Notes of Concepts and Representations in Vision and Cognition

Jihang Li

November 21, 2019

Contents

Part I

Low and Mid Level Vision

Part II

Stochastic Grammars in Vision

Contents

Chapter 1

Overview of Stochastic Grammar

Statistical grammar is a framework with **probabilistic notion** of grammaticality.

1.1 The Origin of Grammars

Signals s in real world that tend to occur together more frequently than by chance can be grouped together to form higher order parts of the signal, and this process can be repeated to form larger parts. This form a vocabulary of **"reusable"** parts.

To measure whether a grouping is a good part:

$$\log_2 \left(\frac{p(s|_{A \cup B})}{p(s|_A) \cdot p(s|_B)} \right) \tag{1.1}$$

where $s|_A$ and $s|_B$ are two parts of signal $s : D \to I$, with $A \subset D$ and $B \subset D$. Two parts of a signal are bound if the probability of their co-occurrence is significantly **greater than** the probability if their occurrence was independent. Example as shown in Figure 1.1.

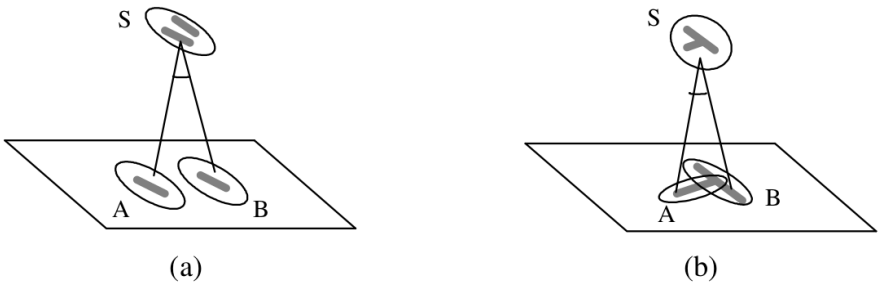


Figure 1.1: (a) Two parallel lines form a reusable part containing as its constituents the two lines (b) A T-junction is another reusable part formed from two lines.

Original Texts

The set of reusable parts that one identifies in some class of signals, e.g. in images, is called the **vocabulary** for this class of signals. Each such reusable part has a name or label. In language, a noun phrase, whose label is "NP", is a common reusable part and an element

of the linguistic vocabulary. In vision, a face is a clear candidate for such a very high-level reusable part. The set of such reusable parts which on encounters in analyzing statistically a specific signal is called the **parse graph** of the signal. Abstractly, one first associate to a signal s the set of subsets $\{A_i\}$ of D such that $s|_{A_i}$ is a reusable part. Then these subsets are made into the vertices or nodes $\langle A_i \rangle$ of the parse graph. In the graph, the proper inclusion of one subset in another, $A_i \subsetneqq A_j$, is shown by a "vertical" directed edge $\langle A_j \rangle \to \langle A_i \rangle$. For simplicity, we prune redundant edges in this graph by adding edges only when $A_i \subsetneqq A_j$ and there is no A_k such that $A_i \subsetneqq A_k \subsetneqq A_j$.

In the ideal situation, parse graph is a tree with the whole signal at the top and the domain D (the letters of the text or the pixels of the image) at the bottom. Moreover, each node $\langle A_i \rangle$ should be the disjoint union of its children, i.e., the parts $\{A_j | A_j \subsetneqq A_i\}$ such that $\cup_j A_j = A_i$.

1.2 The Traditional Formulation of Grammar

Grammar $\mathcal{G} = (V_N, V_T, \mathbf{R}, S)$, where V_N is a finite set of non-terminal nodes, V_T is a finite set of terminal nodes, $S \in V_N$ is a start symbol at the root, and $\mathbf{R}$ is a set of production rules,

$$\mathbf{R} = \{\gamma : \alpha \to \beta\} \tag{1.2}$$

where α, β are strings that $\alpha, \beta \in (V_N \cup V_T)^{+1}$ and including at least one non-terminal symbol. Chomsky classified languages into 4 types according to the form of their production rules:

