i-})\right) P(x;t_{i-}) \tag{S23}$$

where

$$R_i(t) = \lambda_0 + \Delta\lambda \sum_x m_{i-x} P(x;t) \tag{S24}$$

[Eq. (4)] and from Eq. (S18) we find that $m_k(t)$ is updated by

$$m_k(t_{i+}) = \left\{1 + \frac{\Delta\lambda P(x = i-k; t_{i+})\left[1 - m_k(t_{i-})\right]}{\lambda\left(m_k(t_{i-})\right)}\right\} m_k(t_{i-}) \tag{S25}$$

This can be written in the form of Eq. (5) by defining the nonlinear transfer function $\phi(m) = \Delta\lambda m(1-m)/(\lambda_0 + \Delta\lambda m)$.

E. Large Δt

In the above derivation we took the limit $\Delta t \to 0$. This limit is relevant for biological implementation, and it simplifies the equations. However, for the purpose of computer implementation

it requires taking a very small time step, such that at any given time step the probability that any neuron fires a spike is small. This constraint becomes more restrictive as the number of neurons in the model increase. To allow faster computer implementation, here we derive the algorithm without assuming a small time step. When $\lambda_i \Delta t \ll 1$ the implementation of this algorithm should be identical to the original algorithm up to small truncation and roundoff errors.

Our starting point is the equation of the full Bayesian filter (Eq. S5) which applies for non-vanishing Δt. Next, we apply the mean field approximation (see section on mean field approximation above) and derive the update rules:

$$P\left(x;t\right)=\frac{1}{Z}P'\left(x;t\right)\prod_{i}\sum_{s_i}P\left[r_{i+x}\left(t\right)\mid s_i\right]P\left(s_i;t-\Delta t\right) \tag{S26}$$

where

$$P'\left(x;t\right)=\sum_{x'}P\left(x\mid x'\right)P\left(x';t-\Delta t\right) \tag{S27}$$

and

$$P\left(s_i;t\right)=\sum_{x}P\left(x;t\right)\frac{P\left[r_{i+x}\left(t\right)\mid s_i\right]P\left(s_i;t-\Delta t\right)}{\sum_{s_i'}P\left[r_{i+x}\left(t\right)\mid s_i'\right]P\left(s_i';t-\Delta t\right)} \tag{S28}$$

Inserting (S4) into (S26) we obtain the following update rule for $P\left(x;t\right)$

$$P\left(x;t\right)=\frac{1}{Z}P'\left(x;t\right)\exp\left[\sum_{i}r_{i+x}\left(t\right)\log\frac{\rho_i\left(t-\Delta t\right)\Delta t}{1-\rho_i\left(t-\Delta t\right)\Delta t}\right] \tag{S29}$$

where $\rho_i\left(t\right)$ is defined in (S13). Similarly, by inserting (S4) into (S28) we obtain the update rule for $P\left(s_i;t\right)$

$$\begin{aligned}P\left(s_i;t\right) &= P\left(s_i;t-\Delta t\right) \\ &\times \sum_{x}P\left(x;t\right)\left[r_{i+x}\left(t\right)\frac{\lambda\left(s_i\right)}{\rho_i\left(t-\Delta t\right)}+\left(1-r_{i+x}\left(t\right)\right)\frac{1-\lambda\left(s_i\right)\Delta t}{1-\rho_i\left(t-\Delta t\right)\Delta t}\right]\end{aligned} \tag{S30}$$

Keeping terms up to first order in Δt yields

$$\begin{aligned}P\left(s_i;t\right) &= P\left(s_i;t-\Delta t\right)\times\left\{1-\left[\lambda(s_i)-\rho_i(t-\Delta t)\right]\Delta t\right. \\ &+ \left.\sum_{x}P\left(x;t\right)r_{i+x}\left(t\right)\frac{\lambda\left(s_i\right)-\rho_i(t-\Delta t)}{\rho_i\left(t-\Delta t\right)}\right\}\end{aligned}$$

from which the continuous time limit of Eqs. (S12) and (S18) follows by taking the limit $\Delta t\to 0$.

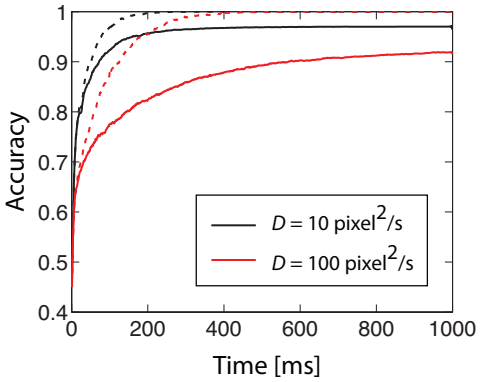

FIG. S1: Comparison of the factorized decoder (full traces) with the ideal Bayesian decoder (dashed traces) for small, one-dimensional images containing 10 pixels. Accuracy (fraction of correctly estimated pixels), averaged over 1000 presentations of random images, is shown as a function of time for two values of the diffusion coefficient (legend).

