Connectionist Grammars for High-Level Vision

Eric Mjolsness

Research Report YALEU/DCS/RR-938
November 1992

Connectionist Grammars for High-Level Vision

Eric Mjolsness
Department of Computer Science
Yale University
New Haven, CT 06520-2158

November 1992

Abstract

We offer a Bayesian foundation and an algebraic design method for neural networks which represent and process visual information at several levels of abstraction. Using this design method we can derive networks which combine symbolic and quantitative computation, such as structural matching and analog parameter optimization. The design method begins with the formulation of a vision problem as Bayesian inference or decision on a comprehensive statistical model of the visual domain given by a probabilistic *grammar*, akin to L-systems which have been used successfully in computer graphics to represent realistic visual environments. Our "connectionist grammars" allow each grammar rule to compute parameter values in a neurally implementable way; the result is *not* the neural network but rather a statistical model of the problem domain.

Subsequent steps in the design method allow us to derive a variety of neural networks for Bayesian inference or decision in the problem domain, with opportunities for using new mathematical methods such as deterministic (Mean Field Theory) annealing on Potts glasses, algebraic transformations that reduce computational cost, correlation matching in scale space, and computational attention mechanisms. One goal of such methods is to achieve acceptable scaling in the performance of the derived neural networks. We summarize several examples of the design method, including the Bayesian derivation of the "Frameville" neural networks for high-level vision, which incorporate both parameter optimization and a variable-binding or graph-matching mechanism to solve correspondance problems.

1 Introduction

It may be possible to close the gulf between symbolic processing, in which expressiveness, abstraction, and representational power are central, and connectionist or neural computation, which is driven by mathematical methods from dynamics, statistics and related fields. This paper describes one approach to doing so. The idea is to blend connectionism and expressive formal languages, not by means of a hybrid computational model (such as a neural network inside a rule-based expert system), but rather in a statistical model of the problem domain (which in this paper will be visual). The resulting model will take the form of a stochastic "connectionist grammar" in which each generative rule has the expressive power of a simple, neurally implementable Boltzmann probability distribution. From this model, and an algebraic design method, we will be able to derive neural networks which perform computations in the problem domain. The resulting networks have some of the representational power of symbolic programs along with some of the mathematical advantages of connectionist network computation.

To construct a connectionist grammar which can model the image-formation process in a complex visual domain, we must generally include heterogeneous sources of image noise, arising from qualitatively different phenomena such as image-level sensor noise, intrinsic variability of single objects, and scene-level statistics. Each such phenomenon will be modeled as a process with its own grammar rule, which influences the stochastic generation of images or other visual data such as line drawings. The structure of the grammar will model relationships between different noise processes. The grammar is a successful model of the visual domain to the extent that it generates pictures or images with an approximately correct probability distribution. It is essential to choose each rule's probability distribution to be accurate enough for visual recognition and yet tractable enough for neural theory and implementation.

Using such grammars, we will derive neural networks as follows: (1) Obtain the grammar, by detailed modelling or by automated learning from examples. (2) Compute the joint Boltzmann probability distribution on images (or pictures) and their grammatical explanations. (3) Optionally, change variables to get an equivalent Boltzmann distribution in a more tractable form. (4) Express desired outputs as averages under this distribution which can be calculated by optimizating an objective function E. This step usually employs Mean Field Theory approximations. (5) Optionally, apply algebraic *transformations* [1] to the objective function which preserve its fixed points (exactly or approximately) but reduce cost, improve network performance or achieve network implementability in some technology. (6) Introduce optimizing neural net dynamics for E. The entire method is sketched in Figure 1, and is mostly to be conducted algebraically, possibly with computer assistance.

The resulting neural networks have a wide variety of forms, but the easiest ones to get are constrained optimization networks. There is currently no general procedure for ensuring that a constrained optimization network will scale well. The scaling properties and practicality of Mean Field Theory networks have been greatly improved by methods that incorporate some constraints exactly [2, 3, 4] in matching problems similar to those we encounter. Also, some of the networks derived from connectionist grammars have a natural hierarchical structure that suggests dividing a large constrained optimization problem into many small sub-problems to get a scalable algorithm. Still, several different algebraic formulations of each objective may have to be tried to achieve acceptable performance.

This paper is a summary of [5], omitting proofs and several applications which may be found in [6]. A somewhat more general view of grammars whose rules posess connectionist models (similar to the Boltzmann distributions attached to rules in this paper) is presented in [7], where such grammars are proposed for modelling the development of biological organisms. Fortunately the large amount of research in computer graphics is a source of accurate generative models for images, some of which are mathematically simple enough to be put in the form of a connectionist grammar or are already close to that form (e.g. [8, 9]). Related directions for generalization of the parallel grammar occur in the extensive literature on L-systems [10] and graph grammars [11].

In the remainder of this section we introduce a very simple connectionist grammar which poses a recognition problem. In Section 2 we demonstrate the algebraic design method for this grammar, and show how the grammar may be generalized to include new noise sources. In Section 3 we consider a visual grammar which achieves an interesting level of generality by placing several objects in a scene, each of which has a hierarchical structure. We show how to derive a neural network architecture [12] which mixes symbolic computing (including structural matching or variable binding) with analog parameter estimation for object recognition.

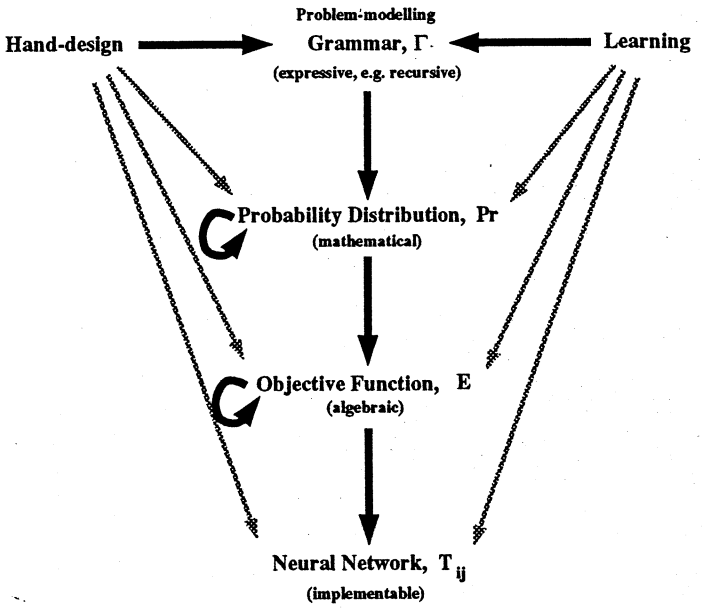


Figure 1: A neural network design methodology. Solid arrows constitute the recommended procedure. The arrow from Pr to E may be realized by approximations from statistical physics, such as Mean Field Theory. Circular arrows represent valid transformations, such as fixed-point preserving transformations of objective functions.

1.1 Example: A Random-Dot Grammar

The first example grammar is a generative model of pictures consisting of a number of dots (e.g. a sum of delta functions) whose relative locations are determined by one out of M stored models. But the dots are subject to unknown independent jitter and an unknown global translation, and the identities of the dots (their numerical labels) are hidden from the perceiver by a random permutation operation. For example each model might represent an imaginary asterism of equally bright stars whose locations have been corrupted by instrument noise. One useful task would be to recognize which model generated the image data.

The random-dot grammar is shown in equation (1).

model and location	Γ^0 :	root	$\rightarrow$	instance of model α at $\mathbf{x}$
		$E_0(\mathbf{x})$	$=$	$\frac{1}{2\sigma_t^2}\lvert\mathbf{x}\rvert^2$
jittered dot locations	Γ^1 :	instance$(\alpha, \mathbf{x})$	$\rightarrow$	$\{\text{dot}(\alpha, m, \mathbf{x}_m)\}$
		$E_1(\{\mathbf{x}_m\})$	$=$	$\frac{1}{2\sigma_{jt}^2}\sum_m(\mathbf{x}_m - \mathbf{x} - \mathbf{u}_m^\alpha)^2$, where $< \mathbf{u}_m^\alpha >_m = 0$
scramble all dots	Γ^2 :	$\{\text{dot}(m, \mathbf{x}_m)\}$	$\rightarrow$	$\{\text{imagedot}(\mathbf{x}_i = \sum_m P_{m,i}\mathbf{x}_m)\}$
		$E_2(\{\mathbf{x}_i\})$	$=$	$-\log[\Pr(P)\prod_i \delta(\mathbf{x}_i - \sum_m P_{m,i}\mathbf{x}_m)]$ where P is a permutation

(1)

In this notation, each rule Γ^r has an "energy function" E_r which determines the relative probabilities of different parameter values according to the Bolzmann distribution associated with E_r. So the probability

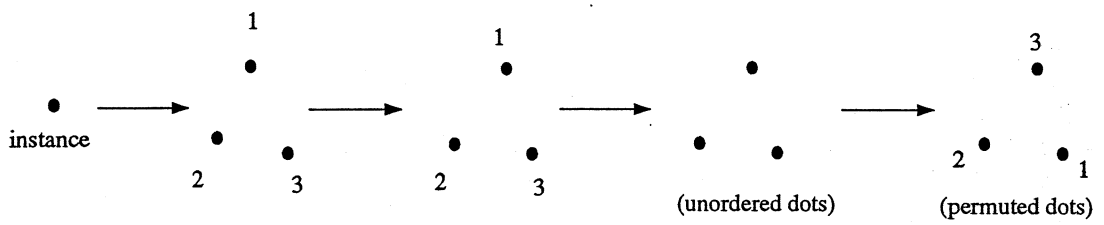


Figure 2: Operation of random dot grammar. The first arrow illustrates dot placement; the next shows dot jitter; the next arrow shows the pure, un-numbered feature locations; and the final arrow is the uninformed renumbering scheme of the perceiver. The last two steps are combined in the algebraic expression of the grammar.

distribution associated with a particular rule Γ^r is

$$\Pr(\text{new terms}, \{\text{new parameters}\}|\text{old terms}, \{\text{old parameters}\}) = e^{-\beta E_r}/Z_r \tag{2}$$

where $\beta \to 1$. Such conditional probabilities can be repeatedly combined in the usual way:

$$\Pr(\xi, x) = \Pr(x|\xi)\Pr(\xi) \tag{3}$$

to yield a final joint probability distribution for the entire grammar. However we are usually interested in computing some average in this distribution, i.e. some moment of this function.

