Bayesian Inference on Visual Grammars by Neural Nets that Optimize

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

May 1, 1990

YALEU-DCS-TR-854

Abstract

We exhibit a systematic way to derive neural nets for vision problems. It involves formulating a vision problem as Bayesian inference or decision on a comprehensive model of the visual domain given by a probabilistic *grammar*. A key feature of this grammar is the way in which it eliminates model information, such as object labels, as it produces an image; correspondance problems and other noise removal tasks result. The neural nets that arise most directly are generalized assignment networks. Also there are transformations which naturally yield improved algorithms such as correlation matching in scale space and the Frameville neural nets for high-level vision. Networks derived this way generally have objective functions with spurious local minima; such minima may commonly be avoided by dynamics that include deterministic annealing, for example recent improvements to Mean Field Theory dynamics. The grammatical method of neural net design allows domain knowledge to enter from all levels of the grammar, including "abstract" levels remote from the final image data, and may permit new kinds of learning as well.

Contents

1 INTRODUCTION

We show how to derive various optimizing neural networks, which represent quantitative visual models and match them to data, from fundamental considerations of Bayesian reasoning. Doing so illustrates a design methodology which starts from first principles, namely a probabilistic model of a visual domain, and proceeds to a neural network which performs visual tasks. The key problem is to choose probability distributions sufficiently intricate to model general visual tasks and yet tractable enough for theory. We do this by means of probabilistic and expressive grammars which model the image-formation process, including heterogeneous sources of noise each modelled with a grammar rule. In particular these grammars include a crucial "relabelling" rule that removes the undetectable internal labels (or indices) of detectable features and substitutes an uninformed labelling scheme used by the perceiver.

For such grammars, in which every rule has a simple Boltzmann probability distribution, it is straightforward to generate a neural network 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) Express desired averages under this distribution in terms of the optimization of an objective function E. This step usually employs Mean Field Theory approximations; the scaling properties and practicality of such approximations have been greatly improved by (Simic, 1990b; Peterson and Soderberg, 1989; Van den Bout and Miller, III, 1990) for matching problems similar to those we encounter. (4) Introduce optimizing neural net dynamics for E.

The procedure becomes more elaborate and malleable by using valid *transformations* (e.g. of probability distributions or objective functions) at each step to reduce network cost, improve network performance or achieve network implementability in some technology. A small catalog of valid objective function transformations for neural nets is presented in (Mjolsness and Garrett, 1990), and the present paper illustrates several transformations of probability distributions. The entire method is sketched in Figure 1.

This paper is organized as follows. In Section 2 we introduce an example grammar which models a simple picture-formation process; we derive a joint probability distribution on models and images (or pictures) and discuss various questions which could be answered by computing moments of this distribution; we derive neural nets for doing such computations, both with and without using "match neurons" which explicitly hypothesize correspondance between model and data objects. The network without such neurons is interpreted as the relatively efficient algorithm of correlation matching in scale space. In Section 3 we demonstrate such nets on an image registration problem. In Section

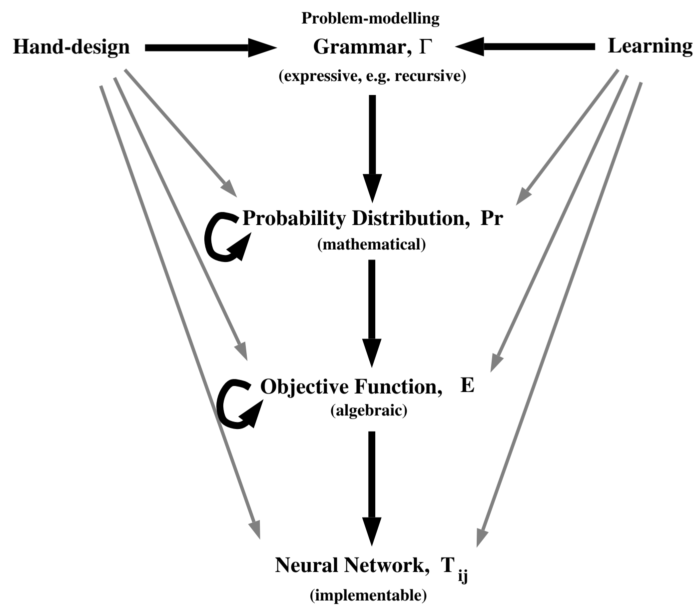


Figure 1: A neural network design methodology. Solid arrows constitute the recommended procedure. The arrow from Pr to E may be realized by approximations from statistical physics, such as Mean Field Theory. Circular arrows represent valid transformations, such as fixed-point preserving transformations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

4 we exhibit a variety of probabilistic grammars and the joint probability distributions they imply, including a network for recognizing simple flexible objects. There we also derive the previously studied "Frameville" neural networks for high-level vision from a simple type of grammar. It results from "pushing" the relabelling operation back to more and more abstract levels of the grammar, thereby breaking up a massive correspondance problem into many small correspondance and grouping problems that interact. To put the latter result in context, it suggests a natural way to derive neural nets that are capable of expressing abstractions usually reserved for symbolic computing, and indicates how symbolic computing in perceptual domains could be improved by derivation from underlying physical models akin to those used in physically-based computer graphics. In Section 5 we speculate on learning algorithms for the types of neural nets we are discussing, illustrated with an example involving inexact graph matching. In Section 6 we discuss the results and conclude.

2 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.

2.1 The Grammar

The random-dot grammar is shown below.

model and its location	$\Gamma^0:$	root	$\rightarrow$	instance of model α at $\mathbf{x}$
		$E_0(\mathbf{x})$	$=$	$\frac{1}{2\sigma_r^2}\lvert\mathbf{x}\rvert^2$
dot locations	$\Gamma^1:$	$\mathrm{instance}(\alpha, \mathbf{x})$	$\rightarrow$	$\{\mathrm{dotloc}(\alpha, m, \hat{\mathbf{x}}_m = \mathbf{x} + \mathbf{u}_m^\alpha)\}$
		$E_1(\{\hat{\mathbf{x}}_m\})$	$=$	$-\log\prod_m \delta(\hat{\mathbf{x}}_m - \mathbf{x} - \mathbf{u}_m^\alpha)$, where $< \mathbf{u}_m^\alpha >_m = 0$
			$\approx$	$\lim_{\sigma_\delta \to 0} \frac{1}{2\sigma_\delta^2}\sum_m \lvert\mathbf{x}_m - \mathbf{x} - \mathbf{u}_m^\alpha\rvert^2 + c(\sigma_\delta)$
dot jitter	$\Gamma^2:$	$\mathrm{dotloc}(\alpha, m, \hat{\mathbf{x}}_m)$	$\rightarrow$	$\mathrm{dot}(m, \mathbf{x}_m)$
		$E_2(\mathbf{x}_m)$	$=$	$\frac{1}{2\sigma_{jt}^2}\lvert\hat{\mathbf{x}}_m - \mathbf{x}_m\rvert^2$
choose random permutation	$\Gamma^3:$	$\{\mathrm{dot}(m, \mathbf{x}_m)\}$	$\rightarrow$	$\{\mathrm{dot}(m, \mathbf{x}_m), P_{m,i}\}$
		$E_3(\{P_{m,i}\})$	$=$	$-\log \Pr(P)$, where P is a permutation
relabel all dots	$\Gamma^4:$	$\{\mathrm{dot}(m, \mathbf{x}_m), P_{m,i}\}$	$\rightarrow$	$\{\mathrm{imagedot}(\mathbf{x}_i = \sum_m P_{m,i}\mathbf{x}_m)\}$
		$E_4(\{\mathbf{x}_i\})$	$=$	$-\log\prod_i \delta(\mathbf{x}_i - \sum_m P_{m,i}\mathbf{x}_m)$

(1)

The operation of this grammar is illustrated in Figure 2. We will show that this grammar is equivalent to a grammar with fewer rules:

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

(2)

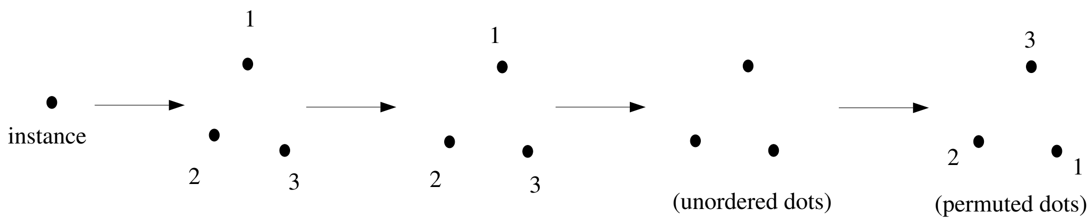


Figure 2: Operation of random dot grammar. The first arrow illustrates dot placement; the next shows dot jitter; the next arrow shows the pure, unlabelled feature locations; and the final arrow is the uninformed relabelling scheme of the perceiver.

The generative process specified by this grammar is to start with nothing (the "root" of a parse tree) and generate 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 $\mathbf{x}$. 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 $\mathbf{x} + \mathbf{u}^\alpha_m$. 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 or "scrambled" set of image dots at the same set of locations. The permutation is not a physical operation; it simply relabels the dots from index m to index i by means of a permutation P_{mi}. The permutation probability distribution $\Pr(P)$ will be specified in the next section. We will show that it is indistinguishable from the uniform distribution on permutations.

2.2 Final Probability Distribution

The probability distribution associated with a particular rule Γ^r is

$$\Pr(\text{new terms}, \{\text{new parameters}\}|\text{old terms}, \{\text{old parameters}\}) = e^{-\beta E_r}/Z_r \tag{3}$$

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

$$\Pr(\xi, x) = \Pr(x|\xi)\Pr(\xi) \tag{4}$$

to yield a final joint probability distribution for the entire grammar. For Grammar 1, we could multiply to get an expression for $\Pr(\alpha, \mathbf{x}, \{\hat{\mathbf{x}}_m\}, \{\mathbf{x}_m\}, \{P_{m,i}\}, \{\mathbf{x}_i\})$. However we are usually interested in computing some average in this distribution, i.e. some moment of this function.

To begin with, the delta functions in rules 1 and 4 completely determine $\hat{\mathbf{x}}_m$ and $\mathbf{x}_m$, respectively, in terms of the other variables; so these are usually integrated out of any interesting average. That leaves $\Pr(\alpha, \mathbf{x}, \{P_{m,i}\}, \{\mathbf{x}_i\})$. For this reason Grammar 2 is obtained by combining Grammar 1's rules 1 and 2 (integrating out $\hat{\mathbf{x}}_m$) and rules 3 and 4 (leaving in $\mathbf{x}_m$). In Section 2.3 we will show that the P variables can often be summed out without losing the answer to real problems. As an example, let us calculate $\Pr(\alpha, \mathbf{x}, \{\mathbf{x}_i\})$ for this grammar.

Calculating the joint probability distribution is especially easy for this grammar because the grammar rules are not recursive. The grammar just consists of a sequence of three stages corresponding to rules 0-2. After rule 0,

$$\Pr^0(\alpha, \mathbf{x}) = \frac{1}{A}\left(\frac{1}{\sqrt{2\pi}\sigma_r}\right)^2 e^{-\frac{1}{2\sigma_r^2}|\mathbf{x}|^2} \tag{5}$$

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

$$\begin{aligned} \Pr^1(\alpha, \mathbf{x}, \{\mathbf{x}_m\}) &= \Pr^1(\{\mathbf{x}_m\}|\alpha, \mathbf{x})\Pr^0(\alpha, \mathbf{x}) \\ &= \frac{1}{A}\left(\frac{1}{\sqrt{2\pi}\sigma_r}\right)^2 \left(\frac{1}{\sqrt{2\pi}\sigma_{jt}}\right)^{2N} e^{-\left(\frac{1}{2\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}\sum_m |\mathbf{x}_m - \mathbf{x} - \mathbf{u}_m^\alpha|^2\right)}. \end{aligned} \tag{6}$$

To finish the calculation we must consider $\Pr(P)$. This is the probability of a given relabelling 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 is to feign ignorance of the permutation and use the maximum entropy distribution on P, namely a uniform distribution. This model seems artificial because there is no actual uniform-probability scrambling mechanism in natural image-generation processes.

Alternatively we could model the relabelling process as deterministically *sorting* the raw dot position data $\mathbf{x}_m$ according to some scalar criterion $f(\mathbf{x}_m)$ so that low values of f are indexed by integers i with low values of some set of weights w_i (e.g. $w_i = i$), and high f's correspond to high w's. This sorting is done by whatever part of the perceiver turns images into some representation of low-level symbols. For example, f could encode the lexicographic ordering of a 2-d array of pixels given by their raster or scan-line sequence, and the dots would be indexed in that order. Or f could order the 2-d array of pixels according to the discretization of any space-filling curve. It is important that f does not depend on quantities like m or α that require perception to deduce. $f(\mathbf{x})$ should also be unique, i.e. f is an invertable function. Then, $\Pr(P)$ is

$$\Pr(P) = \lim_{\beta_{\text{sort}} \to \infty} e^{\beta_{\text{sort}} \sum_{mi} P_{m,i} f(\mathbf{x}_m) w_i} / Z_P \tag{7}$$

and E_2 contributes

$$\begin{aligned}
\sum_{\{P\}} e^{-E_2(\mathbf{x}_i)} &= \lim_{\beta_{\text{sort}}\to\infty} \sum_{\left\{\substack{\text{P}|\ \text{P is a}\\ \text{permutation}}\right\}} \frac{e^{\beta_{\text{sort}} \sum_{mi} P_{m,i} f(\mathbf{x}_m) w_i}}{Z_P} \prod_i \delta(\mathbf{x}_i - \textstyle\sum_m P_{m,i}\mathbf{x}_m) \\
&= \lim_{\beta_{\text{sort}}\to\infty} \sum_{\left\{\substack{\text{P}|\ \text{P is a}\\ \text{permutation}}\right\}} \frac{e^{\beta_{\text{sort}} \sum_i f(\sum_m P_{m,i}\mathbf{x}_m) w_i}}{Z_P} \prod_i \delta(\mathbf{x}_i - \textstyle\sum_m P_{m,i}\mathbf{x}_m) \\
&= \lim_{\beta_{\text{sort}}\to\infty} \sum_{\left\{\substack{\text{P}|\ \text{P is a}\\ \text{permutation}}\right\}} \frac{e^{\beta_{\text{sort}} \sum_i f(\mathbf{x}_i) w_i}}{Z_P} \prod_i \delta(\mathbf{x}_i - \textstyle\sum_m P_{m,i}\mathbf{x}_m) \\
&= \left(\lim_{\beta_{\text{sort}}\to\infty} e^{\beta_{\text{sort}} \sum_i f(\mathbf{x}_i) w_i} \Big/ \sum_{\left\{\substack{\text{P}|\ \text{P is a}\\ \text{permutation}}\right\}} e^{\beta_{\text{sort}} \sum_{mi} P_{m,i} f(\mathbf{x}_m) w_i} \right) \\
&\qquad \times \sum_{\left\{\substack{\text{P}|\ \text{P is a}\\ \text{permutation}}\right\}} \prod_i \delta(\mathbf{x}_i - \textstyle\sum_m P_{m,i}\mathbf{x}_m).
\end{aligned} \tag{8}$$

Here the second term enforces the constraint that $\mathbf{x}_i$ be a permutation of $\mathbf{x}_m$, and in this circumstance the first term constrains $\mathbf{x}_i$ to be in order according to f. Thus we may simplify the first term when it is multiplied by the second:

$$\sum_{\{P\}} e^{-E_2(\mathbf{x}_i)} = \Theta(\{\mathbf{x}_i\}) \sum_{\left\{\substack{P|\ P \text{ is a}\\ \text{permutation}}\right\}} \prod_i \delta(\mathbf{x}_i - \sum_m P_{m,i}\mathbf{x}_m), \tag{9}$$

where $\Theta(\{\mathbf{x}_i\})$ is 1 if $\{\mathbf{x}_i\}$ are in order according to f, and zero otherwise. In the first step of this derivation we have used a crucial property of permutation matrices: that $\sum_m P_{m,i} f(\mathbf{x}_m) = f(\sum_m P_{m,i}\mathbf{x}_m)$ because exactly one element of $\{P_{m,i}|m \in \{1 \ldots N\}\}$ is nonzero.

Since $\Theta(\{\mathbf{x}_i\})$ does not depend on $\{\mathbf{x}_m\}$, it will cancel out when we compute conditional probabilities such as $\Pr^{\text{final}}(\alpha, \mathbf{x}|\{\mathbf{x}_i\})$ (c.f. equation (13) below.) In other words, the sorting model of $\Pr(P)$ is equivalent to the uniform-distribution model as far as E_2 and hence $\Pr^2$ are concerned.

So,

$$
\begin{aligned}
\Pr^{\text{final}}(\alpha, \mathbf{x}, \{\mathbf{x}_i\}) &= \frac{\Theta(\{\mathbf{x}_i\})}{A} \left(\frac{1}{\sqrt{2\pi}\sigma_r}\right)^2 \left(\frac{1}{\sqrt{2\pi}\sigma_{jt}}\right)^{2N} \sum_{\left\{ \substack{\text{P}|\ \text{P is a} \\ \text{permutation}} \right\}} \\
&\quad \int d\{\mathbf{x}_m\} \prod_i \delta(\mathbf{x}_i - \sum_m P_{m,i}\mathbf{x}_m) e^{-\left(\frac{1}{2\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}\sum_m |\mathbf{x}_m - \mathbf{x} - \mathbf{u}_m^\alpha|^2\right)} \\
&= \frac{\Theta(\{\mathbf{x}_i\})}{A} \left(\frac{1}{\sqrt{2\pi}\sigma_r}\right)^2 \left(\frac{1}{\sqrt{2\pi}\sigma_{jt}}\right)^{2N} \\
&\quad \times \sum_{\left\{ \substack{\text{P}|\ \text{P is a} \\ \text{permutation}} \right\}} e^{-\left(\frac{1}{2\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}\sum_m |\sum_i P_{m,i}\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2\right)}
\end{aligned}
\tag{10}
$$

and finally

$$
\begin{aligned}
\Pr^{\text{final}}(\alpha, \mathbf{x}, \{\mathbf{x}_i\}) &= \frac{\Theta(\{\mathbf{x}_i\})}{A} \left(\frac{1}{\sqrt{2\pi}\sigma_r}\right)^2 \left(\frac{1}{\sqrt{2\pi}\sigma_{jt}}\right)^{2N} \\
&\quad \times \sum_{\left\{ \substack{\text{P}|\ \text{P is a} \\ \text{permutation}} \right\}} e^{-\beta \sum_{mi} P_{m,i} \left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2\right)}.
\end{aligned}
\tag{11}
$$

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 local minima during a computation.

In Section 2.4 we review how the Boltzmann distribution (11) and its derivatives may be approximated by a neural net involving quadratic match neurons related to $P_{m,i}$. In Section 2.5 we exhibit a new way to derive even simpler, though sometimes less accurate, neural nets for such problems by eliminating the P variables.

Equation (11) 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" (Bertsekas and Tsitsiklis, 1989) 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 (Kosowsky and Yuille, 1991). In equation (11) 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:

$$E_{\text{final}}(\alpha, P, \mathbf{x}) = \sum_{mi} P_{m,i} \left(\frac{1}{2N\sigma_r^2} |\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2} |\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2 \right) \tag{12}$$

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.

2.3 Inference and Decision Problems

We now show how to pose a variety of problems which could arise in situations modeled by the grammer and whose solution can be expressed as sums over the P configuration space as in the previous section.

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

$$\begin{aligned} \text{argmax}_{\alpha,\mathbf{x}} \Pr(\alpha, \mathbf{x} | \{\mathbf{x}_i\}) &= \text{argmax}_{\alpha,\mathbf{x}} \frac{\Pr(\alpha,\mathbf{x},\{\mathbf{x}_i\})}{\Pr(\{\mathbf{x}_i\})} \qquad \text{(Bayesian inference)} \\ &= \text{argmax}_{\alpha,\mathbf{x}} \Pr(\alpha, \mathbf{x}, \{\mathbf{x}_i\}) \\ &= \text{argmax}_{\alpha,\mathbf{x}} \sum_{\left\{ \begin{array}{c} P|\ P \text{ is a} \\ \text{permutation} \end{array} \right\}} \Pr(\alpha, \mathbf{x}, \{P_{m,i}\}, \{\mathbf{x}_i\}). \end{aligned} \tag{13}$$

Note that the combination of equations (13) and (4) perform Bayesian inference: they determine Pr(model params|data) in terms of forward conditional probabilities including Pr(data|model params). From equation 13, this recognition problem involves computing the function $\Pr(\alpha, \mathbf{x}, \{\mathbf{x}_i\})$ for which we must integrate (sum) out the P variables.

Other problems might require the computation of the average $\langle P_{m,i} \rangle$, which still requires summing over the P configurations. Indeed most if not all inference problems for the grammar can be formulated in terms of such sums. A Mean Field Theory approximation to these high-dimensional sums may be performed by a neural net (Hopfield, 1984; Peterson and Soderberg, 1989; Simic, 1990b; Yuille, 1990; Simic, 1990a; Tresp, 1991), as we will review in Section 2.4.