F. Comparison with the ideal filter

For small one-dimensional images, we can compare the performance of the factorized decoder with that of the ideal Bayesian decoder. Such a comparison is shown for images containing 10 pixels in Fig. S1. As expected, the factorized decoder infers the pixels more slowly than the ideal decoder. Further, the plots demonstrate that after a long presentation of the image, the factorized decoder may converge on an imperfect estimate of the image. In contrast, the ideal decoder infers all the pixels correctly if given enough time.

II. PERFORMANCE AS A FUNCTION OF IMAGE SIZE

A. Accuracy of tracking for a known image

We assume that the image is known and evaluate how well the decoder estimates the position of the image at steady state. The pixels are assumed to be uncorrelated with an equal distribution of *on* and *off* values. This part of the calculation applies to the optimal Bayesian decoder as well as to the factorized one, since both of them have the same dynamics when the image is known.

Specifically, we consider the following quantity

$$\log P(\Delta\mathbf{x}) \equiv \langle \log p(\mathbf{x}(t) + \Delta\mathbf{x}) \rangle_{\mathbf{x}(t),r} \tag{S31}$$

where $\mathbf{x}(t)$ is the true position of the drifting image at time t. The averaging is performed over all possible trajectories $\mathbf{x}(t)$ and over the ganglion cell firing patterns.

For convenience we work with a one-dimensional image containing n pixels and assume that image drift occurs in discrete steps: At each Δt the image moves one step to the left, with probability $D\Delta t$, or to the right, with the same probability. Ultimately we will be interested in the limit of small $\Delta t \to 0$. At this stage we only require that Δt is sufficiently small such that any individual neuron is unlikely to produce two spikes in a single time interval,

$$\lambda_0 \Delta t \ll 1 \quad , \quad \lambda_1 \Delta t \ll 1. \tag{S32}$$

In addition we assume that $D\Delta t \ll 1$.

The update of $p(x,t)$ is:

$$p(x, t+\Delta t) = \frac{1}{Z} p(r \mid x) \left\{ (1 - 2D\Delta t) p(x,t) + D\Delta t \left[p(x+1,t) + p(x-1,t) \right] \right\} \tag{S33}$$

To estimate the steady state behavior of $p\,(x,t)$ we make use of the following approximations. First, we approximate $\log p(r \mid x)$ by replacing it with its average over r. In the limit of large n this quantity is the same for all values of x except for the true position of the image:

$$\langle \log p(r \mid x) \rangle = \begin{cases} c_0 & x = x(t) \\ c_0 - n d_{\mathrm{KL}} \Delta t & x \neq x(t) \end{cases} \tag{S34}$$

where $d_{\mathrm{KL}} \Delta t$ is the Kullback-Leibler distance per pixel between the distribution of firing patterns given the correct image, and the distribution of firing patterns given a shifted version of the image,

$$d_{\mathrm{KL}} = \frac{1}{4} (\lambda_1 - \lambda_0) \log \frac{\lambda_1}{\lambda_0}. \tag{S35}$$

and where c_0 is independent of x. In deriving this expression we made use of the assumption that the pixels are drawn independently from a binary distribution. We thus replace the dynamics of Eq. (S33) by:

$$\begin{aligned} \log p(x, t+\Delta t) \;&=\; -\log Z + n d_{\mathrm{KL}} \Delta t \delta_{\mathbf{x},\mathbf{x}(t)} \\ &+\; \log \left\{ (1 - 2D\Delta t) p(x,t) + D\Delta t \left[p(x+1,t) + p(x-1,t) \right] \right\} \end{aligned} \tag{S36}$$

where Z is determined from the normalization requirement on p. Here the spiking is no longer considered as a stochastic process: The combined influence of all spikes on the Bayesian estimate is encapsulated deterministically in the second term on right hand side of Eq. (S36). In the limit $\Delta t \to 0$,

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} p(x) \;&=\; n d_{\mathrm{KL}} \delta \left[x - x(t) \right] p(x) + D \left[p(x+1) + p(x-1) - 2p(x) \right] \\ &-\; n d_{\mathrm{KL}} p(x^*) p(x) \end{aligned} \tag{S37}$$