- Definition: $A, B \in V_N$, $a \in V_T$
- Type 0: a **phrase structure or free** grammar with no constraint on α and β
- Type 1: a **context sensitive** grammar, where $\xi A \eta \to \xi \beta \eta$ means A is rewritten by β in the context of strings ξ and η
- Type 2: a **context free** grammar, $A \to \beta$
- Type 3: a **finite state or regular** grammar, $A \to aB$ or $A \to a$

The set of all possible strings of terminals ω derived from a $\mathcal{G}$ is called its **language**:

$$\mathbf{L}(\mathcal{G}) = \{\omega : S \overset{\mathbf{R}^*}{\Longrightarrow} \omega, \omega \in V_T^*\} \tag{1.3}$$

$\mathbf{R}^*$ means a sequence of production rules deriving ω from S:

$$S \overset{\gamma_1, \gamma_2, \ldots, \gamma_{n(\omega)}}{\Longrightarrow} \omega \tag{1.4}$$

If $\mathcal{G}$ is of type 1, 2 or 3, then a **parse tree** is obtained for a given ω:

$$\mathbf{p}t(\omega) = (\gamma_1, \gamma_2, \ldots, \gamma_{n(\omega)}) \tag{1.5}$$

For example, in images, V_T can be pixels or a simple set of local structures in the images such as textons and other image primitives. Then V_N will be reusable parts and objects in the image. And a production rule $A \to \beta$ is a template which enables you to expand A. Then $\mathbf{L}(\mathcal{G})$ will be the set of all valid object **configurations**.

The grammar rules represent both structural **regularity** and **flexibility**:

[1] V^* means a string consisting of $n \geq 0$ symbols from V, and V^+ means $n \geq 1$

- Regularity: enforced by the template which decomposes an entity A, such as object into certain elements in β
- Flexibility: reflected by the fact that each structure A has many alternative decompositions

A parse tree:

- Root is an Or-node
- Or-nodes are labelled by $V_N \cup V_T$
- And-nodes are labelled by $\mathbf{R}$

To generate a parse tree:

1. For any Or-node with A, consider all rules have A on the left and create children which have And-nodes labelled by corresponding rules.
2. Step 1 expand to a set of Or-nodes labelled by the symbols on the right of the rule.
3. An Or-node labelled by terminal does not expand further.

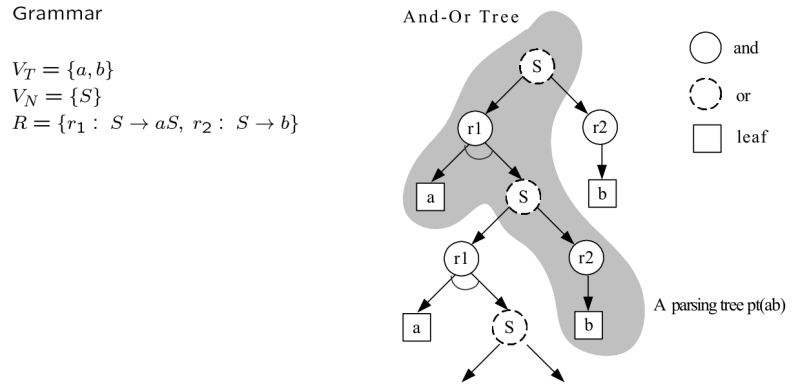


Figure 1.2: A very simple grammar, its universal And-Or tree and a specific parse tree in shadow.

A vision example of an And-Or tree, using the reusable parts in Figure 1.1, is shown in Figure 1.3. B, C are the two ambiguous ways to interpret A. B represents an occlusion configuration with two layers while C represents a butting/alignment configuration at one layer. A is a **frequently observed local structure** in natural images when a long bar (e.g. a tree trunk) occludes a surface boundary (e.g. a fence).

Another example, as shown in Figure 1.4, the 6 leaf nodes can compose a set of configurations for node A, which is called the "language" of A - denoted by $\mathbf{L}(A)$.

Original Texts

The power of composition is crucial for representing visual concepts which have varying structures. For example, if A is an object category, such as car or chair, then $\mathbf{L}(A)$ is a set of valid designs of cars or chairs.

