LETTER **Communicated by Bruno Olshausen**

Bilinear Sparse Coding for Invariant Vision

David B. Grimes
grimes@cs.washington.edu
Rajesh P. N. Rao
rao@cs.washington.edu
Department of Computer Science and Engineering, University of Washington, Seattle, WA 98195-2350, U.S.A

Recent algorithms for sparse coding and independent component analysis (ICA) have demonstrated how localized features can be learned from natural images. However, these approaches do not take image transformations into account. We describe an unsupervised algorithm for learning both localized features and their transformations directly from images using a sparse bilinear generative model. We show that from an arbitrary set of natural images, the algorithm produces oriented basis filters that can simultaneously represent features in an image and their transformations. The learned generative model can be used to translate features to different locations, thereby reducing the need to learn the same feature at multiple locations, a limitation of previous approaches to sparse coding and ICA. Our results suggest that by explicitly modeling the interaction between local image features and their transformations, the sparse bilinear approach can provide a basis for achieving transformation-invariant vision.

1 Introduction

Algorithms for redundancy reduction and efficient coding have been the subject of considerable attention (Olshausen & Field, 1996, 1997; Bell & Sejnowski, 1997; Hinton & Ghahramani, 1997; Rao & Ballard, 1999; Lewicki & Sejnowski, 2000; Schwartz & Simoncelli, 2001). Although the basic ideas can be traced to earlier work (Attneave, 1954; Barlow, 1961), recent techniques such as independent component analysis (ICA) and sparse coding have helped formalize these ideas and have demonstrated the feasibility of efficient coding through redundancy reduction. These techniques produce an efficient code by using appropriate constraints to minimize the dependencies between elements of the code.

One of the most successful applications of ICA and sparse coding has been in the area of image coding. Olshausen and Field (1996, 1997) showed that sparse coding of natural images produces localized, oriented basis filters that resemble the receptive fields of simple cells in primary visual cortex.

Bell and Sejnowski (1997) obtained similar results using their algorithm for ICA. However, these approaches do not take image transformations such as translation into account. Thus, the model cannot take advantage of the fact that certain basis features model the same image features but under different transformations. As a result, for each oriented feature, a number of independent units must code for the same feature at different locations, making it difficult to scale the approach to large image patches and hierarchical networks.

In this letter, we propose an approach to sparse coding that explicitly models the interaction between image features and their transformations. A bilinear generative model is used to learn both the independent features in an image as well as their transformations. Our approach extends Tenenbaum and Freeman's (2000) work on bilinear models for learning content and style by casting the problem within a probabilistic sparse coding framework. Thus, whereas prior work on bilinear models used global decomposition methods such as singular value decomposition (SVD), the approach presented here emphasizes the extraction of local features by removing higher-order redundancies through sparseness constraints. We show that for natural images, this approach produces localized, oriented filters that can be translated by different amounts to account for image features at arbitrary locations. Our results demonstrate how an image can be factored into a set of basic local features and their transformations, providing a basis for transformation-invariant vision. In particular, we focus on the problem of invariance to transformations caused by moving objects or smooth self-motion. We assume that for a given object, the goal is to estimate the motion of the object and learn bilinear features for both the object and its transformation. We conclude by discussing related work and suggest an extension of the approach to parts-based object recognition, wherein an object is modeled as a collection of local features (or "parts") and their relative transformations.

2 Bilinear Generative Models

We begin by considering the standard linear generative model used in algorithms for ICA and sparse coding (Bell & Sejnowski, 1997; Olshausen & Field, 1997; Rao & Ballard, 1999):

$$\mathbf{z} = \sum_{i=1}^{m} \mathbf{w}_i x_i = W\mathbf{x}, \tag{2.1}$$

where $\mathbf{z}$ is a k-dimensional input vector (for instance, an image), $\mathbf{w}_i$ is a k-dimensional basis vector, and x_i is its scalar coefficient. Given the linear generative model above, the goal of ICA is to learn the basis vectors $\mathbf{w}_i$ (i.e., the matrix W) such that the x_i are as independent as possible, while the goal in sparse coding is to make the distribution of x_i highly kurtotic given equation 2.1.

Now consider adding a transformation parameterized by λ to the generative process of an image $\mathbf{z}$, so that $\mathbf{z} = \mathcal{T}_\lambda(W\mathbf{x})$. Given the linear model as described above, as λ varies, the $\mathbf{x}$ will need to change accordingly. Transformation-invariance methods seek to model the image formation process in such a way that $\mathbf{x}$ is independent (or at least uncorrelated) with λ.

A simple method for achieving invariance is to introduce another variable $\mathbf{y}$, which accounts for the changes in the image due to the transformation $\mathcal{T}_\lambda$. Invariance is achieved once $\mathbf{y}$ is known because $\mathbf{x}$ and λ are conditionally independent given $\mathbf{y}$. Thus, the key requirement of any such model is that $\mathbf{y}$ can easily be inferred.

Using a probabilistic approach, we specify the form of the image likelihood function $P(\mathbf{z}|\mathbf{x}, \mathbf{y})$. To model this likelihood, we introduce an interaction function $f(\mathbf{x}, \mathbf{y})$ that models the interactions between $\mathbf{x}$ and $\mathbf{y}$ in the image formation process. For an additive gaussian noise model, the likelihood becomes $P(\mathbf{z}|\mathbf{x}, \mathbf{y}) = \mathcal{G}(\mathbf{z}; f(\mathbf{x}, \mathbf{y}), \sigma^2)$.

The function $f(\mathbf{x}, \mathbf{y})$ should not only be able to represent the transformations of interest, but must also be invertible in the sense that $\mathbf{x}$ and/or $\mathbf{y}$ can be inferred given $\mathbf{z}$ and possibly one of $\mathbf{x}$ or $\mathbf{y}$. Perhaps the simplest function f is the linear function: $f(\mathbf{x}, \mathbf{y}) = W\mathbf{x} + W'\mathbf{y}$. Unfortunately, this model is too impoverished to represent most common classes of transformations such as affine transformations in the image plane. A logical next step is to consider multiplicative interactions between $\mathbf{x}$ and $\mathbf{y}$. In this work, we explore the use of the bilinear function, which is the simplest form of f allowing multiplicative interactions.

The linear generative model in equation 2.1 can be extended to the bilinear case by using two sets of coefficients x_i and y_j (or equivalently, two vectors $\mathbf{x}$ and $\mathbf{y}$) (Tenenbaum & Freeman, 2000):

$$\mathbf{z} = f(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbf{w}_{ij} x_i y_j. \tag{2.2}$$

The coefficients x_i and y_j jointly modulate a set of basis vectors $\mathbf{w}_{ij}$ to produce an input vector $\mathbf{z}$. For this study, the coefficient x_i can be regarded as encoding the presence of object feature i in the image while the y_j values determine the transformation present in the image. In the terminology of Tenenbaum and Freeman (2000), $\mathbf{x}$ describes the content of the image, while $\mathbf{y}$ encodes its style.

Equation 2.2 can also be expressed as a linear equation in $\mathbf{x}$ for a fixed $\mathbf{y}$:

$$\mathbf{z} = f(\mathbf{x})_{|\mathbf{y}} = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} \mathbf{w}_{ij} y_j \right) x_i = \sum_{i=1}^{m} \mathbf{w}_i^{\mathbf{y}} x_i. \tag{2.3}$$

The notation $\mathbf{w}_i^{\mathbf{y}}$ signifies a transformed feature computed by the weighted sum shown above of the bilinear features $\mathbf{w}_{i,*}$ by the values in a given $\mathbf{y}$

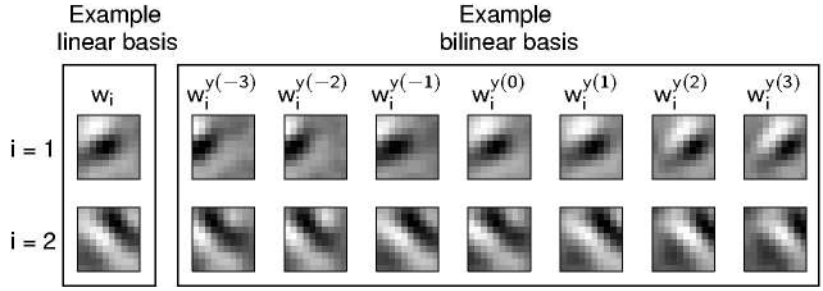

Figure 1: Examples of linear and bilinear features. A comparison of learned features between a standard linear model and a bilinear model, both trained using sparseness constraints to obtain localized, independent features. The two rows in the bilinear case depict the translated object features $\mathbf{w}_i^y$ (see equation 2.3) for different $\mathbf{y}$ vectors corresponding to translations of $-3, \ldots, 3$ pixels.

vector. Likewise, for a fixed $\mathbf{x}$, one obtains a linear equation in $\mathbf{y}$. Indeed, this is the definition of bilinear: given one fixed factor, the model is linear with respect to the other factor. The power of bilinear models stems from the rich nonlinear interactions that can be represented by varying both $\mathbf{x}$ and $\mathbf{y}$ simultaneously.

Note that the standard linear generative model (see equation 2.1) can be seen as a special case of the bilinear model when $n = 1$ and $\mathbf{y} = 1$. A comparison between examples of features used in the linear generative model and the bilinear model is given in Figure 1. The features in the linear model represent a single instance within the range of features that can be learned by the bilinear model.

