Second-Order Statistics of Natural Images

Hauke Bartsch and Klaus Obermayer

Dept. of Electrical Engineering and Computer Science,
Technical University Berlin, Germany, E-mail: hauke@cs.tu-berlin.de

Abstract. Assuming adaptation of the visual cortex to its environment, we analyse the invariance found in natural images to explain the selective response of visual cortical neurons.

We argue that the invariant structure of images can be formally expressed by dot-products. Utilizing the specific structure of the proposed model we show how non-linear functions can be learned efficiently from natural images.

In addition to localized edge detectors we found model neurons that respond to changes in texture and cells that detect edge curvature. The analysis suggests new types of features neurons in primary visual cortex may be selective to.

Key words: symmetry, invariance, texture, curvature

1 Introduction

Under the assumption that the visual cortex adapts to its environment, the invariances found in natural images offer a likely explanation for the emergence of cortical wiring patterns and the operations performed therein [1, 4, 2, 5].

By referring to an object as invariant or symmetric we usually indicate that the object appears unchanged if observed from different perspectives. Symmetry of an object must therefore be defined with respect to an operation or transformation. A high similarity of an object with its transformed counterpart will indicate high symmetry of this object given the transformation.

2 One-dimensional Symmetry Detectors

Transformations will be expressed by a square symmetric matrix A applied onto a data vector $\mathbf{x}$ ($A\mathbf{x} = \mathbf{y}$). Whereas x represents an image, throughout

this paper we will assume that $\mathbf{x}$ is a vector, appending successive columns. If the transformation A leaves the structure in $\mathbf{x}$ unchanged (with respect to its position, form and orientation), we will claim A as coding the symmetry found in $\mathbf{x}$. A well chosen A with respect to symmetry is indicated by $A\mathbf{x} \approx x$. Searching (non-trivial) solutions for A can therefor be done by learning a quadratic form $\mathbf{x}^T A\mathbf{x}$.

Assuming an ensemble X of gray level images ($\mathbf{x} \in X$), what is the information about the image that is explored by A? Using an eigenvalue decomposition on A it is simple to show that calculating the expectation of the quadratic form we arrive at

$$\langle x^T A x \rangle_x = \langle \sum_i^N \lambda_i \mathbf{n}_i^T \mathbf{x}\mathbf{x}_i^T \mathbf{n} \rangle_x = \sum_i^N \lambda_i \mathbf{n}_i^T \langle \mathbf{x}\mathbf{x}^T \rangle_x \mathbf{n}_i = \sum_i^N \lambda_i \mathbf{n}_i^T C \mathbf{n}_i. \tag{1}$$

Only the correlation matrix C of the data will be of importance for defining A which codes therefor statistics of second order.

2.1 Rotation, Scaling and Translation by a binary quadratic form

Before learning the quadratic form from natural images we now try to implement symmetries connected with the rigid transformations rotation, scaling and translation. In analyzing the resulting covariance structure of A we find sets of orthogonal basis functions.

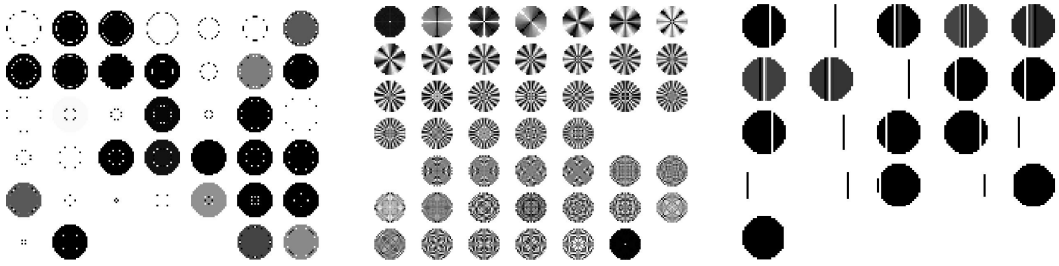

Fig. 1. Orthogonal basis functions for rotation (left), scaling (center) and shift (right) computed as eigenvectors of a quadratic form.

In the following, the coordinates of a pixel i will be denoted by (π-periodic) polar coordinates (θ_i, r_i) in order to simplify the definition of binary matrices A. Because we are only interested in illustrating the representation power of the model we will incorporate the respective covariance structure by a crude approximation into the model. In particular this explains the indefiniteness of the matrix A.

Rotate. The goal is to let A^{rot} encode rotations of the image pixels around their mean position. We have to set the matrix A to the covariance matrix

C obtained from a data set X that contains rotationally symmetric figures. In a first approximation we will assume that a_{ij}^{rot} $(i \neq j)$ will be set to one if $|r_i - r_j| < \varepsilon$ $(i \neq j)$ and otherwise to zero. ε was arbitrarily chosen to be the distance of two neighboring pixels on the outer perimeter of the circular receptive field.