An even simpler recognition problem would be to infer the most likely model α from data $\{\mathbf{x}_i\}$ without regard to its position. This is just like the previous problem except we want to integrate out $\mathbf{x}$. The integral with respect to arbitrary translations $\mathbf{x}$ could be done analytically, but corresponding integrals over 2-d rotation or other Lie group distortions that could be added to the grammar (see Section 4.1) must be approximated and we could treat translations the same way. In particular, a saddle point approximation to the integral over $\mathbf{x}$ (involved in the MFT method) requires finding the most likely value of $\mathbf{x}$ but not reporting it as part of the answer.

Here are two more examples of problems posed in terms of the grammar which could be solved by summing over all P configurations. One might want to find the single configuration of $\alpha, \mathbf{x}, \{\mathbf{x}_m\}$, and $\{P_{m,i}\}$ which is most probable given observations $\{\mathbf{x}_i\}$, i.e. the *maximum a posteriori* (MAP) estimate of all these variables. We may compute the total partition function

$$\begin{aligned} Z(\beta, \lambda, \boldsymbol{\mu}, \{\boldsymbol{\nu}\}, \{\pi\}) \quad = \quad & \textstyle\sum_\alpha \int d\mathbf{x} \int d\{\mathbf{x}_m\} \sum_{\left\{ \begin{array}{c} P| \;\; P \text{ is a} \\ \text{permutation} \end{array} \right\}} \\ & \exp\left[-\beta E_{\text{final}} + \lambda\alpha + \boldsymbol{\mu}\cdot\mathbf{x} + \textstyle\sum_m \boldsymbol{\nu}_m\cdot\mathbf{x}_m + \sum_{m,i}\pi_{m,i}P_{m,i}\right]. \end{aligned} \tag{14}$$

Then, average values of P and the other variables are derivatives of $\log Z$ evaluated at $\lambda, \boldsymbol{\mu}, \boldsymbol{\nu}, \pi = 0$. As $\beta \to \infty$, these averages approach the MAP configuration if it is unique. Note the departure from $\beta \to 1$ which would be used for the other problems discussed in this section. In fact the argmax taken in equation (13) could also be done by a zero-temperature limit, but that would involve a *different β parameter* than the one used in integrating out P.

A second type of problem involves *deciding what to do* based on $\Pr(\alpha, \mathbf{x}|\{\mathbf{x}_i\})$. For example, each model α might have a known interesting part whose position relative to the object center of mass is $\mathbf{c}^\alpha$. The problem is to set the variable $\mathbf{y}$ near this location (e.g. to point a telescope), under uncertainty in the values of α and $\mathbf{x}$. This could be formalized as maximizing the probable reward

$$\begin{aligned} F(\mathbf{y}|\{\mathbf{x}_i\}) \quad &= \quad \textstyle\sum_\alpha \int d\mathbf{x} e^{-\frac{1}{2\sigma_y^2}(\mathbf{x}+\mathbf{c}^\alpha-\mathbf{y})^2} \Pr(\alpha, \mathbf{x}|\{\mathbf{x}_i\}) \\ &\propto \quad \textstyle\sum_\alpha \int d\mathbf{x} \sum_{\left\{ \begin{array}{c} P| \;\; P \text{ is a} \\ \text{permutation} \end{array} \right\}} \exp -\beta \left[E_{\text{final}}(\alpha, P, \mathbf{x}) - \dfrac{1}{2\sigma_y^2}(\mathbf{x} + \mathbf{c}^\alpha - \mathbf{y})^2\right] \end{aligned} \tag{15}$$

with respect to $\mathbf{y}$. Again we use Bayesian inference to get $\Pr(\alpha, \mathbf{x}|\{\mathbf{x}_i\})$, and again a sum over configurations of P is involved. The integral over $\mathbf{x}$ may be done analytically or approximately, as previously discussed.

Thus a wide variety of Bayesian inference and probabilistic decision problems may be reduced to calculating moments of $\Pr(\alpha, \mathbf{x}, \{P_{m,i}\}, \{\mathbf{x}_i\})$ that involve summing out the P configurations and integrating or maximizing with respect to the other random variables.

2.4 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:

$$\begin{aligned}
\Pr^f(\alpha, \mathbf{x}|\{\mathbf{x}_i\}) &= C \sum_{\left\{ \begin{array}{l} P|\sum_i P_{m,i}=1 \\ \wedge \sum_m P_{m,i}=1 \end{array} \right\}} \exp - \sum_{mi} P_{m,i} \left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2 \right) \\
&= C \sum_{\left\{ P|\sum_i P_{m,i}=1 \right\}} \prod_i \delta^K(\sum_m P_{m,i}, 1) \int_{-\infty}^{+\infty} d\{V_{mi}\} \prod_{m,i} \delta(V_{m,i} - P_{m,i}) \\
&\quad \exp - \textstyle\sum_{mi} V_{m,i} \left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2 \right)
\end{aligned} \tag{16}$$

where $\delta^K(n, m)$ is the Kronecker delta function on integers and $\delta(x)$ is the Dirac delta function on real numbers. Both have Gaussian representations, but we'll use an integral representation of the Dirac delta and the Gaussian representation $\delta^K(n, m) = \lim_{A\to\infty} \exp -(A/2)(n - m)^2$. Continuing,

$$\begin{aligned}
\Pr^f(\alpha, \mathbf{x}|\{\mathbf{x}_i\}) &= C \int_{-\infty}^{+\infty} d\{V_{mi}\} \sum_{\left\{ P|\sum_i P_{m,i}=1 \right\}} \lim_{A\to\infty} \exp \left[-(A/2) \sum_i (\sum_m P_{m,i} - 1)^2 \right] \\
&\quad \int_{-i\infty}^{+i\infty} d\{U_{mi}\} e^{-\sum_{m,i} U_{m,i}(V_{m,i} - P_{m,i})} \exp -\beta \sum_{mi} V_{m,i} \left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2 \right) \\
&= C \lim_{A\to\infty} \int_{-\infty}^{+\infty} d\{V_{mi}\} \int_{-i\infty}^{+i\infty} d\{U_{mi}\} \exp \left[-\beta \sum_{mi} V_{m,i} \left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2 \right) \right] \\
&\quad \exp \left[-\beta(A/2) \sum_i (\sum_m V_{m,i} - 1)^2 - \sum_{m,i} U_{m,i} V_{m,i} \right] \sum_{\left\{ P|\sum_i P_{m,i}=1 \right\}} \exp \sum_{m,i} U_{m,i} P_{m,i} \\
&= C \lim_{A\to\infty} \int_{-\infty}^{+\infty} d\{V_{mi}\} \int_{-i\infty}^{+i\infty} d\{U_{mi}\} \exp -\beta E_{eff}(\alpha, \mathbf{x}, \{U\}, \{V\})
\end{aligned} \tag{17}$$

where

$$\begin{aligned}
E_{eff}(\alpha, \mathbf{x}, \{U\}, \{V\}) \equiv & \textstyle\sum_{mi} V_{m,i} \left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2 \right) + (A/2) \sum_i (\sum_m V_{m,i} - 1)^2 \\
& + (1/\beta) \sum_{m,i} U_{m,i} V_{m,i} - (1/\beta) \sum_m \log \left(\sum_i \exp U_{m,i} \right).
\end{aligned} \tag{18}$$

Up to this point the expression is exact; no approximations have been made. Now we approximate both the U and V integrals by way of the saddle points $(\{U^*\},\{V^*\})$ of the objective function E_{eff}:

$$\begin{aligned} \mathrm{argmax}_{\alpha,\mathbf{x}} \Pr{}^f(\alpha,\mathbf{x}|\{\mathbf{x}_i\}) &= \mathrm{argmax}_{\alpha,\mathbf{x}} C \lim_{A\to\infty} \exp -\beta E_{eff}(\alpha,\mathbf{x},\{U^*\},\{V^*\}) \\ &= (\mathrm{argmax}_\alpha \lim_{A\to\infty} \exp -\beta E_{eff}(\alpha,\mathbf{x}^*,\{U^*\},\{V^*\}),\mathbf{x}^*) \end{aligned} \tag{19}$$

where the saddle points satisfy the neural net fixed point equations

$$\begin{aligned} (\partial E_{eff}/\partial U = 0:) \quad V_{m,i} &= \exp U_{m,i} / \sum_j \exp U_{m,j} \\ (\partial E_{eff}/\partial V = 0:) \quad U_{m,i} &= -\beta\left[\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2 + A(\sum_n V_{n,i} - 1)\right] \\ (\partial E_{eff}/\partial \mathbf{x} = 0:) \quad \mathbf{x} &= \frac{1}{N(1+(\sigma_{jt}/\sigma_r)^2)} \sum_{mi} V_{mi}(\mathbf{x}_i - \mathbf{u}_m^\alpha). \end{aligned} \tag{20}$$

Convergent descent dynamics for such networks may be found in (Hopfield, 1984; Peterson and Soderberg, 1989) and many others. The maximization with respect to α can be handled by making one copy of this neural net for each model and adding a winner-take-all circuit.

The method introduces an asymmetry between m and i indices by imposing the $\sum_i P_{m,i} = 1$ constraint exactly but imposing the $\sum_m P_{m,i} = 1$ constraint only in the limit of infinite A. This asymmetry may be removed by *changing variables* before beginning the above calculation: let $P_{m,i} = \sum_a R_{m,a} S_{a,i}$ where R and S are two permutation matrices. Summing over P configurations is equivalent to summing over R and S configurations, for there is a 1-to-$N!$ correspondance. Then impose the $\sum_m$ and $\sum_i$ constraints exactly, as above, or else impose both $\sum_a$ constraints exactly. Either scheme preserves i-m symmetry. To finally reduce the products of continuous versions of the discrete R and S variables to linear form, one may use the objective function transformations of (Mjolsness and Garrett, 1990), e.g.:

$$\sum_{mai} \hat{R}_{ma}\hat{S}_{ai}\mathbf{x}_i \cdot \mathbf{u}_m \to \sum_{ma} \hat{R}_{ma}\mathbf{u}_m \cdot (\boldsymbol{\sigma}_a - \boldsymbol{\tau}_a) + \sum_{ai} \hat{S}_{ai}\mathbf{x}_i \cdot (\boldsymbol{\sigma}_a - \boldsymbol{\omega}_a) + \text{ symmetric potential terms.} \tag{21}$$

The result is a symmetric neural net architecture for the same problem, posessing the same type of Mean Field Theory derivation from a grammar as the does the previous, asymmetric network for approximately summing over P configurations.

2.5 Approximate Neural Network without Match Variables

Short of approximating a P configuration sum via Mean Field Theory and neural nets (Section 2.4 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 (11) and (13),

$$\begin{aligned}
\Pr{}^f(\alpha, \mathbf{x}|\{\mathbf{x}_i\}) &= C \sum_{\left\{\begin{array}{c}\mathrm{P|\ P\ is\ a}\\ \mathrm{permutation}\end{array}\right\}} \exp - \sum_{mi} P_{m,i}\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2\right) \\
&\approx C \sum_{\left\{\begin{array}{c}\mathrm{P|P_{m,i}} \in \{0,1,\ldots \mathrm{N}\}\\ \wedge \sum_{\mathrm{m,i}} \mathrm{P_{m,i}} = \mathrm{N}\end{array}\right\}} \exp - \sum_{mi} P_{m,i}\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2\right) \\
&\approx \frac{C}{N!}\sum_{\{P|\sum P = N\}}\begin{pmatrix} N \\ P_{11}\ldots P_{NN}\end{pmatrix}\prod_{mi}\left[\exp - \left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2\right)\right]^{P_{m,i}} \\
&\qquad \text{(since almost all the multinomials are } = N!) \\
&= C'\left[\textstyle\sum_{m,i}\exp - \left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2\right)\right]^N
\end{aligned} \tag{22}$$

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,

$$\mathrm{argmax}_{\alpha,\mathbf{x}}\Pr(\alpha, \mathbf{x}|\{\mathbf{x}_i\}) \approx \mathrm{argmax}_{\alpha,\mathbf{x}}\sum_{m,i}\exp - \left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathbf{u}_m^\alpha|^2\right). \tag{23}$$

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:

$$
\begin{array}{rl}
& \mathrm{argmin}_{\alpha,\mathbf{x}} \int d\mathbf{z}\, |G * I_1(\mathbf{z}) - G * I_2(\mathbf{z}-\mathbf{x})|^2 \\
= & \mathrm{argmin}_{\alpha,\mathbf{x}} \int d\mathbf{z} \left| G_{\sigma/\sqrt{2}} * \sum_i \delta(\mathbf{z}-\mathbf{x}_i) - G_{\sigma/\sqrt{2}} * \sum_m \delta(\mathbf{z}-\mathbf{x}-\mathbf{u}^{\alpha}_m) \right|^2 \\
= & \mathrm{argmin}_{\alpha,\mathbf{x}} \int d\mathbf{z} \left| \sum_i \exp\left(-\frac{1}{\sigma^2}|\mathbf{z}-\mathbf{x}_i|^2\right) - \sum_m \exp\left(-\frac{1}{\sigma^2}|\mathbf{z}-\mathbf{x}-\mathbf{u}^{\alpha}_m|^2\right)\right|^2 \\
= & \mathrm{argmin}_{\alpha,\mathbf{x}} \left[C_1 - 2\sum_{mi} \int d\mathbf{z} \exp -\frac{1}{\sigma^2}\left[|\mathbf{z}-\mathbf{x}_i|^2 + |\mathbf{z}-\mathbf{x}-\mathbf{u}^{\alpha}_m|^2\right]\right] \\
= & \mathrm{argmax}_{\alpha,\mathbf{x}} \sum_{mi} \int d\mathbf{z} \exp -\frac{1}{\sigma^2}\left[|\mathbf{z}-\mathbf{x}_i|^2 + |\mathbf{z}-\mathbf{x}-\mathbf{u}^{\alpha}_m|^2\right] \\
= & \mathrm{argmax}_{\alpha,\mathbf{x}} \sum_{mi} \exp -\frac{1}{2\sigma^2}|\mathbf{x}_i-\mathbf{x}-\mathbf{u}^{\alpha}_m|^2
\end{array}
\qquad (24)
$$

Furthermore, note that multiplying the objective in (23) 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 (23), 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 (Witkin et al., 1987).

The approximation (22) 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 (Durbin and Willshaw, 1987), which is a less aggressive and probably more accurate approximation in which the sum over all permutation matrices is extended to a sum over all 0/1 matrices with exactly one nonzero element in each row but any number of nonzero entries in each column (Simic, 1990b; Yuille, 1990). Again the sum can be performed exactly. The elastic net's set of allowed matrices are intermediate in generality between permutation matrices (required in the original problem) and the far larger set of nonnegative integer matrices whose elements sum to N, used in our method. Compared to the elastic net, the present objective function is simpler, more symmetric between rows and columns, has a nicer interpretation in terms of known algorithms (correlation in scale space), and is expected to be much less accurate.

A neural net for performing the maximization of (23) with respect to $\mathbf{x}$ has been reported in (Mjolsness and Garrett, 1990). (As mentioned in Section 2.4, the maximization with respect to α can be handled by making one copy of this neural net for each model and adding a winner-take-all circuit.) σ_r

was infinite. The equations of motion were

$$\begin{array}{rcl} \dot{\mathbf{x}} & = & (1/\sigma^2)\sum_{im}\tau_{im}(\mathbf{x}_i - \mathbf{u}_m - \mathbf{x}) \\ \dot{\omega}_{im} & = & -\omega_{im} - (1/2\sigma^2)\sum_{im}|\mathbf{x}_i - \mathbf{u}_m - \mathbf{x}|^2 \\ \tau_{im} & = & \exp\omega_{im}. \end{array} \tag{25}$$

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.

Similar networks can be derived if the grammar includes other distortions such as dot deletion or a global rotation (see Section 4).

The networks of both this and the previous section superficially have space complexity (cost) proportional to NM where $i \in \{1 \ldots N\}$ and $m \in \{1 \ldots M\}$. It may be possible to reduce one of these cost factors, perhaps to a logarithmic term, by using more complicated architectures and approximations.

3 EXPERIMENTS IN IMAGE REGISTRATION

To demonstrate the robustness of the objective function derived in the last section, we tested it outside its formal domain of validity. This kind of robustness is often required of vision algorithms, since their (possibly implicit) mathematical models of the visual problem domain are hardly ever complete. Our grammatical models are intended to make it easier to model heterogeneous noise sources in a complex domain, but robustness would allow the grammars to remain small by modelling just the most important visual phenomena.

Equation (23) is an objective function for recovering the global two-dimensional (2D) translation of a model consisting of arbitrarily placed dots, to match up with similar dots with jittered positions. We use it instead to find the best 2D *rotation* and horizontal translation, for two images which actually differ by a horizontal *3D* translation with roughly constant camera orientation. The images consist of *line segments* rather than single dots, some of which are *missing or extra* data. In addition, there are strong *boundary effects* due to parts of the scene being translated outside the camera's field of view. The jitter is replaced by whatever positional inaccuracies come from an actual camera producing an 128×128 image (Williams and Hanson, 1988) which is then processed by a high quality line-segment finding algorithm (Burns, 1986). Better results would be expected of objective functions derived from grammars which explicitly model more of these noise processes, such as the grammars studied in Section 4.

Since the data to be matched are not dots but line segments, one could alter the grammar and rederive the various objective functions including (23). However one could also consider a line segment to be a dense set of dots (admittedly *not* jittered randomly) and replace the sum over dot pairs in (23) with a sum over line segment pairs, each of which is an integral over dot pairs. For line-line or line-dot matches the integrals can be done exactly (lines are unbounded in both directions). For segment-segment (or segment-dot) matches the integrals can be approximated. First note that

$$\Theta(t) \equiv \begin{cases} 1 & \text{if } t \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{26}$$

$$\Theta(t)\Theta(1-t) \approx \sum_{i=1}^{3} A_i \exp -\frac{1}{2}\frac{(c_i - t)^2}{\sigma_i^2} \tag{27}$$

where by numerical minimization of the Euclidean distance between these two functions of t, the parameters may be chosen as $A_1 = A_3 = 0.800673$, $A_2 = 1.09862$, $\sigma_1 = \sigma_3 = 0.0929032$, $\sigma_2 = 0.237033$, $c_1 = 1 - c_3 = 0.116807$, and $c_2 = 0.5$. Using this approximation, the objective function becomes a double integral along the two line segments:

$$\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \exp -\frac{1}{2}\frac{(\mathbf{q}+t\mathbf{r}-u\mathbf{s})^2}{w^2}\,\Theta(t)\Theta(1-t)\Theta(u)\Theta(1-u)\,dt\,du$$

$$\approx \sum_{i,j} A_i\,A_j \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \exp -\frac{1}{2}\frac{(c_i-t)^2}{\sigma_i^2}\,\exp -\frac{1}{2}\frac{(c_j-u)^2}{\sigma_j^2}\,\exp -\frac{1}{2}\frac{(\mathbf{q}+t\mathbf{r}-u\mathbf{s})^2}{w^2}\,dt\,du. \tag{28}$$

Each of these nine Gaussian integrals can be done exactly. Defining

$$\mathbf{v}_{ij} = \mathbf{q} + c_i\mathbf{r} - c_j\mathbf{s} \tag{29}$$

we have a term of the form

$$\sum_{i,j=1}^{3} A_i\,A_j\,\frac{2\pi w^2\sigma_i\sigma_j}{\sqrt{(w^2+\mathbf{r}^2\sigma_i^2)(w^2+\mathbf{s}^2\sigma_j^2)-\sigma_i^2\sigma_j^2(\mathbf{r}\cdot\mathbf{s})^2}}\,\exp -\frac{1}{2}\frac{\mathbf{v}_{ij}^2 w^2 + (\mathbf{v}_{ij}\times\mathbf{r})^2\sigma_i^2 + (\mathbf{v}_{ij}\times\mathbf{s})^2\sigma_j^2}{(w^2+\mathbf{r}^2\sigma_i^2)(w^2+\mathbf{s}^2\sigma_j^2)-\sigma_i^2\sigma_j^2(\mathbf{r}\cdot\mathbf{s})^2} \tag{30}$$

added to the objective for each pair of line segments, as was calculated by Charles Garrett (Garrett, 1990).

We experimented with minimizing this objective function with respect to unknown global translations and (sometimes) rotations, using the continuation method and sets of line segments derived from real images. The optimization method used at each scale σ was (a) for recovering translation alone, the Conjugate Gradient method with line search; and (b) for recovering translation and rotation, coordinate relaxation by repeated line searches along the x, y, and θ directions, cyclically. The results are shown in Figures 3, 4 and 5.