3 Learning Sparse Bilinear Models

3.1 Learning Bilinear Models. Our goal is to learn from image data an appropriate set of basis vectors $\mathbf{w}_{ij}$ that effectively describe the interactions between the feature vector $\mathbf{x}$ and the transformation vector $\mathbf{y}$. A commonly used approach in unsupervised learning is to minimize the sum of squared pixel-wise errors over all images:

$$E_1(\{\mathbf{w}_{ij}\}, \mathbf{x}, \mathbf{y}) = \left\| \mathbf{z} - \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbf{w}_{ij} x_i y_j \right\|^2 \tag{3.1}$$

$$= \left(\mathbf{z} - \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbf{w}_{ij} x_i y_j \right)^T \left(\mathbf{z} - \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbf{w}_{ij} x_i y_j \right), \tag{3.2}$$

where $\| \cdot \|$ denotes the L_2 norm of a vector. A standard approach to minimizing such a function is to use gradient descent and alternate between minimization with respect to $\{\mathbf{x}, \mathbf{y}\}$ and minimization with respect to $\mathbf{w}_{ij}$. Unfortunately, the optimization problem as stated is underconstrained. The function E_1 has many local minima, and results from our simulations indicate that convergence is not obtainable with data drawn from natural images. There are many different ways to represent an image, making it difficult for the method to converge to a basis set that can effectively represent images that were not in the training set.

A related approach is presented by Tenenbaum and Freeman (2000) in their article dealing with style and content separation. Rather than using gradient descent, their method estimates the parameters directly by computing the SVD of a matrix A containing input data corresponding to each content class in every style. Their approach can be regarded as an extension of methods based on principal component analysis (PCA) applied to the bilinear case. The SVD approach avoids the difficulties of convergence that plague the gradient-descent method and is much faster in practice. Unfortunately, the learned features tend to be global and nonlocalized, similar to those obtained from PCA-based methods based on second-order statistics. As a result, the method is unsuitable for the problem of learning local features of objects and their transformations.

The underconstrained nature of the problem can be remedied by imposing constraints on $\mathbf{x}$ and $\mathbf{y}$. In particular, we cast the problem within a probabilistic framework and impose specific prior distributions on $\mathbf{x}$ and $\mathbf{y}$ with higher probabilities for values that achieve certain desirable properties. We focus here on the class of sparse prior distributions for several reasons: (1) by forcing most of the coefficients to be zero for any given input, sparse priors minimize redundancy and encourage statistical independence between the various x_i and between the various y_j (Olshausen & Field, 1997); (2) there is some evidence for sparse representations in the brain (Földiák & Young, 1995): the distribution of neural responses in visual cortical areas is typically highly kurtotic, that is, cells exhibit little activity for most inputs but respond vigorously for a few inputs, causing a distribution with a high peak near zero and long tails; (3) previous approaches based on sparseness constraints have obtained encouraging results (Olshausen & Field, 1997); and (4) enforcing sparseness on the x_i encourages the parts and local features shared across objects to be learned while imposing sparseness on the y_j allows object transformations to be explained in terms of a small set of basis vectors.

3.2 Probabilistic Bilinear Sparse Coding. Our probabilistic model for bilinear sparse coding follows a standard Bayesian MAP (maximum a posteriori) approach. Thus, we begin by factoring the posterior probability of

the parameters given the data as

$$P(\mathbf{x}, \mathbf{y}, \{\mathbf{w}_{ij}\}|\mathbf{z}) = P(\mathbf{z}|\mathbf{x}, \mathbf{y}, \mathbf{w}_{ij}) \prod_{i=1}^{m} P(x_i) \prod_{j=1}^{n} P(y_j) P(\{\mathbf{w}_{ij}\}) \tag{3.3}$$

$$\propto P(\mathbf{z}|\mathbf{x}, \mathbf{y}, \mathbf{w}_{ij}) \prod_{i=1}^{m} P(x_i) \prod_{j=1}^{n} P(y_j). \tag{3.4}$$

Equation 3.3 assumes independence between $\mathbf{x}$, $\mathbf{y}$, and $\{\mathbf{w}_{ij}\}$ as well as independence within the individual dimensions of $\mathbf{x}$ and $\mathbf{y}$. Equation 3.4 assumes a uniform prior for $P(\{\mathbf{w}_{ij}\})$, which is thus ignored.

We assume the following priors for x_i and y_j:

$$P(x_i) = \frac{1}{Q_\alpha} e^{-\alpha S(x_i)} \tag{3.5}$$

$$P(y_j) = \frac{1}{Q_\beta} e^{-\beta S(y_j)}, \tag{3.6}$$

where Q_α and Q_β are normalization constants, α and β are parameters that control the degree of sparseness, and S is a "sparseness function." For this study, we used $S(a) = \log(1 + a^2)$. As shown in Figure 2, our choice of $S(a)$ corresponds to a Cauchy prior distribution, which exhibits a useful nonlinearity in the derivative $S'(a)$.

The squared error function E_1 in equation 3.2 can be interpreted as representing the negative log likelihood ($-\log P(\mathbf{z}|\mathbf{x}, \mathbf{y}, \mathbf{w}_{ij})$) under the assumption of gaussian noise with unit variance (see, e.g., Olshausen & Field, 1997).

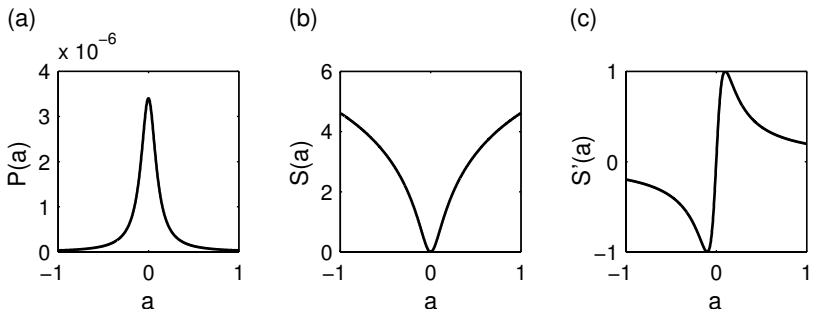

Figure 2: A probabilistic sparse coding prior. (a) The probability distribution function for the Cauchy sparse coding prior. Although the distribution appears similar to a gaussian distribution, the Cauchy is supergaussian (highly kurtotic). (b) The derived sparseness error function. (c) The nonlinearity introduced in the derivative of the sparseness function. Note that the function differentially forces small coefficients toward zero, and only at some threshold are large coefficients made larger.

Maximizing the posterior in equation 3.3 is thus equivalent to minimizing the following log posterior function over all input images:

$$E(\{\mathbf{w}_{ij}\}, \mathbf{x}, \mathbf{y}) = \left\| \mathbf{z} - \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbf{w}_{ij} x_i y_j \right\|^2 + \alpha \sum_{i=1}^{m} S(x_i) + \beta \sum_{j=1}^{n} S(y_j). \quad (3.7)$$

The gradient of E can be used to derive update rules at time t for the components x_a and y_b of the feature vector $\mathbf{x}$ and transformation vector $\mathbf{y}$, respectively, for any image $\mathbf{z}$, assuming a fixed basis $\mathbf{w}_{ij}$:

$$\frac{dx_a}{dt} = -\frac{1}{2}\frac{\partial E}{\partial x_a} = \sum_{q=1}^{n} \mathbf{w}_{aq}^T \left(\mathbf{z} - \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbf{w}_{ij} x_i y_j \right) y_q + \frac{\alpha}{2} S'(x_a) \quad (3.8)$$

$$\frac{dy_b}{dt} = -\frac{1}{2}\frac{\partial E}{\partial y_b} = \sum_{q=1}^{m} \mathbf{w}_{qb}^T \left(\mathbf{z} - \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbf{w}_{ij} x_i y_j \right) x_q + \frac{\beta}{2} S'(y_b). \quad (3.9)$$

Given a training set of inputs $\mathbf{z}_l$, the values for $\mathbf{x}$ and $\mathbf{y}$ for each image after convergence can be used to update the basis set $\mathbf{w}_{ij}$ in batch mode according to

$$\frac{d\mathbf{w}_{ab}}{dt} = -\frac{1}{2}\frac{\partial E}{\partial \mathbf{w}_{ab}} = \sum_{l=1}^{q} \left(\mathbf{z}_l - \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbf{w}_{ij} x_i y_j \right) x_a y_b. \quad (3.10)$$

One difficulty in the sparse coding formulation of equation 3.7 is that the algorithm can trivially minimize the sparseness function by making $\mathbf{x}$ or $\mathbf{y}$ very small and compensate by increasing the $\mathbf{w}_{ij}$ basis vector norms to maintain the desired output range. Therefore, as previously suggested (Olshausen & Field, 1997), in order to keep the basis vectors from growing without bound, we adapt the L_2 norm of each basis vector in such a way that the variance of x_i (and y_j, as discussed below) were maintained at a fixed desired level (σ_g^2). Simply forcing the basis vectors to have a certain norm can lead to instabilities; therefore, a "soft" variance normalization method was employed. The element-wise variance of the $\mathbf{x}$ vectors inferred during a single batch iteration was tracked in the vector $\mathbf{x}_{var}$ and adapted at a rate given by the parameter ε (see algorithm 1, line 18). A gain term $\mathbf{g}_\mathbf{x}$ is computed (see lines 19–20 in algorithm 1), which determines the multiplicative factor for adapting the norm of a particular basis vector:

$$\hat{\mathbf{w}}_{ij} = \mathbf{g}_{\mathbf{x},i} \frac{\mathbf{w}_{ij}}{\|\mathbf{w}_{ij}\|^2}. \quad (3.11)$$