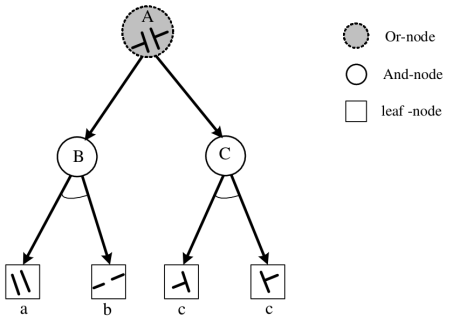


Figure 1.3: An example of binding elements a,b,c into a larger structures A in two alternative ways, represented by an And-Or tree.

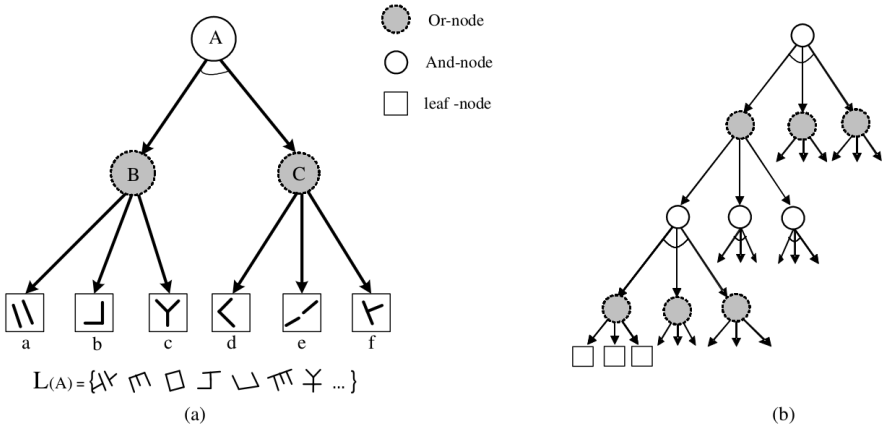


Figure 1.4: (a) An And-node A is composed of two Or-nodes B and C, each of which includes three alternative leaf nodes. The 6 leaf nodes can compose a set of configurations for node A, which is called the "language" of A. (b) An And-Or tree (5-level branch number = 3) with 10 And-nodes, 30 Or-nodes, and 81 leaf nodes, can produce $3^{12} = 531441$ possible configurations, though some may be repeated.

1.3 Overlapping Reusable Parts

If there exists a string $\omega \in \mathbf{L}(\mathcal{G})$ that has more than one parse tree, then $\mathcal{G}$ is said to be an **ambiguous grammar**.

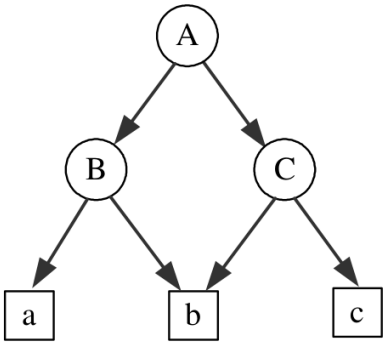


Figure 1.5: Parts sharing and the diamond structure in And-Or graphs.

For example, Figure 1.6 shows two parse trees for a classic ambiguous sentence, which has two distinct reusable parts which overlap in the "the man". In context, the sentence is always spoken with only one of these meanings, so one parse is **right** while the other one is **wrong**. **One reusable part is accepted and the other one is rejected. If we reject one, the remaining parts do not overlap.**

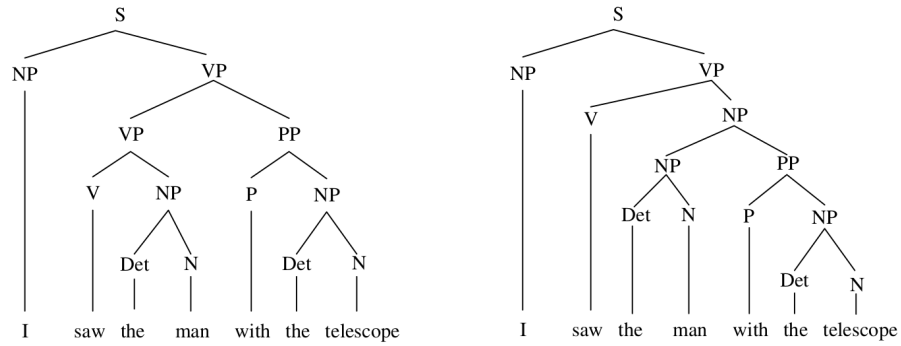


Figure 1.6: An example of ambiguous sentence with two parse trees. The non-terminal nodes S, V, NP, VP denotes sentence, verbal, noun phrase, and verbal phrase respectively. Note that if the two parses are merged, we obtain a graph, not a tree, with a "diamond" in it as above.