The operation of this grammar is illustrated in Figure 2. The generative process starts with nothing (the "root" of a parse tree) and generates one instance of a model chosen randomly from a list of known models. Let the chosen model number be α. Rule Γ^0 also places the instance on the image plane with a Gaussian distribution of locations $\mathbf{x}$. (Hence the objective function E_0 is a quadratic in $\mathbf{x}$.) Given such an instance, the only applicable rule is Γ^1 which replaces it with a set of dots whose locations $\mathbf{x}_m$ are Gaussian-distributed displacements of ideal locations given by $\mathbf{x} + \mathbf{u}_m^\alpha$. The final rule is special: its input is the set of all dots generated by the grammar, and it replaces them with a permuted set of image dots at the same set of locations. In other words it relabels the dots from index m to index i by means of a permutation P_{mi}. The permutation probability distribution $\Pr(P)$ will be taken to be the uniform distribution on permutations. In [5] we show that a plausible general form of $\Pr(P)$ is equivalent to the uniform distribution we assume here.

2 The Algebraic Design Method

We have just performed the first step of the design method of Figure 1: formulating a connectionist grammar which generates "pictures" for a simple problem domain. Now we continue the procedure by deriving the associated probability distribution on such pictures and their labels (Section 2.1), and neural networks capable of performing recognition given the pictures but not the labels (Section 2.2). In Sections 2.3 and 2.4 we illustrate the derivation of alternative networks for the same problem. In Section 2.5 we exhibit network objective functions for a more complex grammar.

2.1 Final Probability Distribution

Calculating the joint probability distribution is especially easy for the grammar (1) because the grammar rules are not recursive. The grammar just consists of a sequence of three stages corresponding to rules $\Gamma^0 - \Gamma^2$. After rule Γ^0,

$$\Pr^0(\alpha, \mathbf{x}) = \frac{1}{A}\left(\frac{1}{\sqrt{2\pi}\sigma_r}\right)^2 e^{-\frac{1}{2\sigma_r^2}|\mathbf{x}|^2} \tag{4}$$

where A is the number of models to choose from. The probability after rule Γ^1 is

$$\begin{aligned}\Pr^1(\alpha, \mathbf{x}, \{\mathbf{x}_m\}) &= \Pr^1(\{\mathbf{x}_m\}|\alpha, \mathbf{x})\Pr^0(\alpha, \mathbf{x}) \\ &= \frac{1}{A}\left(\frac{1}{\sqrt{2\pi}\sigma_r}\right)^2 \left(\frac{1}{\sqrt{2\pi}\sigma_{jt}}\right)^{2N} e^{-\left(\frac{1}{2\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}\sum_m |\mathbf{x}_m - \mathbf{x} - \mathbf{u}_m^\alpha|^2\right)}.\end{aligned} \tag{5}$$

To finish the calculation we must consider $\Pr(P)$. This is the probability of a given renumbering from object-generation indices m to image indices i. This part of the grammar models the fact that the object-generation indices m are generally inaccessable to the perceiver, though if they were known the perception problem would be nearly solved. One model for P, justified in [5], is to feign ignorance of the permutation and use the maximum entropy distribution on P, namely a uniform distribution.

Then,

$$\Pr^{\text{final}}(\alpha, \mathbf{x}, \{\mathbf{x}_i\}) = c_1 \sum_{\left\{\begin{smallmatrix}\text{P| P is a} \\ \text{permutation}\end{smallmatrix}\right\}} \int d\{\mathbf{x}_m\} \prod_i \delta(\mathbf{x}_i - \sum_m P_{m,i}\mathbf{x}_m) e^{-\left(\frac{1}{2\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}\sum_m |\mathbf{x}_m - \mathbf{x} - \mathbf{u}_m^\alpha|^2\right)} \tag{6}$$

whence, using $\sum_i P_{m,k} = 1$,

$$\Pr^{\text{final}}(\alpha, \mathbf{x}, \{\mathbf{x}_i\}) = c_1 \sum_{\left\{\begin{smallmatrix}\text{P| P is a} \\ \text{permutation}\end{smallmatrix}\right\}} e^{-\beta \sum_{mi} P_{m,i}\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2\right)}. \tag{7}$$

The inverse temperature β just introduced must of course be set to one. But for Bayesian inference algorithms this may be done by gradually increasing β from zero, a procedure called "annealing", which often has the effect of avoiding spurious local minima during a computation.

Equation (7) is representative of most of the grammatical probability distributions we will derive in one important way: it is a Boltzmann distribution whose objective function is a generalized "assignment" objective function. The "assignment problem" [13] is to minimize $E = \sum_{\alpha a} P_{\alpha a} W_{\alpha a}$ over permutations P, for constant weights $W \geq 0$. A neural net approach to this problem is analysed in [14]. In equation (7) the assignment problem objective is generalized because the weights W are now functions of real-valued parameters, as will generally be the case for grammatical probability distributions:

$$E_{\text{final}}(\alpha, P, \mathbf{x}) = \sum_{mi} P_{m,i}\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2\right) \tag{8}$$

The sum over permutations may be approximated by an optimization over near-permutations, as we will see, and the fact that P appears only linearly in E_{final} makes such optimization problems easier.

A simple recognition problem might involve looking at data $\{\mathbf{x}_i\}$ and inferring the most likely model (α) and its position ($\mathbf{x}$). We must find

$$\begin{aligned}
\text{argmax}_{\alpha,\mathbf{x}}\Pr(\alpha,\mathbf{x}|\{\mathbf{x}_i\}) &= \text{argmax}_{\alpha,\mathbf{x}}\frac{\Pr(\alpha,\mathbf{x},\{\mathbf{x}_i\})}{\Pr(\{\mathbf{x}_i\})} \quad \text{(Bayesian inference)} \\
&= \text{argmax}_{\alpha,\mathbf{x}}\Pr(\alpha,\mathbf{x},\{\mathbf{x}_i\}) \\
&= \text{argmax}_{\alpha,\mathbf{x}} \sum_{\left\{ \begin{smallmatrix} \text{P}|\ \text{P is a} \\ \text{permutation} \end{smallmatrix} \right\}} \Pr(\alpha,\mathbf{x},\{P_{m,i}\},\{\mathbf{x}_i\}). \qquad (9)
\end{aligned}$$

Note that the combination of equations (9) and (3) perform Bayesian inference: they determine Pr(model params|data) in terms of forward conditional probabilities including Pr(data|model params).

2.2 Neural Network with Match Variables

We review how configuration-space sums over P (along with other variables) may be approximated by quadratic match neural nets. For example we may compute $\Pr^f(\alpha,\mathbf{x}|\{\mathbf{x}_i\})$ as follows:

$$\begin{aligned}
\Pr^f(\alpha,\mathbf{x}|\{\mathbf{x}_i\}) &= c_1 \sum_{\left\{ \begin{smallmatrix} \text{P}|\sum_\text{i}\text{P}_{\text{m,i}}=1 \\ \wedge\sum_\text{m}\text{P}_{\text{m,i}}=1 \end{smallmatrix} \right\}} \exp -\sum_{mi} P_{m,i}\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2+\frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i-\mathbf{x}-\mathbf{u}_m^\alpha|^2\right) \\
&= c_1 \lim_{A\to\infty}\int_{-\infty}^{+\infty} d\{V_{mi}\}\int_{-i\infty}^{+i\infty} d\{U_{mi}\}\exp -\beta F(\alpha,\mathbf{x},\{U\},\{V\})
\end{aligned} \qquad (10)$$

where

$$\begin{aligned}
F(\alpha,\mathbf{x},\{U\},\{V\}) \equiv & \sum_{mi} V_{m,i}\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2+\frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i-\mathbf{x}-\mathbf{u}_m^\alpha|^2\right)+(A/2)\sum_i(\sum_m V_{m,i}-1)^2 \\
& +(1/\beta)\sum_{m,i}U_{m,i}V_{m,i}-(1/\beta)\sum_m\log\left(\sum_i\exp U_{m,i}\right).
\end{aligned} \qquad (11)$$

Up to this point the expression is exact; no approximations have been made. Now the Mean Field Theory approximation replaces the U and V integrals with the problem of finding the saddle points $(\{U^*\},\{V^*\})$ of the objective function F:

$$\begin{aligned}
\text{argmax}_{\alpha,\mathbf{x}}\Pr^f(\alpha,\mathbf{x}|\{\mathbf{x}_i\}) &\approx \text{argmax}_{\alpha,\mathbf{x}} c_1 \lim_{A\to\infty}\exp -\beta F(\alpha,\mathbf{x},\{U^*\},\{V^*\}) \\
&= \left(\text{argmax}_\alpha \lim_{A\to\infty}\exp -\beta F(\alpha,\mathbf{x}^*,\{U^*\},\{V^*\}),\ \mathbf{x}^*\right)
\end{aligned} \qquad (12)$$

where the saddle points satisfy the neural net fixed point equations

$$\begin{aligned}
(\partial E_{eff}/\partial U=0:)\quad V_{m,i} &= \exp U_{m,i}/\sum_j \exp U_{m,j} \\
(\partial E_{eff}/\partial V=0:)\quad U_{m,i} &= -\beta\left[\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2+\frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i-\mathbf{x}-\mathbf{u}_m^\alpha|^2+A(\sum_n V_{n,i}-1)\right] \qquad (13) \\
(\partial E_{eff}/\partial \mathbf{x}=0:)\quad \mathbf{x} &= \frac{1}{N(1+(\sigma_{jt}/\sigma_r)^2)}\sum_{mi} V_{mi}(\mathbf{x}_i-\mathbf{u}_m^\alpha).
\end{aligned}$$

Convergent descent dynamics for such networks may be found in [15, 3] and many others. The maximization with respect to α in equation (9) can be handled by making one copy of this neural net for each model and adding a winner-take-all circuit.