An additional complication in the bilinear case is that $\|\mathbf{w}_{ij}\|^2$ is related to the variance of both x_i and y_j. One possible solution is to compute a joint

gain matrix G (which specifies a gain $G_{i,j}$ for each $\mathbf{w}_{ij}$ basis vector) as the geometric mean of the elements in the gain vectors $\mathbf{g_x}$ and $\mathbf{g_y}$:

$$G = \sqrt{\mathbf{g_x}\mathbf{g_y}^T}. \tag{3.12}$$

However, in the case where sparseness is desired for $\mathbf{x}$ but not $\mathbf{y}$ (i.e., $\beta = 0.0$), the variance of $\mathbf{y}$ will rapidly increase, as the variance of $\mathbf{x}$ rapidly decreases, and no perturbations to the norm of the basis vectors $\mathbf{w}_{ij}$ will solve this problem. To avoid this problem, the algorithm performs soft variance normalization directly on the evolving $\mathbf{y}$ vectors, and scales the basis vectors $\mathbf{w}_{ij}$ based only on the variance of x_i (see algorithm 1, lines 12–14).

3.3 Algorithm for Learning Bilinear Models of Translating Image Patches. This section describes our unsupervised learning algorithm that uses the update rules (see equations 3.8–3.10) to learn localized bilinear features in natural images for two-dimensional translations. Figure 3 presents

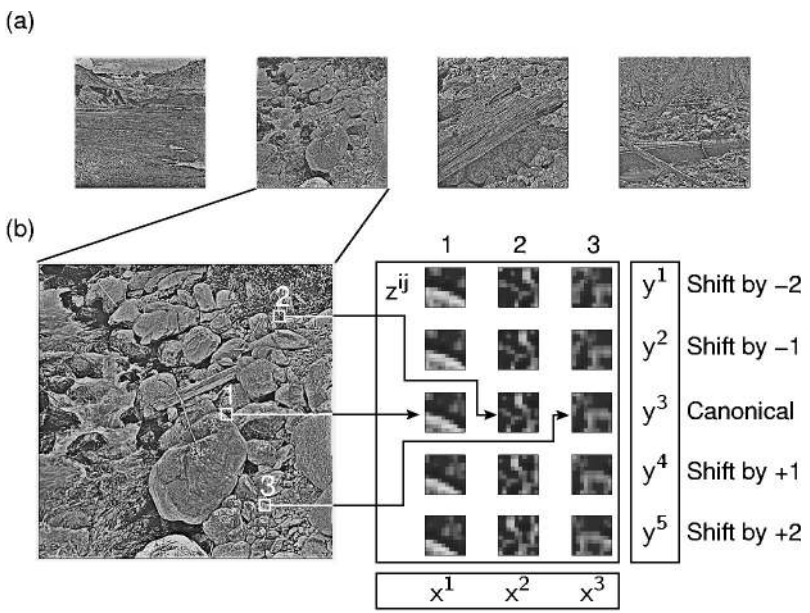

Figure 3: Example training data from natural scene images. Training data are formed by randomly selected patch locations from a set of natural images (a). (b) The patch is then transformed to form the training set of patches $\mathbf{z}^{ij}$. In this case, the patch is shifted using horizontal translations of ± 2 pixels. To learn a model of style/content separation, a single $\mathbf{x}$ vector is used to represent each image in a column, and a single $\mathbf{y}$ vector represents each image in a row.

a high-level view of the training paradigm in which patches are randomly selected from larger images and subsequently transformed.

We initially tested the application of the gradient-descent rules simultaneously to estimate $\{\mathbf{w}_{ij}\}$, $\mathbf{x}$, and $\mathbf{y}$. Unfortunately, obtaining convergence reliably was rather difficult in this situation, due to a degeneracy in the model in the form of an unconstrained degree of freedom. Given a constant c, there is ambiguity in the model since $P(\mathbf{z}|c\mathbf{x}, \frac{1}{c}\mathbf{y}) = P(\mathbf{z}|\mathbf{x}, \mathbf{y})$. Our use of the priors $P(\mathbf{x}), P(\mathbf{y})$ largely mitigates problems stemming from this degeneracy, yet oscillations are still possible when both $\mathbf{x}$ and $\mathbf{y}$ are adapted simultaneously.

Fortunately, we found that minimizing $E(\{\mathbf{w}_{ij}\}, \mathbf{x}, \mathbf{y})$ with respect to a single variable until near convergence yields good results, particularly when combined with a batch derivative approach. This approach of iteratively performing MAP estimation with respect to a single variable at a time is known within the statistics community as iterated conditional modes (ICM) (Besag, 1986). ICM is a deterministic method shown to generally converge quickly, albeit to a local minimum. In our implementation, we use a conjugate gradient method to speed up convergence, minimizing E with respect to $\mathbf{x}$ and $\mathbf{y}$.

The algorithm we have developed for learning the model parameters $\{\mathbf{w}_{ij}\}$ is essentially an incremental expectation-maximization (EM) algorithm applied to randomly selected subsets ("batches") of training image patches. The algorithm is incremental in the sense that we use a single step in the parameter spaces, increasing the log likelihood of the model but not fully maximizing it. We have observed that an incremental M-step often produces better results than a full M-step (equivalent to fully minimizing equation 3.7 with respect to $\{\mathbf{w}_{ij}\}$ for fixed $\mathbf{x}$ and $\mathbf{y}$)). We believe this is because performing a full M-step on each batch of data can potentially lead to many shallow local minima, which may be avoided by taking the incremental M-steps.

The algorithm can be summarized as follows (see also the pseudocode labeled algorithm 1). First, randomly initialize the bilinear basis W, and the matrix Y containing a set of vectors describing each style ($\mathbf{y}^s$). For each batch of training image patches, estimate the per patch (indexed by i) vectors $\mathbf{x}^i$ and $\mathbf{y}^{i,s}$ for a set of transformations (indexed by s). This corresponds to the E-step in the EM algorithm. In the M-step, take a single gradient step with respect to W using the estimated $\mathbf{x}^i$ and $\mathbf{y}^{i,s}$ values. In order to regularize Y and avoid overfitting to a particular set of patches, we slowly adapt Y over time as follows. For each particular style s, adapt $\mathbf{y}^s$ toward the mean of all inferred vectors $\mathbf{y}^{*,s}$ corresponding to patches transformed according to style s (line 11 in algorithm 1). Averaging $\mathbf{y}^s$ across all patches for a particular transformation encourages the style representation to be invariant to particular patch content.

Without additional constraints, the algorithm above would not necessarily learn to represent content only in $\mathbf{x}$ and transformations only in $\mathbf{y}$. In order to learn style and content separation for transformation invariance,

Algorithm 1: LearnSparseBilinearModel($I, T, l, \alpha, \beta, \epsilon, \eta, \varepsilon$)

$W \Leftarrow$ RandNormalizedVectors(k, m, n)
$Y \Leftarrow$ RandNormalizedVectors(n, r)
for $iter \Leftarrow 1, \cdots, maxIter$ **do**
  $P \Leftarrow$ SelectPatchLocations(I, q)
  $Z \Leftarrow$ ExtractPatches(I, P)
  $X \Leftarrow$ InferContent($W, Y(:, c), Z, \alpha$)
  $dW \Leftarrow$ Zeros(k, m, n)
  for $s \Leftarrow 1, \cdots, r$ **do**
    $Z \Leftarrow$ ExtractPatches(I, Transform($P, T(:, s)$))
    $Y_{batch} \Leftarrow$ InferStyle(W, X, Z, β)
    $Y(:, s) \Leftarrow (1 - \epsilon)Y(:, s) + \epsilon \cdot$ SampleMean(Y_{batch})
    $\mathbf{y}_{var} \Leftarrow (1 - \varepsilon)\mathbf{y}_{var} + \varepsilon \cdot$ SampleVar(Y_{batch})
    $\mathbf{g_y} \Leftarrow \mathbf{g_y} \star \left(\frac{\mathbf{y}_{var}}{\sigma_g^2}\right)^{\gamma}$
    $Y(:, s) \Leftarrow$ NormalizeVectors($Y(:, s)$, $\mathbf{g_y}$)
    $dW \Leftarrow dW + \left(\frac{1}{r}\right) dEdW(Z, W, X, Y(:, s))$
  end for
  $W = W + \eta dW$
  $\mathbf{x}_{var} \Leftarrow (1 - \varepsilon)\mathbf{x}_{var} + \varepsilon \cdot$ SampleVar(X)
  $\mathbf{g_x} \Leftarrow \mathbf{g_x} \star \left(\frac{\mathbf{x}_{var}}{\sigma_g^2}\right)^{\gamma}$
  $W \Leftarrow$ NormalizeVectors(W, $\mathbf{g_x}$)
end for

Algorithm 2: InferStyle(W, X, Z, β)

$Y \Leftarrow$ ConjGradSparseFit(W, X, Z, β)

we estimate $\mathbf{x}$ and $\mathbf{y}$ in a constrained fashion. We first infer $\mathbf{x}^i$. Finding the initial $\mathbf{x}$ vector relies on having an initial $\mathbf{y}$ vector. Thus, we refer to one of the transformations as "canonical," corresponding to the identity transformation. This transformation vector $\mathbf{y}^c$ is used for initially bootstrapping the content vector $\mathbf{x}^i$, but besides this use is adapted exactly like the rest of the style vectors in the matrix Y. We then use the same $\mathbf{x}^i$ to infer all $\mathbf{y}^{i,s}$ vectors for each transformed patch $\mathbf{z}^{i,s}$. This ensures that a single content vector $\mathbf{x}^i$ codes for the content in the entire set of transformed patches $\mathbf{z}^{i,s}$.

