Grammatical Inference by n-gram Modeling of Convex Groups: Representation of Visual Random Polytopes*

Peter Michael Goebel
gp@acin.tuwien.ac.at

Markus Vincze
vincze@acin.tuwien.ac.at

Bernard Favre-Bulle
favre@acin.tuwien.ac.at

ACIN – Automation & Control Institute
University of Technology Vienna, A-1040 Vienna, Austria

Abstract

In this paper, a joint solution to the problem of finding appropriate abstract representations for visual polytopes is given. By using support from convex and stochastic geometry, collecting information of views from different viewpoints, perceptual grouping of 3D point-cloud image points into halfplanes with probabilistic robust fitting and the segmentation of edges and corners by intersecting halfplanes yields an aggregation of visual primitives into object prototypes by Bayes' belief networks. In order to build object prototypes, a n-gram model is trained by edge and corner primitives, derived from Monte-Carlo simulations and processing of real 3D point-clouds. Finally, we use perplexity to find out the best performing network and define a Dirichlet distribution model of the n-grams.

Keywords: Bayesian Belief Nets, Grammatical Inference, n-grams, Dirichlet distribution.

1 Introduction

Within two recently proposed approaches, the cognitive vision framework [3] and the modeling of semantic aggregation of object prototypes from visual primitives [2], we utilized predefined recipes, however, in this paper we explain construction and learning of recipes by a stochastic grammar approach.

The paper is organized as follows: the remainder of this section presents required requisites to the methods used herein. Section 2 shows related work to our proposed approach; Section 3 explains grammatical inference by deriving the n-gram model, and Section 4 presents its realization by a Bayes network; we conclude in Section 5 together with an outlook on further work.

1.1 Grammatical Inference

Grammatical inference (GI) aims at learning regular language from examples, i.e. by determining the structure of a finite state automaton and by estimating its inter junction transition probabilities into a statistical model [5, 2], which is represented by a family of probability distributions $\mathbb{P} = \{P_\theta \mid \theta \in \Theta\}$. Here, θ is a parameter vector, ($\Theta \subseteq \mathbb{R}^n$) a parameter space, and P_θ is a probability distribution function. Hence, to each entity considered, a random variable is assigned and P_θ represents their joint probability distribution. Given random observations $X = (x_1, \ldots, x_n)$, the posterior distribution on θ is calculated by Bayes' formula $P(\theta|X) = \frac{P(X|\theta)\,P(\theta)}{P(X)}$, where the uncertainty about θ is interpreted by the subjective probability $P(\theta)$ as the prior and $P(X|\theta)$ as the likelihood of the samples, when given θ. In practice, three principal methodologies are used: *(i) discriminative*[1], which

[*] This work was supported by project S9101 "Cognitive Vision" of the Austrian Science Foundation.

[1] e.g. Support Vector Machines, traditional neural networks, or conditional random fields

directly model the posterior distribution; *(ii) generative*[2], which model both, the likelihood and the prior by sampling from a distribution; and *(iii) hybrid*, as a mixture of i) and ii).

The bigger the model, the more it appears too complex for finding a solution by discriminative methods, and therefore, the distributions are better factorized into manageable parts by the support of a generative model, such as: *(i) the naive Bayes method* - which assumes strongly naive independence between random variables; *(ii) hidden Markov models* - which are Markov processes with unobservable parameters determined from the observed ones; *(iii) probabilistic context free grammars* - which are context-free grammars, in which each production is augmented with an explicit probability, or *(iv) n-grams* - which model symbol sequences, using state transition probabilities.

Therefore, the most widely and practically used statistical model in language recognition is the n-gram model, in which an estimate of the likelihood of a word is made solely on the identity of the preceding $n-1$ words of an expression, firstly proposed by Jelinek and Mercer [6] as a naive Markov model. Thus, given n, it is to assign a probability $\mathbb{P}(W)$ to a conceivable word sequence $W = \{w_{t=1}, w_{t=2}, \ldots, w_{t=m}\}$ of length m, at observation time step $t = \{1 \ldots m\}$ yielding

$$\mathbb{P}(W) = \prod_{i=n}^{m} p(w_i|w_{i-1}, \ldots, w_{i-n+1}) \qquad (1)$$

where $p(w_i|w_{i-1}, w_{i-2}, \ldots, w_{i-n+1})$ is the probability[3] that word w_i will be spoken, given the words $w_{i-1} \ldots w_{i-n+1}$.

