Factoring movies into ‘what’ and ‘where’

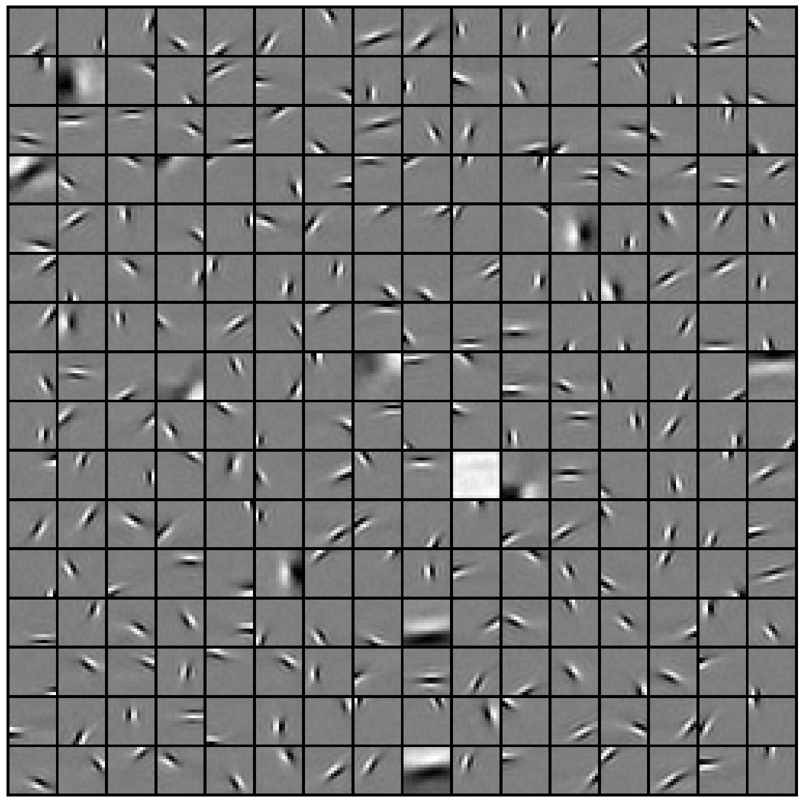
Jack Culpepper
Advisor: Bruno Olshausen

bjc@cs.berkeley.edu

CIAR Summer School
University of Toronto

August 15, 2006

Sparsenet is wasteful because it 'copies' basis functions



Why? Because sparsenet codes aren't translation invariant

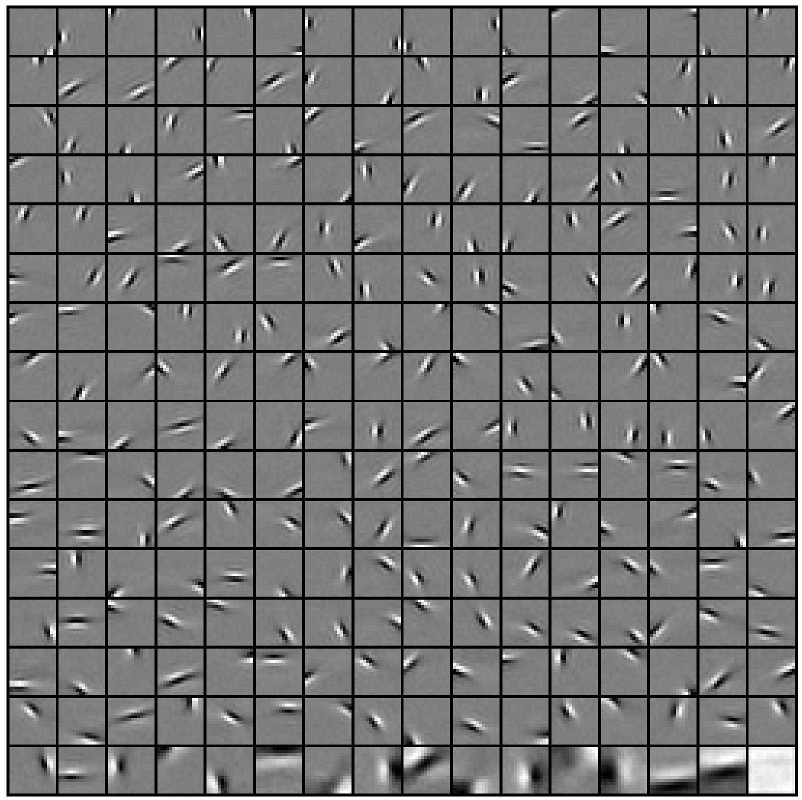
- Ok, actually they are, *sort of*: the activations of low spatial frequency basis functions don't change much for small shifts (they don't need to, because they are so big and boring)

