Supporting information

Introduction

Our goal is to understand how the brain performs marginalization. This is a probabilistic computation of the form

$$p\left(\mathbf{s}'\right)=\int d\mathbf{s}\, p\left(\mathbf{s}',\mathbf{s}\right)=\int d\mathbf{s}\, p\left(\mathbf{s}'\,|\,\mathbf{s}\right)p\left(\mathbf{s}\right) \tag{S1}$$

where $\mathbf{s}'$ is a variable (possibly vector valued) of interest and $\mathbf{s}$ is the set of variables that are "marginalized" out. The classic example is a sensorimotor transformation, in which $\mathbf{s}'$ might be the body centered position of an object and $\mathbf{s}$ the joint angles that would place one's hand on the object. In this case, $\mathbf{s}'=\mathbf{F}(\mathbf{s})$ is a deterministic function, since given joint angles one knows exactly the position of one's hand. However, it is not uniquely invertible, as there are many combinations of joint angles that lead to the same hand positions. Consequently, to find the probability distribution over the position of one's hand, it is necessary to integrate (marginalize) over the probability distribution of all sets of joint angles that lead to each hand position. Another way to see this is to note that we can replaced $p(\mathbf{s}'|\mathbf{s})$ with $\delta(\mathbf{s}'-\mathbf{F}(\mathbf{s}))$, where $\delta(\cdot)$ is the Dirac delta-function, but we still have to do an integral. Of course, the class of computations which require marginalization is much larger than the set of deterministic transformations; it can include non-deterministic ones in which the relationship between $\mathbf{s}'$ and $\mathbf{s}$ is probabilistic.

Our starting point is an encoding model for a probability distribution $\mathbf{s}$, which is captured by the likelihood function, $p(\mathbf{r}|\mathbf{s})$, where $\mathbf{r}$ is the neural response. Typically $\mathbf{r}$ is a vector of spike counts, although when we represent time dependent processes, as in the Kalman filter example below, neural representation will be denoted by $\boldsymbol{\rho}(t)$ and will indicate a population of spike trains. We also assume that the prior is known, which means we can use Bayes' theorem to recover the posterior, $p(\mathbf{s}|\mathbf{r})$. This allows us to write an expression for the true posterior over $\mathbf{s}'$,

$$p_{true}(\mathbf{s}'\,|\,\mathbf{r})=\int d\mathbf{s}\, p\left(\mathbf{s}'\,|\,\mathbf{s}\right)p\left(\mathbf{s}\,|\,\mathbf{r}\right)=\int d\mathbf{s}\,\delta(\mathbf{s}'-\mathbf{F}(\mathbf{s}))p\left(\mathbf{s}\,|\,\mathbf{r}\right)\ , \tag{S2}$$

where $p(\mathbf{s}'|\mathbf{s})$ describes the (known) probabilistic relationship between $\mathbf{s}'$ and $\mathbf{s}$. For a deterministic transformation this can be replaced with a Dirac delta-function, as in the above expression.

The goal is to find a transformation of the form

$$\overline{\mathbf{r}}'=\mathbf{f}\left(\mathbf{r}\right) \tag{S3}$$

such that $p\left(\mathbf{s}'\,|\,\overline{\mathbf{r}}'=\mathbf{f}(\mathbf{r})\right)$ is as close as possible to $p_{true}\left(\mathbf{s}'\,|\,\mathbf{r}\right)$. This is, of course, an ill-defined problem – letting $\overline{\mathbf{r}}'=\mathbf{r}$ would imply that $p\left(\mathbf{s}\,|\,\overline{\mathbf{r}}'\right)$ is exactly equal to $p_{true}\left(\mathbf{s}'\,|\,\mathbf{r}\right)$.

However, because **r** represents **s′** only indirectly (one still has to do the integral in equation (S2)), such a transformation buys us nothing. To make this problem well-defined, and at the same time force the transformation to be useful, we demand that all variables be encoded using a linear probabilistic population code (linear PPC); that is, a code of the form

$$p(\mathbf{r}\,|\,\mathbf{s},\mathbf{g}) = \phi(\mathbf{r},\mathbf{g})\exp\left(\mathbf{h}(\mathbf{s})\cdot\mathbf{r}\right). \tag{S4}$$

We have introduced an additional set of nuisance parameters, denoted **g**, into the measure function $\phi(\mathbf{r},\mathbf{g})$. Here, **g** is intended to play the role of a nuisance parameter which controls the quality of the representation of **s** provided by the induced pattern of activity, **r**; in practice, **g**, can be thought of as image contrast. Note that $\phi(\mathbf{r},\mathbf{g})$ must be chosen so as to normalize the probability distribution, but is otherwise arbitrary. The vector **h**(**s**) is called the linear kernel, and "·" denotes the standard dot product. Thus, the goal is to find both the transformation $\overline{\mathbf{r}}' = \mathbf{f}\left(\mathbf{r}\right)$ *and* a linear kernel, $\mathbf{h}'\left(\mathbf{s}'\right)$, such that the posterior of **s′** given $\overline{\mathbf{r}}'$, which is proportional to $\exp\left(\mathbf{h}(\mathbf{s}')\cdot\overline{\mathbf{r}}'\right)$, is as close as possible to $p_{true}\left(\mathbf{s}\,|\,\mathbf{r}\right)$ when $\overline{\mathbf{r}}' = \mathbf{f}\left(\mathbf{r}\right)$.

An important feature of the linear PPC encoding is that the posterior over **s** is independent of the nuisance parameters **g**, and has an especially simple form,

$$p\left(\mathbf{s}\,|\,\mathbf{r}\right) = \frac{\exp\left(\mathbf{h}\left(\mathbf{s}\right)\cdot\mathbf{r} + h_p(\mathbf{s})\right)}{Z\left(\mathbf{r},h_p\right)}, \tag{S5}$$

where the prior is proportional to log $h_p(s)$, $Z(\mathbf{r},h_p)$ is the normalization constant, and the h_p dependence in Z is shorthand for a dependence on the parameters of the function $h_p(\mathbf{s})$.

Note that while **r** is a vector of spike counts, it is not necessarily the case that the optimal transformation, $\overline{\mathbf{r}}' = \mathbf{f}\left(\mathbf{r}\right)$, will result in an integer valued quantity. This is a concern because we would like to create a self consistent neural code which utilises only spikes and spike counts. Our solution to this problem is to turn $\overline{\mathbf{r}}'$ into a vector of spike counts (or spike trains in the case of a Kalman filter), by sampling from a Poisson distribution (or process). This additional Poisson step does lead to some loss of information, but we will show that the information loss is relatively small.

In the next several sections, we consider specific examples of marginalization for a set of common problems encountered by biological organisms.

Coordinate transformations

A coordinate transformation is a mapping from one set of scalar variables to another; for a mapping from two variables to one, which is the case we focus on here, a general coordinate transformation has the form

$$s_3 = F\left(s_1, s_2\right). \tag{S6}$$

As usual, s_1 and s_2 are encoded, probabilistically – in this case in the spike counts $\mathbf{r}_1$ and $\mathbf{r}_2$, respectively – and we seek a transformation to a new population, $\overline{\mathbf{r}}_3$, that codes for s_3. For the latter transformation, we write, as in equation (S3),

$$\overline{\mathbf{r}}_3 = f\left(\mathbf{r}_1, \mathbf{r}_2\right). \tag{S7}$$

As discussed above, this transformation should lose as little information as possible; that is, $p(\mathrm{s}_3 \mid \overline{\mathbf{r}}_3 = \mathbf{f}(\mathbf{r}_1, \mathbf{r}_2))$ should be as close as possible to the true posterior. That posterior, denoted $p_{true}(s_3|\mathbf{r}_1, \mathbf{r}_2)$, is given by (assuming s_1 and s_2 are encoded independently)

$$p_{true}\left(s_3 \mid \mathbf{r}_1, \mathbf{r}_2\right) = \int ds_1 ds_2 \delta\left(s_3 - F\left(s_1, s_2\right)\right) p\left(s_1 \mid \mathbf{r}_1\right) p\left(s_2 \mid \mathbf{r}_2\right). \tag{S8}$$

In general, the integral in equation (S8) cannot be computed exactly, and so must be approximated. As discussed above, the approximation we investigate here is one in which $p\left(s_i \mid \mathbf{r}_i\right)$, $i = 1 \ldots 3$, are encoded as linear PPCs (see equations (S4) and(S5)). As such we seek to approximate the true posterior, $p_{true}\left(s_3 \mid \mathbf{r}_1, \mathbf{r}_2\right)$, by

$$p\left(s_3 \mid \overline{\mathbf{r}}_3 = \mathbf{f}\left(\mathbf{r}_1, \mathbf{r}_2\right)\right) = \frac{\exp\left(\mathbf{h}_3\left(s_3\right) \cdot \overline{\mathbf{r}}_3\right)}{Z\left(\overline{\mathbf{r}}_3\right)}. \tag{S9}$$

Because all variables are encoded in linear PPCs, the posteriors for s_1 and s_2 are given by

$$p\left(s_i \mid \mathbf{r}_i\right) = \frac{\exp\left(\mathbf{h}_i\left(\mathrm{s}_i\right) \cdot \mathbf{r}_i + h_{pi}\left(\mathrm{s}_i\right)\right)}{Z\left(\mathbf{r}_i, h_{pi}\right)}, \tag{S10}$$

i=1,2 (see equation (S5)). Combining equation (S8)-(S10), and recalling that the goal is to have the approximate and true posteriors as close as possible, we have

$$\begin{aligned} &\frac{\exp\left(\mathbf{h}_3\left(s_3\right) \cdot \mathbf{f}\left(\mathbf{r}_1, \mathbf{r}_2\right)\right)}{Z\left(\mathbf{f}\left(\mathbf{r}_1, \mathbf{r}_2\right)\right)} \approx_{KL} \\ &\int ds_1 ds_2 \delta\left(s_3 - F\left(s_1, s_2\right)\right) \frac{\exp\left(\mathbf{h}_1\left(s_1\right) \cdot \mathbf{r}_1 + h_{p1}\left(s_1\right) + \mathbf{h}_2\left(s_2\right) \cdot \mathbf{r}_2 + h_{p2}\left(s_2\right)\right)}{Z(\mathbf{r}_1, s_1) Z(\mathbf{r}_2, s_2)} \end{aligned} \tag{S11}$$

where the subscript "KL" indicates that the metric we use to measure the quality of the approximation is the Kullback–Leibler divergence between the true and approximate distributions. The goal is to choose $\mathbf{f}\left(\mathbf{r}_1, \mathbf{r}_2\right)$ and $\mathbf{h}_3\left(s_3\right)$ so that the KL divergence between the right and left hand sides in equation (S11) is as small as possible.

Linear transformations

We first consider a case in which the left and right hand sides of equation (S11) can be made identical: the posterior distributions over both s_1 and s_2 are Gaussian and $F(s_1, s_2)$ is a linear function of s_1 and s_2. Without loss of generality, we consider addition, $s_3 = s_1 + s_2$. Our analysis, however, applies to any linear transformation. (Note that in the main text, we used x^R, x^E and x^A for s_1, s_2 and s_3, respectively, and $\mathbf{r}^R$, $\mathbf{r}^E$ and $\mathbf{r}^A$ for $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$, respectively. We use the s notation here for consistency with the rest of Supporting Information.)

Because s_1 and s_2 are Gaussian, their distributions are determined solely by their means and variances, which we denote μ_i and σ_i^2, i=1, 2. In addition, s_3 is also Gaussian, and its mean and variance, μ_3 and σ_3^2, respectively, are given by

$$\begin{aligned}\mu_3 &= \mu_1 + \mu_2 \\ \sigma_3^2 &= \sigma_1^2 + \sigma_2^2.\end{aligned} \tag{S12}$$

The fact that the posterior distributions on s_i are Gaussian makes the kernels, $\mathbf{h}_i(s_i)$ especially simple,

$$p(s_i \mid \mathbf{r}_i, g_i) \propto \exp\left(\mathbf{h}_i(s_i) \cdot \mathbf{r}_i\right) = \exp\left(-\frac{s_i^2}{2}\left(\mathbf{a}_i \cdot \mathbf{r}_i + \alpha_{pi}\right) + s_i \mathbf{b}_i \cdot \mathbf{r}_i\right). \tag{S13}$$