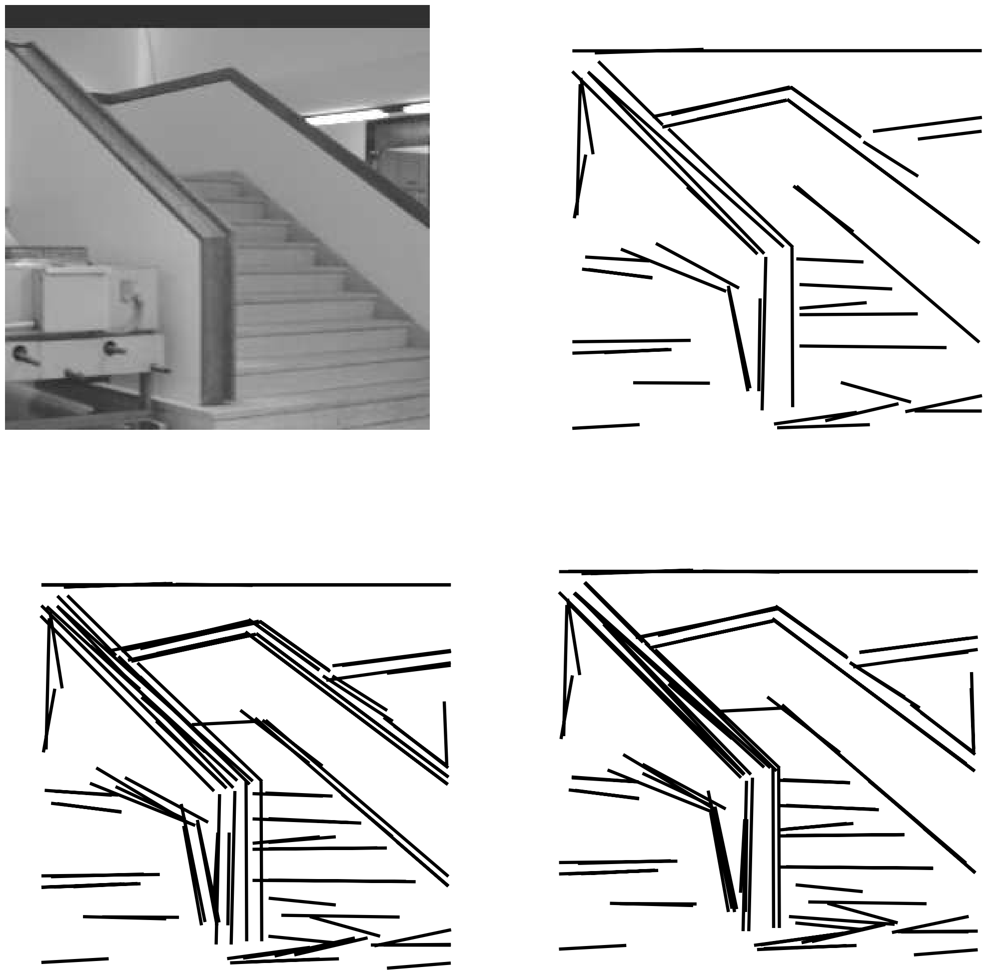


Figure 3: A simple image registration problem. (a) Stair image.(b) Long line segments derived from stair image. (c) Two unregistered line segment images derived from two images taken from two horizontally translated viewpoints in three dimensions. The images are a pair of successive frames in an image sequence. (d) Registered viersions of same data: superposed long line segments extracted from two stair images (taken from viewpoints differing by a small horizontal translation in three dimensions) that have been optimally registered in two dimensions.

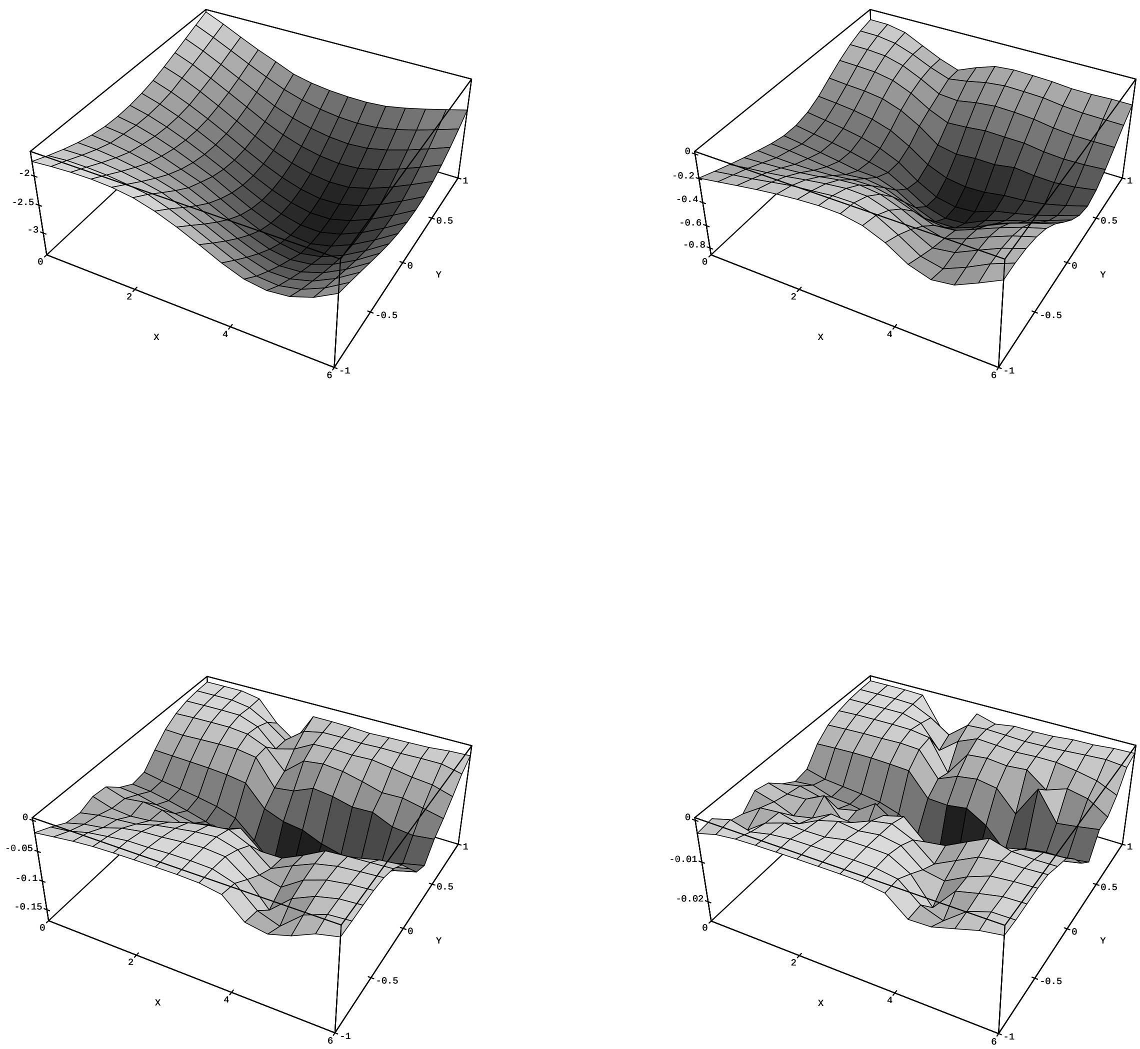


Figure 4: Continuation method (deterministic annealing). (a) Objective function at $\sigma = .0863$. (b) Objective function at $\sigma = .300$. (c) Objective function at $\sigma = .105$. (d) Objective function at $\sigma = .0364$.

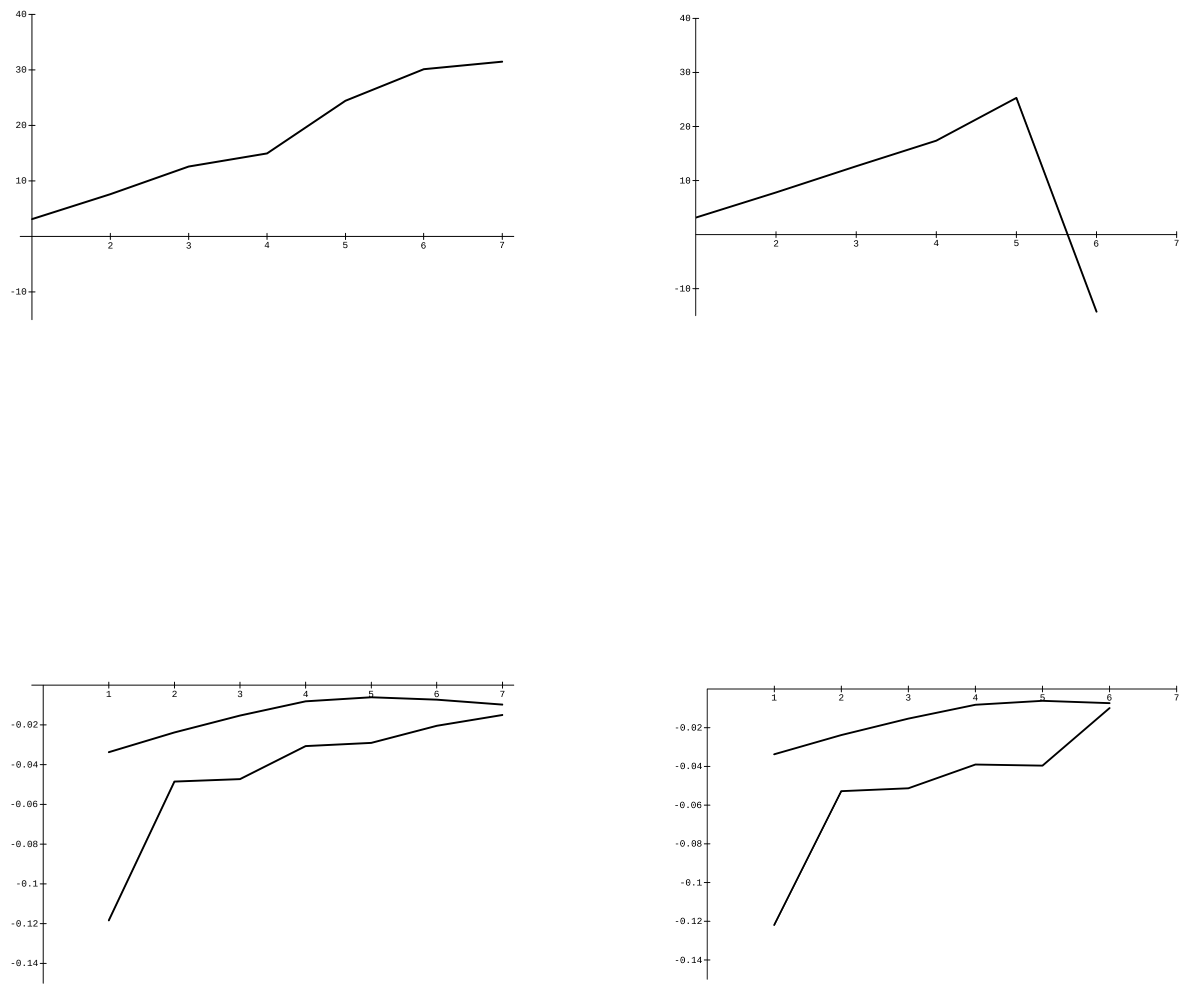


Figure 5: Image sequence displacement recovery. Frame 2 is matched to frames 3-8 in the stair image sequence. Horizontal displacements are recovered. Other starting frames yield similar results except for frame 1, which was much worse. (a) Horizontal displacement recovered, assuming no 2-d rotation. Recovered dispacement as a function of frame number is monotonic. (b) Horizontal displacement recovered, along with 2-d rotation which is found to be small except for the final frame. Displacements are in qualitative agreement with (a), more so for small displacements. (c) Objective function before and after displacement is recovered (upper and lower curves) without rotation. Note gradual decrease in ΔE with frame number (and hence with displacement). (d) Objective function before and after displacement is recovered (upper and lower curves) with rotation.

4 MORE GRAMMARS

To illustrate the generality of the grammatical method for posing vision problems in a form from which neural networks can be derived, we exhibit several grammars of increasing complexity. In Section 4.1 we add rotation and dot deletion as new sources of noise for the random-dot grammar considered in Section 2. Section 4.2 introduces a two-level hierarchy, in which models are sets of clusters of dots. Section 4.3 introduces a grammar for multiple curves in a single image, each of which is represented in the image as a set of dots that may be hard to group into their original curves. This grammar illustrates how flexible objects can be handled in our formalism.

We approach a modest plateau of generality with the grammar of Section 4.4 which again describes hierarchical dot patterns but adds multiple objects in a single scene. This degree of complexity is sufficient to introduce many interesting questions of knowledge representation in high-level vision, such as multiple instances of a model in a scene, as well as requiring segmentation and grouping as part of the recognition process. We prove that 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 permutation matrix as far back into the grammar as possible. This transformation should perhaps be done as a matter of course if one is looking for a modular decomposition of a large vision problem into smaller, more homogeneous pieces to which special methods such as scale-space correlation are most likely to apply.

4.1 A Grammar with 2-D Rotation and Dot Deletion

We can easily add two-dimensional rotations to the previous grammar (and similarly, other parameterized group distortions as well). The grammar, which also adds a dot deletion rule which changes the constraints on P_{im}, is shown below.

model locations	$\Gamma^0:$	root	$\longrightarrow$	instance of model α at $\mathbf{x}$
		$E_0(\mathbf{x})$	$=$	$\frac{1}{2\sigma_r^2}\lvert\mathbf{x}\rvert^2$
rotated, jittered dot locations	$\Gamma^1:$	$\text{instance}(\alpha, \mathbf{x})$	$\longrightarrow$	$\{\text{predot}(\alpha, m, \mathbf{x}_m)\}$
		$E_1(\{\mathbf{x}_c\})$	$=$	$\frac{1}{2\sigma_{jt}^2}\sum_c \lvert\mathbf{x}_m - \mathbf{x} - \mathsf{R}(\theta)\cdot \mathbf{u}_m^\alpha\rvert^2, \quad \text{where} \quad < \mathbf{u}_m^\alpha >_m = 0$
dot deletion	$\Gamma^2:$	$\text{predot}(\alpha, m, \mathbf{x}_m)$	$\longrightarrow$	$\begin{cases} \text{dot}(m, \mathbf{x}_m) & \text{if } \omega_m = 1; \\ \text{nothing} & \text{if } \omega_m = 0. \end{cases}$
		$E_2(\omega_m)$	$=$	$\mu_m \omega_m$
scramble all dots	$\Gamma^3:$	$\{\text{dot}(m, \mathbf{x}_m)\}$	$\longrightarrow$	$\{\text{imagedot}(\mathbf{x}_i = \sum_m P_{m,i}\mathbf{x}_m)\}$
		$E_3(\{\mathbf{x}_i\})$	$=$	$-\log \prod_i \delta(\mathbf{x}_i - \sum_m P_{m,i}\mathbf{x}_m)$
				where $\sum_i P_{m,i} = \omega_m \ \wedge \ \sum_m P_{m,i} = 1$

(31)

The corresponding probability distribution is

$$\Pr{}^f(\alpha, \{\mathbf{x}_i\}) = \frac{1}{A}\left(\frac{1}{\sqrt{2\pi}\sigma_r}\right)^2 \left(\frac{1}{\sqrt{2\pi}\sigma_{jt}}\right)^{2N} \sum_{\left\{ \begin{array}{l} \mathrm{P} \mid \sum_{\mathrm{i}} \mathrm{P}_{\mathrm{m,i}} \leq 1 \\ \text{and} \sum_{\mathrm{m}} \mathrm{P}_{\mathrm{m,i}} = 1 \end{array} \right\}} e^{-\sum_{mi} P_{m,i}\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathsf{R}(\theta)\cdot\mathbf{u}_m^\alpha|^2 + \mu_m\right)}. \tag{32}$$

This is closely related to the objective function recommended for rigid body feature matching in (Yuille, 1990).

As in Section 2.4 one may approximate the maximization of the integrated probability $\Pr^3(\alpha, \mathbf{x}, \theta | \{\mathbf{x}_i\})$ with respect to α, $\mathbf{x}$, and θ via a neural net objective function

$$\begin{aligned} E_{eff}(\alpha, \mathbf{x}, \theta, \{U\}, \{V\}) \equiv & \sum_{m,i} V_{m,i}\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathsf{R}(\theta)\cdot\mathbf{u}_m^\alpha|^2\right) + (A/2)\sum_i\left(\sum_m V_{m,i} - 1\right)^2 \\ & + (1/\beta)\sum_{m,i} U_{m,i}V_{m,i} - (1/\beta)\sum_m \log\left(\sum_i \exp U_{m,i}\right). \end{aligned} \tag{33}$$

Alternatively, as in Section 2.5 one could use an objective function without match variables:

$$E_{eff}(\alpha, \mathbf{x}, \theta) = \sum_{mi} \exp - \left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x} - \mathsf{R}(\theta)\cdot\mathbf{u}_m^\alpha|^2\right). \tag{34}$$

4.2 A Two-Level Random Dot Grammar

A fundamental capability gained by using a grammar is the ability to describe complex objects and scenes. As a first step in this direction, 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 images will generally be clusters of dots with unpredictable jitter of both the dot and the cluster positions. A careful model of such an object is given by this grammar:

model locations	Γ^0 :	root	$\rightarrow$	instance of model α at $\mathbf{x}$
		$E_0(\mathbf{x})$	$=$	$\frac{1}{2\sigma_r^2}\lvert\mathbf{x}\rvert^2$
cluster locations	Γ^1 :	instance$(\alpha, \mathbf{x})$	$\rightarrow$	$\{\text{clusterloc}(\alpha, c, \hat{\mathbf{x}}_c = \mathbf{x} + \mathbf{u}_c^\alpha)\}$
		$E_1(\{\hat{\mathbf{x}}_c\})$	$=$	$-\log \prod_c \delta(\hat{\mathbf{x}}_c - \mathbf{x} - \mathbf{u}_c^\alpha)$, where $< \mathbf{u}_c^\alpha >_c = 0$
cluster jitter	Γ^2 :	clusterloc$(\alpha, c, \hat{\mathbf{x}}_c)$	$\rightarrow$	cluster$(\alpha, c, \mathbf{x}_c)$
		$E_2(\mathbf{x}_c)$	$=$	$\frac{1}{2\sigma_{cd}^2}\lvert\mathbf{x}_c - \hat{\mathbf{x}}_c\rvert^2$
dot locations	Γ^3 :	cluster$(\alpha, c, \mathbf{x}_c)$	$\rightarrow$	$\{\text{dotloc}(\alpha, c, m, \hat{\mathbf{x}}_{cm} = \mathbf{x}_c + \mathbf{u}_{cm}^\alpha)\}$
		$E_3(\{\hat{\mathbf{x}}_{cm}\})$	$=$	$-\log \prod_m \delta(\hat{\mathbf{x}}_{cm} - \mathbf{x} - \mathbf{u}_{cm}^\alpha)$, where $< \mathbf{u}_{cm}^\alpha >_m = 0$
dot jitter	Γ^4 :	dotloc$(\alpha, c, m, \hat{\mathbf{x}}_{cm})$	$\rightarrow$	dot$(c, m, \mathbf{x}_{cm})$
		$E_4(\hat{\mathbf{x}}_{cm})$	$=$	$\frac{1}{2\sigma_{jt}^2}\lvert\mathbf{x}_{cm} - \hat{\mathbf{x}}_{cm}\rvert^2$
scramble all dots	Γ^5 :	$\{\text{dot}(c, m, \mathbf{x}_{cm})\}$	$\rightarrow$	$\{\text{imagedot}(\mathbf{x}_i = \sum_{cm} P_{cm,i}\mathbf{x}_{cm})\}$
		$E_5(\{\mathbf{x}_i\})$	$=$	$-\log \Pr(P) \prod_i \delta(\mathbf{x}_i - \sum_{cm} P_{cm,i}\mathbf{x}_{cm})$
				where $\sum_i P_{m,i} = 1 \;\wedge\; \sum_m P_{m,i} = 1$

(35)

which is equivalent, by integrating out $\hat{\mathbf{x}}_c$ and $\hat{\mathbf{x}}_{cm}$, to a simpler grammar:

model locations	Γ^0 :	root	$\rightarrow$	instance of model α at $\mathbf{x}$
		$E_0(\mathbf{x})$	$=$	$\frac{1}{2\sigma_r^2}\lvert\mathbf{x}\rvert^2$
jittered cluster locations	Γ^1 :	instance$(\alpha, \mathbf{x})$	$\rightarrow$	$\{\text{cluster}(\alpha, c, \mathbf{x}_c)\}$
		$E_1(\{\mathbf{x}_c\})$	$=$	$\frac{1}{2\sigma_{cd}^2}\sum_c \lvert\mathbf{x}_c - \mathbf{x} - \mathbf{u}_c^\alpha\rvert^2$, where $< \mathbf{u}_c^\alpha >_c = 0$
jittered dot locations	Γ^2 :	cluster$(\alpha, c, \mathbf{x}_c)$	$\rightarrow$	$\{\text{dot}(c, m, \mathbf{x}_{cm})\}$
		$E_2(\{\mathbf{x}_{cm}\})$	$=$	$\frac{1}{2\sigma_{jt}^2}\sum_m \lvert\mathbf{x}_{cm} - \mathbf{x}_c - \mathbf{u}_{cm}^\alpha\rvert^2$, where $< \mathbf{u}_{cm}^\alpha >_m = 0$
scramble all dots	Γ^3 :	$\{\text{dot}(c, m, \mathbf{x}_{cm})\}$	$\rightarrow$	$\{\text{imagedot}(\mathbf{x}_i = \sum_{cm} P_{cm,i}\mathbf{x}_{cm})\}$
		$E_3(\{\mathbf{x}_i\})$	$=$	$-\log \prod_i \delta(\mathbf{x}_i - \sum_{cm} P_{cm,i}\mathbf{x}_{cm})$
				where $\sum_i P_{m,i} = 1 \;\wedge\; \sum_m P_{m,i} = 1$

(36)

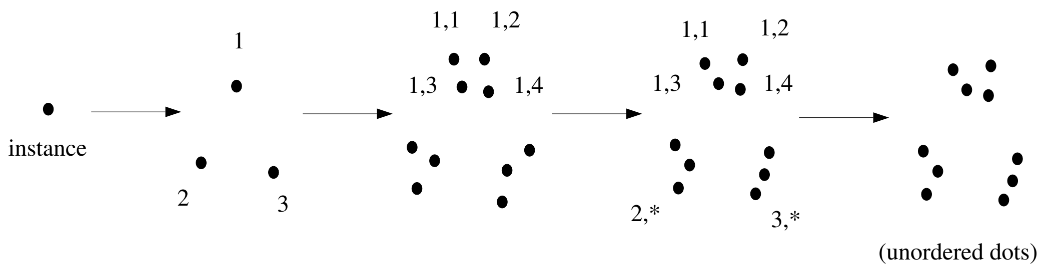


Figure 6: Operation of two-level random dot grammar. The first arrow illustrates cluster placement and cluster jitter; the next shows dot placement; the next shows dot jitter; and the final arrow is the scrambling or relabelling operation.