- That's why there are more 'copies' of high frequency BFs

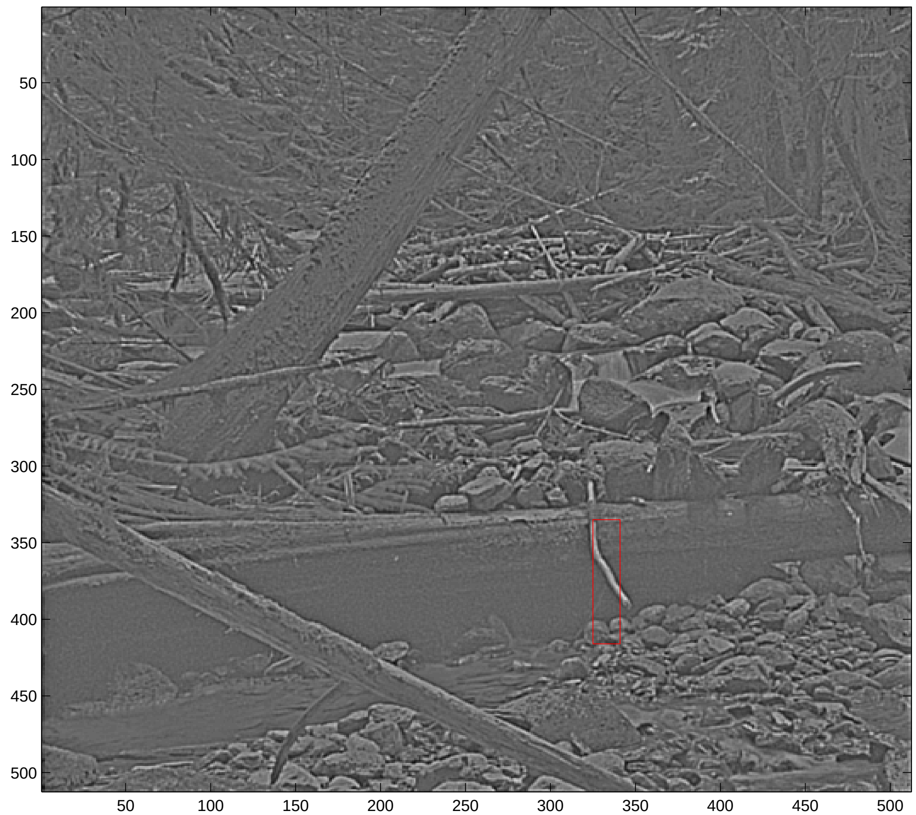
- ..but we'd like a code that gives the same output when the input slightly translated, scaled, rotated, or occluded.

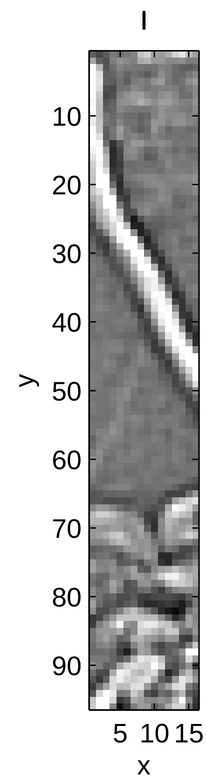
- I.e., we'd like to account for slight transformations of the image with another set of variables.