As a further simplification we consider a particular trajectory, where the image remains at $x(t) = 0$ at all times. In other words, the image is static, but the estimator is tuned to a randomly drifting image with statistics characterized by the diffusion coefficient D. At steady state we then have,

$$D\left[p(x+1) + p(x-1) - 2p(x)\right] + nd_{\mathrm{KL}}p(0)\left[\delta_{x,0} - p(x)\right] = 0 \tag{S38}$$

This equation describes the steady state distribution of particles which are created by a point source at $x = 0$ and undergo one-dimensional random diffusion. The particles are created at a rate $nd_K Lp(0)$ and are randomly removed, independent of their position, at the same rate such that their total number remains constant in time. The parameter $p(0)$ must be obtained self consistently from the requirement that

$$\sum_{-\infty}^{\infty} p(x) = 1. \tag{S39}$$

By dividing this equation by D we see that the solution depends only on the ratio nd_{KL}/D. Therefore doubling the diffusion coefficient has the same effect as reducing the number of pixels by a factor of two. For two-dimensional images, however, n is replaced everywhere by n^2. Hence performance depends on the diffusion coefficient, scaled by the number of pixels. This is the main result of this section. We proceed to analyze the form of the solution to Eq. (S38) in the 1-d case. The solution to Eq. (S38) is found by assuming the ansatz

$$P(x) \propto \exp(-\alpha|x|). \tag{S40}$$

Inserting this expression in Eq (S38), we get

$$2\left[\cosh(\alpha) - 1\right] = \frac{nd_{\mathrm{KL}}p(0)}{D} \tag{S41}$$

From the normalization requirement (S39), $p(0) = \mathrm{tgh}(\alpha/2)$. Therefore

$$\alpha = \sinh^{-1}\left(\frac{nd_{\mathrm{KL}}}{2D}\right) \tag{S42}$$

Fig. S2 A shows a measurement of $\log P(\Delta x)$ (Eq. S31) from a long simulation of the factorized decoder tracking a known one-dimensional image containing 1000 pixels. These results compare well with the approximate analytical expression [Eqs. (S40),(S42)].

B. Accuracy of tracking for an unknown image

Here we consider the opposite limit where the image is completely unknown to the decoder. This is the situation when the image is first presented to the decoder. We consider an approximate

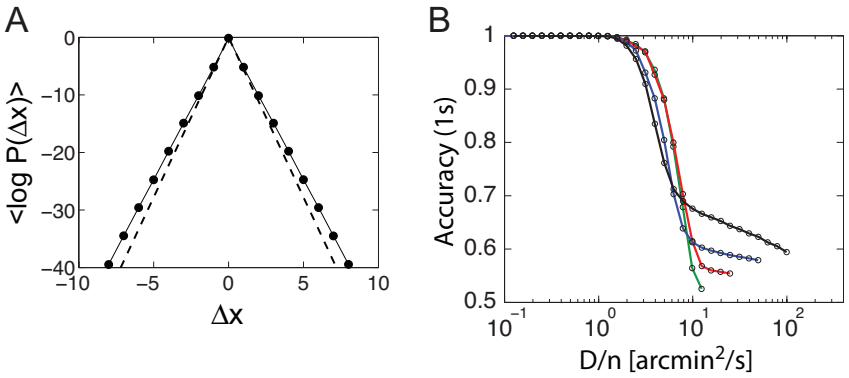

FIG. S2: **A** For large images the probability of position inferred by the decoder is sharply distributed around the true position. Symbols and solid trace show $\log P(\Delta x)$ [Eq (S31)] as a function of Δx, for a one-dimensional image containing 1000 pixels. These results are compared with the analytical estimate, Eqs. (S40) and (S42) (dashed trace). Parameters: $\lambda_{0,1} = 10/100\,\mathrm{Hz}$, $D = 200$ pixels^2/s. **B** Accuracy of decoded pixels after a long presentation (1 s) becomes roughly independent of the number of pixels n^2 when plotted as a function of D/n. Data is shown for $n = 10$ (black), 20 (blue), 40 (red), and 80 (green), which corresponds to 5 x 5, 10 x 10, 20 x 20, and 40 x 40 arcmin. All parameters are as in Fig. 3B.

decoder which assumes that the image moves only at discrete times, separated by regular intervals of duration Δt. We choose

$$\Delta t = \frac{1}{4D} \tag{S43}$$

because over this time scale it is reasonable to assume that the image is static. In order to track the position of the image accurately, the decoder must be able to infer the relative position of the image in the second interval, compared to its position during the first interval, which we denote by Δx^*. The inference is based on the RGC spike counts observed during the first and second intervals, which we denote by r and r'. In order for the decoder to successfully distinguish between a shift Δx and the true shift Δx^*, the following quantity must be large compared to unity for any $\Delta x \neq \Delta x^*$:

$$d_{\mathrm{KL}}\left[p(r, r' \mid \Delta x) \parallel p(r, r' \mid \Delta x^*)\right] \tag{S44}$$