Taking image parsing in vision, there seem to be 4 ways overlap can occur:

1. Ambiguous scenes where distinct parses suggest themselves.
2. High level patterns which incorporate multiple partial patterns.
3. "Joints" between two high level parts where some sharing of pixels or edges occurs.
4. Occlusion where a background object is completed behind a foreground object, so the two objects overlap.

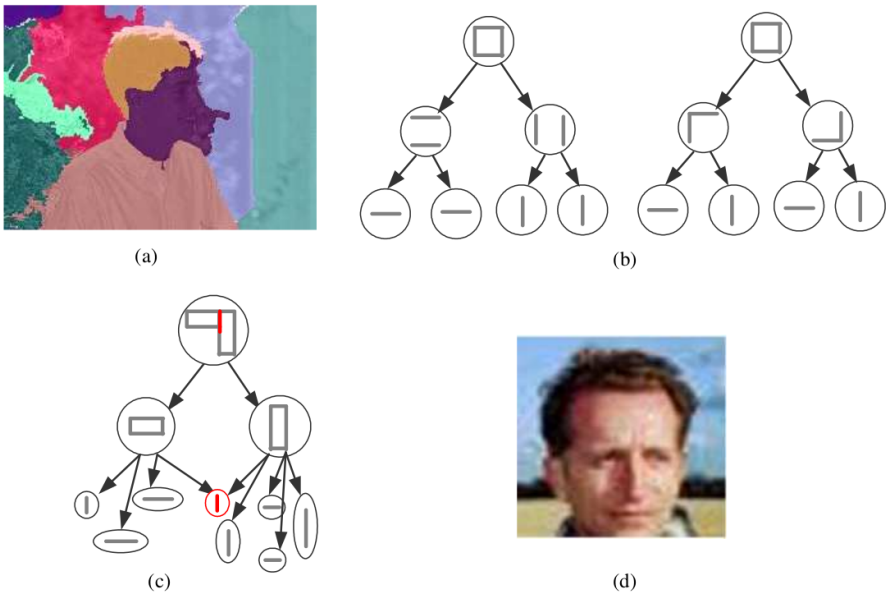


Figure 1.7: Four types of images in which "reusable parts" overlap. (a) The Pinnocio nose is a part of the background whose gray level is close to the face, so it can be grouped with the face or the background. This algorithm chose the wrong parse. (b) The square can be parsed in two different ways depending on which partial patterns are singled out. Neither parse is wrong but the mid-level units overlap. (c) The two halves of a butt joint have a common small edge. A way to solve this, is to duplicate the shared edge to restore a tree-like parse. (d) The reconstructed complete sky, trees and field overlap with the face.

For the overlap example as shown in Figure 1.7.d, we form duplicate images planes carrying the two objects: this is crucial when we want to use priors to reconstruct as much as possible of the occluded object. The right parse for such objects should **add extra leaves** at the bottom to represent the occluded object. The new leaves carry **colors, textures** etc. extrapolated from the visible parts of the object. Their occluded boundaries were that *gestalt school* called **amodal** contours.

1.4 Stochastic Grammar

To connect with real world signals, we augment $\mathcal{G}$ with a set of probabilities $\mathcal{P}$ as a fifth component. For example, stochastic context free grammar (SCFG) supposes $A \in V_N$ has a number of alternative rewriting rules,

$$A \to \beta_1 | \beta_2 | \ldots | \beta_{n(A)}, \gamma_i : A \to \beta_i \tag{1.7}$$

Each production rule is associated with a probability $p(\gamma_i) = p(A \to \beta_i)$ such that:

$$\sum_{i=1}^{n(A)} p(A \to \beta_i) = 1 \tag{1.8}$$

This correspond to a **random branching process**. Similarly stochastic regular grammar corresponds to a **Markov chain process**.