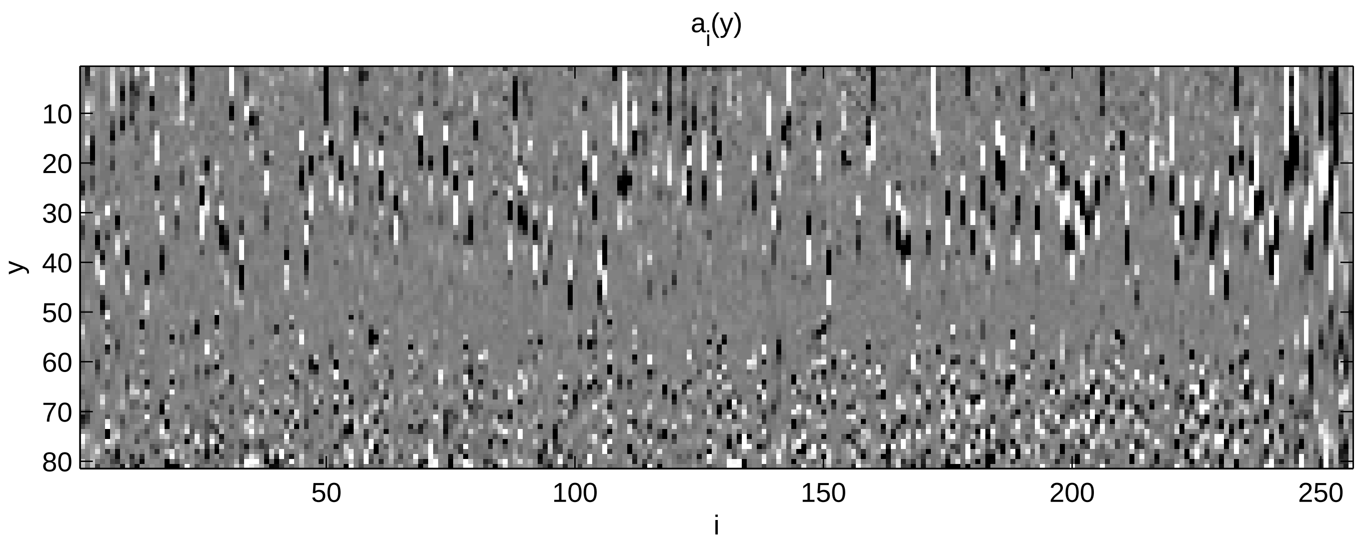
Sparsenet BFs ordered by spatial frequency



What happens as we slide our window?




I
10
20
30
40
50
60
70
80
90
y
5
10
15
x



a_i(y)
y
10
20
30
40
50
60
70
80
50
100
150
200
250
i

Morphable basis functions increase representational power

- We don't have to 'copy' each orientation and frequency to each position
- There are a small number of useful morphs, each of which can be applied to each template basis function
- We can use M basis functions and N morphs to gain the expressiveness of $M \times N$ basis functions

Morphable BFs can be used to separate ‘what’ from ‘where’

- Modeling this way allows us to read out the two factors as separate variables

- The model then captures the separate functions of the ventral (‘what’) and dorsal (‘where’) processing streams in mamallian visual cortex

  - Strong support for this idea of two streams comes from lesion studies in monkey, in which damage to dorsal or ventral cortical regions impairs spatial abilities or object identification, respectively

Image transformations

- Consider a 1-dimensional, discretely sampled image of a 2-dimensional world.

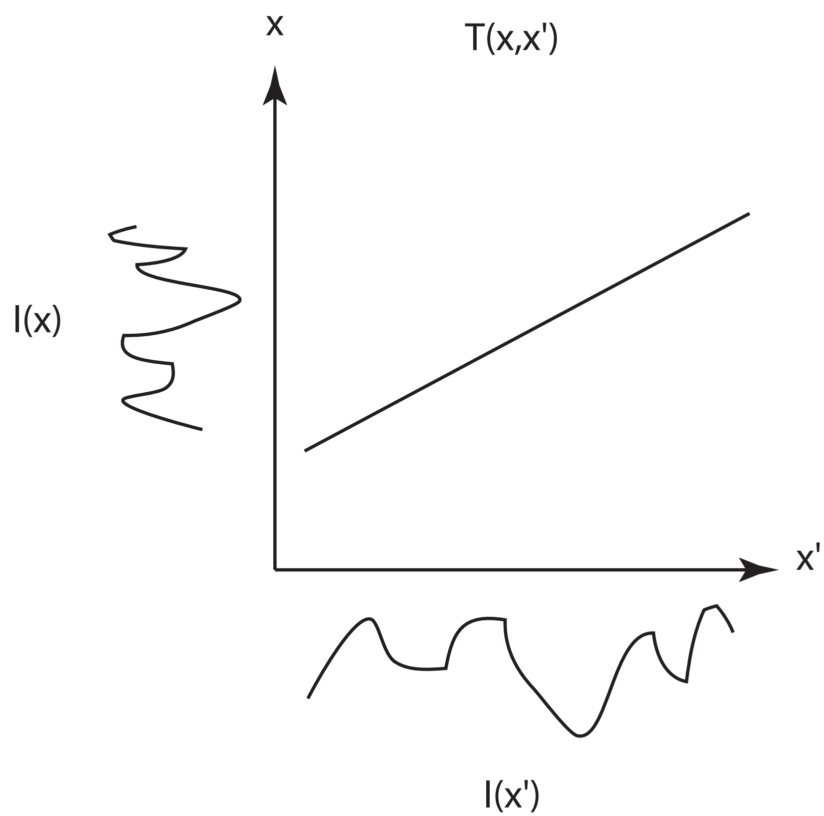
- We can express arbitrary transformations from $I(x)$ to $I'(x')$, $x \in \mathcal{X}$, $x' \in \mathcal{X}'$ as follows.