This is the Kullback-Leibler divergence between the probability distribution of spike counts given Δx and their distribution given Δx^*. Assuming that pixels are statistically independent, and using the instantaneous response property of the neurons, we obtain:

$$d_{\mathrm{KL}}\left[p(r, r' \mid \Delta x) \parallel p(r, r' \mid \Delta x^*)\right] = n^2 \begin{cases} 0 & , \quad \Delta x = \Delta x^* \\ \hat{d}_{\mathrm{KL}} & , \quad \Delta x \neq \Delta x^* \end{cases} \tag{S45}$$

where

$$\hat{d}_{\mathrm{KL}} = \sum_{r_1=0}^{\infty}\sum_{r_2=0}^{\infty}\left[\sum_{s} p(s)p(r_1 \mid s)p(r_2 \mid s) - \sum_{s_1,s_2} p(s_1)p(s_2)p(r_1 \mid s_1)p(r_2 \mid s_2)\right] \times \log\left[\sum_{s'} p(s')p(r_1 \mid s')p(r_2 \mid s')\right] \tag{S46}$$

In this expression $p(s)$ is the prior probability for pixels intensities which, in the following, we assume is uniform, and the spike statistics are Poisson:

$$p(r_i \mid s) = \frac{\mathrm{e}^{-\lambda(s)\Delta t}\left[\lambda(s)\Delta t\right]^{r_i}}{r_i!} \tag{S47}$$

For binary images we can write $\hat{d}_{\mathrm{KL}}$ as

$$\hat{d}_{\mathrm{KL}} = \frac{1}{4}\left[J_{0,0} + J_{1,1} - 2J_{0,1}\right] \tag{S48}$$

where

$$J_{s_1,s_2} \equiv \sum_{r_1=0}^{\infty}\sum_{r_2=0}^{\infty} p(r_1 \mid s_1)p(r_2 \mid s_2)\log\left[\frac{p(r_1 \mid 0)p(r_2 \mid 0) + p(r_1 \mid 1)p(r_2 \mid 1)}{2}\right] \tag{S49}$$

The decoder estimates whether the image has moved by correlating the spike trains in the two time intervals. In the limit of small Δt these firing patterns are sparse, and we expect the information coming from the correlations to scale as Δt^2. A precise expansion in powers of Δt yields

$$\hat{d}_{\mathrm{KL}} = \alpha(\lambda_0, \lambda_1)\Delta t^2 + \ldots \tag{S50}$$

where

$$\alpha(\lambda_0, \lambda_1) = \frac{1}{4}\log\left[\frac{2(\lambda_0^2 + \lambda_1^2)}{(\lambda_0 + \lambda_1)^2}\right](\lambda_1 - \lambda_0)^2. \tag{S51}$$

This expression is valid if $\lambda_{0,1}\Delta t \lesssim 1$. The decoder can track the image accurately if $n^2\hat{d}_{\mathrm{KL}} \gg 1$. We thus obtain the requirement that

$$D \ll \frac{n\alpha^{1/2}}{4} \tag{S52}$$

This result suggests that the value of D, beyond which performance starts to degrade, should scale as n, the square root of the number of pixels, rather than by n^2 as suggested by the tracking of a known image (Sec. II A). Indeed, when the accuracy of pixels inferred by the factorized decoder is plotted as a function of D/n, the traces are seen to be roughly independent of D/n, Fig. S2 B.

III. TEMPORAL RESPONSE OF RETINAL GANGLION CELLS

We consider the situation where a temporal filter is involved in the response of RGCs. As before, we assume that each RGC fires as an inhomogeneous Poisson process but instead of Eq. (S4) we have

$$P\left[r_i\left(t\right) \mid s, X\right] = \left(1 - r_i\right)\left[1 - \lambda_i\left(s, X\right)\Delta t\right] + r_i \lambda_i\left(s, X\right)\Delta t \tag{S53}$$

where the rate $\lambda_i(s, X)$ is given by

$$\lambda_i(s, X) = \lambda_0 + \Delta\lambda \int d\tau f(\tau) s_{i-x(t-\tau)}. \tag{S54}$$

Here $f(\tau)$ is the temporal filter and we adopt the notation that X with a capital letter denotes a full trajectory, and $x(t)$ denotes the image position at a particular time t.