Note that we have used a Gaussian prior with zero mean and variance $1/\alpha_{pi}$. For this choice of kernels, the mean and variance of s_i are given by

$$\begin{aligned}\mu_i &= \frac{\mathbf{b}_i \cdot \mathbf{r}_i}{\mathbf{a}_i \cdot \mathbf{r}_i + \alpha_{pi}} \\ \sigma_i^2 &= \frac{1}{\mathbf{a}_i \cdot \mathbf{r}_i + \alpha_{pi}},\end{aligned} \tag{S14}$$

i=1, 2, and 3. Note that $\alpha_{p3} = 0$ despite the fact that the prior on s_3 is not flat. This is because the population patterns of activity, $\mathbf{r}_1$ and $\mathbf{r}_2$, provide a direct representation the likelihood function, while the population $\mathbf{r}_3$ provides a direct representation of the posterior distribution. Combining equations (S12) and (S14), we see that an optimal transformation $\overline{\mathbf{r}}_3 = \mathbf{f}\left(\mathbf{r}_1, \mathbf{r}_2\right)$ should satisfy

$$\begin{aligned}\frac{\mathbf{b}_3 \cdot \overline{\mathbf{r}}_3}{\mathbf{a}_3 \cdot \overline{\mathbf{r}}_3} &= \frac{\mathbf{b}_1 \cdot \mathbf{r}_1}{\mathbf{a}_1 \cdot \mathbf{r}_1 + \alpha_{p1}} + \frac{\mathbf{b}_2 \cdot \mathbf{r}_2}{\mathbf{a}_2 \cdot \mathbf{r}_2 + \alpha_{p2}} \\ \frac{1}{\mathbf{a}_3 \cdot \overline{\mathbf{r}}_3} &= \frac{1}{\mathbf{a}_1 \cdot \mathbf{r}_1 + \alpha_{p1}} + \frac{1}{\mathbf{a}_2 \cdot \mathbf{r}_2 + \alpha_{p1}},\end{aligned} \tag{S15}$$

or, equivalently,

$$
\begin{aligned}
\mathbf{a}_3 \cdot \overline{\mathbf{r}}_3 &= \frac{\left(\mathbf{a}_1 \cdot \mathbf{r}_1 + \alpha_{p1}\right)\left(\mathbf{a}_2 \cdot \mathbf{r}_2 + \alpha_{p1}\right)}{\mathbf{a}_1 \cdot \mathbf{r}_1 + \mathbf{a}_2 \cdot \mathbf{r}_2 + \alpha_{p1} + \alpha_{p2}} \\
\mathbf{b}_3 \cdot \overline{\mathbf{r}}_3 &= \frac{\left(\mathbf{b}_1 \cdot \mathbf{r}_1\right)\left(\mathbf{a}_2 \cdot \mathbf{r}_2 + \alpha_{p2}\right) + \left(\mathbf{b}_2 \cdot \mathbf{r}_2\right)\left(\mathbf{a}_1 \cdot \mathbf{r}_1 + \alpha_{p1}\right)}{\mathbf{a}_1 \cdot \mathbf{r}_1 + \mathbf{a}_2 \cdot \mathbf{r}_2 + \alpha_{p1} + \alpha_{p2}}.
\end{aligned}
\tag{S16}
$$

A transformation, $\overline{\mathbf{r}}_3 = \mathbf{f}\left(\mathbf{r}_1, \mathbf{r}_2\right)$, that satisfies equation (S16) is

$$
\begin{aligned}
\mathbf{f}(\mathbf{r}_1, \mathbf{r}_2) = \mathbf{a}_3^{\dagger} & \frac{\left(\mathbf{a}_1 \cdot \mathbf{r}_1 + \alpha_{p1}\right)\left(\mathbf{a}_2 \cdot \mathbf{r}_2 + \alpha_{p2}\right)}{\mathbf{a}_1 \cdot \mathbf{r}_1 + \mathbf{a}_2 \cdot \mathbf{r}_2 + \alpha_{p1} + \alpha_{p2}} \\
& + \mathbf{b}_3^{\dagger} \frac{\left(\mathbf{b}_1 \cdot \mathbf{r}_1\right)\left(\mathbf{a}_2 \cdot \mathbf{r}_2 + \alpha_{p1}\right) + \left(\mathbf{b}_2 \cdot \mathbf{r}_2\right)\left(\mathbf{a}_1 \cdot \mathbf{r}_1 + \alpha_{p1}\right)}{\mathbf{a}_1 \cdot \mathbf{r}_1 + \mathbf{a}_2 \cdot \mathbf{r}_2 + \alpha_{p1} + \alpha_{p2}} \\
& + \mathbf{c}_3^{\dagger} f_3(\mathbf{r}_1, \mathbf{r}_2)
\end{aligned}
\tag{S17}
$$

where $\mathbf{a}_3^{\dagger}$, $\mathbf{b}_3^{\dagger}$, and $\mathbf{c}_3^{\dagger}$ obey the relationships

$$
\begin{aligned}
&\mathbf{a}_3 \cdot \mathbf{a}_3^{\dagger} = \mathbf{b}_3 \cdot \mathbf{b}_3^{\dagger} = 1 \\
&\mathbf{a}_3 \cdot \mathbf{b}_3^{\dagger} = \mathbf{a}_3 \cdot \mathbf{c}_3^{\dagger} = \mathbf{b_3} \cdot \mathbf{a}_3^{\dagger} = \mathbf{b_3} \cdot \mathbf{c}_3^{\dagger} = 0,
\end{aligned}
\tag{S18}
$$

and $f_3(\mathbf{r}_1, \mathbf{r}_2)$ is an arbitrary scalar function of $\mathbf{r}_1$ and $\mathbf{r}_2$.

Because equations (S17) and (S18) give us the optimal transformation, it immediately gives us the true posterior over s_3 given $\mathbf{r}_1$ and $\mathbf{r}_2$,

$$
p_{true}(s_3 \mid \mathbf{r}_1, \mathbf{r}_2) = \frac{1}{Z\left(\mathbf{f}(\mathbf{r}_1, \mathbf{r}_2)\right)} \exp\left(-\frac{s_3^2}{2} \mathbf{a}_3 \cdot \mathbf{f}(\mathbf{r}_1, \mathbf{r}_2) + s_3 \mathbf{b}_3 \cdot \mathbf{f}(\mathbf{r}_1, \mathbf{r}_2) \right). \tag{S19}
$$

A network that implements $\overline{\mathbf{r}}_3 = \mathbf{f}(\mathbf{r}_1, \mathbf{r}_2)$ leads to a posterior over s_3 that is exactly equal to the true posterior. Networks, however, communicate via spikes, not spike count. Thus, the actual transformation is probabilistic, and has the form

$$
r_{3i} = Poisson\left(\overline{r}_{3i}\right) = Poisson\left(f_i(\mathbf{r}_1, \mathbf{r}_2)\right), \tag{S20}
$$

where the notation *Poisson* indicates a Poisson distribution.

The noisy transformation given by equation (S20) loses information, but if the number of neurons is large *and* the information is order(1), it doesn't lose much. To see why, note that for Poisson noise added to some fixed population pattern of activity $\mathbf{r}$,

$$
\mathrm{Var}[\mathbf{h}(s) \cdot \left(\mathbf{r} + \delta \mathbf{r}\right)] = \sum_i h_i^2(s) \mathrm{Var}\left[\delta r_i\right] = \sum_i h_i^2(s) r_i. \tag{S21}
$$

For information to be order(1), $h_i(s)$ must scale as 1/*N* where *N* is the number of neuron. Thus, the term on the right hand side of equation (S21) also scales as 1/*N*. This, in turn, implies that the additional noise added to $\mathbf{h}(s)\cdot\mathbf{r}$ must scale as 1/*N*.

There is a great deal of freedom in choosing both $\mathbf{h}_3(s_3)$ (because $\mathbf{a}_3$ and $\mathbf{b}_3$ are only required to be non-parallel; see equation (S18)) and $f(\mathbf{r}_1,\mathbf{r}_2)$ (because $f_3(\mathbf{r}_1,\mathbf{r}_2)$ is arbitrary). In addition, to completely specify the encoding model, we need to choose the prefactors in the likelihood functions, $\phi(\mathbf{r}_i, g_i)$, *i*=1, 2 (see equation(S4)).

We now consider a concrete example in which the encoding model is a conditionally independent Poisson,

$$p(\mathbf{r}_i \mid s_i, g_i) = \prod_j \frac{\left(g_i f_j(s_i)\right)^{r_{ij}}}{r_{ij}!}\exp\left(-g_i f_j(s_i)\right), \tag{S22}$$

i=1, 2. Here r_{ij} is the spike count of the *j*th neuron in population *i* and $f_j(s_i)$ is the tuning curve of the *j*th neuron in population *i*. To ensure that the likelihood functions are Gaussian distributed, we use Gaussian tuning curves,

$$f_j(s) = \exp\left(-\frac{\left(s - s_j^0\right)^2}{2\sigma_w^2}\right). \tag{S23}$$

The preferred orientations, s_j^0, are uniformly distributed on a finite interval and symmetric around 0. Note, therefore, that $\sum_j s_j^0 = 0$, a fact we make use of below. For tuning curves of this shape we can rewrite the likelihood functions for the input populations ($\mathbf{r}_1$ and $\mathbf{r}_2$) as

$$p(\mathbf{r}_i \mid s_i, g_i) = \left(\prod_j \frac{g_i^{n_{ij}}}{r_{ij}!}\right)\exp\left(-\sum_j r_{ij}(s_j^0)^2/2\sigma_w^2\right)\exp\left(-g_i\sum_j f_j(s_i)\right)\exp\left(\mathbf{h}_i(s_i)\cdot\mathbf{r}_i\right)$$

(S24)

where

$$\mathbf{h}_i(s) = -\frac{s^2}{2}\mathbf{a}_i + s\mathbf{b}_i, \tag{S25}$$

with

$$\begin{aligned}\mathbf{a}_i &= \frac{\mathbf{1}}{\sigma_w^2}\\ \mathbf{b}_i &= \frac{\mathbf{s}^0}{\sigma_w^2}.\end{aligned} \tag{S26}$$

Here **1** is a vector consisting of all 1's. Because the preferred orientations, s_j^0, must be distributed over a finite range, the gain parameter, g_i, interacts with the stimulus, s_i, via the third term in equation (S24). Therefore, the likelihood function given in that equation is not a linear PPC. We sidestep this issue by placing a prior on s_1 and s_2 that effectively restricts them to values for which $\sum_j f_j\left(s_i\right)$ is nearly flat.

The parameters of the output kernel, $\mathbf{a}_3$ and $\mathbf{b}_3$, are essentially arbitrary. However, it is convenient to make them orthogonal, and to have the same general shape as **a** and **b**. Therefore, we choose a_{3i} to be an even function of its index i and b_{3i} to be an odd function, and we choose them so they are approximately constant and linear over the range spanned by the tuning curves. One such choice is

$$\begin{aligned} a_{3i} &= \theta_1\left[\exp\left(-2\left((i-i_0)/N\right)^2/\sigma_{3w}^2\right)-\frac{1}{N}\sum_i \exp\left(-2\left((i-i_0)/N\right)^2/\sigma_{3w}^2\right)\right] \\ b_{3i} &= \theta_1\left((i-i_0)/N\right)\exp\left(-2\left((i-i_0)/N\right)^2/\sigma_{3w}^2\right) \end{aligned} \tag{S27}$$

where $i_0=(N+1)/2$, θ_1 is an arbitrary scale parameter, σ_{3w} controls the width of the tuning curves in the output population, and i goes from 1 to N. Because $\mathbf{a}_3$ and $\mathbf{b}_3$ are orthogonal, they are self adjoint, up to a normalizing constant so that

$$\begin{aligned} a_{3i}^{\dagger} &= \frac{a_{3i}}{Z_a} \\ b_{3i}^{\dagger} &= \frac{b_{3i}}{Z_b} \end{aligned} \tag{S28}$$

with Z_a and Z_b chosen to ensure that $\mathbf{a}_3^{\dagger}\cdot\mathbf{a}_3=\mathbf{b}_3^{\dagger}\cdot\mathbf{b}_3=1$. Finally, we choose

$$c_{3i}^{\dagger}=\frac{1}{\theta_2}. \tag{S29}$$

With this choice of parameters we have, via equation (S17) and the fact that $\bar{\mathbf{r}}_3=\mathbf{f}\left(\mathbf{r}_1,\mathbf{r}_2\right)$,

$$\begin{aligned} \bar{r}_{3k} = {} & \frac{a_{3k}}{Z_a}\sum_{ij}\frac{\left(a_{1i}r_{1i}+\frac{\alpha_{p1}}{N_1}\right)\left(a_{2j}r_{2j}+\frac{\alpha_{p2}}{N_2}\right)}{\mathbf{a}_1\cdot\mathbf{r}_1+\mathbf{a}_2\cdot\mathbf{r}_2+\alpha_{p1}+\alpha_{p2}} \\ & +\frac{b_{3k}}{Z_b}\sum_{ij}\frac{b_{1i}r_{1i}\left(a_{2j}r_{2j}+\frac{\alpha_{p2}}{N_2}\right)+b_{1i}r_2\left(a_{2j}\cdot r_{1j}+\frac{\alpha_{p1}}{N_1}\right)}{\mathbf{a}_1\cdot\mathbf{r}_1+\mathbf{a}_2\cdot\mathbf{r}_2+\alpha_{p1}+\alpha_{p2}} \\ & +\frac{f_3(\mathbf{r}_1,\mathbf{r}_2)}{\theta_2} \end{aligned} \tag{S30}$$

where N_1 and N_2 are the number of neurons associated with $\mathbf{r}_1$ and $\mathbf{r}_2$, respectively. Taking advantage of the fact that $\mathbf{a}_i \cdot \mathbf{b}_j = 0$, $i,j=1,2$, and using $\mathbf{a}_i = \mathbf{1}/\sigma_w^2$, equation (S30) can be rewritten as

$$\begin{aligned} \overline{r}_{3k} &= \sum_{ij} \frac{\left(\frac{a_{3k}}{Z_a} a_{1i} a_{2j} + \frac{b_{3k}}{Z_b} \left(b_{1i} a_{2j} + a_{1i} b_{2j} \right) \right) \left(r_{1i} + \sigma_w^2 \frac{\alpha_{p1}}{N_1} \right) \left(r_{2j} + \sigma_w^2 \frac{\alpha_{p2}}{N_2} \right)}{\mathbf{a} \cdot \mathbf{r}_1 + \mathbf{a} \cdot \mathbf{r}_2 + \alpha_{p1} + \alpha_{p2}} + \frac{f_3(\mathbf{r}_1, \mathbf{r}_2)}{\theta_2} \\ &= \sum_{ij} w_{ij}^k \varphi_{ij} \left(\mathbf{r}_1, \mathbf{r}_2 \right) + \frac{f_3(\mathbf{r}_1, \mathbf{r}_2)}{\theta_2} \end{aligned} \tag{S31}$$

where

$$\begin{aligned} w_{ij}^k &= \frac{a_{3k}}{Z_a} a_{1i} a_{2j} + \frac{b_{3k}}{Z_b} \left(b_{1i} a_{2j} + a_{1i} b_{2j} \right) \\ \varphi_{ij} \left(\mathbf{r}_1, \mathbf{r}_2 \right) &= \frac{\left(r_{1i} + \sigma_w^2 \frac{\alpha_{p1}}{N_1} \right) \left(r_{2j} + \sigma_w^2 \frac{\alpha_{p2}}{N_2} \right)}{\mathbf{a}_1 \cdot \mathbf{r}_1 + \mathbf{a}_2 \cdot \mathbf{r}_2 + \alpha_{p1} + \alpha_{p2}}. \end{aligned} \tag{S32}$$

Written in this way it becomes easy to see why we refer to this network has having a quadratic non-linearity, with divisive normalization. Note that equation (S31) corresponds to equation (8) of the main text (except that in the main text, α_{p1}, α_{p2} and $f_3(\mathbf{r}_1,\mathbf{r}_2)$ are all set to zero).