The match variables V_{mi} bear a strong resemblance to the outer product representation commonly used in variable-binding networks but are not yet in the same league of expressive power since only actual data items $\mathbf{x}_i$ are bound to anything. Instead the V variables simply express the answer to a visual correspondance problem.

2.3 Algebraic Transformations

The direct translation of equations (11) or (13) into a neural network results in a number of connections proportional to N^3, where N is the number of data vectors and also the number of model vectors. We can reduce this cost to $\mathcal{O}(N^2)$ by using algebraic transformations of F that preserve its fixed points, as described in [1]. The new objective is $\hat{F}(\alpha, \mathbf{x}, \{U\}, \{V\})$, where

$$\begin{aligned}\hat{F} \equiv & \sum_{mi} V_{m,i}\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2\right) + A\sum_i(\sum_m V_{m,i} - 1)\tau_i - (A/2)\sum_i \tau_i^2 \\ & + (1/\beta)\sum_{m,i} U_{m,i}V_{m,i} - (1/\beta)\sum_{mi}\sigma_m \exp U_{m,i} + (1/\beta)\sum_m \log \sigma_m,\end{aligned} \tag{14}$$

and $\hat{F}$ explicitly has only $\mathcal{O}(N^2)$ interactions (neural connections) between different variables. Some of the connections are, however, of a new type $\sigma \exp U$ which is discussed in [1]. Such transformations as from equation (11) to (14) are represented in Figure 1 by a circular arrow.

Further algebraic transformations of the objective function may also be beneficial. For large problems, N^2 connections may be too many to implement (in software simulation or in neural hardware) at one time. We would like a way to "pay attention" to just part of the optimization problem at any moment. For example one could partition the original image, and hence the data set $\{\mathbf{x}_i\}$, into roughly equal-sized blocks of nearby data, and only relax the neurons associated with one or a few of these blocks at any given time. That would induce a partition of all the neurons into blocks which pertain to local regions of the image. By analogy with "virtual memory" and "virtual processors" in computer science, we may say that such a domain decomposition allows a few "physical neurons" to simulate many "virtual neurons". The active set of neuron blocks can be controlled dynamically by another objective function, as in [16, 17]. In this way we find a computational attention mechanism which serves the purpose of trading off temporal and spatial costs of a computation.

2.4 Correlation Matching in Scale Space

Short of approximating a P configuration sum via Mean Field Theory and neural nets (Section 2.2 above), there is a simpler, cheaper, less accurate approximation that we have used on matching problems similar to the model recognition problem (find α and $\mathbf{x}$) for the dot-matching grammar. From equations (7) and (9),

$$\Pr^f(\alpha, \mathbf{x}|\{\mathbf{x}_i\}) = C \sum_{\left\{\begin{smallmatrix}\text{P}|\ \text{P is a} \\ \text{permutation}\end{smallmatrix}\right\}} \exp -\sum_{mi} P_{m,i}\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2\right), \text{ i.e.} \tag{15}$$

$$\begin{aligned}
\Pr^f(\alpha,\mathbf{x}|\{\mathbf{x}_i\}) &\approx C \sum_{\left\{\begin{smallmatrix} P|P_{m,i}\in\{0,1,\ldots N\} \\ \wedge\sum_{m,i}P_{m,i}=N \end{smallmatrix}\right\}} \exp -\sum_{mi} P_{m,i}\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2+\frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i-\mathbf{x}-\mathbf{u}_m^\alpha|^2\right) \\
&\approx \frac{C}{N!}\sum_{\{P|\sum P=N\}}\left(\begin{array}{c} N \\ P_{11}\ldots P_{NN}\end{array}\right)\prod_{mi}\left[\exp-\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2+\frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i-\mathbf{x}-\mathbf{u}_m^\alpha|^2\right)\right]^{P_{m,i}} \\
&\qquad \text{(since almost all the multinomials are } = N!) \\
&= C'\left[\textstyle\sum_{m,i}\exp-\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2+\frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i-\mathbf{x}-\mathbf{u}_m^\alpha|^2\right)\right]^N .
\end{aligned} \tag{16}$$

The key step is the approximation of the sum over permutation matrices with a sum over a superset, namely all $N \times N$ nonnegative-integer-valued matrices whose entries sum to N. Among such matrices, the vast majority have low occupancy for most rows and columns. This is an entropy argument in favor of the approximation. There is also an energy argument: multiple assignments are allowed but discouraged by the effective energy term $(1/2\sigma_{jt}^2)\sum_{m,i}|\mathbf{x}_i-\mathbf{x}-\mathbf{u}_m^\alpha|^2$ unless two values of $\mathbf{x}_i$ or two values of $\mathbf{u}_m$ happen to be within σ_{jt} of each other. Finally notice that the insertion of the multinomial factor improves this approximation rather than further degrading it, since configurations with $P_{mi}>1$, not present in the original sum over permutation matrices, are weighted less strongly than those in which every P element is 0 or 1.

Under this approximation,

$$\text{argmax}_{\alpha,\mathbf{x}}\Pr(\alpha,\mathbf{x}|\{\mathbf{x}_i\}) \approx \text{argmax}_{\alpha,\mathbf{x}}\sum_{m,i}\exp-\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2+\frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i-\mathbf{x}-\mathbf{u}_m^\alpha|^2\right). \tag{17}$$

This objective function has a simple interpretation when $\sigma_r \to \infty$: it minimizes the Euclidean distance between two Gaussian-blurred images containing the $\mathbf{x}_i$ dots and a shifted version of the $\mathbf{u}_m$ dots respectively:

$$\begin{aligned}
&\text{argmin}_{\alpha,\mathbf{x}}\textstyle\int d\mathbf{z}\,|G*I_1(\mathbf{z})-G*I_2(\mathbf{z}-\mathbf{x})|^2 \\
=\;&\text{argmin}_{\alpha,\mathbf{x}}\textstyle\int d\mathbf{z}\left|G_{\sigma/\sqrt{2}}*\sum_i\delta(\mathbf{z}-\mathbf{x}_i)-G_{\sigma/\sqrt{2}}*\sum_m\delta(\mathbf{z}-\mathbf{x}-\mathbf{u}_m^\alpha)\right|^2 \\
=\;&\text{argmax}_{\alpha,\mathbf{x}}\textstyle\sum_{mi}\int d\mathbf{z}\exp-\frac{1}{\sigma^2}\left[|\mathbf{z}-\mathbf{x}_i|^2+|\mathbf{z}-\mathbf{x}-\mathbf{u}_m^\alpha|^2\right] \\
=\;&\text{argmax}_{\alpha,\mathbf{x}}\textstyle\sum_{mi}\exp-\frac{1}{2\sigma^2}|\mathbf{x}_i-\mathbf{x}-\mathbf{u}_m^\alpha|^2
\end{aligned} \tag{18}$$

Furthermore, note that multiplying the objective in (17) by a temperature factor $\beta = 1/T$ simply rescales σ_{jt}. From this fact we conclude that deterministic annealing from $T=\infty$ down to $T=1$, which is a good strategy for finding global maxima in equation (17), corresponds to a coarse-to-fine correlation matching algorithm: the global shift $\mathbf{x}$ is computed by repeated local optimization while gradually decreasing the Gaussian blurr parameter σ down to σ_{jt}. The output of a coarse-scale optimization is taken as the input to the next finer-scale optimization, as in deterministic annealing and other continuation methods. The resulting coarse-to-fine correlation matching algorithm is similar to the scale-space matching procedure of [18].

The approximation (16) has the effect of eliminating the discrete P_{mi} variables, rather than replacing them with continuous versions V_{mi}. The same can be said for the "elastic net" method [19], which can also be derived from an assignment problem [2, 20]. Compared to the elastic net, the present objective function is simpler but is expected to be less accurate.