Algorithm 1 presents the pseudocode for the learning algorithm. It makes use of algorithms 2 and 3 for inferring style and content, respectively. Table 1 describes the variables used in the learning of the sparse bilinear model from natural images. Capital letters indicate matrices containing column vectors. Individual columns are indexed using Matlab style "slicing," for example, $Y(:, s)$ is the ith column of Y $\mathbf{y}^i$. The $\star$ indicates the element-wise product.

Algorithm 3: InferContent(W, Y, Z, α)

1: $X \Leftarrow$ ConjGradSparseFit(W, Y, Z, α)

Table 1: Variables for Learning and Inference.

Name	Size	Description	Name	Size	Description
m	*scalar*	**x** (content) dim.	n	*scalar*	**y** (style) dim.
k	*scalar*	**z** (patch) dim.	l	*scalar*	Number of patches in batch
I	$[k', l']$	Full-sized images	Z	$[k, l]$	Image patches
α	scalar	x_i prior weight	β	scalar	y_j prior weight
η	scalar	W adaptation rate	ϵ	scalar	Y adaptation rate
ε	scalar	$\mathbf{y}_{var}$ adaptation rate	γ	scalar	Variance penalty
σ_g^2	scalar	x_i goal variance	c	scalar	Canon. transform idx.
r	scalar	Number of transforms	T	$[2, r]$	Transform parameters

4 Results

4.1 Training Methodology. We tested the algorithms for bilinear sparse coding on natural image data. The natural images we used are distributed by Olshausen and Field (1997) along with the code for their algorithm. Except where otherwise noted in an individual experiment, the training set of images consisted of 10×10 pixel patches randomly extracted from 10 512×512 pixel source images. The images are prewhitened to equalize large variances in frequency, helping to speed convergence. To assist convergence, all learning occurs in batch mode, where each batch consists of $l = 100$ image patches.

The effects of varying the number of bilinear basis units m and n were investigated in depth. An obvious setting for m, the dimension of the content vector **x**, is to use the value corresponding to a complete basis set in the linear model (m equals the number of pixels in the patch). As we will demonstrate, this choice for m yields good results. However, one might imagine that because of the ability to merge representations of features that are equivalent with respect to the transformations, m can be set to a much lower value and still effectively learn the same basis. As we discuss later, this is true only if the features can be transformed independently.

An obvious choice for n, the number of dimensions of the style vector **y**, is simply the number of transformations in the training set. However, we found that the model is able to perform substantial dimensionality reduction, and we present the case where the number of transformations is 49 and a basis of $n = 10$ is used to represent translated features.

Experiments were also performed to determine values for the sparseness parameters α and β. Settings between 25 to 35 for α and between 0 and 5 for β appear to reliably yield localized, oriented features. Ranges are given

here because optimum values seem to depend on n for two reasons. First, as n increases, the number of prior terms in equation 3.8 is significantly less than the mn likelihood terms. Thus, α must be increased to force sparseness on the posterior distribution on $\mathbf{x}$. This intuitive explanation is reinforced by noting that α is approximately a factor of n larger than those found previously (Olshausen & Field, 1997). Second, if dimensionality reduction is desired (i.e., n is reduced), β must be lowered or set to zero, as the elements of $\mathbf{y}$ cannot be coded in an independent fashion. For instance, in the case of dimensionality reduction from 49 to 10 transformation basis vectors $\beta = 0$.

The W step size η for gradient descent using equation 3.10 was set to 0.25. Variance normalization used $\varepsilon = 0.25$, $\gamma = 0.05$, and $\sigma_g^2 = 0.1$.

These parameters are not necessarily the optimum parameters, and the algorithm is robust with respect to significant parameter changes. Generally, only the amount of time required to find a good sparse representation changes with untuned parameters. In the cases presented here, the algorithm converged after approximately 2500 iterations, although to ensure that the representations had converged, we ran several runs for 10,000 iterations.

The transformations for most of the experiments were chosen to be two-dimensional translations in the range $[-3, 3]$ pixels in both the axes. The experiments measuring transformation invariance (see section 4.4) considered one-dimensional translations in the range of $[-8, 8]$ in 12×12 sized patches.

4.2 Bilinear Sparse Coding of Natural Images. Experimental results are analyzed as follows. First, we study the qualitative properties of the learned representation, then look quantitatively at how model parameters affect these and other properties, and finally examine the learned model's invariance properties.

Figures 4 and 5 show the results of training the sparse bilinear model on natural image data. Both show localized oriented features resembling Gabor filters. Qualitatively, these are similar to the features learned by the model for linear sparse coding. Some features appear to encode more complex features than is common for linear sparse coding. We offer several possible explanations here. First, we believe that some deformations of a Gabor response are caused by the occlusion introduced by the patch boundaries. This effect is most pronounced when the feature is effectively shifted out of the patch based on a particular translation and its canonical (or starting) location. Second, we believe that closely located and similarly oriented features may sometimes be representable in the same basis feature by using slightly different transformations representations. In turn, this may "free up" a basis dimension for representing a more complex feature.

The bilinear method is able to model the same features under different transformations. In this case, horizontal translations in the range $[-3, 3]$ were used for training the model. Figure 4 provides an example of how

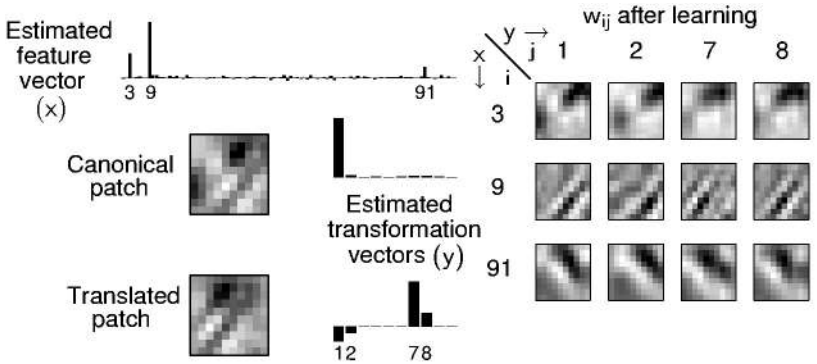

Figure 4: Representing natural images and their transformations with a sparse bilinear model. The representation of an example natural image patch and of the same patch translated to the left. Note that the bar plot representing the **x** vector is indeed sparse, having only three significant coefficients. The code for the translation vectors for both the canonical patch and the translated one is likewise sparse. The $\mathbf{w}_{ij}$ basis images are shown for those dimensions that have nonzero coefficients for x_i or y_j.

the model encodes a natural image patch and the same patch after it has been translated. Note that in this case, both the **x** and **y** vectors are sparse. Figure 5 displays the transformed features for each translation represented by a learned $\mathbf{y}^s$ vector.

Figure 6 shows how the model can account for a given localized feature at different locations by varying the **y** vector. As shown in the last column of the figure, the translated local feature is generated by linearly combining a sparse set of basis vectors $\mathbf{w}_{ij}$. This figure demonstrates that the bilinear form of the interaction function $f(\mathbf{x}, \mathbf{y})$ is sufficient for translating features to different locations.

4.3 Effects of Sparseness on Representation. The free parameters α and β play an important role in deciding how sparse the coefficients in the vectors **x** and **y** are. Likewise, the sparseness of the vectors is intertwined with the desired local and independent properties of the $\mathbf{w}_{ij}$ bilinear basis features. As noted in other research on sparseness (Olshausen & Field, 1996), both the attainable sparseness and independence of features also depend on the model dimensionality—in our case, the parameters m and n. In all of our experiments, we use a complete basis (in which $m = k$) for content representation, assuming that the translations do not affect the number of basis features needed for representation. We believe this is justified also by the very idea that changes in style should not change the intrinsic content.