The corresponding probability distribution is:

$$\begin{aligned}
\Pr{}^3(\alpha, \mathbf{x}, \{\mathbf{x}_c\}, \{\mathbf{x}_i\}) \;=\; & \frac{1}{A}\left(\frac{1}{\sqrt{2\pi}\sigma_r}\right)^2 \left(\frac{1}{\sqrt{2\pi}\sigma_{cd}}\right)^{2C} \left(\frac{1}{\sqrt{2\pi}\sigma_{jt}}\right)^{2N} \sum_{\left\{ \begin{array}{c} \mathrm{P|\ P\ is\ a} \\ \mathrm{permutation} \end{array} \right\}} \\
& e^{-\sum_{cmi} P_{cm,i}\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{C}{2N\sigma_{cd}^2}|\mathbf{x}_c - \mathbf{x} - \mathbf{u}_c^\alpha|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x}_c - \mathbf{u}_{cm}^\alpha|^2\right)}
\end{aligned} \quad (37)$$

where C is the number of clusters and N/C is the number of dots in each cluster. As in Section 2.4 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

$$\begin{aligned}
E_{eff}(\alpha, \mathbf{x}, \{\mathbf{x}_c\}, \{U\}, \{V\}) \;\equiv\; & \sum_{cm,i} V_{cm,i}\left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{C}{2N\sigma_{cd}^2}|\mathbf{x}_c - \mathbf{x} - \mathbf{u}_c^\alpha|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x}_c - \mathbf{u}_{cm}^\alpha|^2\right) \\
& +(A/2)\sum_i (\sum_{cm} V_{cm,i} - 1)^2 + (1/\beta)\sum_{cm,i} U_{cm,i}V_{cm,i} \\
& -(1/\beta)\sum_{cm} \log\left(\sum_i \exp U_{cm,i}\right).
\end{aligned} \quad (38)$$

Alternatively, as in Section 2.5 one could use an objective function without match variables:

$$E_{eff}(\alpha, \mathbf{x}, \{\mathbf{x}_c\}) = \sum_{cmi} \exp - \left(\frac{1}{2N\sigma_r^2}|\mathbf{x}|^2 + \frac{C}{2N\sigma_{cd}^2}|\mathbf{x}_c - \mathbf{x} - \mathbf{u}_c^\alpha|^2 + \frac{1}{2\sigma_{jt}^2}|\mathbf{x}_i - \mathbf{x}_c - \mathbf{u}_{cm}^\alpha|^2 \right). \tag{39}$$

Intermediate designs result from *changing variables* by separately considering the cluster and the member to which a data item is assigned: $P_{cm,i} = P^1_{ci} P^2_{mi}$. 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, and supposing that σ_{cd} is not small with respect to the variance of cluster locations $\mathbf{u}_c^\alpha$.

4.3 Multiple Curves

This problem involves perceptual organization: one must extract multiple curves from a random-dot pattern. The grammar generates such a pattern by sequentially generating a number of curves that start with random locations and directions. Then, for each curve, the grammar sequentially generates new dot locations and curve directions according to a Markov process which favors continuity. The final picture consists of all the dots generated. This grammar is "tail-recursive", that is, a rule in the grammar can replace a term by just one term of the same type.

make set of curves	$\Gamma^0: \quad \text{root} \to \text{curveset}(0)$ $E_0 = 0$ No alternatives $\Rightarrow$ certainty.
extend curve set	$\Gamma^1: \quad \text{curveset}(c) \to \begin{cases} \{\text{curveset}(c+1), \text{curve}(c+1, s=1, \mathbf{x}, \theta)\} & \text{if } \omega_c = 1; \\ \text{nothing} & \text{if } \omega_c = 0. \end{cases}$ $E_1(\mathbf{x}) = \mu\omega_c + \frac{1}{2\sigma_0^2}|\mathbf{x}|^2$
extend curve by one dot	$\Gamma^2: \quad \text{curve}(c, s, \mathbf{x}, \theta) \to \begin{cases} \{\text{curve}(c, s+1, \mathbf{x}', \theta'), \text{dot}(c, s, \mathbf{x}, \theta)\} & \text{if } \omega_{cs} = 1; \\ \text{dot}(c, s, \mathbf{x}, \theta)\} & \text{if } \omega_{cs} = 0. \end{cases}$ $E_2(\mathbf{x}, \theta, \mathbf{x}', \theta') = \nu\omega_{cs} + e_2(\mathbf{x} - \mathbf{x}', \theta - \theta')$, where $e_2(\mathbf{\Delta x}, \Delta\theta) \equiv \frac{1}{2\sigma_\theta^2}\left(\arctan\left(\frac{\Delta x_2}{\Delta x_1}\right) - \Delta\theta\right)^2 + \frac{1}{2\sigma_r^2}\left(|\mathbf{\Delta x}|^2 - l^2\right)^2 + \frac{1}{2\sigma_{\text{bend}}^2}(\Delta\theta)^2$
scramble all dots	$\Gamma^3: \quad \{\text{dot}(c, s, \mathbf{x}_{cs})\} \to \{\text{imagedot}(\mathbf{x}_i = \sum_{cs} P_{cs,i}\mathbf{x}_{cs})\}$ $E_3(\{\mathbf{x}_i\}) = -\log \sum_{\left\{ \begin{array}{l} P \mid \sum_i P_{cs,i} = A_{cs} \\ \text{and} \sum_{cs} P_{cs,i} = 1 \end{array} \right\}} \Pr(P) \prod_i \delta(\mathbf{x}_i - \sum_{cs} P_{cs,i}\mathbf{x}_{cs})$

(40)

This grammar may be compared to the somewhat different curve grammars of (Milios, 1989).

In (Mjolsness et al., 1991a) we show that this grammar has the joint probability distribution

$$\Pr(\mathbf{x}_i, \theta_i, P_{cs,i}, C, (S_c, c = 1, .., C)|N) = (1 - q_1)\left[\frac{q_1(1-q_2)}{Z_1}\right]^C \left[\frac{q_2}{Z_2}\right]^N \exp(-\beta E(\{P\}, \{\mathbf{x}_i\}, \{\theta_i\})) \tag{41}$$

Table 1: Syntactical Constraints for Multiple Curves

Syntactical Constraints	Expressions
Total number of curves= C	$C = N - \sum_{i=1}^{N}\sum_{j=1}^{N} \mathrm{next}_{ij}$
No-loop constraint	$\sum_{i=1}^{N} \mathrm{next}_{ij}\mathrm{mbr}_i = \mathrm{mbr}_j - 1$
Start-element constraint	$\sum_{c=1}^{C} \mathrm{start}_{cj} = 1 - \sum_{i=1}^{N} \mathrm{next}_{ij}$
End-element constraint	$\sum_{j=1}^{N} \mathrm{next}_{ij} \leq 1$
Presence-absence constraint	$\{\mathrm{next}_{ij}\}, \{\mathrm{start}_{ci}\} \in \{0,1\}$ and $\{\mathrm{mbr}_i\} \in \{1,..,N\}$

where Z_1 and Z_2 are normalization constants and

$$E(\{P\},\{\mathbf{x}_i\},\{\theta_i\}) = \sum_{i=1}^{N}\left(\sum_{c=1}^{C} P_{c1,i}\right) E_1(\mathbf{x}_i) + \sum_{i=1}^{N}\sum_{j=1}^{N}\left(\sum_{c=1}^{C}\sum_{s=1}^{S_c-1} P_{cs,i}P_{c(s+1),j}\right) E_2(\mathbf{x}_j - \mathbf{x}_i, \theta_i, \theta_j - \theta_i) \quad (42)$$

The notation is as follows: $c = 1,..,C$ is the curve index, $s = 1,..,S_c$ is the dot index along a curve, S_c is the number of dots in curve c. The image dot locations and orientations are $\{\mathbf{x}_i\}$ and $\{\theta_i\}$. Also $q_1 = \exp(\mu)$, $q_2 = \exp(\nu)$, C is the number of curves, $P_{cs,i}$ is the permutation matrix introduced in Rule 3 of the grammar and N is the number of perceived image dots. The distribution function contains the free parameter β corresponding to the inverse of a temperature. β may by varied in a deterministic annealing process.

The energy function in equation (42) can be further simplified by a suitable change of variables. Consider the following transformations.

$$\mathrm{next}_{ij} = \sum_{c=1}^{C}\sum_{s=1}^{S_c-1} P_{cs,i}P_{c(s+1),j}, \;\; \mathrm{start}_{ci} = P_{c1,i}, \;\; \mathrm{mbr}_i = \sum_{c=1}^{C}\sum_{s=1}^{S_c} sP_{cs,i} \quad (43)$$

The choice of the new variables is not arbitrary. $\{\mathrm{next}_{ij}\}$ tracks the membership of the data elements i and j in the same curve with the constraint that j follows i as the *next member* in the chain. $\{\mathrm{start}_{ci}\}$ turns on, i.e. $\mathrm{start}_{ci} = 1$, if i is the starting element of curve c. $\{\mathrm{mbr}_i\}$ reindexes the data element i in terms of its membership number (mbr) in a curve. The membership number of the starting element in any curve c is one and for the last element in the chain, it is S_c.

The constraints needed to adequately characterize the problem undergo a transformation as well. They are given in Table 1.

In terms of the new variables, the joint probability distribution becomes

$$\Pr(\{\theta_i\}, \{\text{next}_{ij}\}, \{\text{start}_{ci}\}, \{\text{mbr}_i\} | \{\mathbf{x}_i\}, N) = \frac{1}{Z}\left[\exp\left(\left\{N - \sum_{i=1}^{N}\sum_{j=1}^{N}\text{next}_{ij}\right\}\log G - \beta E(\{\text{next}_{ij}\}, \{\mathbf{x}_i\}, \{\theta_i\})\right)\right]$$

where $G = \frac{q_1(1-q_2)}{(2\pi\sigma_0^2)}$ and

$$E(\{\text{next}_{ij}\}, \{\mathbf{x}_i\}, \{\theta_i\}) = \sum_{i=1}^{N}\left(1 - \sum_{j=1}^{N}\text{next}_{ji}\right)E_1(\mathbf{x}_i) + \sum_{i=1}^{N}\sum_{j=1}^{N}\text{next}_{ij}E_2(\mathbf{x}_j - \mathbf{x}_i, \theta_i, \theta_j - \theta_i). \tag{44}$$

This objective function may be transformed to a neural network as in Section 2.4, resulting in a network analogous the the Traveling Salesman network of (Hopfield and Tank, 1985) which, because of the change of variables, has the advantage that a curve can change "phase" (s-numbering as specified by $\{\text{mbr}\}$) gradually and locally as the network runs, without changing the curve's connectedness (as specified by $\{\text{next}\}$).

4.4 Frameville from a Grammar

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. Just as "perceptrons" were originally intended to be minimal models of percepts, related by parameterized interconnections, one might try to invent a more abstract computational unit to model small concepts. Such a "conceptron" could only result from the combined action of many perceptrons or artificial neurons, and in this way would be a collective, large-scale phenomenon in a neural network. A conceptron 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; (d) could enter into part-whole hierarchies with other conceptrons; (e) could enter into geometric relationships with other conceptrons; (f) could enter into generalization and specialization relationships with other conceptrons; and so forth.

The goal of the "Frameville" type of neural network architecture (Mjolsness et al., 1989; Anandan et al., 1989) is to satisfy such constraints in much the way they can be satisfied within a frame system as used in Artificial Intelligence programming (Fahlman, 1979), 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 $INA_{\alpha\beta} \in \{0, 1\}$:

$$E = \sum_{\alpha\beta} \sum_{ij} INA_{\alpha\beta} ina_{ij} M_{\alpha i} M_{\beta j} H^{\alpha\beta}(\mathbf{F}_i, \mathbf{F}_j) \tag{45}$$

subject to constraints including

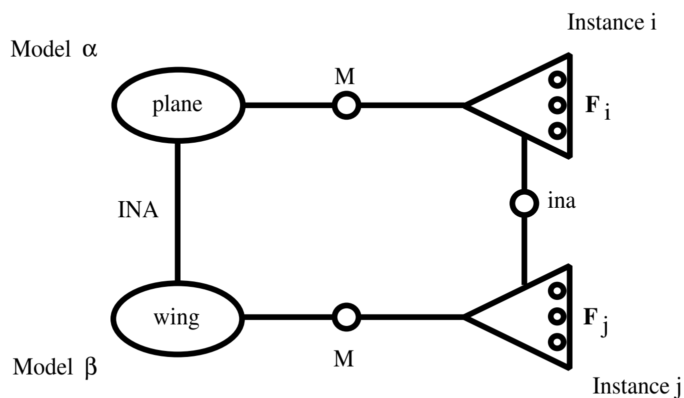
$$\begin{array}{rclr} \sum_\alpha INA_{\alpha\beta} M_{\alpha i} & = & \sum_j ina_{ij} M_{\beta j} & (a) \\ \sum_\beta INA_{\alpha\beta} M_{\beta j} & = & \sum_i ina_{ij} M_{\alpha i} & (b). \end{array} \tag{46}$$

(See figure 7.) Here α or β index the "frame", which could also be called the "object model", "prototype object", or "conceptron", 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 (Zemel, 1989). Other networks are related to Frameville by virtue of the use of graph-matching or arrays of match neurons for visual object recognition (von der Malsburg and Bienenstock, 1986; von der Malsburg, 1988; Cooper, 1989; Feldman et al., 1988; Bienenstock and Doursat, 1991) or using objective functions for high-level knowledge representation (Derthick, 1988; Stolcke, 1989).

4.4.1 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 below.



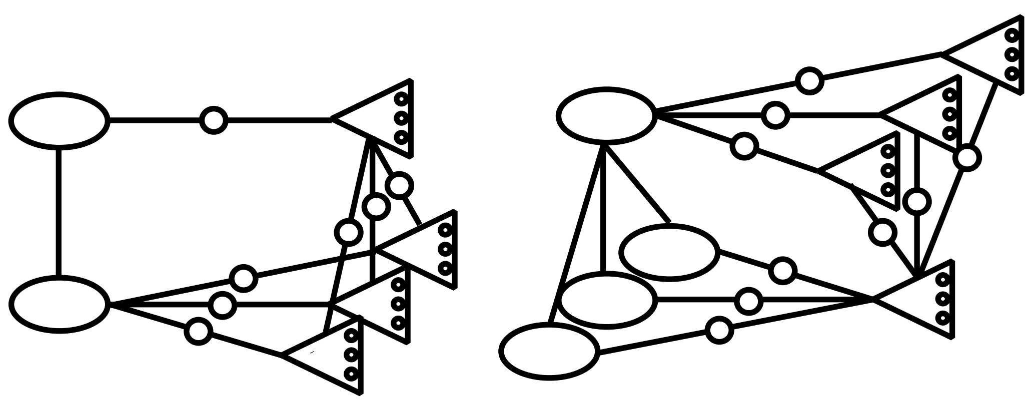


Figure 7: Frameville neural network. (a) The objective function, $E = \sum_{\alpha\beta} \sum_{ij} INA_{\alpha\beta} ina_{ij} M_{\alpha i} M_{\beta j} \times H^{\alpha\beta}(\mathbf{F}_i, \mathbf{F}_j)$. Circles are neurons, ovals are models (or frames) and triangles are model (frame) instances containing 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.

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

(47)

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

$$A_{a\alpha s_2 s_3} = C_{a\alpha} INA_{\alpha, s_2} INA_{\alpha s_2, s_3} \omega_{a\alpha s_2 s_3} \tag{48}$$

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 (45); this however is a much less substantive restriction.

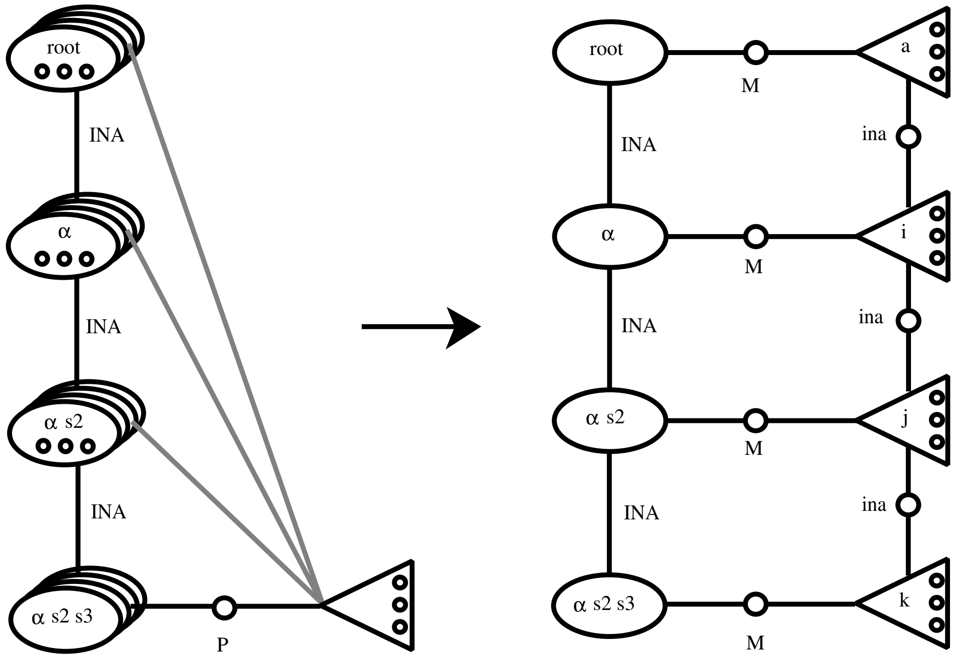


Figure 8: Change of variables, and the corresponding change in objective function, from global permutation variables P to local correspondance variables M and grouping variables ina. Note that analog parameters move from the models (where they must be present in multiple copies) to the instances. Left of arrow: objective of equation (50). Right of arrow: objective of equation (45) or (92).

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

$$\Pr^f(\{\mathbf{x}_{a\alpha}\},\{\mathbf{x}_{a\alpha s_2}\},\{C_{a\alpha}\}|\{\mathbf{x}_k\}) = \frac{1}{Z}\sum_{\left\{\begin{array}{l} P| \ \sum_k P_{a\alpha s_2 s_3,k} \leq 1 \ \wedge \ \sum_{a\alpha s_2 s_3} P_{a\alpha s_2 s_3,k} \leq 1 \\ \quad \wedge \ P_{a\alpha s_2 s_3,k} = P_{a\alpha s_2 s_3,k} C_{a\alpha} INA_{\alpha,s_2} INA_{\alpha s_2,s_3} \end{array}\right\}} \exp -\hat{\beta} E(\{\mathbf{x}_{a\alpha}\},\{\mathbf{x}_{a\alpha s_2}\},\{C_{a\alpha}\},\{\mathbf{x}_k\}) \tag{49}$$

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

$$\begin{aligned} E(\ldots) &= \sum_{a\alpha s_2 s_3}\sum_k P_{a\alpha s_2 s_3,k}\left[H^{\alpha s_2 s_3}(\mathbf{x}_{a\alpha s_2},\mathbf{x}_k) - \mu_{\text{extra}} - \mu^{\alpha s_2 s_3}\right] \\ &\quad + \sum_{a\alpha s_2} C_{a\alpha} H^{\alpha s_2}(\mathbf{x}_{a\alpha},\mathbf{x}_{a\alpha s_2}) + \sum_{a\alpha} C_{a\alpha} H^{\text{root}\alpha}(\mathbf{x}_{a\alpha}). \end{aligned} \tag{50}$$

This objective is illustrated in figure 8(a).

4.4.2 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:

$$P_{a\alpha s_2 s_3,k} = M_{\alpha s_2 s_3,k} \hat{ina}_{a\alpha s_2,k} \tag{51}$$

where

$$M_{\alpha s_2 s_3,k} = \sum_a P_{a\alpha s_2 s_3,k} \quad \text{and} \quad \hat{ina}_{a\alpha s_2,k} = \sum_{s_3} P_{a\alpha s_2 s_3,k}. \tag{52}$$

The constraints on P are consistently translated into new constraints on M and $\hat{ina}$ by Lemma 1.