The probability of a parse tree:

$$p(\mathbf{p}t(\omega)) = \prod_{j=1}^{n(\omega)} p(\gamma_j) \tag{1.9}$$

The probability for a string (in language) or configuration (in image) $\omega \in \mathbf{L}(\mathcal{G})$:

$$p(\omega) = \sum_{\mathbf{p}t(\omega)} p(\mathbf{p}t(\omega)) \tag{1.10}$$

Therefore a $\mathcal{G} = (V_N, V_T, \mathbf{R}, S, \mathcal{P})$ produces a probability distribution on its language

$$\mathbf{L}(\mathcal{G}) = \{(\omega, p(\omega)) : S \stackrel{\mathbf{R}^*}{\Longrightarrow} \omega, \omega \in V_T^*\} \tag{1.11}$$

A stochastic grammar is said to be **consistent** if $\sum_{\omega \in \mathbf{L}(\mathcal{G})} p(\omega) = 1$. This is not necessarily true even when Equation 1.8 is satisfied for each $A \in V_N$. The complication is caused by cases when there is a positive probability that the parse tree may not end in a finite number of steps. For example, if we have a production rule that expands A to AA or terminates to a,

$$A \to AA|a \quad \text{with prob. } \rho|(1-\rho)$$

If $\rho > \frac{1}{2}$ then A expands faster than it terminates and keeps replicating. This poses constraints for designing $\mathcal{P}$.

$\mathcal{P}$ can be learned in a supervised way from a set of **observed parse trees** $\{\mathbf{p}t_m, m = 1, 2, \ldots, M\}$ by maximum likelihood estimation (MLE),

$$\mathcal{P}^* = \arg\max \prod_{m=1}^{M} p(\mathbf{p}t) \tag{1.12}$$

The solution: the probability for each A in Equation 1.7 is

$$p(A \to \beta_i) = \frac{\#(A \to \beta_i)}{\sum_{j=1}^{n(A)} \#(A \to \beta_i)} \tag{1.13}$$

where $\#(A \to \beta_i)$ is the number of times a rule is used in all the M parse trees.

In a unsupervised learning case, when the observation is **a set of strings** without parse trees, one can still follow the MLE above with an expectation-maximization (ME) algorithm. It was shown in[3] that the MLE of $\mathcal{P}$ can rule out infinite expansion and produce a consistent grammar.

In Figure 1.3, one can augment the two parses by probabilities ρ and $1-\rho$, written as a stochastic production rule:

$$A \to a \cdot b|c \cdot c; \quad \rho|(1-\rho) \tag{1.14}$$

1.5 Stochastic Grammar with Context

Later on, the proposed image grammar, an **And-Or tree**, will be augmented to an **And-Or graph** by adding relations and contexts constraints as horizontal links, which results in probabilistic models to represent a stochastic **context sensitive** grammar for images.

(NOTE: An example about natural language is left out.)

Original Texts

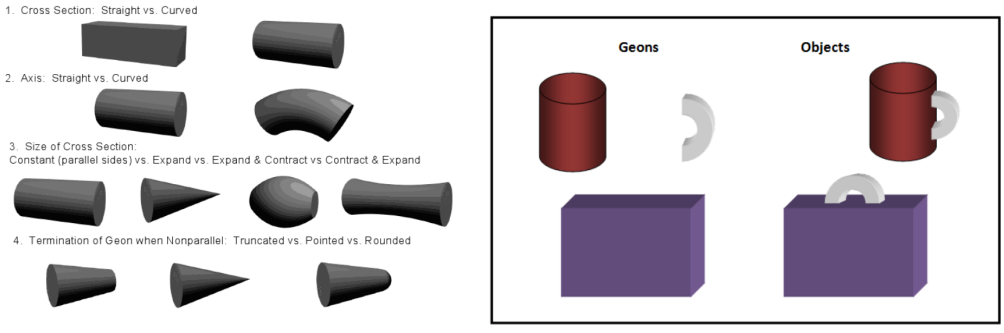
In vision, these non-local relations occur much more frequently. These relationships represent the spatial context at all levels of vision from pixels, primitives to parts, objects and scenes, and lead to various graphical models, such as **Markov random fields**. Gestalt organizations are popular examples in the middle level and low-level vision. For example, whenever a foreground object occludes part of a background object, with this background object being visible on both sides of the foreground one, these two visible parts of the background object constrain each other. Other non-local connections may reflect functional relations, such as object X is "supporting" object Y.