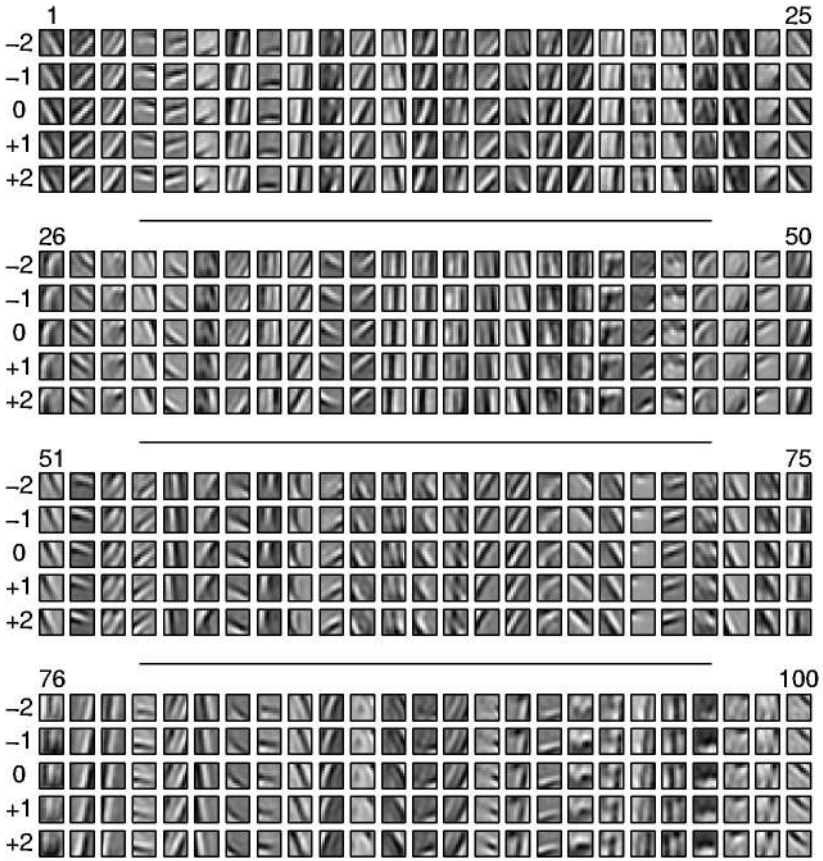

Figure 5: Localized, oriented set of learned basis features. The transformed basis features learned by the sparse bilinear model for a set of five horizontal translations. Each block of transformed features $\mathbf{w}_i^{\psi^s}$ is organized with values of $i = 1, \ldots, m$ across the rows and values of $s = 1, \ldots, r$ down the columns. In this case, $m = 100$ and $r = 5$. Note that almost all of the features exhibit localized and oriented properties and are qualitatively similar to Gabor features.

In theory, the style vector $\mathbf{y}$ could also use a sparse representation. In the case of affine transformations on the plane, using a complete basis for $\mathbf{y}$ means using a large value on n. From a practical perspective, this is undesirable, as it would essentially equate to the tiling of features at all possible transformations. Thus, in our experiments, we set β to a small or zero value and also perform dimensionality reduction by setting n to a fraction of the number of styles (usually between four and eight). This configuration allows

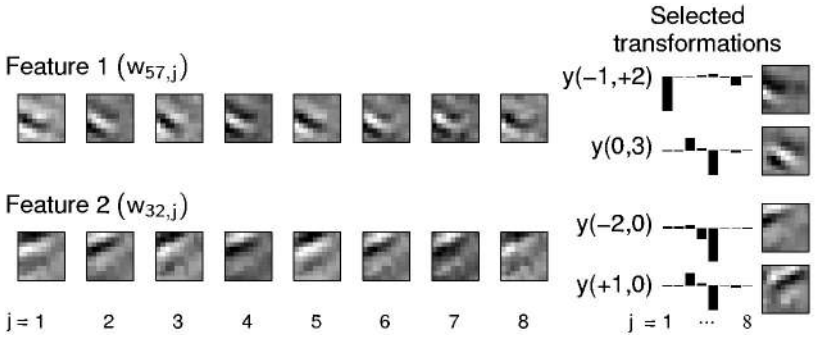

Figure 6: Translating a learned feature to multiple locations. The two rows of eight images represent the individual basis vectors $\mathbf{w}_{ij}$ for two values of i. The y_j values for two selected transformations for each i are shown as bar plots. $y(a, b)$ denotes a translation of (a, b) pixels in the Cartesian plane. The last column shows the resulting basis vectors after translation.

learning of sparse, independent content features while taking advantage of dimensionality reduction in the coding of transformations.

We also analyzed the effects of the sparseness weighting term α. Figure 7 illustrates the effect of varying α on the sparseness of the content representation $\mathbf{x}$ and on the log posterior optimization function E (see equation 3.7). The results shown are based on 1000 $\mathbf{x}$ vectors inferred by presenting the learned model with a random sample of 1000 natural image patches. For each value of α, we also average over five runs of the learning algorithm. Figure 8 illustrates the effect of the sparseness weighting term α on the kurtosis of $\mathbf{x}$ and the basis vectors learned by the algorithm.

4.4 Style and Content Invariance. A series of experiments were conducted to analyze the invariance properties of the sparse bilinear model. These experiments examine how transformation ($\mathbf{y}$) and content representations ($\mathbf{x}$) change when the input patch is translated in the plane. After learning a sparse bilinear model on a set of translated patches, we select a new test patch $\mathbf{z}^0$ and estimate a reference content vector $\mathbf{x}^0$ using InferContent$(W, \mathbf{y}^c, \mathbf{z}^0)$. We then shift the patch according to transformation i: $\mathbf{z}^i = T_i(\mathbf{z}^0)$. Next, we infer the new $\mathbf{y}^i$ using InferStyle$(W, \mathbf{x}^0, \mathbf{z}^i)$ (without using knowledge of the transformation parameter i). Finally, we infer the new content representation $\mathbf{x}^i$ using a call to the procedure InferContent$(W, \mathbf{y}^i, \mathbf{z}^i)$.

To quantify the amount of change or variance in a given transformation or content representation, we use the L_2 norm of the vector difference between the reestimated vector and the initial vector. To normalize our metric and account for different scales in coefficients, we divide by the norm of the

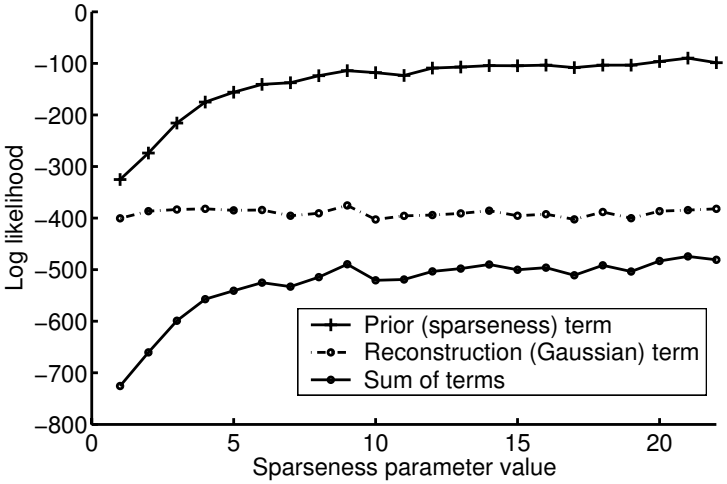

Figure 7: Effect of sparseness on the optimization function. As the sparseness weighting value is increased, the sparseness term ($\log P(\mathbf{x})$) takes on higher values, increasing the posterior likelihood. Note that the reconstruction likelihood ($\log P(\mathbf{z}|\mathbf{x}, \mathbf{y}, \{\mathbf{w}_{ij}\})$) is effectively unchanged, even as the sparseness term is weighted 20 times more heavily, thus illustrating that the sparse code does not cause a loss of representational fidelity.

reference vector:

$$\Delta x_i = \frac{|\mathbf{x}^i - \mathbf{x}^0|}{|\mathbf{x}^0|} \qquad \Delta y_i = \frac{|\mathbf{y}^i - \mathbf{y}^0|}{|\mathbf{y}^0|}. \tag{4.1}$$

For testing invariance, the model was trained on 12×12 patches and vertical shifts in the range $[-8, 8]$, with $m = 144$ and $n = 15$. Figure 9 shows the result of vertically shifting a particular content (image patch at a particular location) and recording the subsequent representational changes. Figure 10 shows the result of selecting random patches (different content vectors $\mathbf{x}$) and translating each in an identical way (same translational offset).

Both figures show a strong degree of invariance of representation: the content vector $\mathbf{x}$ remains approximately unchanged when subjected to different translations, while the translation vector $\mathbf{y}$ remains approximately the same when different content vectors are subject to the same translation.