Nonlinear transformations

We also consider a nonlinear coordinate transformation for which the quantity of interest is the angular location of the 'hand' in 'body' centered coordinates, as shown in Figure S1. We assume that the lengths of the arms are known, and our goal is to find the angular location of the hand (s_3) in terms of the angular locations of the two arms (s_1 and s_2). Mathematically, the relationship between these three angles is given by

$$\begin{aligned} d_3 \sin(s_3) &= d_1 \sin(s_1) + d_2 \sin(s_1 + s_2) \\ d_3 \cos(s_3) &= d_1 \cos(s_1) + d_2 \cos(s_1 + s_2) \\ d_3^2 &= d_1^2 + d_2^2 + 2 d_1 d_2 \cos(s_2) \end{aligned} \tag{S33}$$

Unlike for linear transformations, there is no simple way to determine the optimal network. However, this is some hope that the network architecture we used for the linear transformation will also work for nonlinear ones. To test this notion we use gradient descent learning on the parameters of a nonlinear network transformation which takes the form of equation (S31). The details are described in the Statistical verification section below.

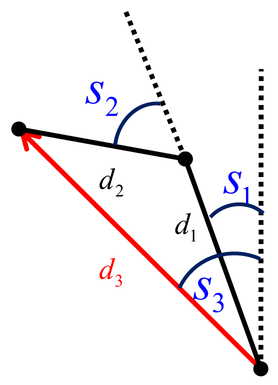

Figure S1. Here the 'shoulder' is depicted by the circle in the lower right and the 'hand' by the circle in the upper left. Proprioception provides information about the angle by which the body upper arm is deflected from a body centric coordinate system, s_1, and the angle by which the lower arm is deflected from the upper arm, s_2. Upper and lower arms are of length d_1 and d_2 respectively. The quantity of interest, s_3, is the angular position of the hand in body centered coordinates.

Kalman filters

A Kalman filter is an algorithm for determining the probability distribution of a variable, or set of variables, based on noisy observations over time. The defining feature of a Kalman filter is that the variables of interest are Gaussian and obey linear dynamics. Given the analysis in the previous section, in which linear PPCs exactly implemented marginalization over Gaussian variables, we expect that linear PPCs will exactly implement a Kalman filter. This is, indeed, what we find.

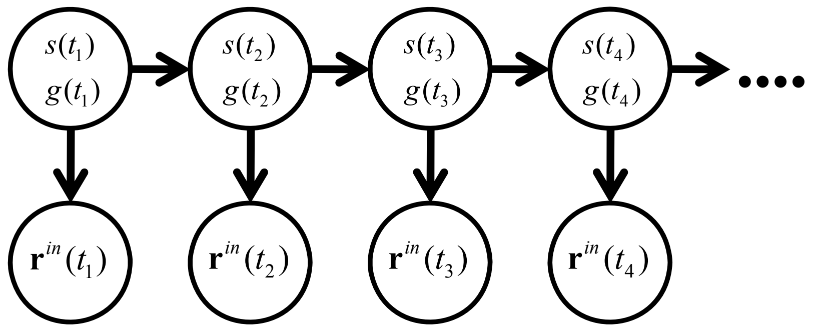

Figure S2. Graphical model for a Kalman filter. The top row contains $s(t)$, a real valued random variable which represents the position of a particle subject to linear dynamics and additive white noise, and $g(t)$, the gain. The bottom row contains the population activity, $\mathbf{r}^{in}(t)$, which provides a noisy estimate of $s(t)$ and $g(t)$. Gain is best thought of as representing

the quality of the population code, and reflects task irrelevant nuisance parameters like the contrast of the stimulus. Here $g_i(t)$ and $s_i(t)$ are assumed to be independent.

The setup we consider is the hidden Markov model shown in Fig. S2. The variable of interest, $s(t)$, evolves under its internal dynamics, as indicated by the arrows connecting the top row of circles. In addition, at each time step we receive noisy information about $\mathbf{s}(t)$ via population activity denoted $\mathbf{r}^{in}(t)$, as indicated by the downward arrows; this information comes via the likelihood function, $p(\mathbf{r}^{in}(t) \mid s(t), g(t))$, specified in equation (S37) below. Here $\mathbf{r}^{in}(t)$ is a vector of spike counts between time t-Δt and t; that is, $r_i^{in}(t)$ is the number of spikes that occurred on neuron i between time t-Δt and t. Note that when Δt is small – the limit of interest here – for most time points, $\mathbf{r}^{in}(t)$ contains no spikes, and so does not provide any information about $s(t)$. Thus, for most time intervals, the probability distribution over s evolves only in accordance with the dynamics of $s(t)$.

The goal is to construct population activity that codes for the probability distribution of $s(t)$ at time t, denoted $p(s(t))$. Clearly, knowledge of $\mathbf{r}^{in}(t)$ for all times up to t will tell us $p(s(t)|\mathbf{r}^{in}(t)$, $\mathbf{r}^{in}(t$-$\Delta t)$,...). However, it is quite inefficient for the brain to store the whole time-history of $\mathbf{r}^{in}(t)$. We seek, therefore, a transformation that takes the time-history of $\mathbf{r}^{in}(t)$ and maps it to population activity whose current set of firing rates tell us $p(s(t)|$ $\mathbf{r}^{in}(t)$, $\mathbf{r}^{in}(t$- $\Delta t)$,...).

To find this transformation, we make use of the update rule for hidden Markov models,

$$p\big(s(t)\big) \propto p\big(\mathbf{r}^{in}(t) \mid s(t), g(t)\big) \int ds(t-\Delta t)\, p\big(s(t) \mid s(t-\Delta t)\big)\, p\big(s(t-\Delta t)\big). \qquad \text{(S34)}$$

The two ingredients we need to compute the right hand side are the likelihood, $p(\mathbf{r}^{in}(t)|s(t))$ (downward arrows in Fig. S2), and the transition probability, $p(s(t)|s(t$-$\Delta t))$ (horizontal arrows in Fig. S1). We start with the latter.

Although $s(t)$ evolves in discrete time in Fig. S1, ultimately we are interested in the continuous time limit, which we access by, eventually, letting Δt go to zero. Therefore, we write a continuous time evolution equation for s,

$$\frac{ds}{dt} = -\gamma s + \eta(t) \qquad \text{(S35)}$$

where $\eta(t)$ is white noise with variance σ_η^2; that is, $<\eta(t)\eta(t')>=\sigma_\eta^2 \delta(t-t')$. With this dynamics, the conditional probability distribution of $s(t)$ given $s(t$-$\Delta t)$ is Gaussian with mean $(1$-$\gamma\Delta t)s(t$-$\Delta t)$ and variance $\sigma_\eta^2 \Delta t$,

$$p\big(s(t) \mid s(t-\Delta t)\big) \sim \mathcal{N}\Big((1-\gamma\Delta t)s(t-\Delta t), \sigma_\eta^2 \Delta t\Big). \qquad \text{(S36)}$$

The second ingredient in equation (S34), $p(\mathbf{r}^{in}(t)|s(t),g(t))$, is encoded via a linear PPC,

$$p\big(\mathbf{r}^{in}(t) \mid s(t), g(t)\big) = \phi\big(\mathbf{r}^{in}(t), g(t)\big) \exp\big(\mathbf{h}^{in}\big(s(t)\big) \cdot \mathbf{r}^{in}(t)\big) \qquad \text{(S37)}$$

where the gain, $g(t)$, is an arbitrary, and possibly random, function of time. We use a likelihood function that is Gaussian in $s(t)$, for which $\mathbf{h}^{in}(s(t))\cdot\mathbf{r}^{in}$ is given by

$$\mathbf{h}^{in}\left(s(t)\right)\cdot\mathbf{r}^{in}(t) = -\frac{s(t)^2}{2}\mathbf{a}^{in}\cdot\mathbf{r}^{in}(t) + s(t)\mathbf{b}^{in}\cdot\mathbf{r}^{in}(t). \tag{S38}$$

With this form for the likelihood, the mean and variance of $s(t)$ given $\mathbf{r}^{in}(t)$ are independent of the gain, $g(t)$ and, assuming a flat prior, are given by

$$\begin{aligned} \mu_{in}(t)/\sigma_{in}^2(t) &= \mathbf{b}^{in}\cdot\mathbf{r}^{in}(t) \\ 1/\sigma_{in}^2(t) &= \mathbf{a}^{in}\cdot\mathbf{r}^{in}(t). \end{aligned} \tag{S39}$$

Note that we are using the so called natural parameters, μ_{in}/σ_{in}^2 and $1/\sigma_{in}^2$, rather than the more standard mean and variance. This is because a linear PPC encodes the natural parameters as a linear combination of neural activity.

Inserting equations (S36)-(S39) into (S34), performing the integral on the right hand side (which is Gaussian, and, therefore, straightforward), carrying out a small amount of algebra, and using equation (S38), we find that the update rule for the mean and variance of $s(t)$, denoted $\mu(t)$ and $\sigma^2(t)$ is given by

$$\begin{aligned} \frac{\mu(t)}{\sigma^2(t)} &= \frac{\mu_{in}(t)}{\sigma_{in}^2(t)} + \frac{(1-\gamma\Delta t)\mu(t-\Delta t)}{\sigma^2(t-\Delta t)(1-\gamma\Delta t)^2 + \sigma_\eta^2\Delta t} \\ \frac{1}{\sigma^2(t)} &= \frac{1}{\sigma_{in}^2(t)} + \frac{1}{\sigma^2(t-\Delta t)(1-\gamma\Delta t)^2 + \sigma_\eta^2\Delta t}. \end{aligned} \tag{S40}$$

To complete our analysis, we take the limit $\Delta t \rightarrow 0$. In this limit, we need to consider two cases. In the first, there are no spikes on $\mathbf{r}^{\text{in}}(t)$ in the time interval between t-Δt and t. Examining equation (S40), we see that in this case $\sigma_{in}^2(t) = \infty$. Consequently, the term $1/\sigma_{in}^2(t)$ in equation (S40) is zero, and so μ/σ^2 and $1/\sigma^2$ evolve according to

$$\begin{aligned} \frac{d(\mu/\sigma^2)}{dt} &= \gamma(\mu/\sigma^2) - (\sigma_\eta^2/\sigma^2)(\mu/\sigma^2) \\ \frac{d(1/\sigma^2)}{dt} &= 2\gamma(1/\sigma^2) - (\sigma_\eta^2/\sigma^2)(1/\sigma^2). \end{aligned} \tag{S41}$$

In the second case, there are spikes on $\mathbf{r}^{\text{in}}(t)$ in the time interval between t-Δt and t. Here μ_{in} and σ_{in}^2 are both order(1), and so both μ and σ^2 exhibit jump discontinuities. In the limit $\Delta t \rightarrow 0$, these are given by

$$\begin{aligned} \mu/\sigma^2 &\rightarrow \mu/\sigma^2 + \mu_{in}/\sigma_{in}^2 \\ 1/\sigma^2 &\rightarrow 1/\sigma^2 + 1/\sigma_{in}^2. \end{aligned} \tag{S42}$$

Combining equations (S41) and (S42) into a single equation, we have

$$\frac{d(\mu/\sigma^2)}{dt} = \gamma(\mu/\sigma^2) - (\sigma_\eta^2/\sigma^2)(\mu/\sigma^2) + \mathbf{b}^{in}\cdot\boldsymbol{\rho}^{in}(t)$$
$$\frac{d(1/\sigma^2)}{dt} = 2\gamma(1/\sigma^2) - (\sigma_\eta^2/\sigma^2)(1/\sigma^2) + \mathbf{a}^{in}\cdot\boldsymbol{\rho}^{in}(t) \tag{S43}$$

where $\boldsymbol{\rho}^{in}(t)$ represents a set of delta function spike trains,

$$\rho_i^{in} = \sum_j \delta\left(t - t_i^j\right) \tag{S44}$$

with t_i^j the time of the *j*th spike of input neuron *i*, so that integrating this quantity over time window Δt would yield $\mathbf{r}^{in}(t)$.