1.6 Compositional World and Grammar Representation

According to Recognition-by-Components (RBC)[2], a theory explaining the bottom-up process of object recognition, humans are able to recognize objects by separating them into geons (the object's main components parts) which are the simple 2D or 3D forms such as **cylinders, bricks, wedges, cones, circles, rectangles**, etc..

Original Texts

It is suggested that the visual inputs of eyes are matched against the structural representation of objects in our brain. Here the structural representation is composed of geons and the relations between them, like **"on top of", "end to end", "end to middle", "next to"** and so on. In other words, when humans see an object, we first recognize the geons. Then the perceived geons are compared to the objects stored in our brain. Further psychological studies also show that we focus on two particular components when see an object: **edges and concavities**. Concavity is the area when two or more edges meet, which is used for recognizing geons and the relations. RBC theory gives us an explanation of object recognition from psychological perspective. In fact, not only the objects, but the **scenes, events or even fluent changes** are also hierarchical and compositional, which is the exact reason why we want to represent them by grammars.



(a) Geons (b) Breaking Objects into Geons

Figure 1.8: Geons and RBC Theory

1.7 Organization of the Book

- S-AOG: serves as a unified framework of representation, learning and recognition for objects or scenes.
- A-AOG: with which constraints can be put in a recursive bottom-up and top-down process, which corresponds to the idea of relations between geons in RBC theory.
- T-AOG: can be used to represent the hierarchical compositions of events and the temporal relations between the sub-events; the terminal node of T-AOG would be atomic actions, which is decomposed into a human pose, one or multiple objects, and the relations between the pose and the objects.
- C-AOG: is given for the **unsupervised learning** of causal relations from video, learning which action can cause objects to change status.

Chapter 2

Spatial And-Or Graph

The proposal grammar integrates three representations:

1. Stochastic grammars for composition.
2. Markov (or graphical) models for contexts.
3. Sparse coding with primitives (wavelets).

2.1 Three New Issues in Image Grammars in Contrast to Language

An image grammar should include two aspects:

1. The hierarchical structures $\mathcal{G}$ which generate a set of valid image configurations $\mathbf{L}(\mathcal{G})$.
2. The context information which makes sure the components in a configuration observe good spatial relationships between object parts, e.g. relative positions, ratio of sizes, consistency of colors.

There are three major differences (and difficulties) between the language grammars and image grammars.

First, the loss of the left-to-right ordering in language. In language, every production rule $A \to \beta$ is assumed to generate a linearly ordered sequence of nodes β and following this down to the leaves, we get a linearly ordered sequences of terminal words. In vision, we have to replace the implicit links of words to their left and right neighbors by the edges of a more complex "region adjacency graph" (RAG). That is, let an image I have a decomposition $D = \cup_{k \in S} R_k$, where a RAG is made with nodes $\langle R_i \rangle$ and edges are represented as $\langle R_k \rangle$—$\langle R_l \rangle$ whenever adjacent, then the nodes of β in $A \to \beta$ are no longer linearly ordered. Instead, β should be made into a configuration which is a set of nodes from $V_N \cup V_T$ plus horizontal edges representing adjacency.

Original Texts

One immediate consequence of the lack of natural ordering is that a region has very ambiguous production rules. Let A be a region and a an atomic region, and let the production rule be $A \to aA|a$. A linear region $\omega = (a, a, a, \ldots, a)$ has a unique parse graph in left-to-right ordering. With the order removed, it has a combinatorial number of parse trees. Figure 2.1 shows an example of parsing an image with a cheetah. It becomes infeasible to estimate the probability $p(\omega)$ by summing over all these parse trees in a grammar.