While Figures 9 and 10 show two sample test sequences, Figures 11 and 12 show the transformation-invariance properties for the average of 100 runs on shifts in the range $[-8, 8]$ in steps of 0.5. The amount of variance in $\mathbf{x}$ to translations of up to three pixels is less than a 2 percent change. These

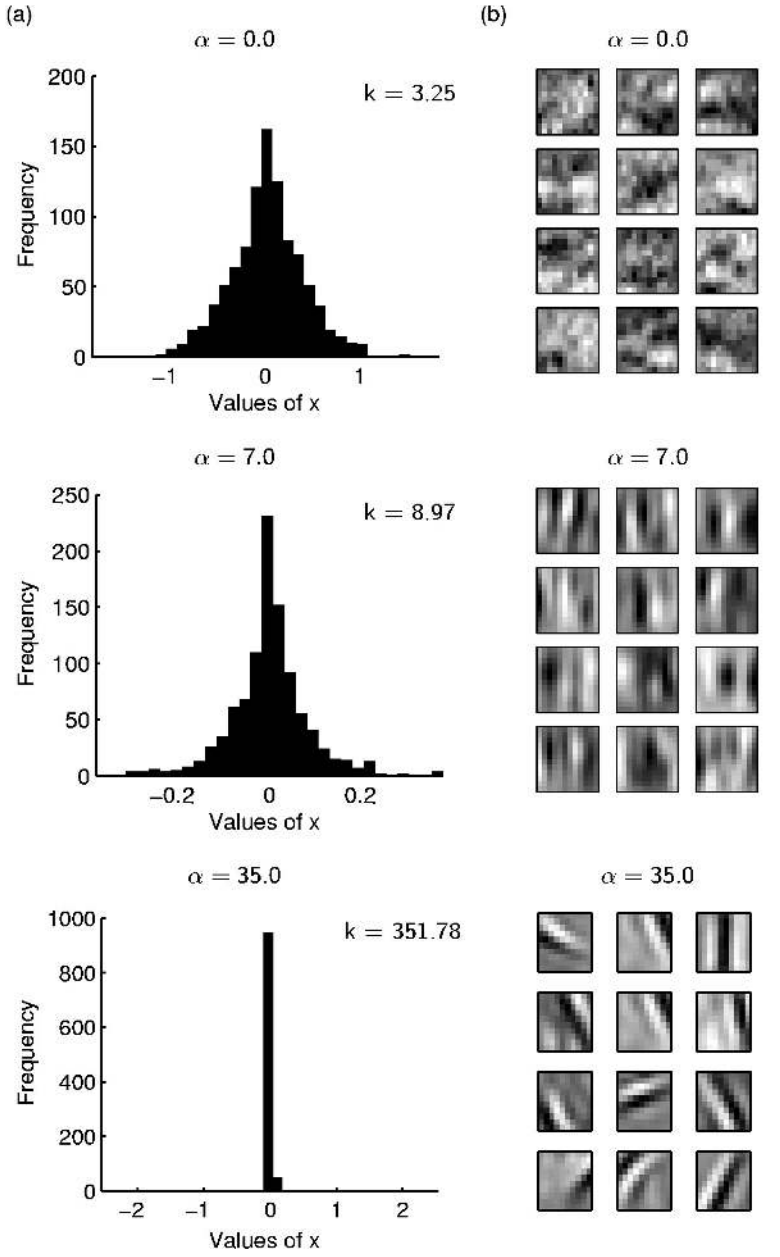

Figure 8: Sparseness prior yields highly kurtotic coefficient distributions. (a) The effect of weighting the sparseness prior for $\mathbf{x}$ (via α) on the kurtosis (denoted by k) of x_i coefficient distribution. (b) A subset of the corresponding bilinear basis vectors learned by the algorithm.

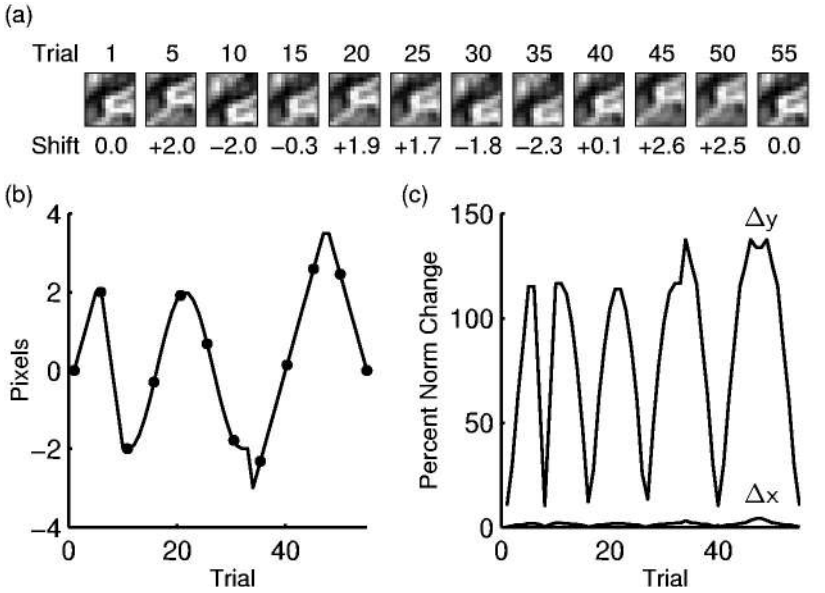

Figure 9: Transformation-invariance property of the sparse bilinear model. (a) Randomly selected natural image patch transformed by an arbitrary sequence of vertical translations. (b) Sequence of vertical pixel translations applied to the original patch location. (c) Effects of the transformations on the transformation (**y**) and patch content (**x**) representation vectors. Note that the magnitude of the change in **y** is well correlated to the magnitude of the vertical translation, while the change in **x** is relatively insignificant (mean $\Delta \mathbf{x} = 2.6$), thus illustrating the transformation-invariance property of the sparse bilinear model.

results suggest that the sparse bilinear model is able to learn an effective representation of translation invariance with respect to local features.

Figure 13 compares the effects of translation in the sparse linear versus sparse bilinear model. We first trained a linear model using the corresponding subset of parameter values used in the bilinear model. The same metric for measuring changes in representation was used by first estimating $\mathbf{x}^0$ for a random patch and then reestimating $\mathbf{x}^i$ on a translated patch. As expected, the bilinear model exhibits a much greater degree of invariance to transformation than the linear model.

4.5 Interpolation for Continuous Translations. Although transformations are learned as discrete style classes, we found that the sparse bilinear model can handle continuous transformations. Linear interpolation in the style space was found to be sufficient for characterizing translations in the continuum between two discrete learned translations. Figure 14a shows

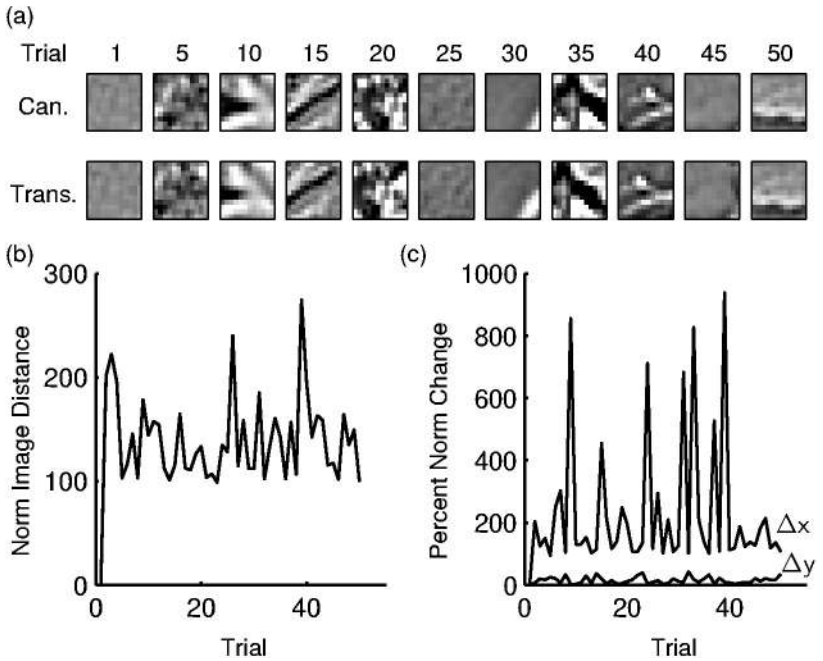

Figure 10: Invariance of style representation to content. (a) Sequence of randomly selected patches (denoted "Can.") and their horizontally shifted versions (denoted "Trans."). (b) Plot of the the L^2 distance in image space between the two canonical images. (c) The change in the inferred $\mathbf{y}$ for the translated version of each patch. Note that the patch content representation fluctuates wildly (as does the distance in image space), while the translation vector changes very little.

the values of the six-dimensional $\mathbf{y}$ vector for each learned translation. The filled circles on each line represent the value of the learned translation vector for that dimension. Plus symbols indicate the resulting linearly interpolated translation vectors. Note that generally the values vary smoothly with respect to the translation amount, allowing simple linear interpolation between translation vectors, similar to the method of locally linear embedding (Roweis & Saul, 2000). Figure 14b shows the style vectors for a model trained with only the transformations $-4, -2, 0, +2, +4$. Figure 14c shows examples of three reconstructed patches for two interpolated style vectors (for translations, -3 and $+0.5$ pixels). Also shown is the mean squared error (MSE) over all image pixels between each reconstructed patch and the actual translated patch. The MSE values for the interpolated cases are somewhat higher than those for translations in the training set (labeled "learned" in the figure) but within the range of MSE values across all image patches.