Finally, we recall that the goal was to obtain a linear PPC encoding for *p*(*s*(*t*)), which we write

$$p\left(s(t)\,|\,\mathbf{v}(t)\right) \propto \exp\left(\mathbf{h}\left(s(t)\right)\cdot\mathbf{v}(t)\right) = \exp\left(-\frac{\left(s(t)\right)^2}{2}\mathbf{a}\cdot\mathbf{v}(t) + s(t)\mathbf{b}\cdot\mathbf{v}(t)\right). \tag{S45}$$

What we need to do is find the time evolution of **v**(*t*) such that *p*(*s*(*t*)| **v**(*t*)) will be Gaussian with the mean and variance one would find by solving equation (S43). When we do that, it will turn out that **v**(*t*) is a vector of time-dependent *rate*s. Eventually, of course, we have to convert to spikes. However, as shown in Fig. 3b-c of the main text, that conversion leads to almost no loss of information.

Our starting point is the observation that, in terms of the parameters of equation (S45), the mean and variance of *s*(*t*) are given by

$$\mu(t)/\sigma^2(t) = \mathbf{b}\cdot\mathbf{v}(t)$$
$$1/\sigma^2(t) = \mathbf{a}\cdot\mathbf{v}(t). \tag{S46}$$

We can now substitute equation (S46) into (S43) to yield continuous time equations for **b·v** and **a·v**,

$$\frac{d\mathbf{b}\cdot\mathbf{v}}{dt} = \gamma\mathbf{b}\cdot\mathbf{v} - \sigma_\eta^2\left(\mathbf{a}\cdot\mathbf{v}\right)\left(\mathbf{b}\cdot\mathbf{v}\right) + \mathbf{b}^{in}\cdot\boldsymbol{\rho}^{in}$$
$$\frac{d\mathbf{a}\cdot\mathbf{v}}{dt} = 2\gamma\mathbf{a}\cdot\mathbf{v} - \sigma_\eta^2\left(\mathbf{a}\cdot\mathbf{v}\right)\left(\mathbf{a}\cdot\mathbf{v}\right) + \mathbf{a}^{in}\cdot\boldsymbol{\rho}^{in}. \tag{S47}$$

An equation for **v** that is consistent with equation (S47) is

$$\frac{d\mathbf{v}}{dt} = \gamma\mathbf{W}\cdot\mathbf{v} - \sigma_\eta^2\left(\mathbf{a}\cdot\mathbf{v}\right)\mathbf{v} + \mathbf{M}\cdot\boldsymbol{\rho}^{in} + \mathbf{c}^\dagger f_c(\mathbf{v},\boldsymbol{\rho}^{in}) \tag{S48}$$

where

$$\begin{aligned}\mathbf{W} &= 2\mathbf{a}^\dagger\mathbf{a} + \mathbf{b}^\dagger\mathbf{b} \\ \mathbf{M} &= \mathbf{a}^\dagger\mathbf{a}^{in} + \mathbf{b}^\dagger\mathbf{b}^{in},\end{aligned} \tag{S49}$$

$\mathbf{a}^\dagger$, $\mathbf{b}^\dagger$ and $\mathbf{c}^\dagger$ obey the orthogonality condition given by equation (S18), but without the subscript "3", and $f_c(\mathbf{v}, \boldsymbol{\rho}^{in})$ is an arbitrary scalar functional. Direct inspection verifies that equation (S48) is the most general equation that is consistent with equation (S47). Equation (S48) corresponds to equation (19) in the main text, except that in the main text, for clarity we do not include $f_c(\mathbf{v}, \boldsymbol{\rho}^{in})$.

To see that equation (S48) implements a divisive normalization, we solve this equation in the limit that $d\mathbf{v}/dt$=0, for which we can set $\boldsymbol{\rho}^{in}$ to its long term average, which we denote $\bar{v}_{in}$. (This limit doesn't, in fact, exist, as $\boldsymbol{\rho}^{in}$ is time-varying; however, it is a useful abstraction.) Setting $d\mathbf{v}/dt$ to zero in equation (S48) (or alternatively, (S47)) and solving for $\mathbf{a}\cdot\mathbf{v}$ and $\mathbf{b}\cdot\mathbf{v}$, we find that

$$\begin{aligned}\mathbf{a}\cdot\mathbf{v} &\sim \frac{\gamma}{\sigma_\eta^2} + \frac{\gamma^2 + \sigma_\eta^2 \mathbf{a}^{in}\cdot\bar{v}_{in}}{\sigma_\eta^2\sqrt{\gamma^2 + \sigma_\eta^2\mathbf{a}^{in}\cdot\bar{v}_{in}}} \\ \mathbf{b}\cdot\mathbf{v} &\sim \frac{\mathbf{b}^{in}\cdot\bar{v}_{in}}{\sqrt{\gamma^2 + \sigma_\eta^2\mathbf{a}^{in}\cdot\bar{v}_{in}}}.\end{aligned} \tag{S50}$$

And so we observe that, in the steady state, the relationship between $\mathbf{v}$ and the average input pattern of activity, $\bar{v}_{in}$, contains a divisive normalization which is related the square root of $\bar{v}_{in}$.

How is the population activity in our model related to known properties of Kalman filters? The answer to this question will be useful for making experimental predictions. One relevant regime is the limit in which there is no input information, $\boldsymbol{\rho}^{in} = 0$, and no internal dynamics (γ=0). In that limit, the variance of s increases linearly with time. What happens to the activity? Rewriting equation (S47) with both $\boldsymbol{\rho}^{in}$ and γ set to zero, we see that $\mathbf{b}\cdot\mathbf{v}$ and $\mathbf{a}\cdot\mathbf{v}$ evolve according to

$$\begin{aligned}\frac{d\mathbf{b}\cdot\mathbf{v}}{dt} &= -\sigma_\eta^2(\mathbf{a}\cdot\mathbf{v})(\mathbf{b}\cdot\mathbf{v}) \\ \frac{d\mathbf{a}\cdot\mathbf{v}}{dt} &= -\sigma_\eta^2(\mathbf{a}\cdot\mathbf{v})(\mathbf{a}\cdot\mathbf{v}).\end{aligned} \tag{S51}$$

Solving for $\mathbf{a}\cdot\mathbf{v}$ and $\mathbf{b}\cdot\mathbf{v}$, we find that

$$\mathbf{a}\cdot\mathbf{v} \propto \mathbf{b}\cdot\mathbf{v} \propto \frac{1}{c+t} \tag{S52}$$

where c is some constant. This indicates that neural activity decreases inversely with time, mirroring the behaviour of the inverse of the variance. When γ is non-zero this relationship holds initially, but eventually neural activity approaches a pattern of activity which corresponds to a zero mean Gaussian distribution with fixed variance.

When the rate parameter, **ν**, evolves according to equation (S48), there is no information loss. However, as with the sensory transformation, we need to convert to spikes. This is slightly more complicated than for the coordinate transformation, since **ν** appears on the right hand side of equation (S48). Thus, we proceed in two steps. The first is to replace the terms $\mathbf{a}\cdot\mathbf{v}$, $\mathbf{W}\cdot\mathbf{v}$ and $f_c\left(\mathbf{v},\boldsymbol{\rho}^{in}\right)$ that appear on the right hand side of equation (S48) with $\mathbf{a}\cdot\boldsymbol{\rho}$, $\mathbf{W}\cdot\boldsymbol{\rho}$ and $f_c\left(\boldsymbol{\rho},\boldsymbol{\rho}^{in}\right)$, where $\boldsymbol{\rho}$ is a spike train generated according to the rate parameter **λ**; that is,

$$\rho_i(t) \sim Poisson\ process\left(\nu_i(t)\right). \tag{S53}$$

Here we use "Poisson process," rather than just "Poisson" (as in equation (S20), the coordinate transformation example) to indicate that **ρ**(t) consists of a set of δ-function spikes.

Second, as equation (S48) is written, there is no guarantee that the rate parameter, **ν**, will be nonnegative. Fortunately, we can insure this by choosing $\mathbf{c}^\dagger$ to be a vector that is composed entirely of ones, and insert a new parameter (denoted ν_0 below) which allows us to control the mean activity of the population.

With these replacements, and letting $\mathbf{c}^\dagger f_c(\boldsymbol{\rho},\boldsymbol{\rho}^{in}) = \mathbf{1}\left(\nu_0 - \mathbf{1}\cdot\boldsymbol{\rho}/N\right)$, the network evolves according to

$$\frac{d\mathbf{v}}{dt} = \gamma\mathbf{W}\boldsymbol{\rho} - \sigma_\eta^2\left(\mathbf{a}\cdot\boldsymbol{\rho}\right)\mathbf{v} + \mathbf{M}\boldsymbol{\rho}^{in} + \mathbf{1}\left(\nu_0 - \frac{\mathbf{1}\cdot\boldsymbol{\rho}}{N}\right), \tag{S54}$$

where **1**=(1, 1, …) and N is the number of neurons in the population (the dimensionality of the rate vector, **ν**). Note that it is still rate that evolves according to equation (S54); this is because we assume that neurons have, in some abstract sense, access to their own internal rates. However, when we compute the posterior, we use spike count, **r**, in an interval δt,

$$\mathbf{r}(t) = \int_t^{t+\delta t} d\tau\ \boldsymbol{\rho}(\tau). \tag{S55}$$

The posterior, as computed by the network, is then given by equation (S45), but with $\mathbf{v}(t)$ replaced by $\mathbf{r}(t)/\delta t$. For fixed δt, as the number of neurons increases, the posterior becomes more and more accurate.

Finally, we need to set the parameters **a**, **b**, $\mathbf{a}^\dagger$, $\mathbf{b}^\dagger$, $\mathbf{a}^{in}$, and $\mathbf{b}^{in}$. For **a**, **b**, $\mathbf{a}^\dagger$, and $\mathbf{b}^\dagger$, we choose

$$\begin{aligned} a_i &= \frac{1}{N\theta}\cos\left(2\pi(i-i_0)/N\right) \\ b_i &= \frac{1}{N\theta}\sin\left(2\pi(i-i_0)/N\right) \\ a_i^\dagger &= \frac{\theta}{2}\cos\left(2\pi(i-i_0)/N\right) \\ b_i^\dagger &= \frac{\theta}{2}\sin\left(2\pi(i-i_0)/N\right). \end{aligned} \tag{S56}$$

As above, $i_0=(N+1)/2$, θ is a scale parameter which, along with ν_0, adjusts the mean firing rate of the output population. Recall that this choice is more or less arbitrary so long as the vectors **a**, **b**, $\mathbf{a}^\dagger$, and $\mathbf{b}^\dagger$ are orthogonal to the vector of ones and span a two dimensional vector space. With this choice for parameters, the mean firing rate, $\bar{\nu} \equiv \mathbf{1}\cdot\boldsymbol{\nu}/N$, evolves according to

$$\frac{d\bar{\nu}}{dt} = -\sigma_\eta^2\left(\mathbf{a}\cdot\boldsymbol{\rho}\right)\bar{\nu} + \nu_0 - \bar{\rho} \tag{S57}$$

where $\bar{\rho} \equiv \mathbf{1}\cdot\boldsymbol{\rho}/N$ is the empirical average of the output population activity. In steady state ($d\bar{\nu}/dt = 0$), with $\bar{\rho}$ replaced by $\bar{\nu}$, $\bar{\nu}$ is given by

$$\bar{\nu} \approx \frac{\nu_0}{1+\sigma_\eta^2\mathbf{a}\cdot\boldsymbol{\nu}}. \tag{S58}$$

Thus, ν_0 can be used to raise the average firing rate, and thus ensure that the minimum firing rate is nonnegative.

Specifying $\mathbf{a}^{in}$ and $\mathbf{b}^{in}$ is not completely straightforward, as our choice must result in Poisson spikes in the limit $\Delta t \to 0$. However, it turns out that a valid choice is, as for equation (S26),

$$\begin{aligned} \mathbf{a}^{in} &= \frac{\mathbf{1}}{\sigma_w^2} \\ \mathbf{b}^{in} &= \frac{\mathbf{s}^0}{\sigma_w^2} \end{aligned} \tag{S59}$$

where s_i^0 are a set of evenly spaced points which will turn out to correspond to the preferred stimuli of the neurons (see equation (S61) below) and, recall, **1** is a vector of all 1's. To see that this is correct, we need to choose the prefactor, $\phi\left(\mathbf{r}^{in}(t), g(t)\right)$, that appears in equation (S37), and then take the limit $\Delta t \to 0$. The appropriate prefactor is (suppressing the time dependence in $\mathbf{r}^{in}$ and g)

$$\phi\left(\mathbf{r}^{in}, g\right) = \exp\left(\sum_i r_i^{in}\log\left(g\Delta t\right) - g\Delta t\sum_i f_i\left(s\right)\right)\prod_i \frac{1}{r_i^{in}!}, \tag{S60}$$

where the $f_i(s)$ are tuning curves, which we take to be Gaussian,

$$f_i\left(s(t)\right) = \exp\left(-\frac{\left(s(t) - s_i^0\right)^2}{2\sigma_w^2}\right). \tag{S61}$$

Here, s_i^0 is the preferred stimulus value of neuron i and σ_w is the tuning curve width. Note that this choice of generative model for the input spikes does not technically lead to evidence which can be modelled by equations (S37) and (S38), unless preferred stimuli s_i^0 are such that $\sum_i f_i(s)$ is approximately independent of s on the interval in which $s(t)$ tends to live. An easy way to insure that this is the case is to choose evenly and closely spaced preferred stimuli, s_i^0, which span the interval which covers the 99% confidence region associated with the prior $p(s)$.