Lemma 1. The following two conjunctions ($a - e$ and $a' - g'$) are equivalent for 0/1 variables P, M, and $\hat{ina}$:

$$\begin{array}{lll|lll}
(a) & & M_{\alpha s_2 s_3,k} = \sum_a P_{a\alpha s_2 s_3,k} & (a') & & P_{a\alpha s_2 s_3,k} = M_{\alpha s_2 s_3,k}\hat{ina}_{a\alpha s_2,k} \\
(b) & \wedge & \hat{ina}_{a\alpha s_2,k} = \sum_{s_3} P_{a\alpha s_2 s_3,k} & (b') & \wedge & \sum_{s_3} M_{\alpha s_2 s_3,k} = \sum_a \hat{ina}_{a\alpha s_2,k} \\
(c) & \wedge & \sum_k P_{a\alpha s_2 s_3,k} \leq 1 & (c') & \wedge & \sum_k M_{\alpha s_2 s_3,k}\hat{ina}_{a\alpha s_2,k} \leq 1 \\
(d) & \wedge & \sum_{a\alpha s_2 s_3} P_{a\alpha s_2 s_3,k} \leq 1 & (d') & \wedge & \sum_{\alpha s_2 s_3} M_{\alpha s_2 s_3,k} \leq 1 \\
 & & & (e') & \wedge & \sum_{a\alpha s_2} \hat{ina}_{a\alpha s_2,k} \leq 1 \\
(e) & \wedge & P_{a\alpha s_2 s_3,k} = P_{a\alpha s_2 s_3,k} C_{a\alpha} INA_{\alpha,s_2} INA_{\alpha s_2,s_3} & (f') & \wedge & \hat{ina}_{a\alpha s_2,k} = \hat{ina}_{a\alpha s_2,k} C_{a\alpha} INA_{\alpha,s_2} \\
 & & & (g') & \wedge & M_{\alpha s_2 s_3,k} = M_{\alpha s_2 s_3,k} INA_{\alpha s_2,s_3}
\end{array} \tag{53}$$

Proof:

$\Rightarrow$:

(a') :

1. $P_{a\alpha s_2 s_3,k} = (P_{a\alpha s_2 s_3,k})^2$ (since $P \in \{0,1\}$) $\leq (\sum_a P_{a\alpha s_2 s_3,k})(\sum_{s_3} P_{a\alpha s_2 s_3,k}) = M_{\alpha s_2 s_3,k}\hat{ina}_{a\alpha s_2,k}$ (by (a, b)).

2. $\sum_{a s_3} P_{a\alpha s_2 s_3,k} \leq \sum_{a\alpha s_2 s_3} P_{a\alpha s_2 s_3,k} \leq 1$ (by (d)).

3. $\sum_{a s_3} M_{\alpha s_2 s_3,k}\hat{ina}_{a\alpha s_2,k} = \sum_{a s_3} \sum_{a' s_3'} P_{a'\alpha s_2 s_3,k} P_{a\alpha s_2 s_3',k}$ (by (a, b)) $= (\sum_{a s_3} P_{a\alpha s_2 s_3,k})^2$ $= \sum_{a s_3} P_{a\alpha s_2 s_3,k}$ (by (2)).

4. $\sum_{a\alpha s_2 s_3 k} |P_{a\alpha s_2 s_3,k} - M_{\alpha s_2 s_3,k}\hat{ina}_{a\alpha s_2,k}| = \sum_{a\alpha s_2 s_3 k}(M_{\alpha s_2 s_3,k}\hat{ina}_{a\alpha s_2,k} - P_{a\alpha s_2 s_3,k})$ (by (1)) $= 0$ (by (3)).

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

(b') : $\sum_{s_3} M_{\alpha s_2 s_3,k} = \sum_{a s_3} P_{a\alpha s_2 s_3,k}$ (by (a)) $= \sum_a \hat{ina}_{a\alpha s_2,k}$ (by (b)).

(c') : $\sum_k M_{\alpha s_2 s_3,k}\hat{ina}_{a\alpha s_2,k} = \sum_k P_{a\alpha s_2 s_3,k}$ (by (a')) ≤ 1 (by (c)).

(d') : $\sum_{a s_2 s_3} M_{\alpha s_2 s_3,k} = \sum_{a\alpha s_2 s_3} P_{a\alpha s_2 s_3,k}$ (by (a)) ≤ 1 (by (d)).

(e') : $\sum_{a\alpha s_2} \hat{ina}_{a\alpha s_2,k} = \sum_{a\alpha s_2 s_3} P_{a\alpha s_2 s_3,k}$ (by (b)) ≤ 1 (by (d)).

(f') : $\hat{ina}_{a\alpha s_2,k} = \sum_{s_3} P_{a\alpha s_2 s_3,k}$ (by (b)) $= C_{a\alpha} INA_{\alpha,s_2} \sum_{s_3} INA_{\alpha s_2,s_3} P_{a\alpha s_2 s_3,k}$ (by (e)) $= (C_{a\alpha})^2 (INA_{\alpha,s_2})^2 \sum_{s_3} INA_{\alpha s_2,s_3} P_{a\alpha s_2 s_3,k} = C_{a\alpha} INA_{\alpha,s_2} \sum_{s_3} P_{a\alpha s_2 s_3,k}$ (by (e)) $= C_{a\alpha} INA_{\alpha,s_2} \hat{ina}_{a\alpha s_2,k}$ (by (b)).

(g') : $M_{\alpha s_2 s_3,k} = \sum_a P_{a\alpha s_2 s_3,k}$ (by (b)) $= INA_{\alpha s_2,s_3} \sum_a P_{a\alpha s_2 s_3,k}$ (by (e)) $= M_{\alpha s_2 s_3,k} INA_{\alpha s_2,s_3}$ (by (b)).

$\Leftarrow$:

(a) : $\sum_a P_{a\alpha s_2 s_3,k} = M_{\alpha s_2 s_3,k} \sum_a \hat{ina}_{a\alpha s_2,k}$ (by (a')) $= M_{\alpha s_2 s_3,k}(\sum_{s_3'} M_{\alpha s_2 s_3',k})$ (by (b')) $= M_{\alpha s_2 s_3,k}$ (since $\sum_{s_3'} M_{\alpha s_2 s_3',k} \leq 1$ by (d')).

(b) : $\sum_{s_3} P_{a\alpha s_2 s_3,k} = \hat{ina}_{a\alpha s_2,k} \sum_{s_3} M_{\alpha s_2 s_3,k}$ (by (a')) $= \hat{ina}_{a\alpha s_2,k} \sum_{a'} \hat{ina}_{a'\alpha s_2,k}$ (by (b')) $= \hat{ina}_{a\alpha s_2,k}$ (by (e')).

(c) : $\sum_k P_{a\alpha s_2 s_3,k} = \sum_k M_{\alpha s_2 s_3,k}\hat{ina}_{a\alpha s_2,k}$ (by (a')) ≤ 1 (by (c')).

(d) : $\sum_{a\alpha s_2 s_3} P_{a\alpha s_2 s_3,k} = \sum_{\alpha s_2}(\sum_{s_3} M_{\alpha s_2 s_3,k})(\sum_a \hat{ina}_{a\alpha s_2,k})$ (by (a')) $= \sum_{\alpha s_2}(\sum_{s_3} M_{\alpha s_2 s_3,k})^2$ (by (b')) $= \sum_{\alpha s_2 s_3} M_{\alpha s_2 s_3,k}$ (by (d')) ≤ 1 (by (d')).

(e) : $P_{a\alpha s_2 s_3,k} = M_{\alpha s_2 s_3,k}\hat{ina}_{a\alpha s_2,k}$ (by (a')) $= M_{\alpha s_2 s_3,k} INA_{\alpha s_2,s_3} \hat{ina}_{a\alpha s_2,k} C_{a\alpha} INA_{\alpha,s_2}$ (by (f', g')) $= P_{a\alpha s_2 s_3,k} C_{a\alpha} INA_{\alpha,s_2} INA_{\alpha s_2,s_3}$ (by (a')).

QED.

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 $P_{a\alpha s_2,j}$:

$$\hat{ina}_{a\alpha s_2,k} = \sum_j ina_{jk} P_{a\alpha s_2,j}. \tag{54}$$

The resulting change of variables is illustrated in figure 8. When augmented by the constraints stated in Lemma 2 below, this change of variables 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, as we will see.

To state the new constraints it is necessary to account for the redundancy inherent in the assignment of j indices to groupings of data indexed by k (i.e. the freedom to permute j in ina_{jk}). To this end, we will define a fixed mapping $R_{a\alpha s_2,j}$ (consistent with $C_{a\alpha}$) from possible copies of high level models (i.e. tuples (a, α, s_2)) to high level frame instances (j), then permute the j's with a matrix Q in order to get $P_{a\alpha s_2,k}$. For example R could be chosen as follows. The number of nonzero $C_{a\alpha} INA_{\alpha s_2}$ entries is $\sum_{a\alpha s_2} C_{a\alpha} INA_{\alpha s_2}$. Lexicographically order these, and let $R_{a\alpha s_2,j} = 1 \iff j$ indexes $(a\alpha s_2)$, for which $C_{a\alpha} INA_{a\alpha s_2} = 1$; otherwise $R = 0$. Then

$$\begin{array}{ll} (r_1) & \sum_j R_{a\alpha s_2,j} = C_{a\alpha} INA_{\alpha,s_2} \\ (r_2) & \sum_{a\alpha s_2} R_{a\alpha s_2,j} \leq 1 \\ (r_3) & R_{a\alpha s_2,j} = R_{a\alpha s_2,j} C_{a\alpha} INA_{\alpha,s_2}. \end{array} \tag{55}$$

Note that (r_2) can actually be refined as follows: due to the lexicographical ordering of $(a\alpha s_2)$ by R, mapping such tuples to index j, we have

$$\begin{aligned} \sum_{a\alpha s_2} R_{a\alpha s_2,j} &= \begin{cases} 1 & \text{if } 1 \leq j \leq n^{(2)}(C, INA) \\ 0 & \text{otherwise} \end{cases} \\ &= \Theta(j-1)\Theta(n^{(2)}(C, INA) - j) \end{aligned} \tag{56}$$

where as before $\Theta(x) \equiv 1$ if $x \geq 0$ and $\Theta(x) \equiv 0$ if $x < 0$. Also $n^{(2)}(C, INA)$ is the number of allowed tuples, namely

$$n^{(2)}(C, INA) = \sum_{a\alpha s_2 j} R_{a\alpha s_2,j} = \sum_{a\alpha s_2} C_{a\alpha} INA_{\alpha,s_2}. \tag{57}$$

For this function $R(C, INA)$ we can now prove

Lemma 2. There is a 0/1-valued function $R_{a\alpha s_2,j}(C_{a\alpha} INA_{\alpha,s_2})$ for which the following two conjunctions ($a-g$ and $a'-h'$) are equivalent for 0/1 variables $\hat{ina}$, Q, P, ina, and M:

$$
\begin{array}{lll|lll}
(a) & & P_{a\alpha s_2,j} = \sum_{j'} R_{a\alpha s_2,j'} Q_{jj'}, & (a') & & \hat{ina}_{a\alpha s_2,k} = \sum_j ina_{jk} P_{a\alpha s_2,j} \\
(b) & \wedge & Q \text{ is an embedded permutation, i.e. :} & (b') & & \sum_j ina_{jk} \leq 1 \\
 & & \begin{array}{lll} (b_1) & \wedge & \sum_j Q_{j'j} \leq 1 \\ (b_2) & \wedge & \sum_{j'} Q_{j'j} = \sum_{a\alpha s_2} R_{a\alpha s_2,j} (\leq 1) \\ (b_3) & \wedge & \sum_{j'j} Q_{j'j} = \sum_{a\alpha s_2} C_{a\alpha} INA_{\alpha,s_2} \end{array} & & & \\
(c) & \wedge & ina_{jk} = \sum_{a\alpha s_2} \hat{ina}_{a\alpha s_2,k} P_{a\alpha s_2,j} & (c') & \wedge & ina_{jk} = ina_{jk} \sum_{a\alpha s_2} P_{a\alpha s_2,j} \\
 & & & (d') & \wedge & \sum_j P_{a\alpha s_2,j} = C_{a\alpha} INA_{\alpha s_2} \\
(d) & \wedge & \sum_{a\alpha s_2} \hat{ina}_{a\alpha s_2,k} \leq 1 & (e') & \wedge & \sum_{a\alpha s_2} P_{a\alpha s_2,j} \leq 1 \\
(e) & \wedge & \hat{ina}_{a\alpha s_2,k} = \hat{ina}_{a\alpha s_2,k} C_{a\alpha} INA_{\alpha s_2} & (f') & \wedge & P_{a\alpha s_2,j} = P_{a\alpha s_2,j} C_{a\alpha} INA_{\alpha s_2} \\
(f) & \wedge & \sum_k M_{\alpha s_2 s_3,k} \hat{ina}_{a\alpha s_2,k} \leq 1 & (g') & \wedge & \sum_k M_{\alpha s_2 s_3,k} \sum_j ina_{jk} P_{a\alpha s_2,j} \leq 1 \\
(g) & \wedge & \sum_a \hat{ina}_{a\alpha s_2,k} = \sum_{s_3} M_{\alpha s_2 s_3,k} & (h') & \wedge & \sum_j ina_{jk} \sum_a P_{a\alpha s_2,j} = \sum_{s_3} M_{\alpha s_2 s_3,k}
\end{array}
\tag{58}
$$

Proof: Suppose ($a-e \iff a'-f'$). Then ($f-g$) and ($g'-h'$) are the same pair of additional constraints expressed in terms of $\hat{ina}$ and (P, ina), respectively, using a'. So it suffices to prove ($a-e \iff a'-f'$).

$\Rightarrow$:

$(d') : \sum_j P_{a\alpha s_2,j} = \sum_{j'} R_{a\alpha s_2,j'} \sum_j Q_{jj'}$ (by (a)) $= \sum_{j'} R_{a\alpha s_2,j'} \sum_{a'\alpha' s'_2} R_{a'\alpha' s'_2,j'}$ (by (b_2)) $= \sum_{j'} R_{a\alpha s_2,j'}$ (by (r_2)) $= C_{a\alpha} INA_{\alpha,s_2}$ (by (r_1)).

$(e') : \sum_{a\alpha s_2} P_{a\alpha s_2,j} = \sum_{j'} \sum_{a\alpha s_2} R_{a\alpha s_2,j'} Q_{jj'}$ (by (a)) $\leq \sum_{j'} Q_{jj'}$ (by (r_2)) ≤ 1 (by (b_1)).

$(a') : \sum_j ina_{jk} P_{a\alpha s_2,j} = \sum_j \sum_{a'\alpha' s'_2} \hat{ina}_{a'\alpha' s'_2,k} P_{a'\alpha' s'_2,j'} P_{a\alpha s_2,j'}$ (by (c))

$= \sum_j \sum_{a'\alpha' s'_2} \hat{ina}_{a'\alpha' s'_2,k} \delta_{aa'} \delta_{\alpha\alpha'} \delta_{s_2 s'_2} P_{a\alpha s_2,j}$ (by (e')) $= \hat{ina}_{a\alpha s_2,k} C_{a\alpha} INA_{\alpha,s_2}$ (by (d')) $= \hat{ina}_{a\alpha s_2,k}$ (by (e)).

$(b') : \sum_j ina_{jk} = \sum_{a\alpha s_2} \hat{ina}_{a\alpha s_2,k} (\sum_j P_{a\alpha s_2,j})$ (by (c)) $\leq \sum_{a\alpha s_2} \hat{ina}_{a\alpha s_2,k}$ (by (d')) ≤ 1 (by (d)).

$(f') : P_{a\alpha s_2,j} = \sum_{j'} R_{a\alpha s_2,j'} Q_{jj'}$ (by (a)) $= C_{a\alpha} INA_{\alpha,s_2} \sum_{j'} R_{a\alpha s_2,j'} Q_{jj'}$ (by (r_3)) $= P_{a\alpha s_2,j} C_{a\alpha} INA_{\alpha,s_2}$

(by (a)).

(c') : $ina_{jk}\sum_{a\alpha s_2} P_{a\alpha s_2,j} = \sum_{a\alpha s_2}\sum_{a'\alpha' s_2'} \hat{ina}_{a'\alpha' s_2',k} P_{a'\alpha' s_2',j} P_{a\alpha s_2,j}$ (by (c)) $= \sum_{a\alpha s_2}\sum_{a'\alpha' s_2'} \hat{ina}_{a'\alpha' s_2',k}\delta_{aa'}\delta_{\alpha\alpha'}\delta_{s_2 s_2'} P_{a\alpha s_2,j}$ (by (e')) $= ina_{j,k}$ (by (c)).

$\Leftarrow$: We exhibited $R(C, INA)$ obeying $(r_1)-(r_3)$. Define

$$Q_{jj'} = \sum_{a\alpha s_2} P_{a\alpha s_2,j} R_{a\alpha s_2,j'}. \tag{59}$$

(a) : $\sum_{j'} R_{a\alpha s_2,j'} Q_{jj'} = \sum_{a'\alpha' s_2'} P_{a'\alpha' s_2',j}\sum_{j'} R_{a'\alpha' s_2',j'} R_{a\alpha s_2,j'}$ (by (59)) $= P_{a\alpha s_2,j}\sum_{j'} R_{a\alpha s_2,j'}$ (by (r_2)) $= P_{a\alpha s_2,j} C_{a\alpha} INA_{\alpha,s_2}$ (by (r_1)) $= P_{a\alpha s_2,j}$ (by (f')).

(b_1) : $\sum_{j'} Q_{jj'} = \sum_{a\alpha s_2} P_{a\alpha s_2,j}\sum_{j'} R_{a\alpha s_2,j'}$ (by (59)) $= \sum_{a\alpha s_2} P_{a\alpha s_2,j} C_{a\alpha} INA_{\alpha,s_2}$ (by (r_1)) $= \sum_{a\alpha s_2} P_{a\alpha s_2,j}$ (by (f')) ≤ 1 (by (e')).

(b_2) : $\sum_j Q_{jj'} = \sum_{a\alpha s_2}(\sum_j P_{a\alpha s_2,j}) R_{a\alpha s_2,j'}$ (by (59)) $= \sum_{a\alpha s_2} R_{a\alpha s_2,j'} C_{a\alpha} INA_{\alpha,s_2}$ (by (d')) $= \sum_{a\alpha s_2} R_{a\alpha s_2,j'}$ (by (r_3)) ≤ 1 (by (r_2)).

(b_3) : $\sum_{jj'} Q_{jj'} = \sum_{a\alpha s_2}(\sum_j P_{a\alpha s_2,j})(\sum_{j'} R_{a\alpha s_2,j'})$ (by (59)) $= \sum_{a\alpha s_2}(C_{a\alpha} INA_{\alpha,s_2})^2$ (by (r_1, d')) $= \sum_{a\alpha s_2} C_{a\alpha} INA_{\alpha,s_2}$.

(c) : $\sum_{a\alpha s_2} \hat{ina}_{a\alpha s_2,k} P_{a\alpha s_2,j} = \sum_{j'} ina_{j'k}\sum_{a\alpha s_2} P_{a\alpha s_2,j} P_{a\alpha s_2,j'}$ (by (a')) $= \sum_{j'} ina_{j'k}\sum_{a\alpha s_2} P_{a\alpha s_2,j}\delta_{jj'}$ (by (d')) $= ina_{jk}\sum_{a\alpha s_2} P_{a\alpha s_2,j} = ina_{jk}$ (by (d')).

(d) : $\sum_{a\alpha s_2} \hat{ina}_{a\alpha s_2,k} = \sum_j ina_{jk}\sum_{a\alpha s_2} P_{a\alpha s_2,j}$ (by (a')) $= \sum_j ina_{jk}$ (by (c')) ≤ 1 (by (b')).

(e) : $\hat{ina}_{a\alpha s_2,k} = \sum_j ina_{jk} P_{a\alpha s_2,j}$ (by (a')) $= C_{a\alpha} INA_{\alpha,s_2}\sum_j ina_{jk} P_{a\alpha s_2,j}$ (by (f')) $= \hat{ina}_{a\alpha s_2,k} C_{a\alpha} INA_{\alpha,s_2}$.

QED

Of the constraints $(58a'-f')$, most can be regarded as constraining the variables M and ina rather than $P_{a\alpha s_2,j}$. That is not true for $d'-f'$ which, however, are analogous to the original constraints on $P_{a\alpha s_2 s_3,k}$, namely $(53c-e)$. Aside from dropping an index, the only change between constraints on $P_{a\alpha s_2 s_3,k}$ and $P_{a\alpha s_2,j}$ is that $\sum_k P_{a\alpha s_2 s_3,k} \leq 1$ becomes $\sum_j P_{a\alpha s_2,j} = C_{a\alpha} INA_{\alpha s_2}$ since the grammar generates extra (spurious) dots but not extra higher-level objects. Because of this close analogy, we can iterate the entire process once to drop the s_2 index from P, and then again to finally eliminate P and C in favor of high-level M and ina variables. What are the resulting constraints on M and ina? The answer is provided by Theorem 1.