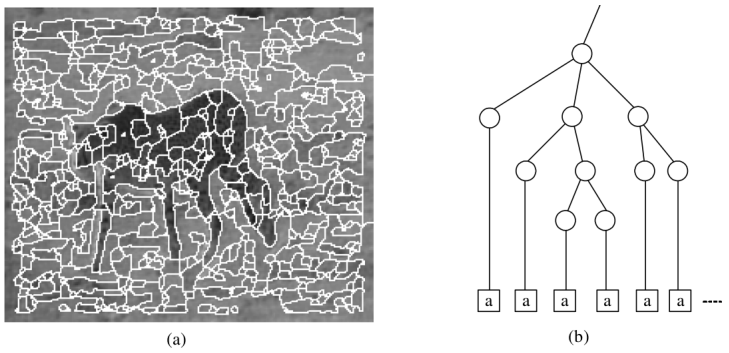


Figure 2.1: A cheetah and the background after local segmentation: both can be described by a RAG. Without the left-to-right order, if the regions are to be merged one at a time, they have a combinatorially explosive number of parse trees.

Therefore we must avoid recursively define $A \to aA$, and treat the grouping of atomic regions into one large region A as a **single computational step**, such as the grouping and partitioning in a graph space[1]. **Thus $p(\omega)$ is assigned to each object as a whole instead of the production rules.**

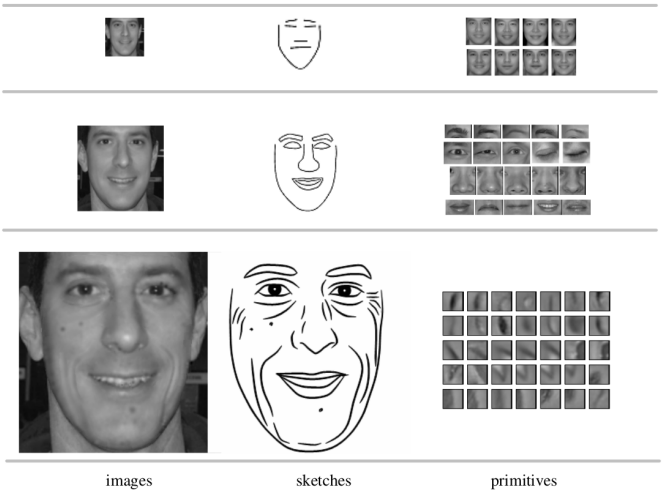


Figure 2.2: A face appears at three resolutions is represented by graph configurations in three scales. The right column shows the primitives used at the three levels.

Second, the issue of scaling is unseen in language grammar. This implies that the parse tree can terminate immediately at any node because no more details is visible.

Figure 2.2 shows an example of such issue. One can add some termination rules to each non-terminal node, e.g., each non-terminal node may exit the production for a low resolution case:

$$\forall A \in V_N, A \rightarrow \beta_1 | \cdots | \beta_{n(A)} | t_1 | t_2 \tag{2.1}$$

where $t_1, t_2 \in V_T$ are image primitives or image templates for A at certain scales. This issue does not complicate the grammar design, except that one must learn the image primitives at multiple scales in developing the visual vocabulary.

Third, natural images contain a much wider spectrum of quite irregular local patterns than in speech signals. Images have: (i) very regular and highly structured objects which could be composed by **production rules**; (ii) very stochastic patterns such as clutter and texture which are better represented by **Markov Random Field (MRF)** models. The spectrum is continuous (?). The structured and textured patterns can transfer from one to the other through continuous scaling[4][5].