Given equations (S59)-(S61), it is straightforward to show that

$$\rho_i^{in}(t) \sim Poisson\,process\left(g(t) f_i\left(s(t)\right)\right). \tag{S62}$$

Generalization to higher dimensions

For the hand-tracking problem that we consider in the main text, we need to know how a network could implement a Kalman filter in higher dimensions. For a D dimensional Kalman filter, we can apply the same analysis, and the result is essentially the same: a network with divisive normalization and a quadratic nonlinearity is sufficient to implement the filter. Here we sketch the analysis. We start with the evolution equation for the vector-valued stimulus, $\mathbf{s}(t)$, which is given by

$$\frac{d\mathbf{s}}{dt} = -\mathbf{\Gamma s} + \mathbf{u}(t) + \boldsymbol{\eta}(t) \tag{S63}$$

where Γ is a DxD, positive-definite matrix, $\boldsymbol{\eta}(t)$ is Gaussian and white with DxD covariance matrix $\mathbf{\Sigma}_{\boldsymbol{\eta\eta}}$ (i.e., $\left\langle \boldsymbol{\eta}(t)\boldsymbol{\eta}(t')\right\rangle = \mathbf{\Sigma}_{\boldsymbol{\eta\eta}}\delta(t-t')$), and, to increase generality, we have added a control signal, $\mathbf{u}(t)$. Equation (S63) is the D dimensional generalization of equation (S35), but with the addition of a control signal, $\mathbf{u}(t)$. The probability distribution for the evidence $\mathbf{r}^{in}$ given $\mathbf{s}(t)$ is written

$$p\left(\mathbf{r}^{in}(t) \mid \mathbf{s}(t), \mathbf{g}(t)\right) = \phi\left(\mathbf{r}^{in}(t), \mathbf{g}(t)\right)\exp\left(\mathbf{h}^{in}\left(\mathbf{s}(t)\right)\cdot\mathbf{r}^{in}(t)\right), \tag{S64}$$

where, analogous to equation (S38), $\mathbf{h}^{in}(\mathbf{s}(t))\cdot\mathbf{r}^{in}(t)$, is given by

$$\mathbf{h}^{in}\left(\mathbf{s}(t)\right)\cdot\mathbf{r}^{in}(t) = -\frac{1}{2}\sum_{ij} s_i(t)s_j(t)\mathbf{a}_{ij}^{in}\cdot\mathbf{r}^{in}(t) + \sum_i s_i(t)\mathbf{b}_i^{in}\cdot\mathbf{r}^{in}(t). \qquad \text{(S65)}$$

Equations (S64) and (S65) are *D* dimensional generalizations of equations (S37) and (S38).

The natural parameters of the posterior are given by $\mathbf{I}$, the inverse of the covariance matrix of **s**, and $\mathbf{I}\cdot\boldsymbol{\mu}$, the product of this matrix with the mean of **s**, the latter denoted $\boldsymbol{\mu}$. (We use **I** for the inverse of the covariance matrix because it corresponds to the *D*x*D* specific Fisher information, also known as the precision matrix.) When these natural parameters are encoded in a linear PPC they are linearly related to a rate parameter, **v**. To see why, note that if we were to write

$$\mathbf{h}\left(\mathbf{s}(t)\right)\cdot\mathbf{v}(t) = -\frac{1}{2}\sum_{ij} s_i(t)s_j(t)\mathbf{a}_{ij}\cdot\mathbf{v}(t) + \sum_i s_i(t)\mathbf{b}_i\cdot\mathbf{v}(t), \qquad \text{(S66)}$$

and $p(\mathbf{v}\,(t)|\mathbf{s}(t)) \sim \exp(\mathbf{h}(\mathbf{s}(t))\cdot\,\mathbf{v}\,(t))$, then we would have

$$\begin{aligned} I_{ij}(t) &= \mathbf{a}_{ij}\cdot\mathbf{v}(t) \\ \sum_j I_{ij}(t)\mu_j(t) &= \mathbf{b}_i\cdot\mathbf{v}(t). \end{aligned} \qquad \text{(S67)}$$

In these equations, $\mathbf{a}_{ij}=\mathbf{a}_{ji}$ so that the precision matrix **I**(*t*) is guaranteed to be symmetric, and the vectors $\mathbf{a}_{ij}$ and $\mathbf{b}_i$ must be chosen so that they span a $\underline{D}+D(\underline{D}+1)/2$ dimensional space (one dimension for each of the natural parameters of the *D* dimensional Gaussian posterior).

Repeating the above analysis for multiple dimensions in the presence of the control signal **u**(*t*) yields evolution equations for the natural parameters, which take the form

$$\begin{aligned} \frac{d\left(\mathbf{I}\cdot\boldsymbol{\mu}\right)}{dt} &= \boldsymbol{\Gamma}^{\mathrm{T}}\cdot\left(\mathbf{I}\cdot\boldsymbol{\mu}\right) + \mathbf{I}\cdot\mathbf{u}(t) - \mathbf{I}\cdot\boldsymbol{\Sigma}_{\boldsymbol{\eta\eta}}\cdot\left(\mathbf{I}\cdot\boldsymbol{\mu}\right) + \left(\mathbf{I}\cdot\boldsymbol{\mu}\right)^{in} \\ \frac{d\mathbf{I}}{dt} &= \mathbf{I}\cdot\boldsymbol{\Gamma} + \boldsymbol{\Gamma}^{\mathrm{T}}\cdot\mathbf{I} - \mathbf{I}\cdot\boldsymbol{\Sigma}_{\boldsymbol{\eta\eta}}\cdot\mathbf{I} + \mathbf{I}^{in}. \end{aligned} \qquad \text{(S68)}$$

This is a generalization of equation (S43). We may then obtain rate equations by inserting equation (S67) into (S68), which yields

$$\begin{aligned} \frac{d\mathbf{a}_{ij}\cdot\mathbf{v}}{dt} &= \left(\sum_k \gamma_{kj}\mathbf{a}_{ik} + \gamma_{ki}\mathbf{a}_{kj}\right)\cdot\mathbf{v} - \mathbf{v}\cdot\left(\sum_{kl}\mathbf{a}_{ik}\sigma_{kl}\mathbf{a}_{lj}\right)\cdot\mathbf{v} + \mathbf{a}_{ij}^{in}\cdot\boldsymbol{\rho}^{in} \\ \frac{d\mathbf{b}_i\cdot\mathbf{v}}{dt} &= \left(\sum_k \gamma_{kl}\mathbf{b}_k\right)\cdot\mathbf{v} + \left(\sum_k u_k(t)\mathbf{a}_{ik}\right)\cdot\mathbf{v} - \mathbf{v}\cdot\left(\sum_{kl}\mathbf{a}_{ik}\sigma_{kl}\mathbf{b}_l\right)\cdot\mathbf{v} + \mathbf{b}_i^{in}\cdot\boldsymbol{\rho}^{in} \end{aligned} \qquad \text{(S69)}$$

where γ_{ij} and σ_{ij} are the components of the matrices $\boldsymbol{\Gamma}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\eta\eta}}$, respectively. Recall that while the subscripts can take on the values *i,j*=1..*D*, these equations are redundant since we have constrained $\mathbf{a}_{ij}=\mathbf{a}_{ji}$. Consequently, for a two dimensional state vector **s**(*t*), equation (S69) consists of five independent equations of evolution.

This set of equations can be written as an evolution equation for $\mathbf{v}$ by defining adjoint vectors $\mathbf{a}_{ij}^{\dagger}$, $\mathbf{b}_i^{\dagger}$ and $\mathbf{c}^{\dagger}$ according to

$$\begin{aligned}
&2\mathbf{a}_{ij}\cdot\mathbf{a}_{kl}^{\dagger}=\delta_{ik}\delta_{jl}+\delta_{il}\delta_{jk}\\
&\mathbf{a}_{lk}^{\dagger}=\mathbf{a}_{kl}^{\dagger}\quad \mathbf{b}_i\cdot\mathbf{b}_j^{\dagger}=\delta_{ij}\\
&\mathbf{a}_{ij}\cdot\mathbf{b}_k^{\dagger}=\mathbf{b}_i\cdot\mathbf{c}^{\dagger}=\mathbf{a}_{ij}\cdot\mathbf{c}^{\dagger}=0
\end{aligned}\tag{S70}$$

where $\mathbf{a}_{ij}^{\dagger}$ and $\mathbf{b}_i^{\dagger}$ are constrained to lie in the D+D(D+1)/2-dimensional subspace spanned by $\mathbf{a}_{ij}$ and $\mathbf{b}_i$. This yields

$$\begin{aligned}
\frac{d\mathbf{v}}{dt}=&\sum_{ij}\mathbf{a}_{ij}^{\dagger}\left(\left(\sum_k\gamma_{kj}\mathbf{a}_{ik}+\gamma_{ki}\mathbf{a}_{kj}\right)\cdot\mathbf{v}-\mathbf{v}\cdot\left(\sum_{kl}\mathbf{a}_{ik}\sigma_{kl}\mathbf{a}_{lj}\right)\cdot\mathbf{v}+\mathbf{a}_{ij}^{in}\cdot\boldsymbol{\rho}^{in}\right)\\
&+\sum_i\mathbf{b}_i^{\dagger}\left(\left(\sum_k\gamma_{ki}\mathbf{b}_k\right)\cdot\mathbf{v}+\left(\sum_k u_k(t)\mathbf{a}_{ik}\right)\cdot\mathbf{v}-\mathbf{v}\cdot\left(\sum_{kl}\mathbf{a}_{ik}\sigma_{kl}\mathbf{b}_l\right)\cdot\mathbf{v}+\mathbf{b}_i^{in}\cdot\boldsymbol{\rho}^{in}\right)\\
&+\mathbf{c}^{\dagger}f_c\left(\mathbf{v},\boldsymbol{\rho}^{in}\right)
\end{aligned}\tag{S71}$$

where $f_c\left(\mathbf{v},\boldsymbol{\rho}^{in}\right)$ is any function of $\mathbf{v}$ and $\boldsymbol{\rho}^{in}$. That equation (S71) satisfies equation (S69) can be verified by direct substitution. As above, $\boldsymbol{\rho}^{in}$ corresponds to delta-functions at the times of the input spikes.

Equivalently, we can define feedforward and recurrent connectivity by

$$\begin{aligned}
\mathbf{M}&=\sum_{ij}\mathbf{a}_{ij}^{\dagger}\mathbf{a}_{ij}^{in}+\sum_i\mathbf{b}_i^{\dagger}\mathbf{b}_i^{in}\\
\mathbf{W}&=\sum_{ijk}\left(\gamma_{kj}\mathbf{a}_{ij}^{\dagger}\mathbf{a}_{ik}+\gamma_{ki}\mathbf{a}_{ij}^{\dagger}\mathbf{a}_{kj}\right)+\sum_{ik}\gamma_{ki}\mathbf{b}_i^{\dagger}\mathbf{b}_k\\
\mathbf{U}(t)&=\sum_{ij}u_j(t)\mathbf{b}_i^{\dagger}\mathbf{a}_{ij}\\
\mathbf{Q}^{(3)}&=\sum_{ijkl}\sigma_{kl}\mathbf{a}_{ik}\mathbf{a}_{ij}^{\dagger}\mathbf{a}_{lj}+\sum_{ikl}\sigma_{kl}\mathbf{a}_{ik}\mathbf{b}_i^{\dagger}\mathbf{b}_l
\end{aligned}\tag{S72}$$

which allows us to write

$$\frac{d\mathbf{v}}{dt}=\mathbf{W}\cdot\mathbf{v}+\mathbf{U}(t)\cdot\mathbf{v}-\mathbf{v}\cdot\mathbf{Q}^{(3)}\cdot\mathbf{v}+\mathbf{M}\cdot\boldsymbol{\rho}^{in}+\mathbf{1}\left(\nu_0-\frac{\mathbf{1}\cdot\mathbf{v}}{N}\right)\tag{S73}$$

where we have chosen $\mathbf{c}^{\dagger}=\mathbf{1}$ and $f_c(\mathbf{v})=\nu_0-\mathbf{1}\cdot\mathbf{v}/N$ (recall that N is the number of neurons, and so the dimensionality of the rate vector, $\mathbf{v}$) and that the parameter ν_0 can be used to adjust the mean firing rate of the population so that all components of $\mathbf{v}(t)$ remain positive.