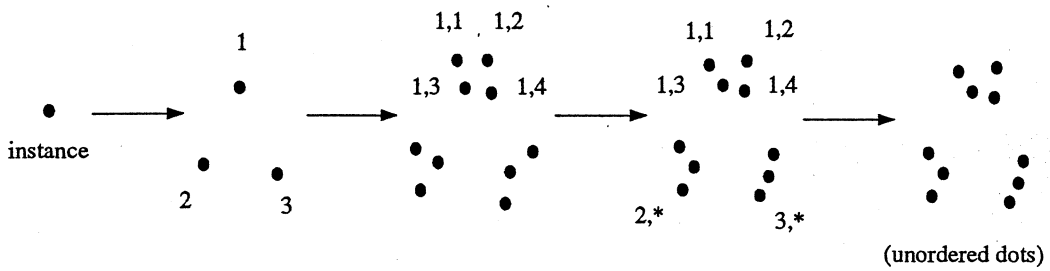


Figure 3: Operation of a two-level random dot grammar. The first arrow illustrates cluster placement and cluster jitter; the next shows dot placement; the next shows dot jitter; and the final arrow is the scrambling or renumbering operation. Not shown: rotation, dot deletion.

Experiments with a neural net for performing the maximization of (17) with respect to $\mathbf{x}$ have been reported in [1]. A continuation from large σ down to σ_{jt} was used, and it greatly reduced the network's susceptibility to finding incorrect local minima of the objective fuction.

2.5 A More Complex Grammar

To illustrate the generality of the grammatical method for posing vision problems in a form from which neural networks can be derived as in Figure 1, we exhibit a more complex grammar. Unlike the simple random-dot grammar considered in Section 1.1, here we add rotation and dot deletion as new sources of noise and introduce a two-level hierarchy, in which models are sets of clusters of dots.

Consider an object with a hierarchical decomposition into parts, with internal degrees of freedom describing the relative positions of the parts. For random dot features the resulting pictures will generally be clusters of dots with unpredictable jitter of both the dot and the cluster positions. We can also easily add two-dimensional rotations to this grammar, and a dot deletion rule which changes the constraints on P. A concise model of such an object is given by the grammar of equation (20) below.

The corresponding probability distribution is $\Pr^3(\alpha, \mathbf{x}, \{\mathbf{x}_c\}, \{\mathbf{x}_i\})$:

$$\Pr^3 = c_2 \sum_{\left\{ \begin{array}{l} \mathrm{P} \mid \sum_i \mathrm{P}_{m,i} \le 1 \\ \text{and} \sum_m \mathrm{P}_{m,i} = 1 \end{array} \right\}} e^{-\sum_{cmi} P_{cm,i} \left(\frac{1}{2N\sigma_r^2} |\mathbf{x}|^2 + \frac{C}{2N\sigma_{cd}^2} |\mathbf{x}_c - \mathbf{x} - \mathrm{R}(\theta)\cdot\mathbf{u}_c^\alpha|^2 + \frac{1}{2\sigma_{jt}^2} |\mathbf{x}_i - \mathbf{x}_c - \mathrm{R}(\theta)\cdot\mathbf{u}_{cm}^\alpha|^2 + \mu \right)} \tag{19}$$

where C is the number of clusters and N/C is the number of dots in each cluster.

model locations	Γ^0 :	root $\rightarrow$	instance of model α at $\mathbf{x}$ and rotated by θ
		$E_0(\mathbf{x})$ =	$\frac{1}{2\sigma_r^2}\|\mathbf{x}\|^2$
rotated, jittered cluster locations	Γ^1 :	instance$(\alpha, \mathbf{x})$ $\rightarrow$	$\{\text{cluster}(\alpha, c, \mathbf{x}_c)\}$
		$E_1(\{\mathbf{x}_c\})$ =	$\frac{1}{2\sigma_{cd}^2}\sum_c \|\mathbf{x}_c - \mathbf{x} - \mathsf{R}(\theta)\cdot\mathbf{u}_c^\alpha\|^2$, where $< \mathbf{u}_c^\alpha >_c = 0$
jittered dot locations	Γ^2 :	cluster$(\alpha, c, \mathbf{x}_c)$ $\rightarrow$	$\{\text{predot}(c, m, \mathbf{x}_{cm})\}$
		$E_2(\{\mathbf{x}_{cm}\})$ =	$\frac{1}{2\sigma_{jt}^2}\sum_m \|\mathbf{x}_{cm} - \mathbf{x}_c - \mathsf{R}(\theta)\cdot\mathbf{u}_{cm}^\alpha\|^2$, where $< \mathbf{u}_{cm}^\alpha >_m = 0$
dot deletion	Γ^3 :	predot$(\alpha, c, m, \mathbf{x}_m)$ $\rightarrow$	$\begin{cases} \text{dot}(c, m, \mathbf{x}_{cm}) & \text{if } \omega_{cm} = 1; \\ \text{nothing} & \text{if } \omega_{cm} = 0. \end{cases}$
		$E_3(\omega_{cm})$ =	$\mu\omega_{cm}$
scramble all dots	Γ^4 :	$\{\text{dot}(c, m, \mathbf{x}_{cm})\}$ $\rightarrow$	$\{\text{imagedot}(\mathbf{x}_i = \sum_{cm} P_{cm,i}\mathbf{x}_{cm})\}$
		$E_4(\{\mathbf{x}_i\})$ =	$-\log\prod_i \delta(\mathbf{x}_i - \sum_{cm} P_{cm,i}\mathbf{x}_{cm})$ where $\sum_i P_{m,i} = \omega_{cm} \ \wedge \ \sum_m P_{m,i} = 1$

(20)

As in Section 2.2 one may approximate the maximization of the integrated probability $\Pr^f(\alpha, \mathbf{x}, \{\mathbf{x}_c\}|\{\mathbf{x}_i\})$ with respect to α, $\mathbf{x}$ and $\mathbf{x}_c$ via a neural net objective function $F(\alpha, \mathbf{x}, \{\mathbf{x}_c\}, \{U\}, \{V\})$ given by

$$F \equiv \sum_{cm,i} V_{cm,i}\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{C}{2N\sigma_{cd}^2}|\mathbf{x}_c - \mathbf{x} - \mathsf{R}(\theta)\cdot\mathbf{u}_c^\alpha|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x}_c - \mathsf{R}(\theta)\cdot\mathbf{u}_{cm}^\alpha|^2 + \mu\right)$$
$$+(A/2)\sum_i(\sum_{cm} V_{cm,i} - 1)^2 + (1/\beta)\sum_{cm,i} U_{cm,i}V_{cm,i} - (1/\beta)\sum_{cm}\log\left(1 + \sum_i \exp U_{cm,i}\right). \tag{21}$$

Alternatively, as in Section 2.4 one could use an objective function without match variables:

$$E_{eff}(\alpha, \mathbf{x}, \{\mathbf{x}_c\}) = \sum_{cmi}\exp -\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{C}{2N\sigma_{cd}^2}|\mathbf{x}_c - \mathbf{x} - \mathsf{R}(\theta)\cdot\mathbf{u}_c^\alpha|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x}_c - \mathsf{R}(\theta)\cdot\mathbf{u}_{cm}^\alpha|^2\right). \tag{22}$$

Intermediate designs result from *changing variables* by separately considering the cluster and the member to which a data item is assigned: $P_{cm,i} = P_{ci}^1 P_{mi}^2$. Then P^1 could be turned into analog match variables and P^2 could be approximately integrated out, assuming that σ_{jt} is small with respect to the variance of model dot locations $\mathbf{u}_{cm}^\alpha$ within a cluster, but not assuming that σ_{cd} is small with respect to the variance of cluster locations $\mathbf{u}_c^\alpha$.

3 Frameville from a Grammar

We reach an interesting level of generality with a new grammar which places several objects in a scene, each of which has a hierarchical structure of feature locations (dots). This degree of complexity is sufficient to introduce questions of knowledge representation in high-level vision, such as representing an unpredictable number of instances of a model in a scene, as well as requiring segmentation and grouping

of data into objects as part of the recognition process. We show how the grammatical approach can yield neural networks nearly identical to the "Frameville" neural networks we have previously studied as a means of mixing simple Artificial Intelligence frame systems (or semantic networks) with optimization-based neural networks. What is more, the transformation leading to Frameville is very natural. It simply pushes the usual permutation matrix P as far back into the grammar as possible, from lower levels of abstraction and scale to higher ones.

3.1 The Frameville Neural Architecture

Most neural net architectures appear inadequate for high-level vision problems because they lack the ability to express, much less use or learn, sufficiently abstract knowledge: knowledge of parameterized classes of shape, or of geometric relationships between objects, or of similarity in topology, shape or function. One might try to use a more abstract computational unit to model small concepts. Such a "frame" could only result from the combined action of many artificial neurons, and in this way would be a collective, large-scale phenomenon in a neural network. A frame would more readily map to the intuitive idea of the "concept" of an object if it: (a) could be instantiated many times in one scene or computation, with different parameters such as position and internal degrees of freedom; (b) could collect feedback from such dynamically allocated instances for use in learning; (c) could express the expected or allowed range of variation from a prototype model; and could enter into various relationships with other frames, including (d) part-whole hierarchies, (e) geometric relationships, and (f) generalization and specialization relationships.

The design goal of the "Frameville" type of neural network architecture [12, 21, 22] is to provide such capabilities in much the way they can be provided within a frame system as used in Artificial Intelligence programming [23], while exhibiting a neural substrate or implementation which provides the kind of inexact matching abilities that objective-function based neural nets are capable of. The Frameville objective function was based on inexact graph-matching applied to a part-whole relationship denoted $INA_{\alpha\beta} \in \{0, 1\}$:

$$E = \sum_{\alpha\beta} \sum_{ij} INA_{\alpha\beta} ina_{ij} M_{\alpha i} M_{\beta j} H^{\alpha\beta}(\mathbf{F}_i, \mathbf{F}_j) \tag{23}$$

subject to constraints including conventional syntactic constraints on M and ina separately, along with new mixed constraints:

$$\begin{array}{lll} \sum_\alpha INA_{\alpha\beta} M_{\alpha i} & = & \sum_j ina_{ij} M_{\beta j} \quad (a) \\ \sum_\beta INA_{\alpha\beta} M_{\beta j} & = & \sum_i ina_{ij} M_{\alpha i} \quad (b). \end{array} \tag{24}$$

(See Figure 4.) Here α or β index the "frame", which could also be called the "object model", or "prototype object", and i or j index an instance tied to α through $M_{\alpha i} \in [0, 1]$. $INA_{\alpha\beta}$ is assumed to be a tree in this paper (so $\sum_\alpha INA_{\alpha\beta} \leq 1$) but may be a directed graph in general Frameville. $\mathbf{F}_i$ are the parameters of the instance, and $H^{\alpha\beta}$ is a distance or parameter-fit function specific to the part-whole relationship $INA_{\alpha\beta}$.