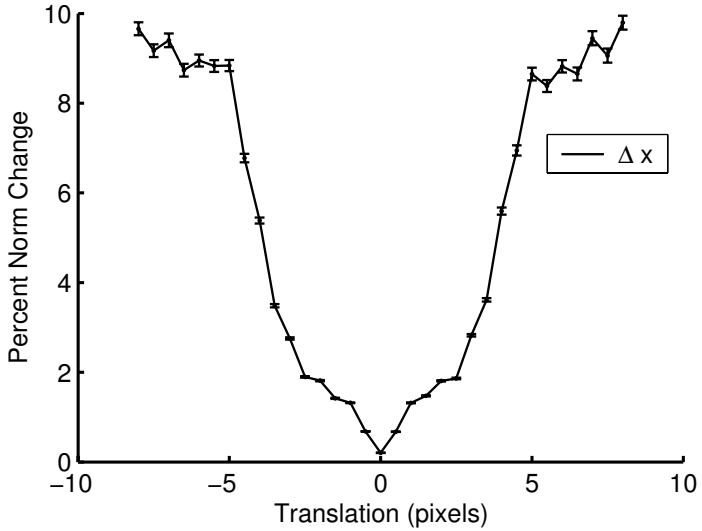

Figure 11: Effects of translation on patch content representation. The average relative change in the content vector **x** versus translation over 100 experiments in which 12×12 image patches were shifted by varying degrees. Note that for translations in the range $(-3, +3)$, the relative change in **x** is small, yet as more and more features are shifted into the patch, the content representation must change. Error bars represent the standard deviation over the 100 experiments.

5 Discussion

5.1 Related Work. Our work is based on a synthesis of two extensively studied tracks of vision research. The first is transformation-invariant object representations, and the second is the extraction of sparse, independent features from images. The key observation is that the combination of the two tracks of research can be synergistic, effectively making each problem easier to solve.

A large body of work exists on transformation invariance in image processing and vision. As discussed by Wiskott (2004), the approaches can be divided into two rough classes: (1) those that explicitly deal with different scales and locations by means of normalizing a perceived image to a canonical view and (2) those that find simple localized features that become transformation invariant by pooling across various scales and regions. The first approach is essentially generative, that is, given a representation of "what" (content) and "where" (style), the model can output an image. The bilinear model is one example of such an approach; others are discussed below. The second approach is discriminative in the sense that the model operates by extracting object information from an image and discards posi-

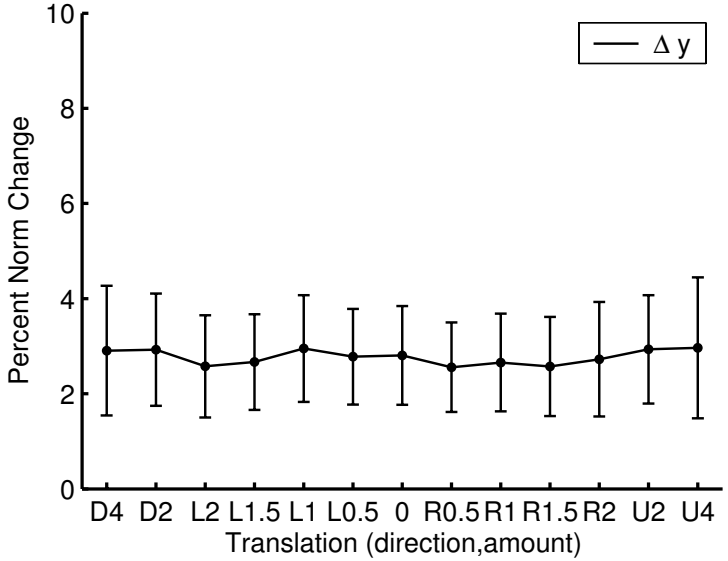

Figure 12: Effects of changing patch content on translation representation. The average relative change in $\mathbf{y}$ for random patch content over 100 experiments in which various transformations were performed on a randomly selected patch. No discernible pattern seems to exist to suggest that some transformations are more sensitive to content than others. The bar shows the mean relative change for each transformation.

tional and scale information. Generation of an image in the second approach is difficult because the pooling operation is noninvertible and discards the "where" information. Examples of this approach include weight sharing and weight decay models Hinton, 1987; Földiák, 1991), the neocognitron (Fukushima, Miyake, & Takayukiito, 1983), LeNet (LeCun et al., 1989), and slow feature analysis (Wiskott & Sejnowski, 2002).

A key property of the sparse bilinear model is that it preserves information about style and explicitly uses this information to achieve invariance. It is thus similar to the shifting and routing circuit models of Anderson and Van Essen (1987) and Olshausen, Anderson, and Van Essen (1995). Both the bilinear model and the routing circuit model retain separate pathways containing "what" and "where" information. There is also a striking similarity in the way routing nodes in the routing circuit model select for scale and shift to mediate routing, and the multiplicative interaction of style and content in the bilinear model. The $\mathbf{y}$ vector in the bilinear model functions in the same role as the routing nodes in the routing circuit model. One important difference is that while the parameters and structure of the routing circuit are selected a priori for a specific transformation, our focus is on learning

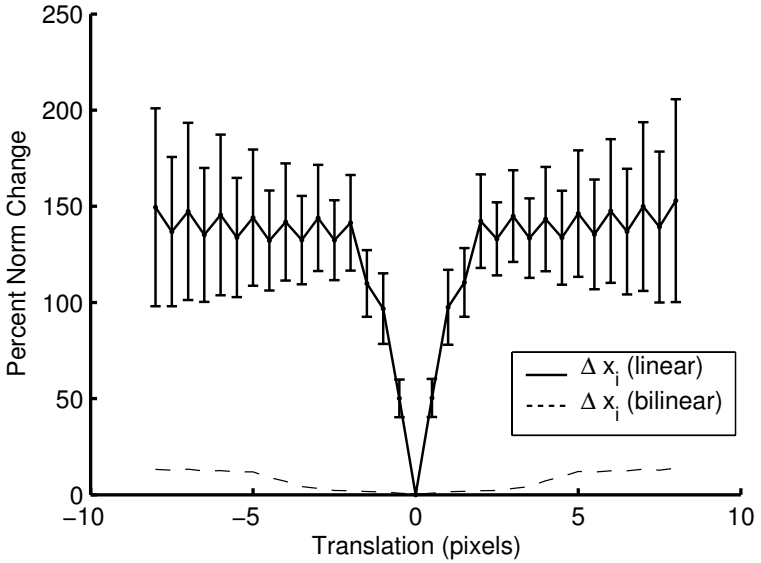

Figure 13: Effects of transformations on the content vector in the bilinear model versus the linear model. The solid line shows, for a linear model, the relative content representation change due to the input patch undergoing various translations. The points on the line represent the average of 100 image presentations per translation, and the error bars indicate the standard deviation. For reference, the results from the bilinear model (shown in detail in Figure 11) are plotted with a dashed line. This shows the high degree of transformation invariance in the bilinear model in comparison to the linear model, whose representation changes steeply with small translations of the input.

arbitrary transformations directly from natural images. The bilinear model is similarly related to the Lie group–based model for invariance suggested in Rao and Ruderman (1999), which also uses multiplicative interactions but in a more constrained way by invoking a Taylor-series expansion of a transformed image.

Our emphasis on modeling both "what" and "where" rather than just focusing on invariant recognition ("what") is motivated by the belief that preserving the "where" information is important. This information is critical not simply for recognition but for acting on visual information, in both biological and robotic settings. The bilinear model addresses this issue in a very explicit way by directly modeling the interaction between "what" and "where" processes, similar in spirit to the "what" and "where" dichotomy seen in biological vision (Ungerleider & Mishkin, 1982).

The second major component of prior work on which our model is based is that of representing natural images through the use of sparse or statisti-

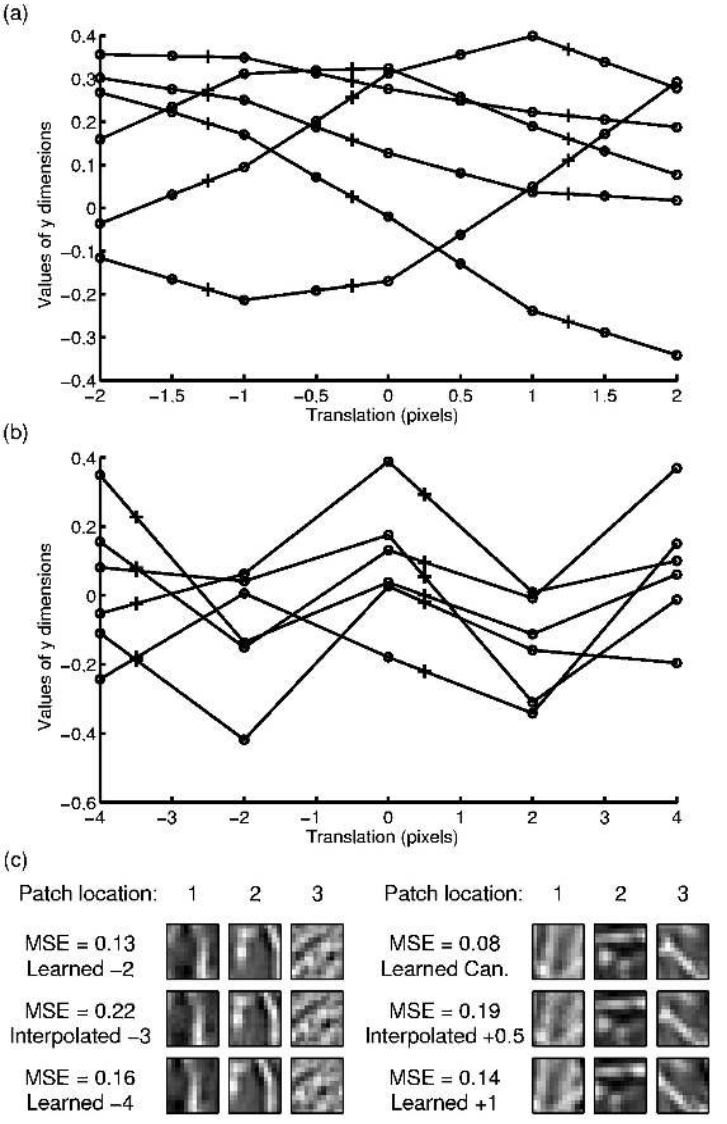

Figure 14: Modeling continuous transformations by interpolating in translation space. (a) Each line represents a dimension of the tranformation space. The values for each dimension at each learned translation are denoted by circles, and interpolated subpixel coefficient values are denoted by plus marks. Note the smooth transitions between learned translation values, which allow interpolation. (b) Tranformation space values as in *a* when learned on coarser translations (steps of 2 pixels). (c) Two examples of interpolated patches based on the training in *b*. See section 4.5 for details.

cally independent features. As discussed in section 1, our work is strongly based on the work of Olshausen and Field (1997), Bell and Sejnowski (1997), and Hinton and Ghahramani (1997) in forming local, sparse distributed representations directly from images.