The network implementation of the two dimensional case mimics the one dimensional example described above, although it is slightly more complicated. For the two dimensional network, we use

$$\mathbf{a}_{11} = \begin{bmatrix} \mathbf{a} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{a}_{22} = \begin{bmatrix} \mathbf{0} \\ \mathbf{a} \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{a}_{12} = \mathbf{a}_{21} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{a} \end{bmatrix},$$
$$\mathbf{b}_1 = \begin{bmatrix} \mathbf{b} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{b}_2 = \begin{bmatrix} \mathbf{0} \\ \mathbf{b} \\ \mathbf{0} \end{bmatrix} \tag{S74}$$

where the vectors **a** and **b** are given in equation (S56). In the 2*D* simulations presented in this work, $\mathbf{a}_{ij}^{in}$ and $\mathbf{b}_i^{in}$ were also obtained by analogue to the one dimensional case (in particular equation (S59)-(S62)). Specifically, we assumed that information about s_1 and s_2 came from two independent population of Poisson spiking neurons with Gaussian tuning curves; i.e.

$$\begin{aligned} \rho_{1,i}^{in}(t) &\sim Poisson\,process\left(gf_i\left(s_1(t)\right)\right) \\ \rho_{2,i}^{in}(t) &\sim Poisson\,process\left(gf_i\left(s_2(t)\right)\right) \end{aligned} \tag{S75}$$

Where

$$f_i\left(s\right) = \exp\left(-\frac{\left(s - s_i^0\right)^2}{2\sigma_w^2}\right). \tag{S76}$$

Comparing equations (S75) and (S76) to (S61) and(S62), we see that $\mathbf{a}_{ij}^{in}$ and $\mathbf{b}_i^{in}$ are given by

$$\mathbf{a}_{11}^{in} = \begin{bmatrix} \mathbf{a}^{in} \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{a}_{22}^{in} = \begin{bmatrix} \mathbf{0} \\ \mathbf{a}^{in} \end{bmatrix}, \quad \mathbf{a}_{12}^{in} = \mathbf{a}_{21}^{in} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix},$$
$$\mathbf{b}_1 = \begin{bmatrix} \mathbf{b}^{in} \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{b}_2 = \begin{bmatrix} \mathbf{0} \\ \mathbf{b}^{in} \end{bmatrix}, \tag{S77}$$

where the components of $\mathbf{a}^{in}$ and $\mathbf{b}^{in}$ are given by equation (S59) with

$$\mathbf{r}^{in}(t) = \begin{bmatrix} \mathbf{r}_1^{in}(t) \\ \mathbf{r}_2^{in}(t) \end{bmatrix}. \tag{S78}$$

As an aside, we note that, while this analysis only considered Gaussian posterior distributions, a derivation based upon the Fokker-Plank equation which does not make this assumption also leads to network with quadratic dynamics in the rate domain, but can require a stimulus dependent kernel, **h**(*s*), which spans a much larger basis.

Olfaction

A simple model of olfaction is shown in Fig. S3. For this model, we modify, and expand, our notion of what a stimulus is: the s_k in Fig. S3 are now binary variables, denoting the presence (s_k=1) or absence (s_k=0) of odors (or anything else; we use odors for concreteness); the c_k denote concentrations. We have also introduced intermediate variables, odorant receptors, denoted o_i in Fig. S3. The activation of these variables is a measure of the actual substances in the air.

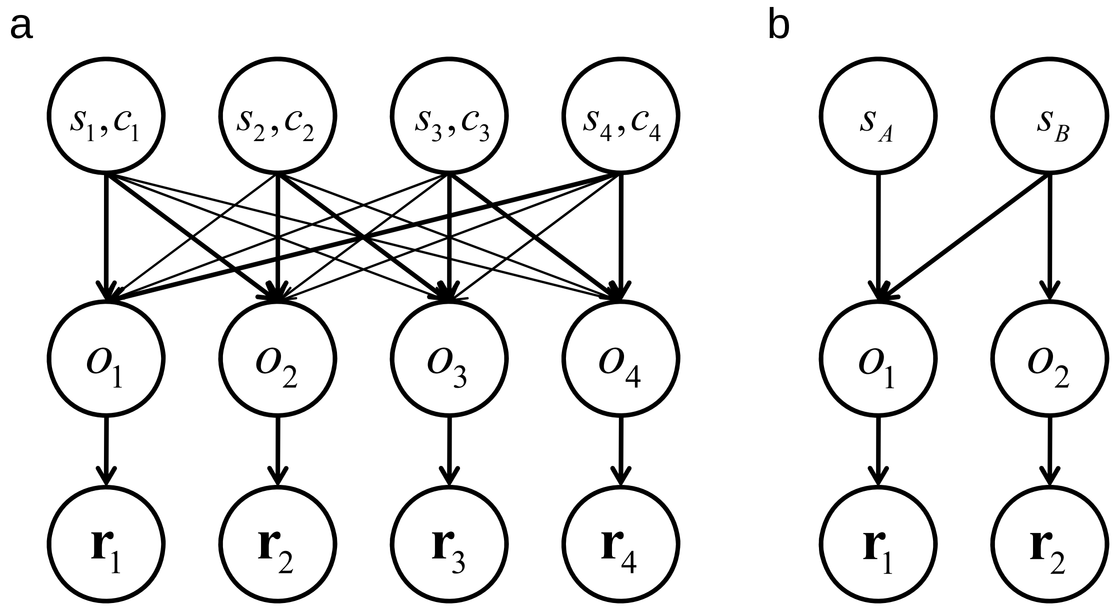

Figure S3. (a) Graphical model for olfaction. Top row: the s_k are binary variables which indicate the presence or absence of an odor. The associated c_k represent the intensity of the odors on each trial. Second row: the odorants, o_i, are real valued quantities which represent actual substances in the air; in this model, they are given by a mixture of odors. Bottom row: the $\mathbf{r}_i$ are the population patterns of activity which provides a linear PPC representation of the odorants. (b) Graphical model describing the blicket experiment. Here, s_A and s_B are binary variables which indicate whether or not the objects A and B are blickets. Two experiments are performed with outcomes o_1 and o_2. In experiment 1 both objects are placed on the detector. In experiment 2 only object B is placed on the detector.

The full encoding model, $p(\mathbf{r}=\{\mathbf{r}_1,\mathbf{r}_2,...\}|\mathbf{s}, \mathbf{c})$, is given by

$$p(\mathbf{r} \mid \mathbf{s},\mathbf{c}) = \int d\mathbf{o} \prod_i p(\mathbf{r}_i \mid o_i) p(\mathbf{o} \mid \mathbf{s},\mathbf{c}), \tag{S79}$$

where bold quantities refer to the whole collection of firing rates, stimuli or concentrations (e.g., $\mathbf{s}=(s_1, s_2, ...)$). We assume, as usual, that the transformation from odorants to spike counts is given by a linear PPC. Here we specialize immediately to Poisson encoding,

$$p\left(\mathbf{r}_i \mid o_i\right) = Poisson\left(o_i \mathbf{f}_i + \mathbf{f}_b\right), \tag{S80}$$

for fixed vectors $\mathbf{f}_i$ and $\mathbf{f}_b$. This equation gives the population response of the neurons which are sensitive to odorant i. We also assume that $\mathbf{r}_i$ is independent of o_j for $i \neq j$. We take the transformation from odors to odorants to be a (deterministic) linear combination of the presence or absence of the odors weighted by the concentrations,

$$o_i = \sum_k w_{ik} c_k s_k . \tag{S81}$$

Because $\mathbf{o}$ is a deterministic function of $\mathbf{c}$ and $\mathbf{s}$, the integral over odorants in equation (S79) is trivial (basically, replace o_i by the sum above wherever it appears). This makes it easy to write down the full posterior, which is given by

$$p\left(\mathbf{s},\mathbf{c} \mid \mathbf{r}\right) = \frac{1}{Z} \exp\left(\sum_{ij} r_{ij} \log\left(f_{ij} \sum_k w_{ik} c_k s_k + f_{bj} \right) - \sum_{ijk} f_{ij} w_{ik} c_k s_k + f_{bj} \right) p(\mathbf{s},\mathbf{c}) \tag{S82}$$

where $p(\mathbf{s},\mathbf{c})$ is the prior on the odors and their concentrations. Note that the posterior is not a linear PPC as the sum over ijk inside the exponent typically depends on the stimulus and the concentration

The quantity of interest is the marginal distribution of the presence of a particular odor, independent of its concentration and the concentrations and presence or absence of the other odors. The true marginal for oder k given by

$$p_{true}\left(s_k \mid \mathbf{r}\right) = \sum_{\mathbf{s} \backslash k} \int d\mathbf{c}\, p\left(\mathbf{s},\mathbf{c} \mid \mathbf{r}\right) \tag{S83}$$

where the notation $\mathbf{s}\backslash k$ means sum over all odors except s_k and $p\left(\mathbf{s},\mathbf{c} \mid \mathbf{r}\right)$ is given in equation (S82). We now seek a transformation,

$$\mathbf{r}_k^{out} = \mathbf{f}_k\left(\mathbf{r}\right), \tag{S84}$$

such that $\mathbf{r}_k^{out}$ codes, probabilistically, for s_k. As usual, we encode the marginal distribution on s_1 using a linear PPC,

$$p(s_k \mid \mathbf{r}_k^{out}) = \frac{\exp\left(\mathbf{h}_k(s_k) \cdot \mathbf{r}_k^{out}\right)}{Z\left(\mathbf{r}_k^{out}\right)}. \tag{S85}$$

Note that because s_k is a binary variable, we may write its posterior as

$$p(s_k \mid \mathbf{r}_k^{out}) = \frac{\exp\left(s_k \mathbf{a}_k \cdot \mathbf{r}_k^{out}\right)}{1+\exp\left(s_k \mathbf{a}_k \cdot \mathbf{r}_k^{out}\right)} \tag{S86}$$

where $\mathbf{a}_k \equiv \mathbf{h}_k(1) - \mathbf{h}_k(0)$.

Unfortunately, the true marginal distribution cannot be represented as a linear PPC. Nor can the parameters of a linear PPC approximation to the true posterior be computed explicitly, as was done for the Gaussian case. What we do, therefore, is test, in the main text, the performance of a variety of network transformations. We find that, as with the two problems considered above, a quadratic nonlinearity with divisive normalization provides the best performance.

The blicket task is a special case of the olfaction task in which there are only two odours, s_A and s_B and two odorants, o_1 and o_2. The difference here is that the two odourants now correspond to the two observations. For the first observation, both objects are placed upon the blicket detector and so $w_{11}=w_{12}=1$. For the second experiment only object B is placed upon the detector and so $w_{22}=1$ and $w_{21}=0$. Note the concentration nuisance parameters c_k appear in this equation despite their absence in the generative model for the blicket detection task. This is because that task had no notion of difficulty. In this version, the c_k are used to model talk difficulty by increasing or decreasing the probability that blicket detector will sound in the presence of a blicket.

Statistical Verification

In this section we assess the performance of the optimal networks derived above, and a set of suboptimal networks that have been proposed in the literature (described in detail below). The natural measure of performance is the KL distance between the true posterior and the posterior recovered by the network. Because this measure provides an upper bound on how much information is lost[1], we normalize it by the true mutual information between the stimulus and response. We also add one additional twist: we compute this normalized ratio for each value of the gain, $\boldsymbol{g}$, and then average. This normalized ratio, denoted $\Delta I(g)/I(g)$, is given by

$$\frac{\Delta I(g)}{I(g)} = \frac{\left\langle \log \dfrac{p_{true}\left(s' \mid \mathbf{r},\mathbf{g}\right)}{p\left(s' \mid \mathbf{r}' = \mathbf{f}(\mathbf{r})\right)} \right\rangle_{p_{true}(s',\mathbf{r}\mid\mathbf{g})}}{\left\langle \log \dfrac{p_{true}\left(s' \mid \mathbf{r},\mathbf{g}\right)}{p_{true}\left(s' \mid \mathbf{g}\right)} \right\rangle_{p_{true}(s',\mathbf{r}\mid\mathbf{g})}}, \tag{S87}$$

where the brackets represent an average over the subscripted probability distribution. Bar plots in the main text report the average of this gain dependent measure over the prior on **g** (or **c** in the explaining away example), a quantity we denote Δ*I*/*I*; it is given by

$$\frac{\Delta I}{I} = \left\langle \frac{\Delta I(g)}{I(g)} \right\rangle_{p(\mathbf{g})} . \tag{S88}$$

Note that equation (S88) is not the only possible measure of performance; an alternative is to separately average the numerator and denominator over gain, rather than averaging their ratio. That alternative, however, effectively suppresses information loss for low gain samples (since they tend to have low information), and this low information/low gain situations is precisely where we expect the probabilistic nature of the code to be most important.

For the coordinate transformation and Kalman filter described above, we know both the network transformation and the associated posterior, so computing the right hand side of equation (S88) is simply a matter of computing, numerically, averages with respect to $p(\mathbf{g})$. When we consider suboptimal networks, however, we don't know the parameterization of the posterior, and typically we know only the *form* of the transformation from **r** to **r′**. Consequently, we have to find both the best estimate of the posterior, and the transformation, numerically. This is accomplished by assuming that the both the posterior and the transformation are in a parameterized class, and choose the parameters to minimizing Δ*I*/*I*.

Specifically, for a suboptimal network that implements the transformation $\mathbf{r}'=\mathbf{f}_{sub}(\mathbf{r},\Theta)$ where Θ is a set of parameter, we assume that the posterior associated with the network is given by

$$p\left(s' \mid \mathbf{r}' = \mathbf{f}_{sub}\left(\mathbf{r},\Theta\right)\right) \propto \exp\left(-\frac{s'^2}{2}\mathbf{a}_{sub}\cdot\mathbf{f}_{sub}\left(\mathbf{r},\Theta\right) + s'\mathbf{b}_{sub}\cdot\mathbf{f}_{sub}\left(\mathbf{r},\Theta\right)\right). \tag{S89}$$

For olfaction and the blicket experiment, on the other hand, because the variables of interest are binary, we assume that

$$p\left(s' \mid \mathbf{r}' = \mathbf{f}_{sub}\left(\mathbf{r},\Theta\right)\right) \propto \exp\left(s'\mathbf{a}_{sub}\cdot\mathbf{f}_{sub}\left(\mathbf{r},\Theta\right)\right). \tag{S90}$$

In both cases we choose the parameters Θ, $\mathbf{a}_{sub}$ and $\mathbf{b}_{sub}$ to minimize Δ*I*/*I*, as defined in equation (S88). Note that the parameters of interest are $\mathbf{a}_{sub}\cdot\mathbf{f}_{sub}(\mathbf{r},\Theta)$ and $\mathbf{b}_{sub}\cdot\mathbf{f}_{sub}(\mathbf{r},\Theta)$.