$$I(x) = \sum_{x'} T(x, x') I'(x')$$

- Or, in vector-matrix notation,

$$\mathbf{I} = \mathbf{T}\mathbf{I}'$$

Visualizing 1-d image transformations



Basis function expansions of image transformations

- Transformations between sequential frames of natural movies are 2-d and not necessarily affine.
  - Add dimensions $y \in \mathcal{Y}$ and $y' \in \mathcal{Y}'$:

$$I(x, y) = \sum_{x'} \sum_{y'} T(x, y, x', y') I'(x', y')$$

  - We can capture the space of all possible transforms by expressing $T(x, y, x', y')$ as a basis function expansion:

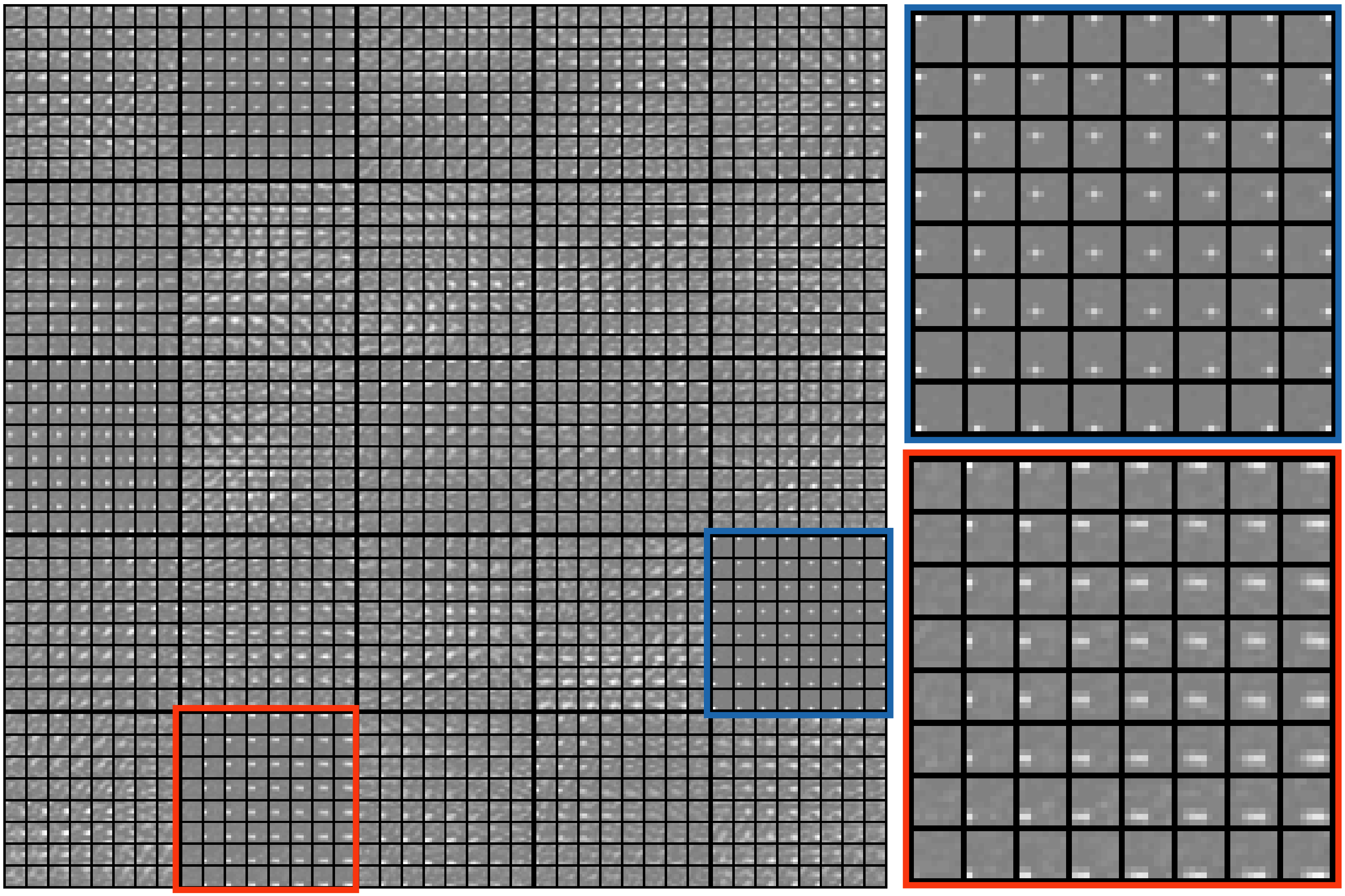
$$T(x, y, x', y') = \sum_{j} c_j \psi_j(x, y, x', y')$$