A. Ideal Bayesian Filter

We denote by $P(s, X; t)$ the posterior probability of the image s and the trajectory X given all the spikes emitted from time 0 up to time t. Between spikes,

$$\frac{\partial P(s, X; t)}{\partial t} = \left(\sum_i \lambda_i(s, X) - R_i\right) P(S, \mathbf{x}; t) + \sum_{X'} T\left(X \mid X'\right) P(s, Y; t) \tag{S55}$$

where

$$R_i = \sum_{X'} \sum_{s'} \lambda_i(s', X') P(s', X'; t). \tag{S56}$$

is the expected firing rate of neuron i. When neuron i spikes at time $t = t_i$,

$$P(s, X; t_i^+) = \frac{\lambda_i(s, X) P(s, X; t_i^-)}{Z} \tag{S57}$$

where Z is a normalization factor, chosen such that the sum

$$\sum_{s'} \sum_{X'} P(s', X'; t_i^+) = 1. \tag{S58}$$

B. Factorized approximation

We can apply the factorized approximation while keeping track of probabilities for full trajectories instead of only the current position:

$$P(s, X; t) \simeq \prod_i P_i(s_i; t) P(X; t). \tag{S59}$$

The update rules for $p_i(s_i;t)$ and $P(X;t)$ are obtained from Eqs (S55), (S57) by marginalizing over s_i and over the trajectory.

We begin with the updates between spikes. Evaluating the update rule for $P(X;t)$ involves averaging of the total firing rate over s given X. While this in general is complicated, it is simple in the linear case,

$$\sum_s P(s;t)\sum_i \lambda_i(s,X) = \lambda_0 + \Delta\lambda \int \mathrm{d}\tau f(\tau)\sum_i \sum_s P(s;t)s_{i-x(t-\tau)} \tag{S60}$$

where we used the notation:

$$P(s;t) = \prod_i P_i(s_i;t) \tag{S61}$$

The last term reduces to

$$\sum_i \sum_s P(s;t)s_{i-x(t-\tau)} = \sum_i m_{i-x(t-\tau)}(t) \tag{S62}$$

where $m_i(t)$ is the mean of s_i with respect to $P(s;t)$. This quantity is independent of X, and this leads to

$$\sum_s P(s;t)\sum_i \lambda_i(s,X) = \lambda_0 + \Delta\lambda f_t \sum_i m_i \tag{S63}$$

where

$$f_t = \int \mathrm{d}\tau f(\tau). \tag{S64}$$

Hence only the diffusion term survives,

$$\frac{\partial P(X,t)}{\partial t} = \sum_Y T(X \mid Y)P(Y,t). \tag{S65}$$

To compute the dynamics of $P_i(s_i;t)$, we denote by S^i the vector of all the s_j except s_i, and write,

$$\int d\tau f(\tau)\sum_{S^i} P(S^i;t)\sum_j s_{j-x(t-\tau)} = f_t\left(\sum_j m_j + s_i - m_i\right) \tag{S66}$$

Hence,

$$\frac{\partial P_i(s_i;t)}{\partial t} = \Delta\lambda f_t(s_i - m_i)P_i(s_i;t) \tag{S67}$$

We next consider the update following a spike in RGC i. Here we need to compute:

$$\sum_s P(s;t)\lambda_i(s,X) = \lambda_0 + \Delta\lambda \int \mathrm{d}\tau\, f(\tau)m_{i-x(t-\tau)} \tag{S68}$$

Hence,

$$P(X, t_i^+) = \frac{\lambda_0 + \Delta\lambda \int \mathrm{d}\tau f(\tau) m_{i-x(t-\tau)}}{R_i} P(X, t_i^-) \tag{S69}$$

where

$$R_i = \lambda_0 + \Delta\lambda \int \mathrm{d}\tau f(\tau) \sum_x m_{i-x} p_\tau(x; t_i^-) \tag{S70}$$

Here $p_\tau(x;t)$ equals the probability with respect to $P(X;t)$ that $x(t-\tau) = x$:

$$p_\tau(x;t) = \sum_X P(X;t) \delta_{X(t-\tau),x} \tag{S71}$$

For the probability of s_k, we write

$$\sum_X P(X;t) \sum_{S^k} P(S^k, t) s_{i-x(t-\tau)} = p_\tau(i-k;t)(s_k - m_k) + R_i \tag{S72}$$

so that,

$$P_k(s_k; t_i^+) = \left[1 + \frac{\Delta\lambda}{R_i} \tilde{P}(i-k; t_i^-)(s_k - m_k)\right] p_k(s_k; t_i^-) \tag{S73}$$

where

$$\tilde{P}(x;t) \equiv \int \mathrm{d}\tau \, f(\tau) p_\tau(x;t) \tag{S74}$$

We note that the update rules for $P_k(s_k;t)$ do not require knowledge of the full distribution over trajectories $P(X,t)$: Only the marginals $p_\tau(x;t)$ are required. Furthermore, the update rules for the pixels have precisely the same form as in the case without temporal filtering, if $P(x;t)$ is replaced by $\tilde{P}(x;t)$. (In the case without temporal filtering we assumed that the firing rate $\lambda(s_i)$ can be written as $\lambda(s_i) = \lambda_0 + \Delta\lambda s_i$). This is seen by comparing Eqs. (S67), (S70), and (S73) with Eqs. (S12), (S15), and (S17), respectively.