Scale. In a similar way A^{sca} is constructed to encode scaling. Entries of matrix a_{ij} will be set to one only if the angular distance between i and j is sufficiently small, i.e. $|\theta_i - \theta_j| < \varepsilon$ $(i \neq j)$. We do not take into account that by this we also code for inversion symmetries: true scaling would not allow for pixels to cross the origin. It follows that the choice of a specific pixel pair subset does not uniquely define a single symmetry operation.

Shift. A^{shift} is constructed to encode the pixel pairs that occur for translations of pixels along one direction (here 90°). a_{ij} $(i \neq j)$ is set to one only if $(x_i = x_j)$ and otherwise to zero.

All orthogonal basis functions found by the analysis (see Figure 1) show responses for pattern with the respective symmetries, circles for rotational symmetries, crosses and fan-like figures for scale invariance and lines for shift invariance.

3 Multi-dimensional Symmetry Detection

An object may be symmetric with respect to more than one transformation. Thus, we expect to find different instantiations of transformations to be mixed in an ensemble of example patches from natural images. Therefore, a more appropriate model is to find a number i of symmetries each one expressed by its own matrix A^i. Let's define Φ as a vector valued function mapping the data into the space of different symmetry operators:

$$\Phi(\mathbf{x}) = \left(x^T A^1 x,\ x^T A^2 x, \ldots,\ x^T A^N x\right)^T \tag{2}$$

Additional assumptions can be made for the distribution of Φ incorporating for example sparseness, independence or uncorrelateness. But before doing this, we first cast the problem into the notion of dot-products. This will give us the vantage of analyzing a linear model in which assumptions can be justified more easily.

Any quadratic form can be written as a linear weighting in the space of dot-products $\mathbf{x}^T A^i \mathbf{x} = (a_{11}^i, \ldots, a_{kk}^i)(x_1 x_1, \cdots, x_n x_m)^T = \mathbf{w}^i \mathbf{s}$. $\mathbf{s}$ consists of monomials of constant order 2. In reducing the number of calculations needed we will restrict ourself to commutative monomials eliminating the respective entries

from $\mathbf{w}$ ($\mathbf{w}$ is of the dimensionality of vech($\mathbf{x}^T\mathbf{x}$)). Adding more dimensions to the model Φ, e.g. more quadratic forms $x^T A^i x$, we arrive at

$$\Phi = W\mathbf{s}. \tag{3}$$

Here, W is a square matrix. It can be viewed as a linear mixture expressed by the mixing matrix W of the signals $\mathbf{s}$ in the space of dot-products.

Linear methods for monomial spaces of constant order: By the model in Equation 3 the input data $\mathbf{x}$ is projected into a manifold of the higher dimensional space of $\mathbf{s}$. Assuming linear sources in two dimensions source directions (u, v) are mapped into $(u^2, uv, v^2) = u^2(1, a, a^2), a = v/u$ which is again a linear direction in the space of dot-products. Thus the detection of linear direction in the manifold of $\mathbf{s}$ selects linear directions in $\mathbf{x}$.

Notably, because the dimension of $\mathbf{s}$ is larger than the dimension of $\mathbf{x}$ in solving 3 we solve an equivalent problem in $\mathbf{x}$ where the goal is to detect more linear sources than observations.

There is no a priori choice for a specific algorithm solving equation 3. The system is ill-posed (W and $\mathbf{s}$ are unknown), thus additional assumptions about the underlying sources are required. Assuming statistical independence of the transformations in space of dot-products lead us in the next section to methods of ICA.

3.1 Results

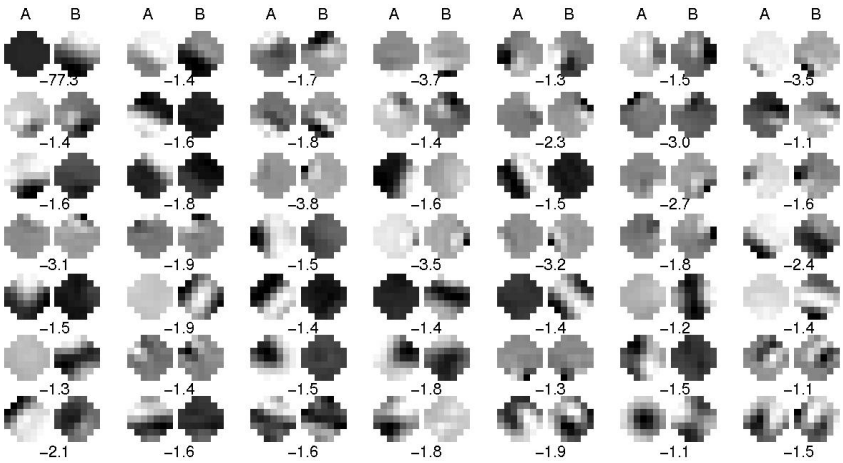

Fig. 2. Learned symmetry detectors by FastICA. For each of 49 found independent component two images (A, B) are shown. The number below each component codes for the ratio of the largest to the second largest eigenvalue.