A typical use of $\mathbf{F}_i$, $\mathbf{F}_j$ and $H^{\alpha\beta}$ would be for $\mathbf{F}$ to hold environment-centered coordinates of the object i and of its part j, along with deduced object-centered coordinates such as translations and orientation angles of each part, and for $\mathbf{H}$ to perform coordinate transformations to deduce such coordinates and to check consistency between the deduced and expected object-centered coordinates of an object's parts.

In this and a number of other important respects, the Frameville networks resemble the "TRAFFIC" system of [24]. Other networks are related to Frameville by virtue of the use of graph-matching or arrays

of match neurons for visual object recognition [25, 26, 27, 28, 29] or using related objective functions for high-level knowledge representation [30, 31, 32, 33].

The basic Frameville architecture was extended in [21] in order to deal with the extra complexities that arise when we apply Frameville to first-order logic instead of high level vision. A very similar representation was used for first-order logic neural networks by Pinkas [34].

From such constrained optimization formulations, it is straightforward to derive neural networks as suggested in the algebraic design procedure of Figure 1. Constraints may be implemented using some combination of penalty terms, Lagrange multiplier neurons [35], or derived Mean Field Theory terms. We now show how to use the first part of the algebraic design method of Figure 1, by deriving Frameville constrained optimization problems from a connectionist grammar.

3.2 The Grammar

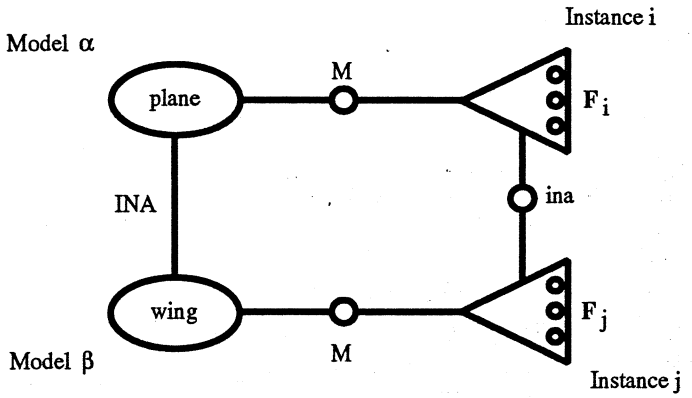
It can now be shown that the Frameville objective function and syntax constraints, as outlined above, can be derived from a random-dot grammar with multiple instances of two-level objects. We use multiple index notation $\beta \leftrightarrow (\alpha, s_2)$ (i.e. model β may occupy the s_2'th "slot" of model α, if $INA_{\alpha\beta} = 1 = INA_{\alpha,s_2}$) and $\gamma \leftrightarrow (\alpha, s_2, s_3)$ (i.e. model γ may occupy the s_3'th slot of model (αs_2), if $INA_{\beta\gamma} = 1 = INA_{\alpha s_2,s_3}$). The multiple-instance grammar is shown in equation (25) below.

N unknown objects	Γ^0 :	root	$\rightarrow$	$\{\text{object}(a) \mid a \in \{1,\ldots N\}\}$
		$E_0(\mathbf{x})$	$=$	0
assign models and locations to objects	Γ^1 :	$\text{object}(a)$	$\rightarrow$	$\text{instance}(a, \alpha, \mathbf{x}_a)$
		$E_1(\alpha, \mathbf{x}_a)$	$=$	$\frac{1}{2\sigma_r^2}\lvert\mathbf{x}_a\rvert^2$
jittered cluster locations	Γ^2 :	$\text{instance}(a, \alpha, \mathbf{x}_a)$	$\rightarrow$	$\{\text{cluster}(a, \alpha, s_2, \mathbf{x}_{as_2}) \mid INA_{\alpha,s_2} = 1\}$
		$E_2(\{\mathbf{x}_{as_2}\})$	$=$	$\frac{1}{2\sigma_{cd}^2}\sum_{s_2} INA_{\alpha,s_2}\lvert\mathbf{x}_{as_2} - \mathbf{x}_a - \mathbf{u}_{s_2}^{\alpha}\rvert^2$
			$=$	$\sum_{s_2} INA_{\alpha,s_2} H^{(\alpha s_2)}(\mathbf{x}_a, \mathbf{x}_{as_2})$
jittered dot locations	Γ^3 :	$\text{cluster}(a, \alpha, s_2, \mathbf{x}_{as_2})$	$\rightarrow$	$\{\text{predot}(a, s_2, s_3, \mathbf{x}_{as_2s_3}) \mid INA_{\alpha s_2,s_3} = 1\}$
		$E_3(\{\mathbf{x}_{as_2s_3}\})$	$=$	$\frac{1}{2\sigma_{jt}^2}\sum_{s_3} INA_{\alpha s_2,s_3}\lvert\mathbf{x}_{as_2s_3} - \mathbf{x}_{as_2} - \mathbf{u}_{s_2s_3}^{\alpha}\rvert^2$
			$=$	$\sum_{s_3} INA_{\alpha s_2,s_3} H^{(\alpha s_2 s_3)}(\mathbf{x}_{as_2}, \mathbf{x}_{as_2s_3})$
dot deletion	Γ^4 :	$\text{predot}(a, \alpha, s_2, s_3, \mathbf{x}_{as_2s_3})$	$\rightarrow$	$\begin{cases} \text{dot}(a, \alpha, s_2, s_3, \mathbf{x}_{as_2s_3}) & \text{if } \omega_{a\alpha s_2 s_3} = 1; \\ \text{nothing} & \text{if } \omega_{a\alpha s_2 s_3} = 0. \end{cases}$
		$E_4(\omega_{a\alpha s_2 s_3})$	$=$	$INA_{\alpha,s_2} INA_{\alpha s_2,s_3} \mu_{\alpha s_2 s_3} \sum_a (1 - C_{a\alpha}\omega_{a\alpha s_2 s_3})$
scramble all dots, and add noise dots	Γ^5 :	$\{\text{dot}(a, \alpha, s_2 s_3, \mathbf{x}_{as_2s_3})\}$	$\rightarrow$	$\{\text{imagedot}(\mathbf{x}_k = \sum_{a\alpha s_2 s_3} P_{a\alpha s_2 s_3, i}\mathbf{x}_{as_2s_3}) \mid \omega_k = 1\}$ $\cup\{\text{imagedot}(\mathbf{x}_k) \mid \omega_k = 0\}$
		$E_5(\{\mathbf{x}_k\})$	$=$	$-\log\Big[\prod_{k\mid\omega_k=1} \delta(\mathbf{x}_k - \sum_{a\alpha s_2 s_3} P_{a\alpha s_2 s_3, k}\mathbf{x}_{a\alpha s_2 s_3})\Big]$ $-\log\delta(\sum_i \omega_k - \sum_{a\alpha s_2 s_3} A_{a\alpha s_2 s_3})$ $+\mu_{\text{extra}}\sum_i(1 - \omega_k)$
		where		$\sum_i P_{a\alpha s_2 s_3, k} = A_{a\alpha s_2 s_3}$ and $\sum_{a\alpha s_2 s_3} P_{a\alpha s_2 s_3, k} = \omega_k$

(25)

We have introduced the "aliveness variables" $A_{\ldots}^r \in \{0, 1\}$:

$$A_{a\alpha s_2 s_3} = C_{a\alpha} INA_{\alpha,s_2} INA_{\alpha s_2,s_3} \omega_{a\alpha s_2 s_3} \tag{26}$$



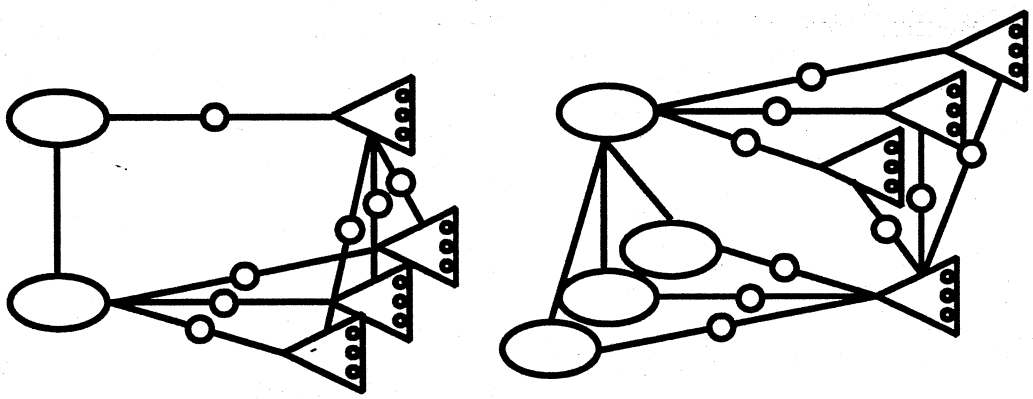


Figure 4: Frameville neural network. (a) The objective function, $E = \sum_{\alpha\beta}\sum_{ij} INA_{\alpha\beta} ina_{ij} M_{\alpha i} M_{\beta j} \times H^{\alpha\beta}(\mathbf{F}_i, \mathbf{F}_j)$. Circles are neurons, ovals are models (or frames) and triangles are model (frame) instances containing analog parameters (internal circles). (b) The constraints, $\sum_\alpha INA_{\alpha\beta} M_{\alpha i} = \sum_j ina_{ij} M_{\beta j}$ and $\sum_\beta INA_{\alpha\beta} M_{\beta j} = \sum_i ina_{ij} M_{\alpha i}$. Since $INA_{\alpha\beta}$ is a tree, the two constraint diagrams are not symmetric.

which is required in the expression for E_5. Here $C_{a\alpha}$ records the choice of model made in rule Γ^1 by object(a); thus $\sum_\alpha C_{a\alpha} = \Theta(a-1)\Theta(N-a)$ and $\sum_{a\alpha} C_{a\alpha} = 1$. $A_{a\alpha s_2 s_3}$ records which combinations of indices survive the whole grammar to account for some data dot.