1.2 Perceptual Grouping and Convex Groups

Perceptual grouping (PG) focuses on the extraction of relations based solely on low-level image features – i.e. points, lines and surfaces – without applying any knowledge concerning the image structure. These relations are then used to group features together in order to obtain a semantic structure map, like humans do. The grouping rules are: *(i) similarity*, *(ii) proximity*, *(iii) common fate*, *(iv) collinearity*, *(v) good continuation*, and *(vi) past experience* according to Gestalt theory [10]. Since we are concerned with the understanding of the topology of visual objects, for the sake of simplicity we have restricted ourselves to convex polyhedral objects. Thus, analysis of random polytopes requires a combination of two fields of research – convex and stochastic geometry. Convex geometry, is positioned between geometry, analysis and discrete mathematics; it dates back to the ancient Greeks, and was systematically developed by Minkowski (1864–1909) within his theorem on mixed volumes. Minkowski considered the determination of convex polytopes by the areas and exterior normal vectors of their facets [4]. Stochastic geometry is concerned with random geometric structures, ranging from simple points or line segments to arbitrary closed sets.

Hence, modeling the topology of polytopes in the limit can be seen from three different points of view: *(i) using convex hulls* of random points – by utilizing volume approximation when the number of generating points tends to infinity; *(ii) random projections* of higher-dimensional polytopes – by using combinatorial properties of cross-polytopes that are projected into lower dimensional random subspaces; and *(iii) intersections of random halfspaces* – by assuming the polytopes at hand are large enough in size.

Figure 1 show the grouping of point clouds by random halfspaces, which were grouped by a RANSAC method [16] that groups triangles in order to get halfplanes. In Figure 2 the resulting intersection lines are shown, which are completed by lines extracted between the object border and object shadow. The method delivers line primitive lists in 3D.

2 Related Work

There exists a vast literature on PG in vision, see for a survey e.g. [15]. Early work in PG dates back to Marr [13], who was the

[2] i.e graphical models akin Bayes nets

[3] t starts at n in order to satisfy causality

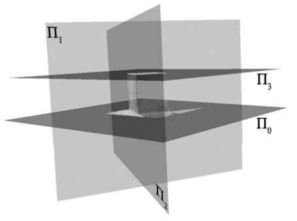

Figure 1: Halfplanes group the point clouds; the plane intersections give the convex extrema of the object.

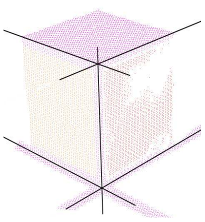

Figure 2: The outlines of a cube are given by the intersection lines and border lines.

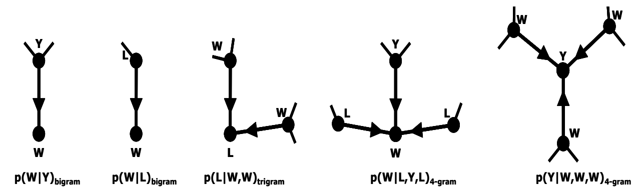

Figure 3: Junction to junction transitions that appear at object corners, for instance as bigram $p(W|Y)$, $p(W|L)$; trigram $p(L|W, W)$; and 4-gram $p(W|L, Y, L)$, $p(Y|W, W, W)$ probabilities.

first to suggest incorporating grouping based on curvilinearity into larger structures by his primal sketch approach; Witkin and Tenenbaum [17] postulated non-accidentalness for spatiotemporal coherence; Lowe [12] derived an expectation estimate for accidental occurrences by assuming a uniform distribution to line segments; Sarkar and Boyer [15] developed a Bayesian network method for geometric knowledge-based representation; Zucker [19] introduced closure as more global feature to better deal with occlusions; and Ackermann et al [1] introduced a Markov random field grouping approach with learning from hand-labeled trainings sets; however, and many more.

More recent work of Procter [14] investigated grouping of edge-triple features, to recognize polyhedrons from 2D image projections, however, the method turned out to be too sensitive to noise and thus failed practical demonstration. Levinshtein et al [11] proposed recovering a Marr-like abstraction hierarchy [13] from a set of examples by applying a multiscale blob and ridge detector for feature extraction, here drawbacks arise from the fact that positional information of the blobs is lost during graph embedding and that a vast number of parameters are to be defined.

Zillich [18] focused on issues concerning complexity and robustness by proposing an incremental processing scheme for the PG of edges in indoor scenes. He proposed using Gestalt principles to support PG and successfully implemented a Markov random field approach in order to deal with real-world objects. Although, in an abstract sense, the approach relates very closely to our approach, Zillich's self-criticism is that he unfortunately relied too much on getting clean edges by local edge detectors, which degraded system performance in complexer setups. The main difference to our approach is that we focus on robustness by applying multiresolution methods [3] and using more global approaches rather than localized ones. We intend to first get only a coarse representation of an object at hand and then only refine the representation afterwards when more information is required.