For the suboptimal networks we also introduce what we call specialized decoders – decoders that "know" the gain or gains. For these decoders, we allow $\mathbf{a}_{sub}$ and $\mathbf{b}_{sub}$ in equation (S89) to depend on gain, **g**. We then choose $\mathbf{a}_{sub}(\boldsymbol{g})$, $\mathbf{b}_{sub}(\boldsymbol{g})$ and Θ to minimize the KL distance between $p_{true}(s' \mid \mathbf{r},\mathbf{g})$ and $p(s' \mid \mathbf{r}' = \mathbf{f}_{sub}(\mathbf{r}),\mathbf{g})$. In this case, the average over gain in the right hand side of equation (S87) is an average over the specialized decoders; that is, there is a different parameterization of the network posterior for each value of the gain. Note that many previous investigations[2-4] which have claimed to have constructed networks that implement Bayes optimal computations have not considered gain as a nuisance parameter. Thus, they have effectively used specialized decoders. Consequently, to maintain optimality, their networks must adjust their parameters whenever gain is altered, while our networks do not.

In what follows, we explicitly compute $\Delta I/I$ for the models described above, and for a set of suboptimal models. The format of the next several sections is as follows:

1. We translate from s', $\mathbf{r}'$ and $\mathbf{r}$ (which are really placeholders; see the Introduction) to the variables of interest.
2. For a given $p(s'|s)$ we compute the true posterior $p_{true}(s'|\mathbf{r},\mathbf{g})$ and, for the coordinate transformation and Kalman filter cases, we compute the optimal network transformation $\mathbf{r}'=\mathbf{f}(\mathbf{r})$.
3. We provide details of the model (e.g., number of neurons, parameters of the model), and we specify the prior over the gain parameters, $\mathbf{g}$.
4. We give details of the suboptimal models.

Linear coordinate transformations, $s_3 = s_1 + s_2$.

1. Translating to the relevant variables.

$s'=s_3$, $\mathbf{r}'=\mathbf{r}_3$, and $\mathbf{r}=(\mathbf{r}_1, \mathbf{r}_2)$.

2. Posteriors.

$p_{true}(s_3|\mathbf{r}_1,\mathbf{r}_2,\mathbf{g})$ is given by equation (S19) (note that it is independent of $\mathbf{g}$). $p(s_3|\mathbf{r}_1,\mathbf{r}_2,\mathbf{g})$ is also given by equation (S19). This posterior, like the true one, is independent of $\mathbf{g}$.

3. Model details.

We used 20 neurons in each input population, 20 neurons in the output population, and the preferred directions, s_0^j (see equation (S23)) ranged from -5 to 5 in steps of 10/19 and the width of the tuning curve was given by $\sigma_w^2 = 1$. The gain parameters, g_i, were uniformly distributed between 1 and 15, corresponding to standard deviations ranging from 0.172 to 0.666 (i.e., population activity generated with gain=1 produced a likelihood with width 0.666, on average). For the priors over s_1 and s_2, we used $\alpha_{p1}=\alpha_{p2}=1$. The stimulus dependent kernel for the output population was given by equations (S27)-(S29) with $\sigma_{3w}^2=1$, $\theta_1 = 1/20$, $\theta_2 = 10$ and $f_3 = 1$.

4. Suboptimal models.

We considered three suboptimal models: quadratic, linear threshold and linear. All three have the form

$$r_{3ij} = \varphi_{ij}(\mathbf{r}_1, \mathbf{r}_2, \Theta). \quad \text{(S91)}$$

The difference between the models lies in the nonlinearity, $\varphi_{ij}(\mathbf{r}_1, \mathbf{r}_2, \Theta)$. For the quadratic mapping,

$$\varphi_{ij}(\mathbf{r}_1, \mathbf{r}_2) = r_{1i} r_{2j}. \quad \text{(S92)}$$

For the linear threshold model,

$$\varphi_{ij}(\mathbf{r}_1, \mathbf{r}_2, \Theta). = [r_{1i} + r_{2j} - \theta]_+. \quad \text{(S93)}$$

where $[\cdot]_+$ is the linear threshold function, $[x]_+ = \max(0, x)$. For the linear model, $\mathbf{r}_3 = [\mathbf{r}_1; \mathbf{r}_2]$, where the bracket notation on the right hand side of this equation indicates a concatenation.

In all three cases, the two quantities that make up the posterior, $\mathbf{a}_{sub} \cdot \mathbf{f}_{sub}(\mathbf{r}, \theta)$ and $\mathbf{b}_{sub} \cdot \mathbf{f}_{sub}(\mathbf{r}, \Theta)$ are given by (dropping the "sub" subscript for clarity),

$$\begin{aligned} \mathbf{a} \cdot \mathbf{f} &= \sum_{jk} A_{jk} \varphi_{ij}(\mathbf{r}_1, \mathbf{r}_2) \\ \mathbf{b} \cdot \mathbf{f} &= \sum_{jk} B_{jk} \varphi_{ij}(\mathbf{r}_1, \mathbf{r}_2). \end{aligned} \quad \text{(S94)}$$

For the three models, A_{jk} and B_{jk} were chosen to minimize $\Delta I/I$; for the threshold nonlinearity, we also optimized with respect to θ. The results are presented in Fig. 3b-c of the main text. We also verified that a nonlinear mapping was necessary by considering the set of all possible linear combinations of neural activity.

Non-linear coordinate transformations (*estimating hand position; see equation* (S33)).

1. Translating to the relevant variables.

$s' = s_3$, $\mathbf{r}' = \mathbf{r}_3$, and $\mathbf{r} = (\mathbf{r}_1, \mathbf{r}_2)$.

2. Posteriors.

In this case, $p_{true}(s_3 \mid \mathbf{r}_1, \mathbf{r}_2, g_1, g_2)$, is obtained by numerical integration of the joint posterior $p_{true}(s_1, s_2 \mid \mathbf{r}_1, \mathbf{r}_2, g_1, g_2)$ subject to the constraints defined by the coordinate transformation given in equation (S33). This joint distribution is defined by the likelihood function for population responses conditioned on s_1 and s_2. These population responses are generated from Poisson spiking neurons with circular Gaussian tuning curves, i.e. $r_{ij} \sim Poisson(f_j(s_i))$, where

$$f_j(s) = g_i \exp\left(K_i \left(\cos(s - s_0^j) - 1\right)\right). \quad \text{(S95)}$$

3. Model details.

We used 10 neurons in each input population. The preferred directions, s_0^j, ranged from -π to π in steps of 2π/10. Concentration parameters K_i were set to 1. The gain parameters, g_i, were uniformly distributed between 5 and 20, corresponding to standard deviations ranging from 0.127 to 0.254 (i.e., population activity generated with gain=5 produced a likelihood with width 2×0.127 radians, on average). For the priors over s_1 and s_2, we used circular Gaussians with mode at zero and concentration parameters K_{pi}= 1, for i=1,2.

4. Suboptimal models.

Since none of the models we considered were optimal for this case, we considered four suboptimal models: quadratic, linear threshold, linear, quadratic with divisive normalization. All four have the same form as in equation (S91), $r_{3ij} = \varphi_{ij}(\mathbf{r}_1,\mathbf{r}_2,\Theta)$. Once again, the difference between the models lies in the nonlinearity, $\varphi_{ij}(\mathbf{r}_1,\mathbf{r}_2,\Theta)$. For the quadratic mapping, $\varphi_{ij}(\mathbf{r}_1,\mathbf{r}_2,\Theta)$ is given by equation (S92); for the linear threshold model by equation (S93), and, as above, for the linear model, $\boldsymbol{\varphi}(\mathbf{r}_1,\mathbf{r}_2,\Theta) = [\mathbf{r}_1;\mathbf{r}_2]$, where again the notation indicates a concatenation. For the quadratic model with divisive normalization we used

$$\varphi_{ij}(\mathbf{r}_1,\mathbf{r}_2,\Theta) = \frac{r_{1i}r_{2j}}{\boldsymbol{\theta}_1\cdot\mathbf{r}_1 + \boldsymbol{\theta}_2\cdot\mathbf{r}_1 + \theta_0}. \tag{S96}$$

In all four cases the posterior was assumed to take the form of a circular Gaussian, i.e.

$$\begin{aligned}\log p(s_3 \mid \mathbf{r}_3) &= \mathbf{a}\cdot\mathbf{r}_3\cos(s_3) + \mathbf{b}\cdot\mathbf{r}_3\sin(s_3) + c\\ \mathbf{a}\cdot\mathbf{r}_3 &= \sum_{ij} A_{ij}\varphi_{ij}(\mathbf{r}_1,\mathbf{r}_2,\Theta)\\ \mathbf{b}\cdot\mathbf{r}_3 &= \sum_{ij} B_{ij}\varphi_{ij}(\mathbf{r}_1,\mathbf{r}_2,\Theta),\end{aligned} \tag{S97}$$

where Θ represents the parameters of the function $\varphi_{ij}(\mathbf{r}_1,\mathbf{r}_2,\Theta)$. For the four models, A_{jk}, B_{jk} and Θ were chosen via gradient descent to minimize Δ*I*/*I*. The results are presented in Fig. 3b-c of the main text. We also verified that a nonlinear mapping was necessary by considering the set of all possible linear combinations of neural activity.

Kalman Filter (*1D*)

1. Translating to the relevant variables.

$s'=s(t)$, $\mathbf{r}'=\mathbf{r}(t)$ (equation (S55) with δt=10 ms), $s=s(t-\delta t)$ and $\mathbf{r}=\mathbf{r}^{in}(t)$.

2. Posteriors.

There is no direct expression for the posterior distribution of $s(t)$ given the input spike train. To find the true posterior, it is necessary to integrate equation (S48) to find $\mathbf{v}(t)$, and then insert that into equation (S45). For the approximate posterior, it is necessary to integrate equation (S54), determine $\mathbf{r}(t)$ from equation (S55), and again insert that into equation (S45) in place of $\mathbf{v}(t)$.

3. Model details.

We used 20 neurons in the input layer (the layer coding for $\mathbf{r}^{in}$) and 200 in the output layer (the layer coding for $\mathbf{r}$). The activity in the input population was very sparse, with neurons firing at an average rate of 1.5 spikes per second. Preferred stimuli, s_i^0, for input neurons were evenly spaced on the interval -4 to 4 and the tuning width, σ_w^2, was set to 1. To sample from the true posterior, we evolved the stochastic differential equation for s, equation (S35), and each millisecond we generated $\mathbf{r}^{in}(t)$ from equation (S62). The model – equation (S54) – was run continuously for 2000 seconds, with a random value for the gain parameter ($g(t)$ in equation (S62)) sampled from a uniform distribution on the interval from 0 to 20 every 250 milliseconds. Each of these runs was then repeated 8000 times. The time step was 1 ms. The parameters of the drift diffusion process were $\sigma_\eta^2 = 2\sec^{-1}$ and $\gamma = 1\sec^{-1}$. The parameter θ in equation (S56) was set to 400 and ν_0 was set to 100. These values ensured that neurons in the output population fired at about 100 spikes per second.

4. Suboptimal models.

The suboptimal network was the linearized form of the optimal network, and corresponds roughly to that of (Huys 2006). To linearize, we replaced $\mathbf{a}\cdot\boldsymbol{\rho}$ with its average, which is the expected Fisher information. Specifically, denoting this average I_F, we used

$$I_F = \langle \mathbf{a}\cdot\mathbf{v}\rangle_{t,g} \tag{S98}$$

where the average is over both time and gain and $\mathbf{v}$ in this expression evolves according to equation (S54); that is, I_F is the expected precision of the true posterior. We then evolved the equation

$$\frac{d\boldsymbol{\rho}_{LIN}}{dt} = \mathbf{W}\cdot\boldsymbol{\rho}_{LIN} - \sigma_\eta^2 I_F \boldsymbol{\rho}_{LIN} + \mathbf{M}\cdot\mathbf{r}^{in} \tag{S99}$$

with $\mathbf{M}$ and $\mathbf{W}$ defined as in equation (S49). The posterior in this case is given by equation (S45) but with $\boldsymbol{\rho}$ replaced by $\boldsymbol{\rho}_{LIN}$. Note that we did not re-express the linear rate model in terms an inhomogeneous Poisson process.

For the specialized decoders, we fixed gain to a single value rather than sampling it every 250 milliseconds. We then evolved equation (S99), but with a different I_F for each gain, with its value selected to minimize information loss.

Olfaction

1. Translating to the relevant variables.

$s' = s_1$, $\mathbf{r}' = \mathbf{r}_k^{out}$, and $\mathbf{r}=(\mathbf{r}_1, \mathbf{r}_2, \ldots)$.