Sparse coding in transform space

- The transformations lie inside a subspace of $\mathcal{X} \times \mathcal{Y} \times \mathcal{X}' \times \mathcal{Y}'$ (not all possible transforms are viable).

- We expect that for a small image patch, the correspondence between most sequential movie frames can be 'explained' by a small number of underlying causes – such as object motion or observer self-motion.

- Thus, it makes sense to learn them via sparse coding!



Putting it all together

- Our complete model is now:

$$I(x,y) = \sum_{x',y'} T(x,y,x',y') I'(x',y')$$

where

$$T(x,y,x',y') = \sum_j \psi_j(x,y,x',y') c_j$$

and

$$I'(x',y') = \sum_i \phi_i(x',y') a_i$$

A bilinear model

- Rearranging the terms

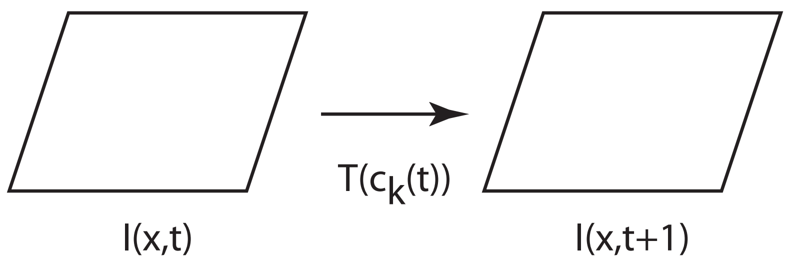
$$I(x,y) = \sum_{x',y',i,j} \psi_j(x,y,x',y')\phi_i(x',y')a_i c_j$$

and precomputing the sums over x' and y'