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

$$\mathcal{A}: \quad \begin{array}{rcl} \sum_k P_{a\alpha s_2 s_3,k} & \leq & 1, \\ \sum_{a\alpha s_2 s_3} P_{a\alpha s_2 s_3,k} & \leq & 1, \\ P_{a\alpha s_2 s_3,k} & = & P_{a\alpha s_2 s_3,k} C_{a\alpha} INA_{\alpha,s_2} INA_{\alpha s_2,s_3}, \end{array} \quad (a) \quad \text{and} \quad \sum_\alpha C_{a\alpha} = 1, \ (b) \qquad (60)$$

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

$$\mathcal{B}: \quad \begin{array}{rclr} \sum_j Q^2_{j'j} & \leq & 1, & (a2) \\ \sum_{j'} Q^2_{j'j} & = & \Theta(n^{(2)}(C) - j)\Theta(j-1), & (b2) \\ \sum_{j'j} Q^2_{j'j} & = & n^{(2)}(C) & (c2) \end{array} \quad \text{and} \quad \begin{array}{rclr} \sum_i Q^1_{i'i} & \leq & 1 & (a1) \\ \sum_{i'} Q^1_{i'i} & = & \Theta(n^{(1)}(C) - i)\Theta(i-1) & (b1) \\ \sum_{i'i} Q^1_{i'i} & = & n^{(1)}(C) & (c1) \end{array} \qquad (61)$$

(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

$$\mathcal{C}: \quad \begin{array}{rclr} P_{a\alpha s_2 s_3,k} & = & M^3_{\alpha s_2 s_3,k} \sum_j ina^3_{jk} P_{a\alpha s_2,j} & (3) \\ P_{a\alpha s_2,j} & = & M^2_{\alpha s_2,j} \sum_i ina^2_{ij} P_{a\alpha,i} & (2) \\ P_{a\alpha,i} & = & M^1_{\alpha,i} ina^1_{ai} & (1) \end{array} \qquad (62)$$

and inversely by

$$\mathcal{D}: \quad \begin{array}{l} \begin{array}{rclr} M^3_{\alpha s_2 s_3,k} & = & \sum_a P_{a\alpha s_2 s_3,k} & (a3) \\ ina^3_{jk} & = & \sum_{\alpha s_1 s_2} \left(\sum_{s_3} P_{a\alpha s_2 s_3,k}\right) P_{a\alpha s_2,j} & (b3) \\ P_{a\alpha s_2,j} & \equiv & \sum_{j'} R^3_{a\alpha s_2,j}(\{C\}) Q^2_{jj'} & (c3) \end{array} \quad \begin{array}{rclr} M^2_{\alpha s_2,j} & = & \sum_a P_{a\alpha s_2,j} & (a2) \\ ina^2_{ij} & = & \sum_{a\alpha} \left(\sum_{s_2} P_{a\alpha s_2,j}\right) P_{a\alpha,i} & (b2) \\ P_{a\alpha,i} & \equiv & \sum_{i'} R^2_{a\alpha,i}(\{C\}) Q^1_{ii'} & (c2) \end{array} \\ \\ \qquad \begin{array}{rclr} M^1_{\alpha,i} & = & \sum_a P_{a\alpha,i} & (a1) \\ ina^1_{ai} & = & \sum_\alpha P_{a\alpha,i} & (b1) \end{array} \end{array} \qquad (63)$$

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

$\mathcal{E}$:

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

(64)

Notes. (1) The proof is given at a very detailed level, and could perhaps be shortened considerably by the invention of new notation. (2) In M, ina space, $\mathcal{E}(d1)$ has the effect of defining C in the change of variables back to P, C space. (3) The theorem asserts the existence of functions $n^{(1,2)}(C)$ and $R(C)$. We have already specified $n^{(2)} = \sum_{a\alpha s_2} C_{a\alpha} INA_{\alpha,s_2}$ and R; they also depend on the constants INA_{α,s_2}. $n^{(1)}$ is analogous to $n^{(2)}$: $n^{(1)} = \sum_{a\alpha} C_{a\alpha} \times (INA_{\mathrm{root},\alpha} = 1)$. (4) The constants $INA_{\mathrm{root},\alpha} = 1$ and $M^0_{\mathrm{root},a} = 1$ (in $\mathcal{E}$) are introduced for uniformity of notation across levels 1, 2, and 3. In particular $INA_{\mathrm{root},\alpha} = 1$ for the *possible* models types that may appear in a scene, but $INA_{\alpha,s_2} = 1$ and $INA_{\alpha s_2,s_3} = 1$ for the *expected* subparts of an object.

Proof of Theorem 1: The goal is to prove $\mathcal{A} \wedge \mathcal{B} \wedge \mathcal{D} \iff \mathcal{C} \wedge \mathcal{E}$, in which each labelled proposition (equations (60)-(64) above) is itself a conjunction. It will be helpful to use the phrase "... in the context of ..." technically: "$\mathcal{X} \iff \mathcal{Y}$ in the context of $\mathcal{Z}$" means $\mathcal{X} \wedge \mathcal{Z} \iff \mathcal{Y} \wedge \mathcal{Z}$.

Step 1. Assumptions $\mathcal{A}(a) \wedge \mathcal{B}(2) \wedge \mathcal{D}(3)$, along with the definition $(53b)$, are equivalent by Lemmas 1 and 2 to $(53a') \wedge (53d') \wedge (53g') \wedge (58a' - h')$. Note that $\mathcal{B}(b2)$ has been refined from $(b2)$ in Lemma 2, according to equation (56), which is valid since Lemma 2 was proved for a particular $R(C, INA)$ satisfying equation (56). Upon eliminating $\hat{ina}$ by $(58a')$, this conjunction may be rewritten as the conjunction of

$$\mathcal{F}: \quad \begin{array}{rcll} P_{a\alpha s_2 s_3,k} & = & M^3_{\alpha s_2 s_3,k} \sum_j ina^3_{jk} P_{a\alpha s_2,j} & (\equiv \mathcal{C}(3) \text{ from } (53a')) \\ \sum_{\alpha s_2 s_3} M^3_{\alpha s_2 s_3,k} & \leq & 1 & (\equiv \mathcal{E}(a3) \text{ from } (53d')) \\ M^3_{\alpha s_2 s_3,k} & = & INA_{\alpha s_2,s_3} M^3_{\alpha s_2 s_3,k} & (\equiv \mathcal{E}(f3) \text{ from } (53g')) \\ \sum_j ina^3_{jk} & \leq & 1 & (\equiv \mathcal{E}(b3) \text{ from } (58b')) \\ \sum_j ina^3_{jk} \sum_a P_{a\alpha s_2,j} & = & \sum_{s_3} M^3_{\alpha s_2 s_3,k} & (\iff \mathcal{E}(c3) \text{ in context of } \mathcal{D}(a2); \text{from } (58h')) \end{array} \tag{65}$$

and

$$\mathcal{G}: \quad \begin{array}{rcll} ina^3_{jk} & = & ina^3_{jk} \sum_{a\alpha s_2} P_{a\alpha s_2,j} & (a) \quad (\text{from } 58(c')) \\ \sum_k M^3_{\alpha s_2 s_3,k} \sum_j ina^3_{jk} P_{a\alpha s_2,j} & \leq & 1 & (b) \quad (\text{from } 58(g')) \end{array} \tag{66}$$

and

$$\mathcal{H}: \quad \begin{array}{rcll} \sum_j P_{a\alpha s_2,j} & = & C_{a\alpha} INA_{\alpha,s_2} & (\text{from } 58(d')) \\ \sum_{a\alpha s_2} P_{a\alpha s_2,j} & \leq & 1 & (\text{from } 58(e')) \\ P_{a\alpha s_2,j} & = & P_{a\alpha s_2,j} C_{a\alpha} INA_{\alpha,s_2} & (\text{from } 58(f')) \end{array} \tag{67}$$

Thus $\mathcal{A}(a) \wedge \mathcal{B}(2) \wedge \mathcal{D}(3)$ is equivalent to $\mathcal{F} \wedge \mathcal{G} \wedge \mathcal{H}$.

Step 2. $\mathcal{F}$ has already been "translated" into parts of $\mathcal{C} \wedge \mathcal{E}$. We must now translate $\mathcal{G}$ and $\mathcal{H}$. Let us first work on $\mathcal{H}$, which is analogous to $\mathcal{A}(a)$ except for the modification to $\sum_j P$. There are corresponding versions of Lemmas 1 and 2, modified only by loss of an index and as follows from $\sum_j P_{a\alpha s_2,j} = C_{a\alpha} INA_{\alpha,s_2}$: (53$c$), (53$c'$) $\equiv$ (58f), and (58g') are all equal to $C_{a\alpha} INA_{\alpha,s_2}$ rather than bounded by unity. Note that for INA_{α,s_2} factors the loss of the s_2 index results in $INA_{\mathrm{root},\alpha} = 1$ which can be omitted from products in which it occurs. This affects (53e, f') and (58e, d', f').

Thus $\mathcal{H} \wedge \mathcal{B}(1) \wedge \mathcal{D}(2)$, along with the definition $\hat{ina}_{a\alpha,j} = \sum_{s_2} P_{a\alpha s_2,j}$, are equivalent by analogs of Lemmas 1 and 2 to the conjunction of

$$\mathcal{I}: \quad \begin{array}{rcll} P_{a\alpha s_2,j} & = & M^2_{\alpha s_2,j} \sum_i ina^2_{ij} P_{a\alpha,i} & (\equiv \mathcal{C}(2)) \\ \sum_{\alpha s_2} M^2_{\alpha s_2,j} & \leq & 1 & (\equiv \mathcal{E}(a2)) \\ M^2_{\alpha s_2,j} & = & INA_{\alpha,s_2} M^2_{\alpha s_2,j} & (\equiv \mathcal{E}(f2)) \\ \sum_i ina^2_{ij} & \leq & 1 & (\equiv \mathcal{E}(b2)) \\ \sum_j ina^2_{ij} \sum_a P_{a\alpha,i} & = & \sum_{s_2} M^2_{\alpha s_2,j} & (\iff \mathcal{E}(c2) \text{ in context of } \mathcal{D}(a1)) \end{array} \tag{68}$$

and

$$\mathcal{J}: \quad \begin{array}{rcll} ina^2_{ij} & = & ina^2_{ij} \sum_{a\alpha} P_{a\alpha,i} & (a) \\ \sum_j M^2_{\alpha s_2,j} \sum_i ina^2_{ij} P_{a\alpha,i} & = & C_{a\alpha} INA_{\alpha,s_2} & (b) \end{array} \tag{69}$$

and

$$\mathcal{K}: \quad \begin{array}{rcll} \sum_i P_{a\alpha,i} & = & C_{a\alpha} & (a) \\ \sum_{a\alpha} P_{a\alpha,i} & \leq & 1 & (b) \\ P_{a\alpha,i} & = & P_{a\alpha,i} C_{a\alpha} & (c) \end{array} \tag{70}$$

Note that $P_{a\alpha,i} = P_{a\alpha,i} C_{a\alpha}$ is actually redundant since $P_{a\alpha,i} C_{a\alpha} = P_{a\alpha,i} \sum_{i'} P_{a\alpha,i'} = P_{a\alpha,i}$.

Thus $\mathcal{H} \wedge \mathcal{B}(1) \wedge \mathcal{D}(2)$ is equivalent to $\mathcal{I} \wedge \mathcal{J} \wedge \mathcal{K}$.

Step 3. At this point we know enough to rewrite $\mathcal{G}$ in terms of $\mathcal{C}$ and $\mathcal{E}$. In the context of $\mathcal{F} \wedge \mathcal{I} \wedge \mathcal{J} \wedge \mathcal{K}$, i.e. (by step 2) in the context of $\mathcal{F} \wedge \mathcal{H} \wedge \mathcal{B}(1) \wedge \mathcal{D}(2)$, $\mathcal{G}$ is equivalent to the conjunction of

$$\mathcal{L}: \quad \begin{array}{rclll} ina^3_{jk} & \leq & \sum_{\alpha s_2} M^2_{\alpha s_2,j} & (a) & (\equiv \mathcal{E}(e3)) \\ \sum_k M^3_{\alpha s_2 s_3,k} ina^3_{jk} & \leq & M^2_{\alpha s_2,j} INA_{\alpha s_2,s_3} & (b) & (\equiv \mathcal{E}(d3)) \end{array} \tag{71}$$

as we now prove.

$\mathcal{G}(a) \Rightarrow \mathcal{L}(a)$: $ina^3_{jk} = ina^3_{jk} \sum_{a\alpha s_2} P_{a\alpha s_2,j} = ina^3_{jk} \sum_{\alpha s_2} M^2_{\alpha s_2,j}$ (by $\mathcal{D}(a2)$) $\leq \sum_{\alpha s_2} M^2_{\alpha s_2,j}$ since $ina^3_{jk} \leq 1$.

$\mathcal{L}(a) \Rightarrow \mathcal{G}(a)$: $ina^3_{jk} \leq \sum_{\alpha s_2} M^2_{\alpha s_2,j} \Rightarrow (ina^3_{jk})^2 \leq ina^3_{jk} \sum_{\alpha s_2} M^2_{\alpha s_2,j} \Rightarrow ina^3_{jk} \leq ina^3_{jk} \sum_{\alpha s_2} M^2_{\alpha s_2,j} \leq ina^3_{jk}$ (since $ina^3_{jk} = (ina^3_{jk})^2$ and $\sum_{\alpha s_2} M^2_{\alpha s_2,j} \leq 1$ by $\mathcal{I}$) $\Rightarrow ina^3_{jk} = ina^3_{jk} \sum_{\alpha s_2} M^2_{\alpha s_2,j}$.

$\mathcal{L}(b) \Rightarrow \mathcal{G}(b)$: $\sum_k M^3_{\alpha s_2 s_3,k} ina^3_{jk} \leq M^2_{\alpha s_2,j} INA_{\alpha s_2,s_3} \Rightarrow \sum_k M^3_{\alpha s_2 s_3,k} \sum_j ina^3_{jk} P_{a\alpha s_2,j}$ $\leq \sum_j M^2_{\alpha s_2,j} P_{a\alpha s_2,j} INA_{\alpha s_2,s_3} = \sum_j M^2_{\alpha s_2,j} \sum_i ina^2_{ij} P_{a\alpha,i} INA_{\alpha s_2,s_3}$ (by $\mathcal{I}$) $= C_{a\alpha} INA_{\alpha,s_2} INA_{\alpha s_2,s_3}$ (by $\mathcal{J}(b)) \leq 1$.

$\mathcal{G} \Rightarrow \mathcal{L}(b)$: It suffices to deduce

$$\sum_k M^3_{\alpha s_2 s_3,k} ina^3_{jk} M^2_{\alpha s_2,j} \leq M^2_{\alpha s_2,j} INA_{\alpha s_2,s_3} \tag{72}$$

because this is $\mathcal{L}(b)$ if $M^2_{\alpha s_2,j} = 1$ and otherwise $M^2_{\alpha s_2,j} = 0 \Rightarrow P_{a\alpha s_2,j} = 0$ (by $\mathcal{I}$(a)) $\Rightarrow ina^3_{jk} = 0$ (by $\mathcal{D}(3)$, which follows from $\mathcal{F} \wedge \mathcal{G} \wedge \mathcal{H}$ and step 1) $\Rightarrow \sum_k M^3_{\alpha s_2 s_3,k} ina^3_{jk} = 0 \Rightarrow \mathcal{L}(b)$. To derive (72), multiply $\mathcal{G}(b)$ by $\sum_a P_{a\alpha s_2,j} = M^2_{\alpha s_2,j}$ (using $\mathcal{D}(2)$) and use $\mathcal{F}(c)$ to introduce *INA*:

$$\forall a', \ \sum_k M^3_{\alpha s_2 s_3,k} \sum_{j'} ina^3_{j'k} P_{a'\alpha s_2,j'} \sum_a P_{a\alpha s_2,j} \leq M^2_{\alpha s_2,j} INA_{\alpha s_2,s_3}. \tag{73}$$

Now $\mathcal{H} \Rightarrow \sum_a P_{a\alpha s_2,j} \in \{0,1\}$. Case 1: There is a unique a for which $P_{a\alpha s_2,j} = 1$. Let a' be that a. Then

$$P_{a'\alpha s_2,j'} \sum_a P_{a\alpha s_2,j} = \sum_a P_{a\alpha s_2,j'} P_{a\alpha s_2,j}. \tag{74}$$

Case 0: $P_{a\alpha s_2,j} = 0 \ \forall a \Rightarrow$ (74) for any a'. Either way the inequality (73) becomes

$$\sum_k M^3_{\alpha s_2 s_3,k} \sum_{j'} ina^3_{j'k} \sum_a P_{a\alpha s_2,j'} P_{a\alpha s_2,j} \leq M^2_{\alpha s_2,j} INA_{\alpha s_2,s_3}. \tag{75}$$

But $\mathcal{H} \Rightarrow \sum_j P_{a\alpha s_2,j} \leq 1 \Rightarrow P_{a\alpha s_2,j'} P_{a\alpha s_2,j} = \delta_{j'j} P_{a\alpha s_2,j} \Rightarrow \sum_a P_{a\alpha s_2,j'} P_{a\alpha s_2,j} = \delta_{j'j} M^2_{\alpha s_2,j}$ (by $\mathcal{D}(2)$) and (75) implies (72), as desired.

So, in the context of $\mathcal{F} \wedge \mathcal{I} \wedge \mathcal{J} \wedge \mathcal{K}$, $\mathcal{G}$ is equivalent to $\mathcal{L}$. We calculate $\mathcal{A}(a) \wedge \mathcal{B}(1,2) \wedge \mathcal{D}(2,3)$ $\equiv [\mathcal{A}(a) \wedge \mathcal{B}(2) \wedge \mathcal{D}(3)] \wedge \mathcal{B}(1) \wedge \mathcal{D}(2) \iff \mathcal{F} \wedge \mathcal{G} \wedge \mathcal{H} \wedge \mathcal{B}(1) \wedge \mathcal{D}(2)$ (by Step 1) $\iff \mathcal{D}(a2) \wedge \mathcal{F} \wedge \mathcal{G} \wedge \mathcal{H} \wedge \mathcal{B}(1) \wedge \mathcal{D}(2) \iff \mathcal{D}(a2) \wedge \mathcal{F} \wedge \mathcal{G} \wedge \mathcal{I} \wedge \mathcal{J} \wedge \mathcal{K}$ (by Step 2) $\iff \mathcal{D}(a2) \wedge \mathcal{F} \wedge \mathcal{I} \wedge \mathcal{J} \wedge \mathcal{K} \wedge \mathcal{L}$ (by Step 3 so far) $\iff \mathcal{C}(3) \wedge \mathcal{E}(3) \wedge \mathcal{I} \wedge \mathcal{J} \wedge \mathcal{K} \wedge \mathcal{D}(a2)$ (by definitions of $\mathcal{F}$ and $\mathcal{L}$, plus use of context $\mathcal{D}(a2)$ for $\mathcal{E}(c3)$.)

So, $\mathcal{A}(a) \wedge \mathcal{B}(1,2) \wedge \mathcal{D}(2,3) \iff \mathcal{C}(3) \wedge \mathcal{E}(3) \wedge \mathcal{I} \wedge \mathcal{J} \wedge \mathcal{K} \wedge \mathcal{D}(a2)$

Step 4. $\mathcal{I}$ has already been translated into the notation of $\mathcal{C} \wedge \mathcal{E}$; it remains to translate $\mathcal{J}$ and $\mathcal{K}$. In this step we work on $\mathcal{K}$. The analog of Lemma 2 breaks down here because we are at the top of the hierarchy. Instead, observe that $\mathcal{K}(b) \wedge \mathcal{D}(1)$ is directly equivalent to the conjunction

$$\mathcal{M}: \begin{array}{rcll} P_{a\alpha,i} & = & M^1_{\alpha,i} ina^1_{ai} & (a) \quad (\equiv \mathcal{C}(1)) \\ \sum_\alpha M^1_{\alpha,i} & = & \sum_a ina^1_{ai} \leq 1 & (b) \quad (\equiv \mathcal{E}(1a,b,c)), \end{array} \tag{76}$$

as we now prove.

$\Rightarrow$: $M^1_{\alpha,i} ina^1_{ai} = (\sum_b P_{b\alpha,i})(\sum_\beta P_{a\beta,i}) \geq (P_{a\alpha,i})^2 = P_{a\alpha,i}$. On the other hand $1 \geq \sum_{b\beta} P_{b\beta,i}$ (by $\mathcal{K}(b)$) $\geq \sum_b P_{b\alpha,i} + \sum_\beta P_{a\beta,i} - P_{a\alpha,i} \Rightarrow P_{a\alpha,i} + 1 \geq \sum_b P_{b\alpha,i} + \sum_\beta P_{a\beta,i} \Rightarrow (P_{a\alpha,i}+1)^2 \geq (\sum_b P_{b\alpha,i} + \sum_\beta P_{a\beta,i})^2 - (\sum_b P_{b\alpha,i} - \sum_\beta P_{a\beta,i})^2 = 4(\sum_b P_{b\alpha,i})(\sum_\beta P_{a\beta,i})$; therefore $M^1_{\alpha,i} ina^1_{ai} = (\sum_b P_{b\alpha,i})(\sum_\beta P_{a\beta,i})$ $= \lfloor (\sum_b P_{b\alpha,i})(\sum_\beta P_{a\beta,i}) \rfloor \leq \lfloor (P_{a\alpha,i}+1)^2/4 \rfloor = P_{a\alpha,i}$ (since $P_{a\alpha,i} \in \{0,1\}$). Thus $M^1_{\alpha,i} ina^1_{ai} = P_{a\alpha,i}$, i.e. we have $\mathcal{M}(a)$. We see $\sum_\alpha M^1_{\alpha i} = \sum_{a\alpha} P_{a\alpha,i} = \sum_a ina^1_{ai}$ directly from $\mathcal{D}(1)$, and $\sum_{a\alpha} P_{a\alpha,i} \leq 1$ is $\mathcal{K}(b)$; this establishes $\mathcal{M}(b)$.