3 The Graphical Model

Objects, seen from a very abstract level, can be represented by graphs. These graphs are representations of the connections between structural elements, such as *(i) corners*; *(ii) edges*; and *(iii) boundaries*, where two areas meet: in a fold[4], or in a blade[5], or in a face[6]. Structural elements and their connections are defined by relations between image primitives. Image primitives are composed by groupings of image features that are extracted from image points. Depending on the viewpoint, corners may appear very differently (see Figure 5, left column). In our approach, we classify corners into four junction types, such as: *(L)-type* which means an occluding line and denotes a blade that is an object region in front of the background; *(Y)-type* means a 3-

[4] when both areas are from the same object

[5] separating object and background

[6] when one area appears closed

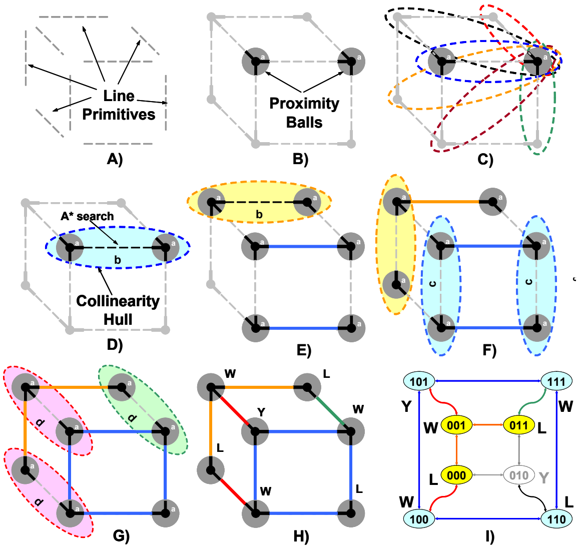

Figure 4: Example of building a planar graph in 3D space: A) line primitives are assumed to be given; B) proximit-balls at the line endings group together up to three lines; C) search for candidates of correspondence; D) determine true edges by A* search within collinearity hulls a and within a halfplane; E) look for similar correspondences b at parallel halfplanes; F) continue search c within halfplanes; G) connect halfplanes by correspondences d; H) label corners with junction types.

junction, where three surfaces intersect with the angles between each pair are $<$ 180 degree; *(T)-type* means a 3-junction with one of the angles has exactly 180 degrees; and *(W)-type* means a 3-junction with one angle $>$ 180 degrees, see Figure 3 for examples of several conditional junction/junction groupings.

Figure 4 shows the object graph builder algorithm for the construction of a cube. It starts with assuming the presence of line primitives in 3D space, which can be skew to each other (A). Thus, either at the intersection or in the middle of the perpendicular shortest distance between such two lines, proximity-balls (B) at the line endings group pairwise together concurrent lines. A search for corresponding candidates (C) find shortest paths by A* search within collinearity hulls (D) within a halfplane. Then, looking for similar correspondences (E) at parallel halfplanes and continuing search (F) within halfplanes, connecting is supported by correspondences (G). Finally, labeling of corners with junction types generates a planar graph (I) [2].

3.1 The Statistical Language n-gram Model

Statistical language models are used for the modeling of sequences of symbols under the assumption that the underlying generation process is an approximate Markov-chain process, where the *approximate Markovian property* of an order n process is that the conditional probability of future states depends only upon the past $n-1$ states, what means that the process is conditionally independent of the $> n-1$ past states. Thus, a statistical language model, in general, defines a probability distribution over the set of symbols sampled from a finite alphabet. Its representation by a Markov chain appears as a simple, but high performing concept [8].

A certain n-tuple of symbols is called a n-gram and is denoted $n - G := \mathbf{y}z$ where $\mathbf{y} = [y_1, y_2, \ldots, y_{n-1}]$ is the context length and z is the predicted symbol [8]. The maximal context symbol length $n-1$ is also called *the symbols history*, where lengths with $n = 1$ denote unigram, $n = 2$ bigram, $n = 3$ trigram, and $n > 3$ generally n-gram models.

A n-gram model is a Markov-chain model, giving the probability definition for non-terminating symbols with a maximal context length of $n-1$ predecessors by Bayes' chain rule

$$p(\omega) \approx \prod_{t=1}^{T} p(\omega_t | \underbrace{\omega_{t-n+1}, \ldots, \omega_{t-1}}_{n-1 \text{ symbols history}}) \qquad (2)$$

When one is able to make sure that the training data contain all objects' data that should be learned, n-gram models have the advantage over hidden Markov language models to allow the calculation of optimum parameters directly from training data. In our present approach, the naive straight ahead solution to the learning problem is to count the absolute frequency $c(\omega_1, \ldots, \omega_N)$ of all symbol tuples and all possible contexts $\omega_1, \ldots, \omega_N$ to define all conditional probabilities by relative frequencies.