$$\Gamma_{ij}(x,y) = \sum_{x',y'} \psi_j(x,y,x',y')\phi_i(x',y')$$

$$I(x,y) = \sum_{i,j} \Gamma_{ij}(x,y)a_i c_j$$

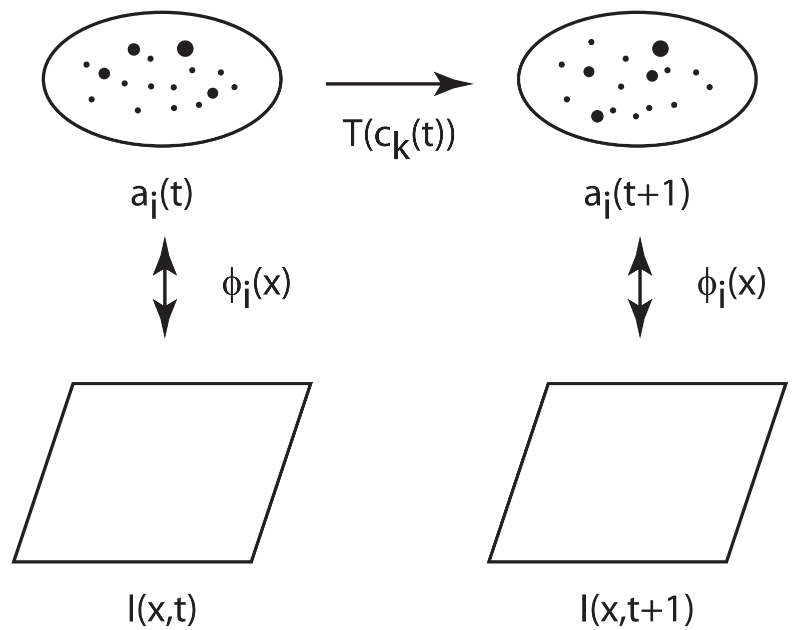
(Grimes & Rao, 2005)


I(x,t)
T(ck(t))
I(x,t+1)

Transforms in coefficient space

- Sparse coding is capable of learning transformations between the pixels in sequential movie frames

- It turns out that it is also capable of learning transformations between the sparse codes for representing sequential movie frames, and this is even easier



T(c_k(t))
a_i(t)
a_i(t+1)
ϕ_i(x)
ϕ_i(x)
I(x,t)
I(x,t+1)

Modeling slow underlying causes

- Edges don't pop into existence in one position, disappear, then pop into existence in a new position. They appear, then translate. We capture this using a latent variable model to describe the frame by frame coefficients:

$$I(x,y,t) = \sum_i \phi_i(x,y) a_i(t) + \nu_I$$

$$a_i(t) = \sum_j T_{ij}(t) b_j(t)$$

$$T_{ij} = \sum_k \Gamma_{ijk} c_k(t) + \nu_a$$

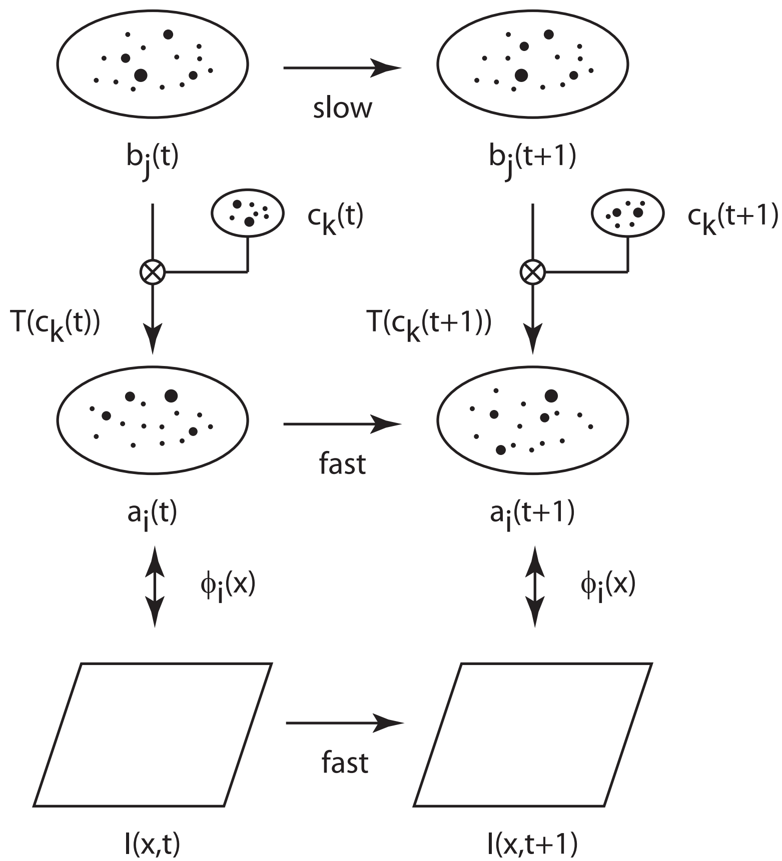
A bilinear model of slow underlying causes