2.2 Visual Vocabulary

2.2.1 The Hierarchical Visual Vocabulary – the "Lego Land"

Definition I. Visual Vocabulary

The visual vocabulary is a set of pairs, each consisting of an image function $\Phi_i(x, y; \alpha_i)$ and a set of $d(i)$ bonds (i.e., its degree), to be eventually connected to with other elements, which are denoted by a vector $\beta_i = (\beta_{i,1}, \ldots, \beta_{i,d(i)})$. We think of $\beta_{i,k}$ as an **address variable or pointer**. α_i is a **vector of attributes** for (a) a geometric transformation, e.g. the central position, scale, orientation and plastic deformation, and (b) appearances, such as intensity contrast, profile or surface albedo. In particular, α_i determines a domain $\Lambda_i(\alpha_i)$ and Φ_i is then defined for $(x, y) \in \Lambda_i$ with values in R (a gray-valued template) or R^3 (a color template). Often each $\beta_{i,k}$ is associated with a subset of the boundary of $\Lambda_i(\alpha_i)$. The whole vocabulary is thus a set:

$$\Delta = \{(\Phi_i(x, y; \alpha_i), \beta_i) : (x, y) \in \Lambda_i(\alpha_i) \subset \Lambda\} \tag{2.2}$$

where i indexes the type of the primitives.

The conventional wavelets, Gabor image bases, image patches, and image fragments are possible examples of this vocabulary except that they don't have bonds.

2.2.2 Image Primitives

Julesz conjectured that textons are the atomic elements in the early stage of visual perception for local structures. Marr extended texton concept to image primitives which called "symbolic tokens" — primal sketch.

As illustrated in Figure 2.3(a), an image primitive is a small image patch with a degree d connections or bonds illustrated by the half circles:

- blobs ($d = 0$)

- terminators ($d = 1$)
- edges/ridges ($d = 2$)
- "L"-junctions ($d = 2$)
- "T"-junctions ($d = 3$)
- cross junctions ($d = 4$)

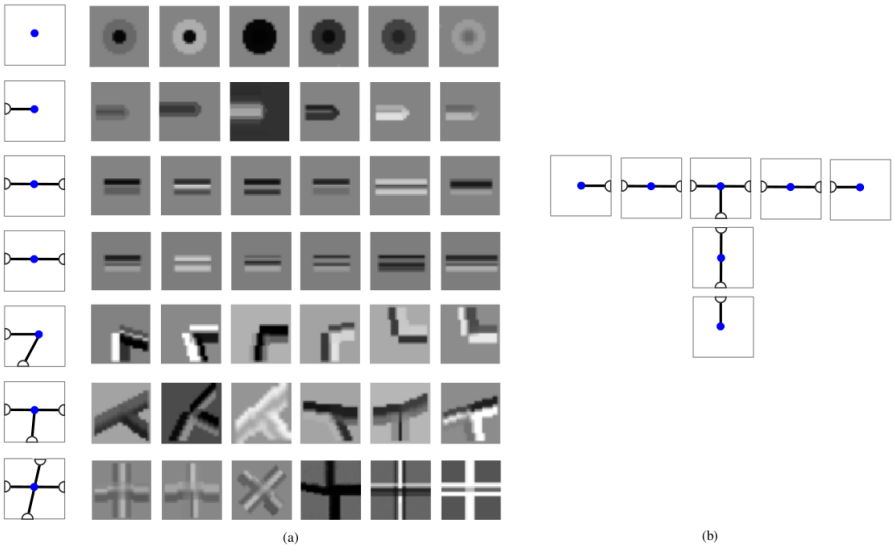


Figure 2.3: Low level visual vocabulary image primitives. (a). Some examples of image primitives: blobs, terminators, edges, ridges, "L"-junctions, "T"-junction, and cross junction etc. These primitives are the elements for composing a bigger graph structure at the upper level of the hierarchy. (b) is an example of composing a big "T"-shape image using 7 primitives. From (Guo, Zhu and Wu, 2003).

Part III

Cognitive Models

Bibliography

[1] Adrian Barbu and Song-Chun Zhu. Generalizing sweden-wang to sampling arbitrary posterior probabilities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1239–1253, 2005.

[2] Irvimg Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94(2):115–147, 1987.

[3] Zhiyi Chi and Stuart Geman. Estimation of probabilistic context-free grammar. *Computational Linguistics*, 24(2):299–305, 1998.

[4] Yizhou Wang and Song-Chun Zhu. Perceptual scale-space and its applications. *International Journal of Computer Vision*, 80(1):143–1165, 2008.

[5] Ying Nian Wu, Gui-Song Xia, and Song-Chun Zhu. Perceptual scale-space and its applications. *CVPR*, pages 1–8, 2007.