The sparse and independent nature of the learned features and their locality enables a simple, essentially linear model (given fixed content) to efficiently represent transformations of that content. Given global eigenvectors such as those resulting from principal component analysis, this would be more difficult to achieve. A second benefit of combining transformation-invariant representation with sparseness is that the multiplicity of the same learned features at different locations and transformations can be reduced by explicitly learning transformations of a given feature.

Finally, our use of a lower-dimensional representation based on the bilinear basis to represent inputs and to interpolate style and content coefficients between known points in the space has some similarities to the method of locally linear embedding (Roweis & Saul, 2000).

5.2 Extension to a Parts-Based Model of Object Representation. The bilinear generative model in equation 2.2 uses the same set of transformation values y_j for all the features $i = 1, \ldots, m$. Such a model is appropriate for global transformations that apply to an entire image region such as a shift of p pixels for an image patch or a global illumination change. Consider the problem of representing an object in terms of its constituent parts (Lee & Seung, 1999). In this case, we would like to be able to transform each part independent of other parts in order to account for the location, orientation, and size of each part in the object image. The standard bilinear model can be extended to address this need as follows:

$$\mathbf{z} = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} \mathbf{w}_{ij} y_j^i \right) x_i. \tag{5.1}$$

Note that each object feature i now has its own set of transformation values y_j^i. The double summation is thus no longer symmetric. Also, note that the standard model (see equation 2.2) is a special case of equation 5.1 where $y_j^i = y_j$ for all i.

We have tested the feasibility of equation 5.1 using a set of object features learned for the standard bilinear model. Preliminary results (Grimes & Rao, 2003) suggest that allowing independent transformations for the different features provides a rich substrate for modeling images and objects in terms of a set of local features (or parts) and their individual transformations relative to an object-centered reference frame. Additionally, we believe that the number of basis features needed to represent natural images could be greatly reduced in the case of independently transformed features. A single "template" feature could be learned for a particular orientation and then

translated to represent a localized, oriented image feature.

6 Summary and Conclusion

A fundamental problem in vision is to simultaneously recognize objects and their transformations (Anderson & Van Essen, 1987; Olshausen et al., 1995; Rao & Ballard, 1998; Rao & Ruderman, 1999; Tenenbaum & Freeman, 2000). Bilinear generative models provide a tractable way of addressing this problem by factoring an image into object features and transformations using a bilinear function. Previous approaches used unconstrained bilinear models and produced global basis vectors for image representation (Tenenbaum & Freeman, 2000). In contrast, recent research on image coding has stressed the importance of localized, independent features derived from metrics that emphasize the higher-order statistics of inputs (Olshausen & Field, 1996, 1997; Bell & Sejnowski, 1997; Lewicki & Sejnowski, 2000). This paper introduces a new probabilistic framework for learning bilinear generative models based on the idea of sparse coding.

Our results demonstrate that bilinear sparse coding of natural images produces localized oriented basis vectors that can simultaneously represent features in an image and their transformation. We showed how the learned generative model can be used to translate a basis vector to different locations, thereby reducing the need to learn the same basis vector at multiple locations, as in traditional sparse coding methods. We also demonstrated that the learned representations for transformations vary smoothly and allow simple linear interpolation to be used for modeling transformations that lie in the continuum between training values. Finally, we showed that the object representation (the "content") in the sparse bilinear model remains invariant to image translations, thus providing a basis for invariant vision. Our current efforts are focused on exploring the application of the model to learning other types of transformations such as rotations, scaling, and view changes. We are also investigating a framework for learning parts-based object representations that builds on the bilinear sparse coding model presented in this article.

Acknowledgments

This research is supported by NSF Career grant 133592, ONR YIP grant N00014-03-1-0457, and fellowships to R.P.N.R. from the Packard and Sloan foundations.

References

Anderson, C. H., & Van Essen, D. C. (1987). Shifter circuits: A computational strategy for dynamic aspects of visual processing. *Proceedings of the National Academy of Sciences*, *84*, 1148–1167.

Attneave, F. (1954). Some informational aspects of visual perception. *Psychological Review*, *61*(3), 183–193.

Barlow, H. B. (1961). Possible principles underlying the transformation of sensory messages. In W. A. Rosenblith (Ed.), *Sensory communication* (pp. 217–234). Cambridge, MA: MIT Press.

Bell, A. J., & Sejnowski, T. J. (1997). The "independent components" of natural scenes are edge filters. *Vision Research*, *37*(23), 3327–3338.

Besag, J. (1986). On the Statistical Analysis of Dirty Pictures. *J. Roy. Stat. Soc. B*, *48*, 259–302.

Földiák, P. (1991). Learning invariance from transformation sequences. *Neural Computation.*, *3*(2), 194–200.

Földiák, P., & Young, M. P. (1995). Sparse coding in the primate cortex. In M. A. Arbib (Ed.), *The handbook of brain theory and neural networks* (pp. 895–898). Cambridge, MA: MIT Press.

Fukushima, K., Miyake, S., & Takayukiito (1983). Neocognitron: A neural network model for a mechanism of visual pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, *13*(5), 826–834.

Grimes, D. B., & Rao, R. P. N. (2003). A bilinear model for sparse coding. In S. Becker, S. Thrün, & K. Obermayer (Eds.), *Advances in neural information processing systems, 15*. Cambridge, MA: MIT Press.

Hinton, G. E. (1987). Learning translation invariant recognition in a massively parallel network. In G. Goos & J. Hartmanis (Eds.), *PARLE: Parallel architectures and languages Europe* (pp. 1–13). Berlin: Springer-Verlag.

Hinton, G. E., & Ghahramani, Z. (1997). Generative models for discovering sparse distributed representations. *Philosophical Transactions Royal Society B*, *352*, 1177–1190.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, *1*(4), 541–551.

Lee, D. D., & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, *401*, 788–791.

Lewicki, M. S., & Sejnowski, T. J. (2000). Learning Overcomplete Representations. *Neural Computation*, *12*(2), 337–365.

Olshausen, B., Anderson, C., & Van Essen, D. (1995). A multiscale routing circuit for forming size- and position-invariant object representations. *Journal of Computational Neuroscience*, *2*, 45–62.

Olshausen, B. A., & Field, D. J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, *381*, 607–609.

Olshausen, B. A., & Field, D. J. (1997). Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, *37*, 33113325.

Rao, R. P. N., & Ballard, D. H. (1998). Development of localized oriented receptive fields by learning a translation-invariant code for natural images. *Network: Computation in Neural Systems*, *9*(2), 219–234.

Rao, R. P. N., & Ballard, D. H. (1999). Predictive coding in the visual cortex: A functional interpretation of some extra-classical receptive field effects. *Nature Neuroscience*, *2*(1), 79–87.

Rao, R. P. N., & Ruderman, D. L. (1999). Learning Lie groups for invariant visual perception. In M. S. Kearns, S. Solla, & D. Cohn (Eds.), *Advances in neural information processing systems, 11* (pp. 810–816). Cambridge, MA: MIT Press.

Roweis, S., & Saul, L. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science, 290*(5500), 2323–2326.

Schwartz, O., & Simoncelli, E. P. (2001). Natural signal statistics and sensory gain control. *Nature Neuroscience, 4*(8), 819–825.

Tenenbaum, J. B., & Freeman, W. T. (2000). Separating style and content with bilinear models. *Neural Computation, 12*(6), 1247–1283.

Ungerleider, L., & Mishkin, M. (1982). Two cortical visual systems. In D. Ingle, M. Goodale, & R. Mansfield (Eds.), *Analysis of visual behavior* (pp. 549–585). Cambridge, MA: MIT Press.

Wiskott, L. (2004). How does our visual system achieve shift and size invariance? In J. L. van Hemmen & T. J. Sejnowski (Eds.), *Problems in systems neuroscience*. New York: Oxford University Press.

Wiskott, L., & Sejnowski, T. (2002). Slow feature analysis: Unsupervised learning of invariances. *Neural Computation, 14*(4), 715–770.

Received November 4, 2003; accepted June 4, 2004.