Known trajectory

If the trajectory is known, the dynamics between spikes are given by Eqs. (S67) and (S73), where in Eq.(S73) $\tilde{P}(x,t)$ is replaced by:

$$\tilde{P}(x,t) = \int \mathrm{d}\tau \, f(\tau) \delta_{X(t-\tau),x}. \tag{S75}$$

Rectification

So far we assumed that λ_0 is sufficiently large that $\lambda_i(t)$ remain positive at all times. In more realistic models of RGC responses, the firing rate involves rectification:

$$\lambda_i(s, X) = \phi\left[\lambda_0 + \Delta\lambda \int d\tau f(\tau) s_{i-x(t-\tau)}\right]. \tag{S76}$$

Here we assume linear rectification:

$$\phi(\lambda) = \begin{cases} \lambda & , \quad \lambda > \lambda_c \\ \lambda_c & , \quad \lambda < \lambda_c \end{cases} \tag{S77}$$

where λ_c is a (typically very small) cutoff firing rate. The precise treatment of rectification within the factorized approach leads to complicated update rules. Instead, we use an approximation, which reduces to the precise update rules derived earlier when there is no rectification.

To explain the approximation we consider the update rule between spikes. To derive the update rule for $p_k(s_k; t)$ we need to calculate the following quantity,

$$A_k = p_k(s_k) \prod_{j \neq k} \left(\sum_{s_j} p_j(s_j) \right) \sum_i \phi\left[\lambda_0 + \Delta\lambda \int \mathrm{d}\tau\, f(\tau) s_{i-x(t-\tau)}\right] \tag{S78}$$

The derivative of $p_k(s_k; t)$ between spikes can then be written in terms of A_k as

$$\frac{\mathrm{d}}{\mathrm{d}t} p_k(s_k; t) = -A_k + \sum_j A_j p_k(s_k; t) \tag{S79}$$

The sum over i is the total firing rate from the whole population of RGCs. Due to the nonlinearity it is difficult to calculate precisely the sum over s_j. Our approximation is to replace, for each i, the argument inside ϕ by an estimate based on the expected firing rate,

$$\phi\left[\lambda_0 + \Delta\lambda \int \mathrm{d}\tau\, f(\tau) s_{i-x(t-\tau)}\right] \simeq \Theta_i \times \left[\lambda_0 + \Delta\lambda \int \mathrm{d}\tau\, f(\tau) s_{i-x(t-\tau)}\right] \tag{S80}$$

where

$$\Theta_i = \Theta\left[\lambda_0 + \Delta\lambda \int \mathrm{d}\tau\, f(\tau) m_{i-x(t-\tau)} - \lambda_c\right] \tag{S81}$$

and Θ is the Heaviside function. In other words, the decoder estimates for each RGC whether its output is rectified, based on its current estimate of the pixels. After making this approximation, it is straightforward to evaluate A_k and in the binary case we get

$$\frac{\partial m_k(t)}{\partial t} = -\Delta\lambda m_k(t) \left[1 - m_k(t)\right] \sum_x \tilde{P}(x; t) \Theta_{x+k} \tag{S82}$$

where we use the notation $m_k(t) = P_k(s_k = 1; t)$. A similar procedure yields an approximation rule for the update following a spike in RGC i,

$$m_k(t_+) = \frac{q_1}{[1 - m_k(t_-)]\, q_0 + q_1 m_k(t_-)} m_k(t_-) \tag{S83}$$

where

$$q_1 = \min\left\{ R_k + \Delta\lambda \tilde{P}(i-k)\,[1 - m_k(t_-)]\,, \lambda_c \right\}, \tag{S84}$$

$$q_0 = \min\left\{ R_k - \Delta\lambda \tilde{P}(i-k) m_k(t_-), \lambda_c \right\} \tag{S85}$$

and where R_k is given by Eq. (S70).

Comparison with the ideal decoder

In the case of a known trajectory [Eq. (S75)] and for a very small image (4 x 4 pixels) we can compare performance of this decoder with the ideal Bayesian decoder, Fig. S3 B. The factorized decoder in this case matches almost precisely the ideal Bayesian decoder, and therefore provides an estimated upper bound for performance in the case of an unknown trajectory. Further, we expect performance for a known trajectory to depend only weakly on image size. This expectation is confirmed by comparing Fig. 3S B (red trace) with Fig. 3 A (dashed trace).