In this grammar, Γ^0 is the essential new ingredient. Γ^4 and the "noise" dots of Γ^5 are just extra types of noise that can be handled. The following restrictions on Frameville apply for this grammar: *INA* is a tree; *ISA* is absent; sibling relationships (hence graph-matching on these relationships) are absent. Also it will turn out that the instance indices k and j are preassigned to either level 3, 2, or 1 of a hierarchy (corresponding to models indexed by α, β and γ respectively) which is not true of the original Frameville objective (23); this however is a much less substantive restriction.

One useful moment of the joint model-image probability distribution is

$$\Pr^f(\{\mathbf{x}_{a\alpha}\}, \{\mathbf{x}_{a\alpha s_2}\}, \{C_{a\alpha}\} | \{\mathbf{x}_k\}) = \frac{1}{Z}\sum \left\{ \begin{array}{l} P | \ \sum_k P_{a\alpha s_2 s_3, k} \le 1 \ \wedge \ \sum_{a\alpha s_2 s_3} P_{a\alpha s_2 s_3, k} \le 1 \\ \wedge \ P_{a\alpha s_2 s_3, k} = P_{a\alpha s_2 s_3, k} C_{a\alpha} INA_{\alpha, s_2} INA_{\alpha s_2, s_3} \end{array} \right\} \exp -\hat{\beta} E(\{\mathbf{x}_{a\alpha}\}, \{\mathbf{x}_{a\alpha s_2}\}, \{C_{a\alpha}\}, \{\mathbf{x}_k\}). \quad (27)$$

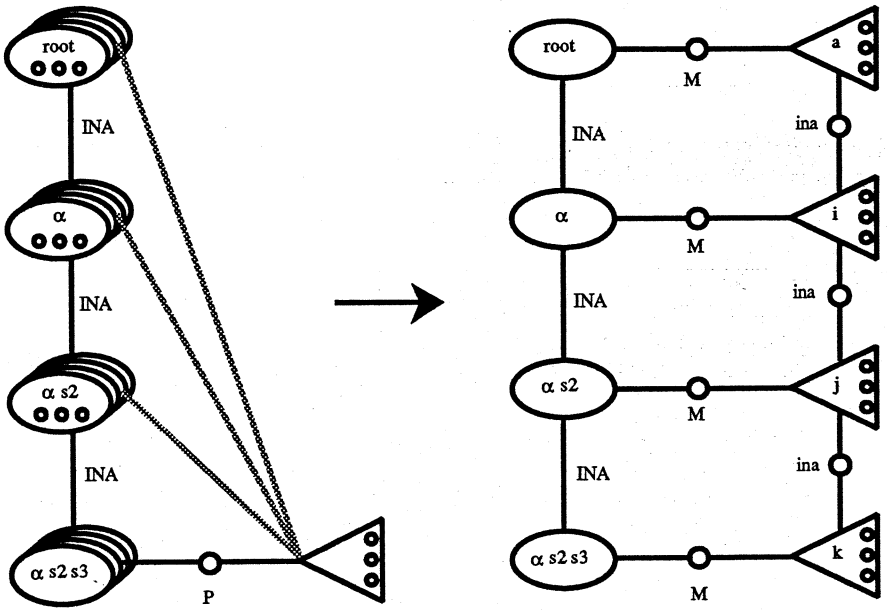


Figure 5: Change of variables, and the corresponding change in objective function, from global permutation variables P to local correspondance variables M and grouping variables *ina*. Note that analog parameters move from the models (where they must be present in multiple copies) to the instances. Left of arrow: objective of equation (28). Right of arrow: objective of equation (23) or (39).

Here $\hat{\beta}$ is the inverse temperature (to be taken to 1) and

$$\begin{aligned} E(\ldots) \;=\; & \sum_{a\alpha s_2 s_3} \sum_{k} P_{a\alpha s_2 s_3,k} \left[H^{\alpha s_2 s_3}(\mathbf{x}_{a\alpha s_2}, \mathbf{x}_k) - \mu_{\text{extra}} - \mu^{\alpha s_2 s_3} \right] \\ & + \sum_{a\alpha s_2} C_{a\alpha} H^{\alpha s_2}(\mathbf{x}_{a\alpha}, \mathbf{x}_{a\alpha s_2}) + \sum_{a\alpha} C_{a\alpha} H^{\text{root}\alpha}(\mathbf{x}_{a\alpha}). \end{aligned} \tag{28}$$

This objective is illustrated in Figure 5(a).

3.3 Changing Variables

To get the Frameville objective and constraints, we must reparameterize $\Pr^f$ and E by *changing variables*. Generally we do this by pushing the permutations, P, farther back into the grammar. A computational advantage is that P's replacements will have fewer indices and hence be less costly. To begin with, for a data item indexed by k we could separately specify its correspondance to s_3 and to (a, α, s_2). Unfortunately E needs to know more than s_3 in order to apply the correct H term, so we instead consistently specify α, s_2, s_3 and a, α, s_2 for each k:

$$P_{a\alpha s_2 s_3,k} = M_{\alpha s_2 s_3,k} \hat{ina}_{a\alpha s_2,k} \tag{29}$$

where

$$M_{\alpha s_2 s_3,k} = \sum_{a} P_{a\alpha s_2 s_3,k} \quad \text{and} \quad \hat{ina}_{a\alpha s_2,k} = \sum_{s_3} P_{a\alpha s_2 s_3,k}. \tag{30}$$

The constraints on P may be consistently translated into new constraints on M and $\hat{ina}$.

This change of variables is one-to-one, so E could simply be rewritten in terms of M and $\hat{ina}$. But we can do better. $\hat{ina}$ has many similarities to P with one index removed, so one could try to change variables again to remove another index. This doesn't quite work because $\hat{ina}_{a\alpha s_2,k}$ relates coarse-scale models to fine-scale data and therefore the constraints on $\hat{ina}$ are tighter than the constraints on P. So before attempting a hierarchical induction step, we factor $\hat{ina}$ into a grouping term ina_{jk} that constructs a data hierarchy, and a coarse-scale matching matrix $P_{a\alpha s_2,j}$:

$$\hat{ina}_{a\alpha s_2,k} = \sum_j ina_{jk} P_{a\alpha s_2,j}. \tag{31}$$

The resulting change of variables is illustrated in Figure 5. It has the effect of pushing the P matrix back one level into the grammar, leaving behind Frameville variables M and ina at the bottom level (the finest scale). This entire process can be repeated inductively.

3.4 The Constraints

Upon changing variables, the original constraints on P and C become almost identical to the Frameville constraints. This is shown in the follwing theorem, which is proved in [5].

Theorem 1. For configurations of 0/1 variables $P_{a\alpha s_2 s_3,k}$ and $C_{a\alpha}$, satisfying the constraints

$$\mathcal{A}: \quad \begin{array}{rcl} \sum_k P_{a\alpha s_2 s_3,k} & \le & 1, \\ \sum_{a\alpha s_2 s_3} P_{a\alpha s_2 s_3,k} & \le & 1, \\ P_{a\alpha s_2 s_3,k} & = & P_{a\alpha s_2 s_3,k} C_{a\alpha} INA_{\alpha,s_2} INA_{\alpha s_2,s_3}, \end{array} \quad (a) \text{ and } \sum_\alpha C_{a\alpha} = 1, \ (b) \tag{32}$$

along with auxiliary variables Q^2 and Q^1 satisfying

$$\mathcal{B}: \quad \begin{array}{rcll} \sum_j Q^2_{j'j} & \le & 1, & (a2) \\ \sum_{j'} Q^2_{j'j} & = & \Theta(n^{(2)}(C) - j)\Theta(j-1), & (b2) \\ \sum_{j'j} Q^2_{j'j} & = & n^{(2)}(C) & (c2) \end{array} \quad \text{and} \quad \begin{array}{rcll} \sum_i Q^1_{i'i} & \le & 1 & (a1) \\ \sum_{i'} Q^1_{i'i} & = & \Theta(n^{(1)}(C) - i)\Theta(i-1) & (b1) \\ \sum_{i'i} Q^1_{i'i} & = & n^{(1)}(C) & (c1) \end{array} \tag{33}$$