$\Leftarrow$: $\sum_a P_{a\alpha,i} = M^1_{\alpha,i} \sum_a ina^1_{ai}$ (by $\mathcal{M}(a)$) $= M^1_{\alpha,i} \sum_\beta M^1_{\beta,i}$ (by $\mathcal{M}(b)$) $= M^1_{\alpha,i}$ (by $\mathcal{M}(b)$). Likewise $\sum_\alpha P_{a\alpha,i} = ina^1_{ai}$, so we have $\mathcal{D}(1)$. From this we calculate $\sum_{a\alpha} P_{a\alpha,i} = \sum_\alpha M^1_{\alpha,i} \leq 1$ (by $\mathcal{M}(b)$).

So $\mathcal{K}(b) \wedge \mathcal{D}(1)$ is equivalent to $\mathcal{M}$. In the context of $\mathcal{M}$, $\mathcal{K}(a)$ is equivalent (by straightforward translation) to

$$\mathcal{N}: \sum_i M^1_{\alpha,i} ina^1_{ai} \quad = \quad C_{a\alpha} \quad (\equiv \mathcal{E}(d1)) \tag{77}$$

whence, in the context of $\mathcal{M}$, $\mathcal{A}(b)$ is equivalent to

$$\mathcal{O}: \begin{array}{rcl} 1 & = & \sum_i (\sum_\alpha M^1_{\alpha,i}) ina^1_{ai} \\ & = & \sum_i (\sum_{i'} ina^1_{ai'}) ina^1_{ai} = \sum_i ina^1_{ai} \quad (\equiv \mathcal{E}(e1)), \end{array} \tag{78}$$

As previously noted, $\mathcal{K}(c)$ is redundant in the context of $\mathcal{K}(a)$ and $\mathcal{K}(b)$.

Therefore, $\mathcal{A}(b) \wedge \mathcal{D}(1) \wedge \mathcal{K}$ is equivalent to $\mathcal{M} \wedge \mathcal{N} \wedge \mathcal{O}$, i.e. to $\mathcal{C}(1) \wedge \mathcal{E}(1)$.

Step 5. Here we translate $\mathcal{J}$ to provide the missing pieces of $\mathcal{C} \wedge \mathcal{E}$. In the context of $\mathcal{C}(1) \wedge \mathcal{D}(1) \wedge \mathcal{D}(2) \wedge \mathcal{E}(1) \wedge \mathcal{I} \wedge \mathcal{K} \wedge \mathcal{L}$, $\mathcal{J}$ is equivalent to the conjunction of

$$\mathcal{P}: \begin{array}{rcll} ina^2_{ij} & \leq & \sum_\alpha M^1_{\alpha,i} & (a) \quad (\equiv \mathcal{E}(e2)) \\ \sum_j M^2_{\alpha s_2,j} ina^2_{ij} & = & M^1_{\alpha,i} INA_{\alpha,s_2} & (b) \quad (\equiv \mathcal{E}(d2)). \end{array} \tag{79}$$

Furthermore, the implication $\mathcal{P} \Rightarrow \mathcal{J}$ doesn't require the entire context; it follows from $\mathcal{D}(1) \wedge \mathcal{E}(a1) \wedge \mathcal{K}$ alone. We now prove these assertions.

$\mathcal{J}(a) \Rightarrow \mathcal{P}(a)$: $ina^2_{ij} = ina^2_{ij} \sum_a (\sum_\alpha P_{a\alpha,i}) = ina^2_{ij} \sum_a M^1_{\alpha,i}$ (by $\mathcal{D}(a1)$, which is part of the context) $\leq \sum_a M^1_{\alpha,i}$ (since $ina^2_{ij} \in \{0,1\}$).

$\mathcal{P}(a) \Rightarrow \mathcal{J}(a)$: $ina^2_{ij} \leq \sum_\alpha M^1_{\alpha,i} \Rightarrow (ina^2_{ij})^2 \leq ina^2_{ij} \sum_\alpha M^1_{\alpha,j} \Rightarrow ina^2_{ij} \leq ina^2_{ij} \sum_\alpha M^1_{\alpha,i} \leq ina^2_{ij}$ (since $ina^2_{ij} = (ina^2_{ij})^2$ and $\sum_\alpha M^1_{\alpha,i} \leq 1$ by $\mathcal{E}(a1)) \Rightarrow ina^2_{ij} = ina^2_{ij} \sum_\alpha M^1_{\alpha,j} = ina^2_{ij} \sum_{a\alpha} P_{a\alpha,i}$ (by $\mathcal{D}(a1)$).

$\mathcal{P}(b) \Rightarrow \mathcal{J}(b)$: $\sum_j M^2_{\alpha s_2,j} ina^2_{ij} = M^1_{\alpha,i} INA_{\alpha,s_2} \Rightarrow \sum_j M^2_{\alpha s_2,j} \sum_i ina^2_{ij} P_{a\alpha,i} = (\sum_i M^1_{\alpha,i} P_{a\alpha,i}) INA_{\alpha,s_2}$ $= (\sum_i M^1_{\alpha,i} M^1_{\alpha,i} ina^1_{ai}) INA_{\alpha,s_2}$ (by $\mathcal{M}$ which follows from context $\mathcal{K}(b) \wedge \mathcal{D}(1)$ by the first half of Step 4) $= C_{a\alpha} INA_{\alpha,s_2}$ (by $\mathcal{N}$ which follows from context $\mathcal{K}(a) \wedge \mathcal{M}$ i.e. $\mathcal{K}(a) \wedge \mathcal{K}(b) \wedge \mathcal{D}(1)$).

$\mathcal{J}(b) \Rightarrow \mathcal{P}(b)$: It suffices to deduce

$$\sum_j M^2_{\alpha s_2,j} ina^2_{ij} M^1_{\alpha,i} = M^1_{\alpha,i} INA_{\alpha,s_2} \tag{80}$$

because this is $\mathcal{P}(b)$ if $M^1_{\alpha,i} = 1$ and otherwise $M^1_{\alpha,i} = 0 \Rightarrow P_{a\alpha,i} = 0$ (by $\mathcal{D}(1)$) $\Rightarrow ina^2_{ij} = 0$ (by $\mathcal{D}(2)$) $\Rightarrow \sum_j M^2_{\alpha s_2,j} ina^2_{ij} = 0 \Rightarrow \mathcal{P}(b)$. To derive (80), multiply $\mathcal{J}(b)$ by $\sum_a P_{a\alpha,i} = M^1_{\alpha,i}$ (again using $\mathcal{D}(1)$)

$$\forall a', \ \sum_j M^2_{\alpha s_2,j} \sum_{i'} ina^2_{i'j} P_{a'\alpha,i'} \sum_a P_{a\alpha,i} = M^1_{\alpha,i} C_{a'\alpha} INA_{\alpha,s_2}. \tag{81}$$

Now $\mathcal{K} \Rightarrow \sum_a P_{a\alpha,i} \in \{0,1\}$.

Case 1: There is a unique a for which $P_{a\alpha,i} = 1$. Let a' be that a. Then

$$P_{a'\alpha,i'} \sum_a P_{a\alpha,i} = \sum_a P_{a\alpha,i'} P_{a\alpha,i}. \tag{82}$$

and $C_{a\alpha} = \sum_{i'} P_{a'\alpha,i'}$ (by $\mathcal{K}(a)$) $= 1$ (since $P_{a\alpha,i} = 1$) implies we can drop $C_{a\alpha}$ from the right hand side of (81).

Case 0: $P_{a\alpha,i} = 0$, $\forall a$. Case $\sum_{a'} C_{a'\alpha} > 0$: pick a' so that $C_{a'\alpha} = 1$. Then we can drop $C_{a\alpha}$ from the right hand side of (82), and $P_{a\alpha,i} = 0 \, \forall a \Rightarrow$ (82). Case $\sum_{a'} C_{a'\alpha} = 0$: $M^1_{\alpha i} = \sum_{a'} P_{a'\alpha,i}$ (by $\mathcal{D}(a1)$) $\leq \sum_{a'i} P_{a'\alpha,i} = \sum_{a'} C_{a'\alpha}$ (by $\mathcal{K}(a)$) $= 0$. This directly implies equation (80) since both sides are zero.

So, without loss of generality we may assume equation (82) and drop $C_{a'\alpha} = 1$ from the right hand side of equation (81). As in the proof of step 3, these suffice to prove (80): (81) becomes

$$\sum_j M^2_{\alpha s_2,j} \sum_{i'} ina^2_{i'j} \sum_a P_{a\alpha,i'} P_{a\alpha,i} = M^1_{\alpha,i} INA_{\alpha,s_2} \tag{83}$$

but $\mathcal{K} \Rightarrow \sum_i P_{a\alpha,i} \leq 1 \Rightarrow P_{a\alpha,i'} P_{a\alpha i} = \delta_{i'i} P_{a\alpha,i} \Rightarrow \sum_a P_{a\alpha,i'} P_{a\alpha,i} = \delta_{i'i} M^1_{\alpha,i}$ (by $\mathcal{D}(1)$) and (83) implies (80), as desired.

So, in the context of $\mathcal{C}(1) \wedge \mathcal{D}(1) \wedge \mathcal{D}(2) \wedge \mathcal{E}(1) \wedge \mathcal{I} \wedge \mathcal{K} \wedge \mathcal{L}$, $\mathcal{J}$ is equivalent to $\mathcal{P}$ which is just $\mathcal{E}(d2, e2)$.

Step 6. Here we prove that $\mathcal{A} \wedge \mathcal{B} \wedge \mathcal{D} \Rightarrow \mathcal{C} \wedge \mathcal{E}$, which is half of Theorem 1. Step 7 will establish the converse.

First note that $\mathcal{A}\wedge\mathcal{B}\wedge\mathcal{D} \iff [\mathcal{A}(a) \wedge \mathcal{B}(1,2) \wedge \mathcal{D}(2,3)] \wedge\mathcal{A}(b)\wedge\mathcal{D}(1)\wedge\mathcal{D}(a2) \iff \mathcal{C}(3)\wedge\mathcal{E}(3)\wedge\mathcal{I}\wedge\mathcal{J}$ $\wedge [\mathcal{K} \wedge \mathcal{A}(b) \wedge \mathcal{D}(1)] \wedge \mathcal{D}(a2)$ (by Step 3) $\iff \mathcal{C}(1,3) \wedge \mathcal{E}(1,3) \wedge \mathcal{I} \wedge \mathcal{J} \wedge \mathcal{D}(a2)$ (by Step 4). This will be useful in Step 7 as well:

$$\mathcal{A} \wedge \mathcal{B} \wedge \mathcal{D} \iff \mathcal{C}(1,3) \wedge \mathcal{E}(1,3) \wedge \mathcal{I} \wedge \mathcal{J} \wedge \mathcal{D}(a2). \tag{84}$$

.

Next, $\mathcal{E}(3) \Rightarrow \mathcal{L}$ (by definition) and $\mathcal{C}(1) \wedge \mathcal{E}(1) \Rightarrow \mathcal{K}$ (Step 4), so we can augment (84): $\mathcal{A} \wedge \mathcal{B} \wedge \mathcal{D}$ $\Rightarrow \mathcal{C}(1,3) \wedge \mathcal{E}(1,3) \wedge \mathcal{D}(1,2) \wedge\mathcal{I} \wedge \mathcal{K} \wedge \mathcal{L} \wedge \mathcal{J} \Rightarrow \mathcal{C}(1,3) \wedge \mathcal{E}(1,3) \wedge \mathcal{D}(1,2) \wedge\mathcal{I} \wedge \mathcal{K} \wedge \mathcal{L} \wedge \mathcal{P}$ (by Step 5) $\Rightarrow \mathcal{C}(1,3) \wedge \mathcal{E}(1,3) \wedge [\mathcal{I} \wedge \mathcal{D}(a1)] \wedge \mathcal{P} \Rightarrow \mathcal{C}(1,3) \wedge \mathcal{E}(1,3) \wedge \mathcal{C}(2) \wedge \mathcal{E}(2)$ (by definition of $\mathcal{I}, \mathcal{P}$).

Thus $\mathcal{A} \wedge \mathcal{B} \wedge \mathcal{D} \Rightarrow \mathcal{C} \wedge \mathcal{E}$.

Step 7. It remains to prove $\mathcal{C} \wedge \mathcal{E} \Rightarrow \mathcal{A} \wedge \mathcal{B} \wedge \mathcal{D}$, using the previous steps. By (84) it suffices to prove $\mathcal{C} \wedge \mathcal{E} \Rightarrow \mathcal{C}(1,3) \wedge \mathcal{E}(1,3) \wedge \mathcal{I} \wedge \mathcal{J} \wedge \mathcal{D}(a2)$, which would be implied by $\mathcal{C} \wedge \mathcal{E} \Rightarrow \mathcal{I} \wedge \mathcal{J} \wedge \mathcal{D}(a2)$. So that's what we will prove.

By Step 4, $\mathcal{C}(1)\wedge\mathcal{E}(1) \equiv \mathcal{M}\wedge\mathcal{N}\wedge\mathcal{O} \Rightarrow \mathcal{D}(1)\wedge\mathcal{K}$. From the definitions of $\mathcal{I}$ and $\mathcal{P}$, $\mathcal{C}(2)\wedge\mathcal{E}(2)\wedge\mathcal{D}(1) \Rightarrow$ $\mathcal{I}\wedge\mathcal{P}$. Thus $\mathcal{C}\wedge\mathcal{E} \Rightarrow \mathcal{D}(1)\wedge\mathcal{E}(1)\wedge\mathcal{I}\wedge\mathcal{K}\wedge\mathcal{P}$. By Step 5, in the context $\mathcal{D}(1)\wedge\mathcal{E}(a1)\wedge\mathcal{K}$ (which has just been established), $\mathcal{P} \Rightarrow \mathcal{J}$. Thus $\mathcal{C} \wedge \mathcal{E} \Rightarrow \mathcal{I} \wedge \mathcal{J} \wedge \mathcal{D}(1)$. We needed to prove $\mathcal{C} \wedge \mathcal{E} \Rightarrow \mathcal{I} \wedge \mathcal{J} \wedge \mathcal{D}(a2)$, so it now suffices to show $\mathcal{C} \wedge \mathcal{E} \wedge \mathcal{D}(1) \Rightarrow \mathcal{D}(a2)$.

Assuming $\mathcal{C} \wedge \mathcal{E} \wedge \mathcal{D}(1)$, we calculate $\sum_a P_{a\alpha s_2,j} = M^2_{\alpha s_2,j} \sum_i ina^2_{ij} \sum_a P_{a\alpha,i}$ (by $\mathcal{C}(2)$) $= M^2_{\alpha s_2,j} \sum_i ina^2_{ij} M^1_{\alpha,i}$ (by $\mathcal{D}(a1)$) $= M^2_{\alpha s_2,j} \sum_{s'_2} INA_{\alpha,s'_2} M^2_{\alpha s'_2,j}$ (by $\mathcal{E}(c2)$) $= M^2_{\alpha s_2,j} \sum_{s'_2} M^2_{\alpha s'_2,j}$ (by $\mathcal{E}(f2)$). But $1 \geq \sum_i ina^2_{ij}$ (by $\mathcal{E}(b2)$) $\geq \sum_i ina^2_{ij} M^1_{\alpha,i}$ (since $M^1_{\alpha,i} \leq 1$) $= \sum_{s'_2} M^2_{\alpha s'_2,j}$ (by $\mathcal{E}(c2)$ and $\mathcal{E}(f2)$). So $\sum_{s'_2} M^2_{\alpha s'_2,j} \leq 1$ which implies $M^2_{\alpha s_2,j} \sum_{s'_2} M^2_{\alpha s'_2,j} = M^2_{\alpha s_2,j}$, as usual. Thus $\sum_a P_{a\alpha s_2,j} =$ $M^2_{\alpha s_2,j}$ which is $\mathcal{D}(a2)$, as desired.

Thus $\mathcal{C} \wedge \mathcal{E} \Rightarrow \mathcal{A} \wedge \mathcal{B} \wedge \mathcal{D}$.

Together, Steps 6 and 7 show that

$$\mathcal{A} \wedge \mathcal{B} \wedge \mathcal{D} \iff \mathcal{C} \wedge \mathcal{E}. \tag{85}$$

Q.E.D.

Equation (64) is to be compared with the Frameville constraints, which include (46). Those constraints are represented at three levels of hierarchical organization by $\mathcal{E}(c, d)$. In addition, constraints $\mathcal{E}(a, b)$ are conventional, appearing for example as penalty terms in "Rule 1, 2, and 3" of (Utans et al., 1989). 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. Conversely previous work has relied mostly on the constraints found here, including $\mathcal{E}(c, d)$; for example (Utans et al., 1989) includes $\mathcal{E}(c, d)$ as "Rules 5 and 6", leaving only the constraint of "Rule 4" to differ from $\mathcal{E}(e)$. (Rules 7-9 of that source just encoded the graph-matching objective function.) Thus, the constraints of Theorem 1 agree with those of Frameville, with a few minor differences (namely $\mathcal{E}(e)$ and the specialization of every M and ina variable to some hierarchical level) which may be improvements on previous Frameville neural networks.

4.4.3 The Objective Function

Proposition $\mathcal{E}$ in Theorem 1 establishes the Frameville syntax constraints, including the subtle constraints of equation (46), as a consequence of the grammar. We must now derive the Frameville objective function, equation (45), from equation (50) 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

$$\mathbf{x}_{a\alpha s_2} = \sum_j P_{a\alpha s_2,j}\mathbf{x}_j \quad \text{and} \quad \mathbf{x}_{a\alpha} = \sum_i P_{a\alpha,i}\mathbf{x}_i. \tag{86}$$

Now translating (50) 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.

The first term of (50) is $E_3 = \sum_{a\alpha s_2 s_3} \sum_k P_{a\alpha s_2 s_3,k} [H^{\alpha s_2 s_3}(\mathbf{x}_{a\alpha s_2}, \mathbf{x}_k) - \mu_{\text{extra}} - \mu^{\alpha s_2 s_3}]$. Without loss of generality we can absorb the μ's into H, and omit them from the algebra. Substituting $P_{a\alpha s_2 s_3,k} = M_{\alpha s_2 s_3,k} \sum_j ina_{jk} P_{a\alpha s_2,j}$ and using (86), $E_3 = \sum_{a\alpha s_2 s_3} \sum_k M_{\alpha s_2 s_3,k} \sum_j ina_{jk} P_{a\alpha s_2,j}$ $\times H^{\alpha s_2 s_3}(\sum_{j'} P_{a\alpha s_2,j'}\mathbf{x}_{j'}, \mathbf{x}_k)$. But $H^{\alpha s_2 s_3}(\sum_{j'} P_{a\alpha s_2,j'}\mathbf{x}_{j'}, \mathbf{x}_k) = \sum_{j'} P_{a\alpha s_2,j'} H^{\alpha s_2 s_3}(\mathbf{x}_{j'}, \mathbf{x}_k)$. Since $P_{a\alpha s_2,j} P_{a\alpha s_2,j'} = \delta_{j,j'} P_{a\alpha s_2,j}$, we find $E_3 = \sum_{\alpha s_2 s_3} \sum_k M_{\alpha s_2 s_3,k} \sum_j ina_{jk}$ $(\sum_a P_{a\alpha s_2,j}) \times H^{\alpha s_2 s_3}(\mathbf{x}_j, \mathbf{x}_k)$. We can also evaluate $\sum_a P_{a\alpha s_2,j} = M_{\alpha s_2,j} \sum_a \hat{ina}_{a\alpha,j}$ (by $(53a')$) $= M_{\alpha s_2,j} \sum_{s_2} M_{\alpha s_2,j}$ (by $(58g)$) $=$

$M_{\alpha s_2,j}$ (by $\mathcal{E}(a2)$). Also $M_{\alpha s_2 s_3,k} = M_{\alpha s_2 s_3,k} INA_{\alpha s_2,s_3}$ (by $\mathcal{E}(f3)$). Then

$$E_3 = \sum_{\alpha s_2 s_3} \sum_{jk} M_{\alpha s_2 s_3,k} M_{\alpha s_2,j} INA_{\alpha s_2,s_3} ina_{jk} H^{\alpha s_2 s_3}(\mathbf{x}_j, \mathbf{x}_k). \tag{87}$$

The second term of (50) is $E_2 = \sum_{a\alpha s_2} C_{a\alpha} H^{\alpha s_2}(\mathbf{x}_{a\alpha}, \mathbf{x}_{a\alpha s_2}) = \sum_{a\alpha s_2} C_{a\alpha} H^{\alpha s_2}(\sum_i P_{a\alpha,i}\mathbf{x}_i, \sum_j P_{a\alpha s_2,j}\mathbf{x}_j)$ $= \sum_{\alpha s_2} \sum_{ij} (\sum_a C_{a\alpha} P_{a\alpha,i} P_{a\alpha s_2,j}) H^{\alpha s_2}(\mathbf{x}_i, \mathbf{x}_j)$. But $\sum_a C_{a\alpha} P_{a\alpha,i} P_{a\alpha s_2,j} = \sum_a P_{a\alpha,i} P_{a\alpha s_2,j}$ (by $(58f')$) $= \sum_a P_{a\alpha,i} M_{\alpha s_2,j} \sum_{i'} ina_{i'j} P_{a\alpha,i'} = M_{\alpha s_2,j} \sum_{i'} ina_{i'j} \sum_a P_{a\alpha,i} P_{a\alpha,i'} = M_{\alpha s_2,j} \sum_{i'} ina_{i'j} \delta_{i,i'} \sum_a P_{a\alpha,i} =$ $M_{\alpha s_2,j} ina_{ij} M_{\alpha,i}$ (by $\mathcal{D}(a1)$). Also $M_{\alpha s_2,j} = M_{\alpha s_2,j} INA_{\alpha,s_2}$ (by $\mathcal{E}(f2)$). Thus

$$E_2 = \sum_{\alpha s_2} \sum_{ij} M_{\alpha s_2,j} M_{\alpha,i} INA_{\alpha,s_2} ina_{ij} H^{\alpha s_2}(\mathbf{x}_i, \mathbf{x}_j). \tag{88}$$

The third term of (50) is $E_1 = \sum_{a\alpha} C_{a\alpha} H^{\text{root } \alpha}(\mathbf{x}_{a\alpha}) = \sum_{a\alpha} C_{a\alpha} H^{\text{root } \alpha}(\sum_i P_{a\alpha,i}\mathbf{x}_i)$ $= \sum_{a\alpha i} C_{a\alpha} P_{a\alpha,i} H^{\text{root } \alpha}(\mathbf{x}_i) = \sum_{a\alpha i} P_{a\alpha,i} H^{\text{root } \alpha}(\mathbf{x}_i)$ (by $\mathcal{K}(c)$) $= \sum_{a\alpha i} M^1_{\alpha,i} ina^1_{ai} H^{\text{root } \alpha}(\mathbf{x}_i)$ (by $\mathcal{C}(1)$). Thus

$$E_1 = \sum_{\alpha} \sum_{ai} M^1_{\alpha,i} M^0_{\text{root},a} INA_{\text{root},\alpha} ina^1_{ai} H^{\text{root } \alpha}(\mathbf{x}_i) \tag{89}$$

(since $M^0_{\text{root},a} = 1$ and $INA_{\text{root},\alpha} = 1$), which is in the desired form. But since $\sum_a P_{a\alpha,i} = M^1_{\alpha,i}$ (by $\mathcal{D}(a1)$), this special case could be more simply written as $E_1 = \sum_{\alpha i} M^1_{\alpha,i} H^{\text{root } \alpha}(\mathbf{x}_i)$.