C. Unknown trajectory - factorized decoder with trajectory filtering

In the full problem where the decoder jointly estimates the trajectory and the filter, we considered an approximate scheme, which we call the factorized decoder with trajectory filtering. The decoder estimates the position of the image using the naive rules of Eqs. (S11) and (S14). Even though the naive decoder ignores the temporal filter, it tracks the position of the image, with a small delay $\delta t \simeq 15\,\text{ms}$ that matches the peak time of $f(\tau)$, Fig. S3 C. The decoder then generates an estimate of $\tilde{P}(x,t)$ as follows,

$$\tilde{P}(x;t) = \int \mathrm{d}\tau\, f(\tau) P(x; t-\tau) \tag{S86}$$

This estimate is used to update the pixel estimates $m_i(t)$ using Eqs. (S67) and (S73) using Eqs. (S82) and (S83). The network architecture that could implement this decoding strategy is shown schematically in Fig S3 D. Because the estimate of $\tilde{P}(x;t)$ is delayed by δt, we introduce a compensating delay in the spikes when updating $P_i(s_i;t)$. Therefore the process of spike estimation

starts only after a delay δt. In order to improve the trajectory estimate during the initial δt period, we update $P_i(s_i)$ during this period using the naive rules, Eqs. (S12) and (S17). After the initial δt period, pixel estimation starts anew using Eqs. (S82) and (S83).

The factorized decoder with trajectory filtering performs significantly better than the naive factorized decoder that ignores temporal filtering altogether, as demonstrated in Fig. S3E.

IV. PIECEWISE STATIC DECODER

The piecewise static decoder (Fig. 4C, gray trace) is defined as follows. Time is split into intervals of duration T. The spikes emitted in each one of these time intervals are analyzed separately to generate a likelihood estimate for each of the patterns s^α (the 26 letters). This estimate is given by

$$p_\alpha(t) = \frac{1}{Z(t)} \sum_x \prod_i p\left[r_i(t) \mid s^\alpha_{i+x}\right] \tag{S87}$$

where s^α_i is the intensity of pixel i in pattern α, and the sum is over all possible translations of the pattern. The decoder assumes that within a time interval the position of the image is static, and all possible locations (represented in the sum by x) are equally likely. The spike count statistics are Poisson,

$$p\left[r \mid s\right] = \frac{\exp[-\lambda(s)T]\left[\lambda(s)T\right]^r}{r!} \tag{S88}$$

Finally, the decoder treats positions in different time intervals as if they are independent. Hence the likelihood for each pattern is given by:

$$\log P_\alpha = \sum_t \log p_\alpha(t). \tag{S89}$$

In a discrimination task, only the relative magnitude of P_α for different α is important. Therefore it is sufficient for the decoder to evaluate the following quantities

$$L_\alpha = \sum_t \log\left\{\sum_x \exp\left[-a_\alpha + \sum_i r_i(t) \log\lambda\left(s^\alpha_{i+x}\right)\right]\right\} \tag{S90}$$

where

$$a_\alpha = \sum_i \lambda(s^\alpha_i)T \tag{S91}$$

which represent the log likelihood up to an additive constant which is independent of α, and to choose the pattern α for which L_α is maximal.

V. ONLINE METHODS

Initialization

At time $t = 0$ all estimates in the *what* population are set to $m_i(0) = 0.5$, *i.e.* to implement a prior that *on* and *off* occur with equal probabilities. We arbitrarily label the position of the image at time 0 as $x(0) = 0$. Therefore we set $p(x = 0, t = 0) = 1$. For all other x, $p(x, t = 0) = 0$.

Accuracy measurements

The representation in *what* cells is stabilized in time, but in different trials with the same image it may converge at different spatial shifts. To accommodate for these shifts when measuring accuracy, we first find the shift x_m such that $p\left(s \mid \{m_{i+x_m}\}\right)$ is maximized, where s is the true image and $p\left(s|\{m\}\right) \equiv \prod_i \left[s_i m_i + (1 - s_i)(1 - m_i)\right]$. To measure accuracy we compare the maximum-likelihood pattern (obtained by rectifying m_i) with the image s at the shift x_m.

Resolution of reconstruction

For reconstruction of images composed of pixels subtending 0.5 arcmins, a diffusion coefficient $D = 100\,\mathrm{arcmin}^2/\mathrm{s}$ corresponds to $400\,\mathrm{pixels}^2/\mathrm{s}$. To estimate performance on reconstruction of pixels spanning 1 arcmin, we modified our simulations in two ways: First, because four RGCs are available to report on the value of each pixel, we increased firing rates by a factor of 4. Second, we decreased the diffusion coefficient, in units of $\mathrm{pixels}^2/\mathrm{s}$, by the same factor.