2. Posteriors.

$p_{true}(s_k \mid \mathbf{r})$ is given by equation (S83). Because the true posterior is not an exponential family distribution, there is no clearly defined optimal model. For all models we used the posterior given in equation (S86).

3. Model details.

Each odor, s_i, was present with a probability of 1/2 and the associated concentrations, c_i, were sampled uniformly on the interval 2 to 10. For simulations with four odors, the bold solid lines in Fig. S3 corresponded to a mixing weight of one while the grey lines corresponded to a random mixing matrix, w_{ij}, sampled uniformly from 0 to 1. The average over $p(\mathbf{c})$ was accomplished by binning each c_i into four equally sized bins and then computing percent information loss for each of the 256 resulting bins of the vector **c**. For the two odor/two odorant or blicket case, one odor gave rise to both odorants with mixing weight one, while the other odor gave rise to only one. Also, in this case, concentrations were sampled from the interval 1 to 5. Each odorant was associated with a population of 10 neurons with mean firing rates given by $f_{ai}o_i + f_{bi}$, where f_{ai} and f_{bi} were randomly selected for each input neuron from a uniform distribution between 0 and 1. The network parameters (the A_{jklm} below) were learned via stochastic gradient descent on $\Delta I/I$.

4. Suboptimal models.

All the models we consider have the form

$$\mathbf{r}_k^{out} = \sum_{jklm} w_{jklm}^{1,i} \varphi_{jklm}(\mathbf{r}_1, \mathbf{r}_2, \ldots, \Theta) \tag{S100}$$

where r_{jl} is the l^{th} spike count in population j (and similarly for r_{km}). Note that we don't need the individual components of $\overline{r}'_{1i}$; we need only $\mathbf{a}_1 \cdot \overline{\mathbf{r}}'_1$ (see equation (S86)). The latter quantity is given by

$$\mathbf{a}_1 \cdot \mathbf{r}_k^{out} = \sum_{jklm} A_{jklm} \varphi_{jklm}(\mathbf{r}_1, \mathbf{r}_2, \ldots, \Theta) \tag{S101}$$

where

$$A_{jklm} \equiv \sum_i a_{1i} w_{jklm}^{i}. \tag{S102}$$

We considered 4 models. The first was quadratic with divisive normalization, for which

$$\varphi_{jklm}(\mathbf{r}_1,\mathbf{r}_2,...,\Theta)=\frac{r_{jl}r_{km}}{\alpha+\sum_i \boldsymbol{\theta}_i^1\cdot\mathbf{r}_i}. \tag{S103}$$

This corresponds to equation (21) of the main text. The second was simply quadratic,

$$\varphi_{jklm}(\mathbf{r}_1,\mathbf{r}_2,...,\Theta)=r_{jl}r_{km}. \tag{S104}$$

The third was threshold linear,

$$\varphi_{jklm}(\mathbf{r}_1,\mathbf{r}_2,...,\Theta)=[r_{jl}+r_{km}-\theta]_+. \tag{S105}$$

And the fourth was linear,

$$\varphi_{jklm}(\mathbf{r}_1,\mathbf{r}_2,...,\Theta)=r_{jl}. \tag{S106}$$

Simulations of 2D Kalman Filter for the Hand Tracking Task

In this section we show how we created a neural network implementation of the inference algorithm used to model the behavioral experiment of Wolpert et al. (Ref. 5). In that experiment, Wolpert and colleagues modelled the responses of subjects performing a linear reaching movement in a virtual environment. In the task, the subjects were cued to move their hand to a specific location. Upon initiation of the movement the visual representation of the hand disappeared, leaving proprioception as the only source of sensory information about arm position. After a variable delay (0-2 seconds), a short tone was played, and subjects ended their movement. At that point, subjects were asked to estimate the position of their (now stationary) hand.

The equations of motion describing this system are given by

$$\begin{aligned} \frac{dx}{dt} &= v + \eta_x \\ m\frac{dv}{dt} &= -\gamma v + \beta u(t) + m\eta_v \end{aligned} \tag{S107}$$

where m is the mass of the hand, γ is the coefficient of friction, $u(t)$ is the force applied by the muscles, and η_x and η_v represent Gaussian white noise with (diagonal) covariance $\mathbf{\Sigma}_{\boldsymbol{\eta\eta}}$. In the notation of equation (S63), $\mathbf{s}=(x, v)$ and

$$\begin{aligned} \Gamma &= \begin{pmatrix} 0 & -1 \\ 0 & \gamma/m \end{pmatrix} \\ \mathbf{u}(t) &= \begin{pmatrix} 0 \\ u(t)/m \end{pmatrix} \\ \boldsymbol{\eta} &= \begin{pmatrix} \eta_x \\ \eta_v \end{pmatrix}. \end{aligned} \tag{S108}$$

Following Wolpert et al., for $u(t)$ we use

$$u(t) = F\left(1 - 2\Theta\left(t - t_{tone}\right)\right) \tag{S109}$$

where F is a constant, $\Theta(t)$ is the Heaviside function, and t_{tone} is the time at which subjects apply the force needed to terminate the movement. The parameters we use in the model are

$$\begin{aligned} \beta &= 1 \\ m &= 4 \text{ kg} \\ \gamma &= 3.9 \text{ kg/s} \\ F &= 1.3, 1.5, 1.9 \text{ Newtons} \\ \sigma_x^2 &= 3.3\times 10^{-4} \text{ m}^2/\text{s} \\ \sigma_v^2 &= 3.3\times 10^{-4} \text{ m}^2/\text{s}^3. \end{aligned} \tag{S110}$$

In addition to the equations which model the evolution of the underlying variables of interest (x and v in this case) we must also specify a model for the evidence. Wolpert et al. modelled the proproceptive evidence as noise corrupted measurements of the hidden variables,

$$\begin{aligned} y_x(t) &= x(t) + \xi_x(t) \\ y_v(t) &= v(t) + \xi_v(t) \end{aligned} \qquad \text{(S111)}$$

where $\xi_x(t)$ and $\xi_y(t)$ are uncorrelated, Gaussian, and white, with variances given by

$$\begin{aligned} \langle \xi_x(t)\xi_x(t') \rangle &= \sigma_x^2 \delta(t-t') \\ \langle \xi_v(t)\xi_v(t') \rangle &= \sigma_v^2 \delta(t-t'). \end{aligned} \qquad \text{(S112)}$$

Equations (S107)-(S112) are sufficient to fully specify a Kalman filter. Since the Kalman filter returns an efficient and unbiased estimate in these conditions, whereas in behavioural experiments subjects are biased[5], Wolpert et al. found it necessary to generate observational evidence using one set of model parameters, but then implement a Kalman filter using a different set of parameters. In particular, Wolpert et al. assumed that humans overestimated the acceleration applied to the hand (they had an approximate internal model). Thus, they implemented a Kalman filter which assumed that the parameter β in equation (S107) was 1.4, rather than its actual value, which was 1.

To replicate their results, we made the same assumption when building our network implementation of a Kalman filter. Specifically, we evolved equation (S73) for the rate $\mathbf{v}$ with the parameters given in equations (S107)-(S112) but with β set to 1.4 (to provide a biased internal model). Unbiased evidence concerning hand position and velocity was provided by input spike trains, $\mathbf{r}^{in}(t)$, which were generated by evolving the stochastic differential equations for $x(t)$ and $v(t)$ (equation (S107) with β set to 1, with the same realization of the noise as was used with β=1.4), and then simulating Poisson processes conditioned on these hidden variables. Specifically, for sample paths $x(t)$ and $v(t)$ from the numerical solution to equation (S107) with β set to 1, we generated $\boldsymbol{\rho}^{in}(t)$ from equations (S75) and (S76) with $s_1(t)$ replaced by $x(t)$ and $s_2(t)$ replaced by $v(t)$.

Finally, we had to insure that $\mathbf{r}^{in}(t)$ corresponds to the noise model utilized by Wolpert et al. (equations (S111) and (S112)). To do that, we note that the time-evolution equations for a Kalman filter with continuous valued evidence (as in equation(S112)) is the same as equation (S68) except that the terms $\mathbf{I}^{in}$ and $(\mathbf{I}\cdot\boldsymbol{\mu})^{in}$ are replaced by terms corresponding to the continuous evidence associated with equations (S111) and (S112). Examining equation (S68), and its counterpart, equation (S69), we see that this induces the following condition on $\mathbf{r}^{in}(t)$,

$$\begin{aligned}\frac{1}{\sigma_x^2} &= \left\langle \mathbf{a}_x^{in} \cdot \boldsymbol{\rho}_x^{in} \right\rangle \\ \frac{x(t)}{\sigma_x^2} &= \left\langle \mathbf{b}_x^{in} \cdot \boldsymbol{\rho}_x^{in} \right\rangle \\ \frac{1}{\sigma_x^2} &= \left\langle \mathbf{a}_v^{in} \cdot \boldsymbol{\rho}_v^{in} \right\rangle \\ \frac{x(t)}{\sigma_x^2} &= \left\langle \mathbf{b}_v^{in} \cdot \boldsymbol{\rho}_v^{in} \right\rangle\end{aligned} \tag{S113}$$

where the angle brackets indicate an average over sample paths of $x(t)$ and $v(t)$ generated from equation (S107) with β set to 1. As with equation (S78), we let

$$\boldsymbol{\rho}^{in} = \begin{bmatrix} \boldsymbol{\rho}_x^{in} \\ \boldsymbol{\rho}_v^{in} \end{bmatrix}. \tag{S114}$$

The noise averages of $\boldsymbol{\rho}_x^{in}$ and $\boldsymbol{\rho}_v^{in}$ are equal to the firing rates, which are given by equation (S75) (with, as above, s_1 replaced by x and s_2 by v), and with tuning curves given by equation (S76). As with equation , we use $a_{x,i}^{in} = a_{v,i}^{in} = 1/\sigma_w^2$ and $b_{x,i}^{in} = b_{v,i}^{in} = s_i^0/\sigma_w^2$. Putting this all together yields

$$\begin{aligned}\frac{1}{\sigma_x^2} &= \frac{1}{\sigma_w^2}\sum_i g \exp\left(-\frac{\left(x(t)-s_i^0\right)^2}{2\sigma_w^2}\right) \\ \frac{x(t)}{\sigma_x^2} &= \frac{1}{\sigma_w^2}\sum_i g\hat{s}_i \exp\left(-\frac{\left(x(t)-s_i^0\right)^2}{2\sigma_w^2}\right) \\ \frac{1}{\sigma_v^2} &= \frac{1}{\sigma_w^2}\sum_i g \exp\left(-\frac{\left(v(t)-s_i^0\right)^2}{2\sigma_w^2}\right) \\ \frac{v(t)}{\sigma_v^2} &= \frac{1}{\sigma_w^2}\sum_i g\hat{s}_i \exp\left(-\frac{\left(v(t)-s_i^0\right)^2}{2\sigma_w^2}\right).\end{aligned} \tag{S115}$$

When the preferred stimulus values, s_i^0, are evenly spaced and concentrated around the typical values of the random variables $x(t)$ and $v(t)$, we can take the continuous limit to transform the sums over i to integrals to obtain

$$\begin{aligned}\frac{1}{\sigma_x^2} &= \frac{(2\pi)^{1/2} g}{\sigma_w \delta\hat{s}} \\ \frac{x(t)}{\sigma_x^2} &= \frac{(2\pi)^{1/2} g x(t)}{\sigma_w \delta\hat{s}} \\ \frac{1}{\sigma_v^2} &= \frac{(2\pi)^{1/2} g}{\sigma_w \delta\hat{s}} \\ \frac{v(t)}{\sigma_v^2} &= \frac{(2\pi)^{1/2} g v(t)}{\sigma_w \delta\hat{s}}\end{aligned} \tag{S116}$$

where δs^0 is the spacing between adjacent values of s_i^0. Because σ_x^2 and σ_v^2 have the same numerical value, all four expressions in equation (S116) yield the same value for the gain,

$$g = \frac{\sigma_x^2 (2\pi)^{1/2}}{\sigma_w^2 \delta\hat{s}}. \tag{S117}$$

We used σ_w^2=1/2 m^2 and $s_i^0 = 8(i - 60.5)/120,\ i = 1,...,120$ (the latter implying that $\delta\hat{s} = 8/120$). Inserting these into equation (S116) yields g=198 Hz. (For interested readers we also note that an appeal to Fisher Information in the input populations could also have been used to obtain this result).

Bias and variance were computed by comparing the mean and variance produced by the biased network $(\beta = 1.4)$ that of the optimal Kalman filter, averaged over 20,000 sample paths. As in Wolpert et al. this was done for three forced feedback conditions corresponding to F={1.3, 1.5, 1.9}.

References

1. Latham, P.E. & Nirenberg, S. Synergy, redundancy, and independence in population codes, revisited. *J Neurosci* 25, 5195-5206 (2005).
2. Bobrowski, O., Meir, R. & Eldar, Y.C. Bayesian filtering in spiking neural networks: noise, adaptation, and multisensory integration. *Neural Comput* 21, 1277-1320 (2009).
3. Huys, Q.J., Zemel, R.S., Natarajan, R. & Dayan, P. Fast population coding. *Neural Comput* 19, 404-441 (2007).
4. Pitkow, X., Sompolinsky, H. & Meister, M. A neural computation for visual acuity in the presence of eye movements. *PLoS Biol* 5, e331 (2007).
5. Wolpert, D., Ghahramani, Z., & Jordan, M. An Internal Model for Sensorimotor Integration. *Science* 269: 5232, (1995).