There are also entropy terms that arise from integrating out Q^1 and Q^2, i.e. the redundancy of M and ina with respect to P. Let $N^2 \geq n^2(C)$ be the maximum number of level 2 instances, i.e. the range of j, and $N^1 \geq n^1(C)$ be the same for level 1. Then the entropy term (from $\mathcal{B}$) is $\log(\begin{pmatrix} N^2 \\ n^2 \end{pmatrix} n^2!) + \log(\begin{pmatrix} N^1 \\ n^1 \end{pmatrix} n^1!) = \log((N^2!) - \log((N^2 - n^2)!) + \log(N^1!) - \log((N^1 - n^1)!)$ i.e.

$$S = \log(N^2!) - \log((N^2 - \sum_{a\alpha s_2 i} ina^1_{ai} M^1_{\alpha i} INA_{\alpha,s_2})!) + \log(N^1!) - \log((N^1 - \sum_{a\alpha i} ina^1_{ai} M^1_{\alpha i})!) \tag{90}$$

There is a further entropy term associated with integrating out unused analog model variables such as $\mathbf{x}_{a\alpha s_2}$, but it may be absorbed into chemical potentials (μ terms in E) and hence into H terms.

Thus the final Frameville objective function is

$$\begin{aligned} E(M, ina, x) \;=\; & \textstyle\sum_{\alpha s_2 s_3} \sum_{jk} M^3_{\alpha s_2 s_3,k} M^2_{\alpha s_2,j} INA_{\alpha s_2,s_3} ina^3_{jk} H^{\alpha s_2 s_3}(\mathbf{x}_j, \mathbf{x}_k) \\ & + \textstyle\sum_{\alpha s_2} \sum_{ij} M^2_{\alpha s_2,j} M^1_{\alpha,i} INA_{\alpha,s_2} ina^2_{ij} H^{\alpha s_2}(\mathbf{x}_i, \mathbf{x}_j) \\ & + \textstyle\sum_{\alpha} \sum_{ai} M^1_{\alpha,i} M^0_{\text{root},a} INA_{\text{root},\alpha} ina^1_{ai} H^{\text{root } \alpha}(\mathbf{x}_i) \\ & + 1/\beta[\log(N^2!) - \log((N^2 - \textstyle\sum_{a\alpha s_2 i} ina^1_{ai} M^1_{\alpha i} INA_{\alpha,s_2})!) \\ & + \log(N^1!) - \log((N^1 - \textstyle\sum_{a\alpha i} ina^1_{ai} M^1_{\alpha i})!)] \end{aligned} \tag{91}$$

where as before $M^0_{\mathrm{root},a} = INA_{\mathrm{root},\alpha} = 1$. This 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:

$$\begin{aligned} E(M, ina, x) \quad = \quad & \textstyle\sum_{\alpha\beta} \sum_{ij} M^{\mathrm{level}(\alpha)}_{\alpha,i} M^{\mathrm{level}(\beta)}_{\beta,j} INA_{\alpha,\beta} ina^{\mathrm{level}(\beta)}_{ij} H^{\alpha\beta}(\mathbf{x}_i, \mathbf{x}_j) \\ & +1/\hat{\beta}[\log(N^2!) - \log((N^2 - \textstyle\sum_{\alpha\beta ij} ina^1_{ij} M^0_{\mathrm{root},i} M^1_{\alpha,j} INA_{\alpha,\beta})!) \\ & + \log(N^1!) - \log((N^1 - \textstyle\sum_{\alpha ij} ina^1_{ij} M^0_{\mathrm{root},i} M^1_{\alpha,i})!)]. \end{aligned} \tag{92}$$

This is to be compared with equation (45). 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 (Mjolsness and Garrett, 1990).

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.

4.4.4 Frameville and High-Level Vision

As mentioned earlier, with the Frameville grammar we approach a modest plateau of generality. From the generalized assignment problem of equation (50) 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 proved, 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 relabelling 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 2.5, 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. Another efficient indexing scheme, not used in the Frameville networks, would be geometric hashing (Lamdan et al., 1988). Either form of indexing could possibly be added as a learned computational shortcut. Learning is briefly discussed in the next section.

5 LEARNING

Three possible methods for learning a grammar are suggested here. They all assume that learning the grammar can be expressed as tuning its parameters, as is the case for unstructured neural networks. First, the kind of grammar we have been studying could be augmented with an initial set of "meta-grammar" rules, which randomly choose the parameters of the permanent models and then generate many images by the usual grammar. The task of inferring the permanent models' parameters is just another Bayesian inference problem, stretched out over many images. Second, one could minimize the Kullback information between the probability distributions of an unknown grammar, images from which the perceiver sees, and a parameterized grammar. This algorithm would be similar to the "Boltzmann machine" for neural network learning (Hinton and Sejnowski, 1986). Finally one could look for *clusters* in model space by defining a distance "metric" D between images and mathematically projecting it back though the grammar.

The latter alternative can be explained by an analogy with the forward-going composition law for probabilities (and hence objectives)

$$e^{-\beta E_{n+1}(\mathbf{x}_{n+1})} \propto \Pr(\mathbf{x}_{n+1}) = \int d\mathbf{x}_n \Pr(\mathbf{x}_{n+1}|\mathbf{x}_n)\Pr(\mathbf{x}_n) \propto \int d\mathbf{x}_n \Pr(\mathbf{x}_{n+1}|\mathbf{x}_n) e^{-\beta E_n(\mathbf{x}_n)}, \tag{93}$$

which uses only those probabilities $\Pr(\mathbf{x}_{n+1}|\mathbf{x}_n)$ directly provided by the grammar. One could define a backward-going composition law for $\hat{D}_n(\mathbf{x}_n, \mathbf{y}_n)$ by, for example,

$$e^{-\beta \hat{D}_n(\mathbf{x}_n,\mathbf{y}_n)} \propto \int d\mathbf{x}_{n+1} \int d\mathbf{y}_{n+1} \Pr(\mathbf{x}_{n+1}|\mathbf{x}_n)\Pr(\mathbf{y}_{n+1}|\mathbf{y}_n) e^{-\beta \hat{D}_{n+1}(\mathbf{x}_{n+1},\mathbf{y}_{n+1})}. \tag{94}$$

Here the proposed distance D is related to $\hat{D}$ by $\hat{D}(\mathbf{x}, \mathbf{y}) = f(D(\mathbf{x}, \mathbf{y}))$ where f is any monotonic, strictly increasing function such as the identity plus a constant, $\hat{D} = D + c$, or the squaring function, $\hat{D} = D^2$.

This expression for D has the advantage that β can be varied as in deterministic annealing, and also that if $\Pr(\mathbf{x}_{n+1}|\mathbf{x}_n)$ is an invariant measure on a group G (e.g. translations, rotations or permutations with uniform probability) and D_{n+1} respects G (i.e. if $\Pr(\mathbf{x}_{n+1}|\mathbf{x}_n) = \int dg \delta(\mathbf{x}_{n+1} - g \cdot \mathbf{x}_n)$ and $D_{n+1}(g \cdot x, y) = D_{n+1}(x, g^{-1} \cdot y)$) then the large-$\beta$ limit of (94) is

$$\begin{aligned} D_n(\mathbf{x}_n, \mathbf{y}_n) &= f^{-1}\left(-(1/\beta) \log \max_{g_1,g_2 \in G} \exp -\beta f\left(D_{n+1}(g_1 \cdot \mathbf{x}_n, g_2 \cdot \mathbf{y}_n)\right)\right) \\ &= \min_{g_1,g_2 \in G} D_{n+1}(g_1 \cdot \mathbf{x}_n, g_2 \cdot \mathbf{y}_n) \\ &= \min_{g_1,g_2 \in G} D_{n+1}(g_2^{-1} g_1 \cdot \mathbf{x}_n, \mathbf{y}_n) \\ &= \min_{g \in G} D_{n+1}(g \cdot \mathbf{x}_n, \mathbf{y}_n) \end{aligned} \tag{95}$$

which actually preserves all the properties of a distance metric including the triangle inequality. That is, if D_{n+1} is a distance metric then D_n is also.

Expressions similar to (94) could define distances between objects at earlier and later stages in the grammar, for example between an image and a model; in this way data can be clustered in the model parameter space. In fact if d is such an asymmetric "distance" and $\Pr(\mathbf{x}_{n+1}|\mathbf{x}_n)$ is an invariant measure on a group G respected by d, then the large-β limit of

$$e^{-\beta d_n(\mathbf{x}_n,\mathbf{y}_m)} \propto \int d\mathbf{x}_{n+1} \Pr(\mathbf{x}_{n+1}|\mathbf{x}_n) e^{-\beta d_{n+1}(\mathbf{x}_{n+1},\mathbf{y}_m)}. \tag{96}$$

is again

$$d_n(\mathbf{x}_n, \mathbf{y}_m) = \min_{g \in G} d_{n+1}(g \cdot \mathbf{x}_n, \mathbf{y}_m) \tag{97}$$

where $m > n$.

5.1 Graph Matching Objectives

For example, we can use (95) with $\hat{D} = f(D) = D^2$ to define the distance between two graphs, G and g, by adding permutation invariance to the Euclidean distance $D_1(G, g)$ between the 0/1 connection matrices $G_{\alpha\beta}$ and g_{ij}. Euclidean distance between matrices respects the adjoint representation of the group of permutations, i.e. permutations that act by permuting the nodes of a graph and hence both indices of a matrix:

$$\begin{aligned}
[D_0(G,g)]^2 &= [\min{}_{\text{Permutations } P} D_1(G, P \circ g)]^2 \\
&= \min{}_{\text{Permutations } P} ||G - P \cdot g \cdot P^T||^2 \\
&= \min{}_{\text{Permutations } P} ||G \cdot P - P \cdot g||^2 \\
&= \min{}_{\text{Permutations } P} \left[||G||^2 + ||g||^2 - 2\text{trace}\left(G \cdot P g^T \cdot P^T\right)\right] \\
&= ||G||^2 + ||g||^2 - 2 \min{}_{\text{Permutations } P} \textstyle\sum_{\alpha\beta ij} G_{\alpha\beta} P_{\beta j} g_{ij} P_{\alpha i}
\end{aligned} \tag{98}$$

which is the standard neural network objective function for inexact graph matching (von der Malsburg and Bienenstock, 1986; Hopfield and Tank, 1986), up to an additive constant. It is also a standard cost metric for missing model links and extra data links in pure graph matching for computer vision (Ballard and Brown, 1982), restricted to the case of permutation correspondances:

$$\begin{aligned}
E(P) &= \text{missing model graph links} + \text{extra data graph links} \\
&= \textstyle\sum_{\alpha\beta} \max\left(0, G_{\alpha\beta} - \sum_{ij} g_{ij} P_{\alpha i} P_{\beta j}\right) + \sum_{\alpha\beta} \max\left(0, \sum_{ij} g_{ij} P_{\alpha i} P_{\beta j} - G_{\alpha\beta}\right) \\
&= \textstyle\sum_{\alpha\beta} \left(G_{\alpha\beta} - \sum_{ij} g_{ij} P_{\alpha i} P_{\beta j}\right)^2
\end{aligned} \tag{99}$$

since G and g are 0/1 matrices and P is a permutation matrix. This is the same expression as that minimized to produce D_0 above, though it has the interpretation of a distance (squared) between model and data rather than between two models.

These objectives for graph-matching have proven useful in neural networks devoid of learning, but they may also be useful in the approach to learning a structured neural network through abstract clustering at a high level in a visual grammar.

6 DISCUSSION

Figure 1 is a diagram of the method used in this paper to derive neural networks from visual grammars via Bayesian inference. Given a grammar of suitable form, one can calculate a joint probability distribution on images and their explanations. This distribution may be transformed, by changing variables, as we showed in the multiple curve grouping and Frameville networks. By using the Mean Field Theory approximation, a Bayesian inference problem on this distribution is transformed into an optimization problem with an algebraic objective function. This function can be further transformed, for example using the techniques of (Mjolsness and Garrett, 1990), to reduce its cost or increase its circuit-level implementability; then a neural network follows from descent dynamics.

We studied grammars that model visual phenomena such as missing and extra data, group invariances, hierarchical objects, multiple instances of an object in a scene, and flexible spline-like objects. 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, or include discrimination or inheritance trees on the set of object models. A somewhat more general view of grammars whose rules posess connectionist models (similar to the objective functions attached to rules in this paper) is presented in (Mjolsness et al., 1991b), where such grammars are proposed for modelling the development of biological organisms. A different way to translate a class of grammars into neural network objective functions is presented in (Miller et al., 1991); it currently applies to "regular" languages and has been demonstrated in a low-level vision problem. Other directions for generalization of the parallel grammar occur in the extensive literature on L-systems (Rozenberg and Salomaa, 1980) and graph grammars (Ehrig et al., 1983).

The grammars examined in this paper do not yet produced realistic images, and one could consider adding new rules to move from the "pictures" we studied to gray-level images. This would allow the preprocessing of images to produce pictures composed of image features, necessary in Section 3, to be replaced with more neural network computation. At another extreme, the idea of relating optimization to Bayesian inference on a probabilistic grammar is in principle not restricted to vision at all, and could perhaps be adapted to other problems to which neural network optimization has been applied.

The complexity of the visual world will certainly demand many grammars of increasing size for success in computer vision by our approach. We have not yet discussed how to obtain such grammars, though Figure 1 suggests hand-design and learning should be directed at producing grammars rather than the subsequent stages. Hand design takes a lot of human labor. Fortunately the large amount of research in computer graphics is a source of generative models for images, some of which are mathematically simple enough to be put in the form of a probabilistic grammar or are already close to that form (e.g. (Smith, 1984; Prusinkiewicz, 1990)). In addition we speculated on possible algorithms for learning the grammars, or at least aspects of them, by using visual experience.

Acknowledgements

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

References

Anandan, P., Letovsky, S., and Mjolsness, E. (1989). Connectionist variable-binding by optimization. In *11th Annual Conference of the Cognitive Science Society*. Lawrence Earlbaum Associates. University of Michigan.

Ballard, D. H. and Brown, C. M. (1982). *Computer Vision*, chapter 11, pages 360–362. Prentice Hall. Equation 11.3, Missing Cost.

Bertsekas, D. P. and Tsitsiklis, J. N. (1989). *Parallel and Distributed Computation*, chapter 5, page 333. Prentice Hall.

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

Burns, J. B. (1986). Extracting straight lines. *IEEE Trans. PAMI*, 8(4):425–455.

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

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

Durbin, R. and Willshaw, D. (1987). An analog approach to the travelling salesman problem using an elastic net method. *Nature*, 326:689–691.

Ehrig, H., Nagl, M., Rosenfeld, A., and Rozenberg, G., editors (1983). *Graph Grammars and their Application to Computer Science; Third International Workshop*, volume 291 of *Lecture Notes in Computer Science*. Springer-Verlag.

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

Feldman, J. A., Fanty, M. A., and Goddard, N. H. (1988). Computing with structured neural networks. *IEEE Computer*, page 91.

Garrett, C. (1990). Private communication.

Hinton, G. E. and Sejnowski, T. J. (1986). Learning and relearning in Boltzmann machines. In Rumelhart, D. E. and McClelland, J. L., editors, *Parallel Distributed Processing*, chapter 7. MIT Press.

Hopfield, J. J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences USA*, vol. 81:3088–3092.

Hopfield, J. J. and Tank, D. W. (1985). 'Neural' computation of decisions in optimization problems. *Biological Cybernetics*, vol. 52:141–152.

Hopfield, J. J. and Tank, D. W. (1986). Collective computation with continuous variables. In *Disordered Systems and Biological Organization*, pages 155–170. Springer-Verlag.

Kosowsky, J. J. and Yuille, A. L. (1991). The invisible hand algorithm: Solving the assignment problem with statistical physics. Technical Report 91-1, Harvard Robotics Laboratory.

Lamdan, Y., Schwartz, J. T., and Wolfson, H. J. (1988). Object recognition by affine invariant matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 335–344.

Milios, E. E. (1989). Shape matching using curvature processes. *Computer Vision, Graphics, and Image Processing*, 47:203–226.

Miller, M. I., Roysam, B., Smith, K. R., and O'Sullivan, J. A. (1991). Representing and computing regular languages on massively parallel networks. *IEEE Transactions on Neural Networks*, 2(1).

Mjolsness, E. and Garrett, C. (1990). Algebraic transformations of objective functions. *Neural Networks*, 3:651–669.

Mjolsness, E., Gindi, G., and Anandan, P. (1989). Optimization in model matching and perceptual organization. *Neural Computation*, 1.

Mjolsness, E., Rangarajan, A., and Garrett, C. (1991a). A neural net for reconstruction of multiple curves with a visual grammar. Manuscript in preparation. Summary in International Joint Conference on Neural Networks Seattle, July 1991.

Mjolsness, E., Sharp, D. H., and Reinitz, J. (1991b). A connectionist model of development. *Journal of Theoretical Biology*. In press. Also available as Yale Computer Science technical report YALEU/DCS/796, June 1990.

Peterson, C. and Soderberg, B. (1989). A new method for mapping optimization problems onto neural networks. *International Journal of Neural Systems*, 1(3).

Prusinkiewicz, P. (1990). *The Algorithmic Beauty of Plants*. Springer-Verlag New York.

Rozenberg, G. and Salomaa, A. (1980). *The Mathematical Theory of L-systems*. Academic Press.

Simic, P. D. (1990a). Constrainted nets for graph matching and other quadrateic assignment problems. Technical Report CALT-68-1672, Caltech Physics Department.

Simic, P. D. (1990b). Statistical mechanics as the underlying theory of 'elastic' and 'neural' optimization. *Network: Computation in Neural Systems*, 1(1):89–103.

Smith, A. R. (1984). Plants, fractals and formal languages. *Computer Graphics*, 18(3):1–10. Proceedings of SIGGRAPH '84.

Stolcke, A. (1989). Unification as constraint satisfaction in structured connectionist networks. *Neural Computation*, 1(4):559–567.

Tresp, V. (1991). A neural network approach for three-dimensional object recognition. In *Neural Information Processing Systems 3*. Morgan Kaufmann.

Utans, J., Gindi, G., Mjolsness, E., and Anandan, P. (1989). Neural networks for object recognition within compositional hierarchies: Initial experiments. Technical Report Report No. 8903, Center for Systems Science, Yale University Department of Electrical Engineering.

Van den Bout, D. E. and Miller, III, T. K. (1990). Graph partitioning using annealed networks. *IEEE Transactions on Neural Networks*, 1(2):192–203.

von der Malsburg, C. (1988). Pattern recognition by labeled graph matching. *Neural Networks*, 1:141–148.

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

Williams, L. R. and Hanson, A. R. (1988). Translating optical flow into token matches and depth from looming. In *Second International Conference on Computer Vision*, pages 441–448. Staircase test image sequence.

Witkin, A., Terzopoulos, D., and Kass, M. (1987). Signal matching through scale space. *International Journal of Computer Vision*, 1:133–144.

Yuille, A. L. (1990). Generalized deformable models, statistical physics, and matching problems. *Neural Computation*, 2(1):1–24.

Zemel, R. S. (1989). Traffic: A connectionist model of object recognition. Technical Report CRG-TR-89-2, University of Toronto Connectionist Research Group.