Temporal filter

In all numerical simulations with a temporal filter, we used a biphasic kernel of the form [15,19]

$$f(t) = \frac{t^n}{\tau_1^{n+1}} \mathrm{e}^{-t/\tau_1} - \rho \frac{t^n}{\tau_2^{n+1}} \mathrm{e}^{-t/\tau_2} \tag{S92}$$

where $\tau_1 = 5\,\mathrm{ms}$, $\tau_2 = 15\,\mathrm{ms}$, $n = 3$, and $\rho = 0.8$.

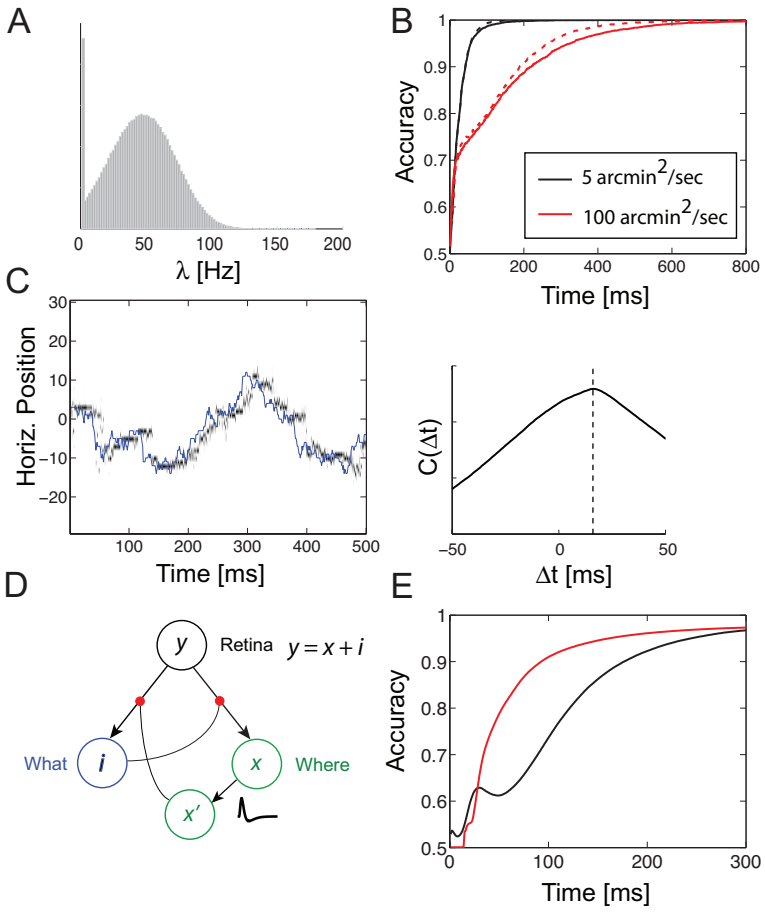

FIG. S3: **A** Distribution of firing rates [Eq. (6) in Methods] measured over a long presentation of an image containing 10 x 10 pixels and averaged over all RGCs. The diffusion coefficient $D = 100\,\mathrm{arcmin}^2/\mathrm{s}$. In all panels in this figure, $\lambda_0 = 20$ and $\Delta\lambda$ is set such that the maximum possible firing rate is 200 Hz. **B** For a known trajectory and spikes generated with a temporal filter we compare performance of the ideal Bayesian filter (dashed traces) and the factorized decoder of Eqs. (S82)–(S85) with known trajectory, Eq. (S75), which takes into account the structure of the temporal filter (solid traces). The full Bayesian decoder can only be implemented for very small images, hence the image contains only 4 x 4 pixels. Results are shown for two values of the diffusion coefficient (legend). The resolution is 0.5 arcmin. **C** Tracking of the image position by the naive factorized decoder which assumes that RGC response is instantaneous, when presented with spikes generated from RGCs with a non-instantaneous response. Left: The true position (blue trace), and tracking by the *where* cells: grayscale intensities represent the inferred position, marginalized over the vertical axis. Right: correlation function of the true position and the mean estimated position. Tracking lags behind the true position by about 16.5 ms (vertical dashed line). This lag corresponds approximately to the sharp peak in the temporal filter (Fig. 4A, inset). **D** Schematic architecture of a neural network that implements the factorized decoder with trajectory filtering (Supporting Text). **E** Performance of the factorized decoder with trajectory filtering (red trace), compared to the naive factorized decoder (black trace, as in Fig. 4C). Parameters: 30 x 30 arcmin image, 1 arcmin resolution, $D = 100\,\mathrm{arcmin}^2/\mathrm{s}$.