(for certain integer-valued functions $n^{(1,2)}(C)$) there is a one-to-one correspondance with constrained configurations of 0/1 variables $M^3, M^2, M^1, ina^3, ina^2, ina^1$. The correspondance is given by

$$\mathcal{C}: \quad \begin{array}{rcll} P_{a\alpha s_2 s_3,k} & = & M^3_{\alpha s_2 s_3,k} \sum_j ina^3_{jk} P_{a\alpha s_2,j} & (3) \\ P_{a\alpha s_2,j} & = & M^2_{\alpha s_2,j} \sum_i ina^2_{ij} P_{a\alpha,i} & (2) \\ P_{a\alpha,i} & = & M^1_{\alpha,i} ina^1_{ai} & (1) \end{array} \tag{34}$$

and inversely by

$$\mathcal{D}: \quad \begin{array}{rcll} M^3_{\alpha s_2 s_3,k} & = & \sum_a P_{a\alpha s_2 s_3,k} & (a3) \\ ina^3_{jk} & = & \sum_{\alpha s_1 s_2} \left(\sum_{s_3} P_{a\alpha s_2 s_3,k}\right) P_{a\alpha s_2,j} & (b3) \\ P_{a\alpha s_2,j} & \equiv & \sum_{j'} R^3_{a\alpha s_2,j}(\{C\}) Q^2_{jj'} & (c3) \end{array} \quad \begin{array}{rcll} M^2_{\alpha s_2,j} & = & \sum_a P_{a\alpha s_2,j} & (a2) \\ ina^2_{ij} & = & \sum_{a\alpha} \left(\sum_{s_2} P_{a\alpha s_2,j}\right) P_{a\alpha,i} & (b2) \\ P_{a\alpha,i} & \equiv & \sum_{i'} R^2_{a\alpha,i}(\{C\}) Q^1_{ii'} & (c2) \end{array}$$

$$\begin{array}{rcll} M^1_{\alpha,i} & = & \sum_a P_{a\alpha,i} & (a1) \\ ina^1_{ai} & = & \sum_\alpha P_{a\alpha,i} & (b1) \end{array} \tag{35}$$

for 0/1 variables $P_{a\alpha s_2,j}$ and $P_{a\alpha,i}$ and certain ordering functions $R(C)$. The constraints on M and ina are

$\mathcal{E}$:

$\sum_{\alpha s_2 s_3} M^3_{\alpha s_2 s_3,k} \leq 1$	$(a3)$	$\sum_j ina^3_{j,k} \leq 1$	$(b3)$
$\sum_{\alpha s_2} M^2_{\alpha s_2,j} \leq 1$	$(a2)$	$\sum_i ina^2_{i,j} \leq 1$	$(b2)$
$\sum_\alpha M^1_{\alpha,i} \leq 1$	$(a1)$	$\sum_a ina^1_{a,i} \leq 1$	$(b1)$
$\sum_j ina^3_{j,k} M^2_{\alpha s_2,j} = \sum_{s_3} INA_{\alpha s_2,s_3} M^3_{\alpha s_2 s_3,k}$	$(c3)$	$\sum_k ina^3_{j,k} M^3_{\alpha s_2 s_3,k} \leq INA_{\alpha s_2,s_3} M^2_{\alpha s_2,j}$	$(d3)$
$\sum_i ina^2_{i,j} M^1_{\alpha,i} = \sum_{s_2} INA_{\alpha,s_2} M^2_{\alpha s_2,j}$	$(c2)$	$\sum_j ina^2_{i,j} M^2_{\alpha s_2,j} = INA_{\alpha,s_2} M^1_{\alpha,i}$	$(d2)$
$\sum_a ina^1_{a,i} M^0_{\text{root},a} = \sum_\alpha INA_{\text{root},\alpha} M^1_{\alpha,i}$	$(c1)$	$\sum_i ina^1_{a,i} M^1_{\alpha,i} = C_{a\alpha} \leq 1$	$(d1)$
$ina^3_{j,k} \leq \sum_{\alpha s_2} M^2_{\alpha s_2,j}$	$(e3)$	$M^3_{\alpha s_2 s_3,k} = INA_{\alpha s_2,s_3} M^3_{\alpha s_2 s_3,k}$	$(f3)$
$ina^2_{i,j} \leq \sum_\alpha M^1_{\alpha,i}$	$(e2)$	$M^2_{\alpha s_2,j} = INA_{\alpha,s_2} M^2_{\alpha s_2,j}$	$(f2)$
$\sum_i ina^1_{a,i} = 1$	$(e1)$		

(36)

Proof: see [5].

Equation (36) (proposition $\mathcal{E}$) is to be compared with the Frameville constraints [12, 22] which include $\mathcal{E}(c)$ and $\mathcal{E}(d)$ (c.f. equation (24) and Figure 4) and conventional syntax constraints $\mathcal{E}(a)$ and $\mathcal{E}(b)$ for match variables. Note that M and ina have been restricted by specializing every variable to some hierarchical level (as has been also been done in some Frameville experiments [22]). The static constraints of $\mathcal{E}(f)$ are usually taken to be obvious: only models present in the model base may have match variables. This leaves $\mathcal{E}(e)$ as the only constraint not clearly accounted for in previous Frameville research.

3.5 The Objective Function

Proposition $\mathcal{E}$ in Theorem 1 establishes the Frameville syntax constraints, including the subtle constraints of equation (24), as a consequence of the grammar. We must now derive the Frameville objective function, equation (23), from equation (28) which is the objective derived from the grammar. This involves changing variables from P to ina and M as in Theorem 1, and also from analog model variables $\mathbf{x}_{a\alpha s_2}$ and $\mathbf{x}_{a\alpha}$, which were redundantly present in multiple copies for every model, to analog instance variables $\mathbf{x}_j$ and $\mathbf{x}_i$ which determine the original variables by

$$\mathbf{x}_{a\alpha s_2} = \sum_j P_{a\alpha s_2,j} \mathbf{x}_j \quad \text{and} \quad \mathbf{x}_{a\alpha} = \sum_i P_{a\alpha,i} \mathbf{x}_i. \tag{37}$$

Now translating (28) is a matter of substituting new variables for old in each term and adding an entropy term that arises from integrating out Q, i.e. the redundancy of M and ina with respect to P.

Then the final Frameville objective function may be calculated as in [5]:

$$\begin{aligned} E(M, ina, x) = & \sum_{\alpha s_2 s_3} \sum_{jk} M^3_{\alpha s_2 s_3,k} M^2_{\alpha s_2,j} INA_{\alpha s_2,s_3} ina^3_{jk} H^{\alpha s_2 s_3}(\mathbf{x}_j, \mathbf{x}_k) \\ & + \sum_{\alpha s_2} \sum_{ij} M^2_{\alpha s_2,j} M^1_{\alpha,i} INA_{\alpha,s_2} ina^2_{ij} H^{\alpha s_2}(\mathbf{x}_i, \mathbf{x}_j) \\ & + \sum_\alpha \sum_{ai} M^1_{\alpha,i} M^0_{\text{root},a} INA_{\text{root},\alpha} ina^1_{ai} H^{\text{root}\,\alpha}(\mathbf{x}_i) \\ & + 1/\beta[\log(N^2!) - \log((N^2 - \textstyle\sum_{a\alpha s_2 i} ina^1_{ai} M^1_{\alpha i} INA_{\alpha,s_2})!) \\ & + \log(N^1!) - \log((N^1 - \textstyle\sum_{a\alpha i} ina^1_{ai} M^1_{\alpha i})!)] \end{aligned} \tag{38}$$

where as before $M^0_{\text{root},a} = INA_{\text{root},\alpha} = 1$. Note that the μ's have been absorbed into the parameter-checking functions H. This objective is a stratified or layered version of the original Frameville objective, as can be seen by rewriting it in terms of model indices $\alpha, \beta, \ldots$ that range over all three levels, and similar modified instance indices $i, j, \ldots$, and using the fact that in this paper *INA* is a tree:

$$\begin{aligned} E(M, ina, x) \;=\; & \textstyle\sum_{\alpha\beta}\sum_{ij} M^{\text{level}(\alpha)}_{\alpha,i} M^{\text{level}(\beta)}_{\beta,j} INA_{\alpha,\beta} ina^{\text{level}(\beta)}_{ij} H^{\alpha\beta}(\mathbf{x}_i, \mathbf{x}_j) \\ & + 1/\hat{\beta}[\log(N^2!) - \log((N^2 - \textstyle\sum_{\alpha\beta ij} ina^1_{ij} M^0_{\text{root},i} M^1_{\alpha,j} INA_{\alpha,\beta})!) \\ & + \log(N^1!) - \log((N^1 - \textstyle\sum_{\alpha ij} ina^1_{ij} M^0_{\text{root},i} M^1_{\alpha,i})!)]. \end{aligned} \tag{39}$$

This is to be compared with equation (23). The graph-matching terms differ just by the new level superscripts on M and *ina*, which preallocate instance indices i, j, k to specific levels of abstraction. Such specialization of instance function could probably be removed at the cost of further entropy terms. The entropy terms are new, and easily implementable with analog neural networks by Stirling's approximation and algebraic transformations of the resulting $X \log X$ forms [1].

Thus we have translated the probability distribution of the Frameville grammar, specified by the objective and the constraints, into the standard Frameville variables, recovering the standard objective function terms and constraints along with a few new ones. This may be regarded as a transformation at the level of the probability distribution, before Mean Field Theory is applied and hence before any approximations are made. It may also be possible to express this derivation as a transformation at the level of the grammar, in which the permutation operation is applied in a limited form at each stage rather than globally at the final stage of the grammar.