3.1.1 Experimental Setup

In order to get a satisfying set of sample data, we have tested 4-gram, trigram and bigram modeling by data from real cube-point-clouds with different viewpoints and also Monte-Carlo (MC) 2-manifold projection simulations of polyhedron objects with 4/4, 8/6, and 20/12 vertices/faces that are tetrahedron, hexahedron, and dodecahedron views, respectively. Figure 5-left, shows a sequence of projections a)...d) of a polyhedron with $n = 8$ vertices, a cube, where the data was generated by MC simulations; at the middle, the respective planar graphs are given, as described in [2]; and at the right, derived Bayes' networks are shown that will be discussed in Section 4. Four bigram transition matrices of MC-samples of Figure 5 are calculated with the observed counts of junction-type to junction-type transitions that are cumulated into training counts for $N_{i,j} \mid i,j : \{L, W, Y, T\}$, as shown at the left of Table 1.

A problem is that the probability for *unseen events* is per definition zero, and in the case of presenting unseen events to the model, the n-gram model runs into *empirical holes* or singularities of its distribution. Therefore, a post-processing step for *smoothing* [8] in order to overcome the problem is indicated. The simplest one would be to apply the *Adding-one*[7] method to all elements in the matrices. Hence, this would overestimate the probability of the unseen events, since we are gaining only few counts per sample.

Thus, we apply smoothing that better deals with low counts by a modified[8] *Good-Turing discounting* (1953) method before normalizing into probabilities, we estimate the probability for n-grams with zero counts and occurrence $N_{C=0}$ by looking on the number of n-grams that occurred with a global minimum count $N_{C_{min}>0}$ and calculate counts $N_{unseen} = N_{C_{min}>0}/N_{C=0}$.

The smoothed counts yield $\omega_{i,j} = (\forall N_{i,j} = 0 : N_{unseen}) \cup N_{i,j}$, and therefore, the transition probabilities τ for the full bigram yields

$$\tau_{(2)} = \hat{p}(\omega_{i,j}) = \omega_{i,j} / \sum_{k=1}^{4} \omega_{i,k} \mid i,j = \{1 \ldots 4\} \tag{3}$$

Similarly to the calculation of the (4 by 4) bigram transition matrix $\tau_{(2)}$, both, a (4 by 4 by 4) trigram transition matrix $\tau_{(3)}$, defining threefoldjunction/junction-type conditional probabilities, and a (4 by 4 by 4 by 4) 4-gram transition matrix $\tau_{(4)}$, defining fourfoldjunction/junction-type conditional probabilities, are calculated.

However, in order to model object prototypes by data of the transition matrices, we defined to have recipes of object prototypes realized by a probabilistic finite state automaton, defined according to [9, 2] $\{Q, \Sigma, \delta, \tau, S_0, F, \varphi\}$, with Q as a finite set of states, Σ the Alphabet, $\delta : Q \times \Sigma \mapsto Q$ the transition function, $\tau : Q \times \Sigma \mapsto]0,1]$ the transition probabilities, S_0 the initial state, $F \subset Q$ is a subset of final states from the set Q and $\varphi : Q \times \Sigma \mapsto]0,1]$ the probability for a state to be final.

In Section 4, we define the states of the automaton of an object at hand, by Bayes' belief nets with the probabilities calculated so far.

4 Belief Networks

Belief networks model firstly the independence relationships between groups of random variables and secondly reflect their topology graphically in a directed acyclic graph (DAG). The edges of the DAG show the conditioning variables in their expansions and represent the recipes for object construction. As in our approach the network starts in S_0 at an arbitrary junction, and the DAG gets assigned directions only in order to satisfy Bayes chain rule, it may end in an also arbitrary final state $F_{Terminate}$. Thus, it gets possible to remodel the belief network for optimization purposes.

In Figure 5, four results of MC simulations of a cube are given: in the first and second column, the views and their plane graphs are shown; the belief nets with coloring the

[7] also referred to as *Laplace's Law*

[8] The modification is to use the minimum count not equal zero rather than a count equal 1 [8]

Table 1: Bigram transitions according to a Monte-Carlo single run sampling of Figure 5.