- This gives a bilinear model for the frame by frame coefficients:

$$I(x,y,t) = \sum_i \phi_i(x,y) a_i(t) + \nu_I$$

$$a_i(t) = \sum_{j,k} \Gamma_{ijk} b_j(t) c_k(t) + \nu_a$$

- We'd like to learn $\mathbf{\Gamma}$ that produces sparse $\mathbf{c}$ and $\mathbf{b}$, and *slowly varying* $\mathbf{b}$


slow
$b_j(t)$
$b_j(t+1)$
$c_k(t)$
$c_k(t+1)$
$T(c_k(t))$
$T(c_k(t+1))$
fast
$a_i(t)$
$a_i(t+1)$
$\phi_i(x)$
$\phi_i(x)$
fast
$I(x,t)$
$I(x,t+1)$

Probabilistic framework

- Assuming that reconstruction errors in $\mathbf{a}$ are due to i.i.d. Gaussian noise, the probability of $\mathbf{a}$ given the coefficients and model parameters is a Gaussian distribution:

$$P(\mathbf{a}|\mathbf{b}, \mathbf{c}, \mathbf{\Gamma}) = \frac{1}{\sqrt{2\pi}\sigma_n} \exp\{-\frac{1}{2\sigma_n^2} \sum_{i,t} [e(i,t)]^2\}$$

where

$$e(i,t) = a_i(t) - \hat{a}_i(t)$$

$$\hat{a}_i(t) = \sum_{jk} \Gamma_{ijk} b_j(t) c_k(t)$$

$Z_n = \sqrt{2\pi}\sigma_n$, and σ_n^2 is the variance of the noise.

Prior knowledge

$$P(\mathbf{b}|\mathbf{\Gamma}) = \prod_{j,t} \frac{1}{Z_S} \mathsf{exp}\{-S(b_j(t))\} \prod_j \frac{1}{Z_R} \mathsf{exp}\{-R(b_j(t))\}$$

$$P(\mathbf{c}|\mathbf{\Gamma}) = \prod_{k,t} \frac{1}{Z_S} \mathsf{exp}\{-S(c_k(t))\}$$

- Wish to adapt our model parameters so that (1) the coefficients follow distributions that are peaked at zero with heavy tails, because we believe that our images can be 'explained' by a small number of independent, additive features, and (2) the 'what' coefficients vary slowly in time.

Slow prior

- The time-varying model imposes a smoothness constraint over time on the 'what' variables using the following cost function:

$$R(b_j(t)) = \frac{1}{2}\sum_{t}^{\tau-1}[b_j(t) - b_j(t+1)]^2$$

$$\frac{\partial R}{\partial b_l(u)} = 2b_l(u) - b_l(u+1) - b_l(u-1)$$

Probabilistic bilinear sparse coding

- Using Bayes' rule, we can compute the probability of different explanations for an image's sparse code:

$$P(\mathbf{b}, \mathbf{c}|\mathbf{a}, \mathbf{\Gamma}) = P(\mathbf{a}|\mathbf{b}, \mathbf{c}, \mathbf{\Gamma})P(\mathbf{b}, \mathbf{c}|\mathbf{\Gamma})/P(\mathbf{a}, \mathbf{\Gamma})$$

- Given an image's code $\mathbf{a}$ and a set of basis functions $\mathbf{\Gamma}$, we select the most likely coefficients for describing it by computing the maximum a-posteriori (MAP) estimates $\mathbf{b}^*$ and $\mathbf{c}^*$.

$$P(\mathbf{b}, \mathbf{c}|\mathbf{a}, \mathbf{\Gamma}) \propto P(\mathbf{a}|\mathbf{b}, \mathbf{c}, \mathbf{\Gamma})P(\mathbf{b}|\mathbf{\Gamma})P(\mathbf{c}|\mathbf{\Gamma})$$

Objective function

- From this, we derive our objective function, minimization of which is equivalent to finding the MAP estimates:

$$
\begin{aligned}
\mathcal{L} &= -\log P(\mathbf{b}, \mathbf{c} | \mathbf{a}, \mathbf{\Gamma}) \\
&\propto \frac{1}{2\sigma_n^2} \sum_{i,t} [e(i,t)]^2 + \sum_{j,t} S(b_j(t)) + \sum_{k,t} S(c_k(t)) + \sum_{j} R(b_j(t))
\end{aligned}
$$