3.6 Frameville and High-Level Vision

With the Frameville grammar, we approach a modest plateau of generality. From the generalized assignment problem of equation (28) we have derived a network which explicitly has problems of *recognition* (find $M_{\alpha i}$), *segmentation* or *grouping* (find ina_{ij}), *correspondance* between data and the expected parts of an object (find $M_{\alpha s_2 s_3, k}$), *multiple instances* of a model (find $\mathbf{x}_i$ rather than, say, $\mathbf{x}_\alpha$), at multiple *levels of abstraction* (levels 3, 2, and 1 in the hierarchical grammar). These processes arise from Bayesian inference on a constrained Boltzmann probability distribution which, we have shown, is equivalent to the distribution generated by a simple grammar. The transformation to Frameville is natural: it simply pushes the permutation matrix as far back into the grammar as is possible, so that each grammar rule can be regarded as having its own renumbering processes even at abstract levels.

The resulting Frameville objective is different from the original generalized assignment objective in several important ways. Where the assignment objective is linear in its binary-valued match variables, the Frameville objective is cubic in far fewer variables. (The linear or cubic terms are multiplied by analog parameter-check objectives $H(\mathbf{x}, \mathbf{x})$ in both cases.) This increase in polynomial order may create more local minima in a smaller net. It is not clear whether this is a net gain or loss for practical optimization. On the other hand, further simplifying transformations such as the correlation method of Section 2.4, which have special conditions of applicability, are far more likely to apply to small, single-object correspondance problems (e.g. find $M_{\alpha s_2 s_3, k}$ given $M_{\alpha s_2, j}$) that can arise in Frameville than to the original monolithic assignment problem.

Thus the Frameville formulation suggests a modular decomposition of a large vision problem into smaller, more homogeneous pieces to which special methods are most likely to apply. The decomposition follows the lines indicated by the hierarchical and heterogeneous grammar.

One important aspect of model-based vision, and of the original Frameville networks, is still missing: the use of an indexing scheme such as a discrimination tree or graph composed of *ISA*-links to organize the set of models into a data base. An alternative efficient indexing scheme, not used in the Frameville networks, is geometric hashing [36].

We have studied grammars that model visual phenomena such as missing and extra data, group invariances, hierarchical objects, and multiple instances of an object in a scene. The rudiments of a frame system for knowledge representation emerged naturally from one such grammar, by pushing the matching process from low levels to high levels in a hierarchical, multiple-instance grammar. Nevertheless the full representational capacities of such grammars were hardly used: it remains to design networks from grammars that generate trees recursively, or are context-dependent (perhaps with several grammatical terms interacting to produce new terms as in [7]), or include discrimination or property inheritance trees on the set of object models.

Acknowledgements

The author benefitted from discussions with P. Anandan, Gene Gindi, Greg Hager, Chien Ping Lu, Drew McDermott, Anand Rangarajan, Joachim Utans and Alan Yuille. C. Garrett performed computer simulations of dot-matching networks.

References

[1] E. Mjolsness and C. Garrett, "Algebraic transformations of objective functions," *Neural Networks*, vol. 3, pp. 651–669, 1990.

[2] P. D. Simic, "Statistical mechanics as the underlying theory of 'elastic' and 'neural' optimization," *Network: Computation in Neural Systems*, vol. 1, pp. 89–103, January 1990.

[3] C. Peterson and B. Soderberg, "A new method for mapping optimization problems onto neural networks," *International Journal of Neural Systems*, vol. 1, no. 3, 1989.

[4] D. E. Van den Bout and T. K. Miller, III, "Graph partitioning using annealed networks," *IEEE Transactions on Neural Networks*, vol. 1, pp. 192–203, June 1990.

[5] E. Mjolsness, "Bayesian inference on visual grammars by neural nets that optimize," Tech. Rep. YALEU/DCS/TR-854, Yale Computer Science Department, May 1991. (Temporarily available on neuroprose as "mjolsness.grammar.ps.Z").

[6] E. Mjolsness, "Visual grammars and their neural networks," in *Science of Artificial Neural Networks* (D. W. Ruck, ed.), vol. 1710, pp. 63–85, SPIE, April 1992.

[7] E. Mjolsness, D. H. Sharp, and J. Reinitz, "A connectionist model of development," *Journal of Theoretical Biology*, vol. 152, pp. 429–453, 1991.

[8] P. Prusinkiewicz, *The Algorithmic Beauty of Plants.* Springer-Verlag New York, 1990.

[9] A. R. Smith, "Plants, fractals and formal languages," *Computer Graphics*, vol. 18, pp. 1–10, July 1984. Proceedings of SIGGRAPH '84.

[10] G. Rozenberg and A. Salomaa, *The Mathematical Theory of L-systems.* Academic Press, 1980.

[11] H. Ehrig, M. Nagl, A. Rosenfeld, and G. Rozenberg, eds., *Graph Grammars and their Application to Computer Science; Third International Workshop*, vol. 291 of *Lecture Notes in Computer Science.* Springer-Verlag, 1983.

[12] E. Mjolsness, G. Gindi, and P. Anandan, "Optimization in model matching and perceptual organization," *Neural Computation*, vol. 1, 1989.

[13] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation*, ch. 5, p. 333. Prentice Hall, 1989.

[14] J. J. Kosowsky and A. L. Yuille, "The invisible hand algorithm: Solving the assignment problem with statistical physics," Tech. Rep. 91-1, Harvard Robotics Laboratory, 1991.

[15] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons," *Proceedings of the National Academy of Sciences USA*, vol. vol. 81, pp. 3088–3092, May 1984.

[16] E. Mjolsness and W. L. Miranker, "A Lagrangian approach to fixed points," in *Neural Information Processing Systems 3* (R. P. Lippmann, J. E. Moody, and D. S. Touretzky, eds.), Morgan Kaufmann, 1991.

[17] E. Mjolsness, "Control of attention in neural networks," in *Proc. of First International Conference on Neural Networks*, vol. vol. II, pp. 567–574, IEEE, 1987.

[18] A. Witkin, D. Terzopoulos, and M. Kass, "Signal matching through scale space," *International Journal of Computer Vision*, vol. 1, pp. 133–144, 1987.

[19] R. Durbin and D. Willshaw, "An analog approach to the travelling salesman problem using an elastic net method," *Nature*, vol. 326, pp. 689–691, 1987.

[20] A. L. Yuille, "Generalized deformable models, statistical physics, and matching problems," *Neural Computation*, vol. 2, no. 1, pp. 1–24, 1990.

[21] P. Anandan, S. Letovsky, and E. Mjolsness, "Connectionist variable-binding by optimization," in *11th Annual Conference of the Cognitive Science Society*, Lawrence Earlbaum Associates, August 1989. University of Michigan.

[22] J. Utans, G. Gindi, E. Mjolsness, and P. Anandan, "Neural networks for object recognition within compositional hierarchies: Initial experiments," Tech. Rep. Report No. 8903, Center for Systems Science, Yale University Department of Electrical Engineering, February 1989.

[23] S. E. Fahlman, *NETL: A System for Representing and Using Real-World Knowledge.* MIT Press, 1979.

[24] R. S. Zemel, "TRAFFIC: A connectionist model of object recognition," Tech. Rep. CRG-TR-89-2, University of Toronto Connectionist Research Group, February 1989.

[25] C. von der Malsburg and E. Bienenstock, "Statistical coding and short-term synaptic plasticity: A scheme for knowledge representation in the brain," in *Disordered Systems and Biological Organization*, pp. 247–252, Springer-Verlag, 1986.

[26] C. von der Malsburg, "Pattern recognition by labeled graph matching," *Neural Networks*, vol. 1, pp. 141–148, 1988.

[27] P. R. Cooper, *Parallel Object Recognition from Structure (The Tinkertoy Project).* PhD thesis, University of Rochester Department of Computer Science, July 1989. Technical Report 301.

[28] J. A. Feldman, M. A. Fanty, and N. H. Goddard, "Computing with structured neural networks," *IEEE Computer*, p. 91, March 1988.

[29] E. Bienenstock and R. Doursat, "Issues of representation in neural networks," in *Representations of Vision: Trends and Tacit Assumptions in Vision Research* (A. Gorea, ed.), Cambridge University Press, 1991.

[30] D. H. Ballard, "Parallel logical inference and energy minimization," in *Proceedings of the 5th National Conference on Artificial Intelligence*, pp. 203–208, 1986.

[31] M. Derthick, *Mundane Reasoning by Parallel Constraint Satisfaction.* PhD thesis, Carnegie Mellon University, September 1988. Available as CMU-CS-88-182 from the Computer Science Department.

[32] P. Smolensky, "Tensor product variable binding and the representation of symbolic structures in connectionist systems," in *Connectionist Symbol Processing* (G. Hinton, ed.), pp. 159–216, MIT Press, 1991. Special issue of Artificial Intelligence: An International Journal, v. 46, nos. 1-2, 1990.

[33] A. Stolcke, "Unification as constraint satisfaction in structured connectionist networks," *Neural Computation*, vol. 1, no. 4, pp. 559–567, 1989.

[34] G. Pinkas, "First-order logic proofs using connectionist constraint relaxation," Tech. Rep. WUCS-91-54, Washington University Department of Computer Science, December 1991.

[35] J. C. Platt and A. H. Barr, "Constrained differential optimization," in *Neural Information Processing Systems* (D. Z. Anderson, ed.), American Institute of Physics, 1987.

[36] Y. Lamdan, J. T. Schwartz, and H. J. Wolfson, "Object recognition by affine invariant matching," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 335–344, June 1988.