To perform an ICA we used the FastICA Matlab package described in [3] (symmetric approach with tanh as non-linearity). $100,000$ example image patches $\tilde{y}$ of size 7×7 where extracted at random positions from images depicting natural scenes. We performed no pre-processing of the images except a shift to zero mean for each image separately. We assumed that $\tilde{y}_i\tilde{y}_j = \tilde{y}_j\tilde{y}_i$ and $\tilde{y}_i\tilde{y}_i = 0$ together with a circular rather than rectangular receptive field which reduces the number of valid pixel combinations to $7\pi(7\pi - 1)/2 \approx 700$. A dimension reduction was performed by doing an eigenvalue decomposition of Φ and keeping only the eigenvectors belonging to the 49 largest eigenvalues (this keeps 94% of the variance).

Doing ICA in the space of dot products results in finding independent sources in dot product space. To visualize each component found by ICA we plotted examples from the resulting orthogonal basis function set (see Figure 2). The figure displays in each case only the absolute largest and second largest eigenvector of the 7×7 matrices A^i, which where found as lines in the mixing matrix W.

Among others, edges with different orientation, spatial frequency and position are prominent structures found by the model (see Figure 2). A further analysis of the filters is given in the discussion.

4 Discussion

Whereas coding of edges is trivially implemented by a linear filter, our model relies solely on statistics of second order, e.g. products of pixel values. Edges are found to be implemented by a simple mechanism of 'be the same' and 'be different'. The pixels on each side of the edge are pooled together by assigning only positive weights between the pixels. Together they code for the same coherent region of space. Contrarily, the weights between the two groups are negative, thus forcing the two regions to code for different structure.

Fig. 3. Texture dependent sub-fields. Eigenvectors with negative eigenvalues concentrate onto the lower right part of the receptive field, eigenvectors with positive eigenvalues onto the upper left part.

Coding of texture properties: Figure 3 displays more eigenvectors of a selected component. Sub-structure is concentrated either on the upper or lower integration field of the filter. The filter distinguishes in its response non-textured regions (objects) from textured regions (background) or vice versa.

Detecting curvature. More difficult to interpret are filters which show a *yin-yang*-like pattern. They are stable features in our simulations. We offer an interpretation to their function in terms of forming curvature detection units. We consider the orbit that is spanned by the largest positive and negative eigenvector $\tilde{n}(\gamma) = \gamma \mathbf{n}_0 + (1-\gamma)\mathbf{n}_N$, $\gamma = [0 \dots 1]$. As shown in Figure 4, (first row), maximum (left, $\gamma = 0$) and minimum (right, $\gamma = 1$) together constitute an edge detector (center, $\gamma \approx 0.5$). Because of orthogonal eigenvectors the unit's response to each image $\tilde{n}(\gamma)$ is the sum of weighted (by γ) individual responses. Increasing or decreasing γ 'bends' the line along the edge in directions of positive and negative curvature– larger curvature results in larger (absolute) output.

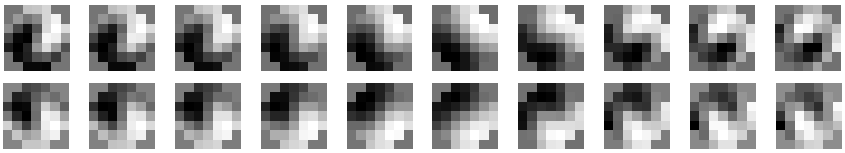

Fig. 4. Two dot-product filters that code for curvature of a line. The **top row** shows the orbit that is spanned by the largest and smallest eigenvector of the ICA-component nr. 47 shown in Figure 2. In the **bottom row** same is done for the ICA-component nr. 49.

References

[1] P. C. Bressloff, J. D. Cowan, M. Golubitsky, P. J. Thomas, and M. C. Wiener. Geometric visual hallucinations, euclidean symmetry, and the functional architecture of striate cortex. *Phil. Trans. Royal Soc. London B*, 356:299–330, 2001.

[2] W. Einhäuser, C. Kayser, P. König, and K. P. Körding. Extracting slow subspaces from natural videos leads to complex cells. In G. Dorffner, H. Bischof, and K. Hornik, editors, *Artificial Neural Networks - ICANN 2001*, volume 2130 of *Lecture notes in computer science*. Springer Berlin, 2001.

[3] A. Hyvärinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks*, 10(3):626–634, 1999.

[4] Rajesh P. N. Rao and Daniel L. Ruderman. Learning lie groups for invariant visual perception. In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Info Processing Systems*, volume 11, pages 810–816. MIT Press, 1999.

[5] L. Wiskott and T. Sejnowski. Slow feature analysis: unsupervised learning of invariances. *Neural Computation*, 14:715–770, 2002.