Typ	Training Counts $N_{i,j}$					Smoothed Counts $\omega_{i,j}$					Transition Probabilities $\tau_{(2)} = \hat{p}(\omega_{i,j})$				
		L	W	Y	T		L	W	Y	T		L	W	Y	T
Te	L	3	6	0	2	L	3	6	0.143	2	L	0.269	0.538	0.013	0.179
tra	W	6	5	3	0	W	6	5	3	0.143	W	0.424	0.354	0.212	0.010
hed	Y	0	3	0	0	Y	0.143	3	0.143	0.143	Y	0.042	0.875	0.042	0.042
ron	T	2	1	0	0	T	2	1	0.143	0.143	T	0.609	0.304	0.044	0.044
Cu	L	4	12	0	8	L	4	12	0.125	8	L	0.166	0.497	0.005	0.332
be	W	12	0	6	0	W	12	0.125	6	0.125	W	0.658	0.007	0.329	0.007
	Y	0	6	0	0	Y	0.125	6	0.125	0.125	Y	0.020	0.941	0.020	0.020
	T	8	0	0	2	T	8	0.125	0.125	2	T	0.780	0.012	0.012	0.195
Dod	L	12	30	0	4	L	12	30	0.600	4	L	0.258	0.644	0.013	0.086
eca	W	30	2	16	0	W	30	2	16	0.600	W	0.617	0.041	0.329	0.012
hed	Y	0	16	18	2	Y	0.600	16	18	2	Y	0.016	0.437	0.492	0.055
ron	T	4	0	2	0	T	4	0.600	2	0.600	T	0.556	0.083	0.278	0.083

nodes by its costs[9] are shown in column three. When it turns out that there is a trigram used by the network, we use perplexity to test if we can minimize the order of the n-grams by recreating the network with changing link directions and preserving joint probability (see Figure 5-column four). Perplexity is a related measure of the uncertainty of a language event. The perplexity of a language model is the reciprocal of the geometric average of the symbol probabilities of a test set $\Omega = \{\omega_1, \omega_2, \ldots, \omega_N\}$ of the predictions [8]:

$$PP(\Omega) = \left[\prod_{i=1}^{|\Omega|} p\left(\omega_i | \omega_1 \ldots \omega_{i-1}\right) \right]^{-\frac{1}{|\Omega|}} . \quad (4)$$

Thus, the higher the conditional probability of the symbol sequence, the lower the perplexity, and therefore, minimizing the perplexity is the optimization criteria used.

4.1 Training of the Model

For training the model, we split given data in three disjoint sets: *(i) the training set* $\mathbb{T}$, used for stepwise learning; *(ii) the validation set* $\mathbb{V}$, used to verify an order change of the model; and *(iii) the test set* $\mathbb{A}$, used to assess the performance of the model.

Hence, in every MC *training step*, we firstly select a training object t randomly from the training set $\mathbb{T}$. Secondly, we repeat for $i = 1 \ldots N$ times a random selection of viewpoint positions $P_{P(x,y,z)}(t)$ around each test object t, and calculate transition counts for bigrams, trigrams, and 4-grams of the junction to junction connectivity from the set of junctions $\mathbb{J} = \langle W, L, Y, T \rangle$ as defined as in Section 3.1. Thirdly, we apply smoothing and calculate the transition probability matrices $\tau_{(2)}$, $\tau_{(3)}$, and $\tau_{(4)}$ that are used to define the belief network of the new recipe candidate r, representing the conditional probabilities of the dependencies between all junctions of the object given. Finally, it is checked if a variant recipe can be found that provides the same or lower perplexity with using lower ordered n-grams, which is then selected for replacement of the recipe at hand. This new order n-gram recipe candidate is verified with the validation set $\mathbb{V}$ by the *verification step* in order to preserve performance, and the selection result is stored as a new recipe r to the set R of known recipes.

The *inference step* is designated to find the best recipe $r \in R$ that fits to a given observation O. We find a solution by calculating the likelihood $p(O, r)$ and classifying the observation O into a class that maximizes the posterior probability

$$p(r^*, O) = max_i \frac{p(O, r_i) p(r_i)}{p(O)} \quad (5)$$

Since $p(O)$ is independent from r, we only have to consider the nominator of Equ.5 to find the optimum

$$r^* = \text{argmax}\, p(r|O) = \text{argmax}\, p(O|r) p(r) \quad (6)$$

Despite to common use of dropping $p(r)$, we though use $p(r)$ to ensure high selectivity in cases where the observation O is only partly given. The inference step is validated by the test set $\mathbb{A}$ within an confidence of 95%.

[9] The computationally costs are increasing from using bigrams, to trigrams and 4-grams for conditioning.

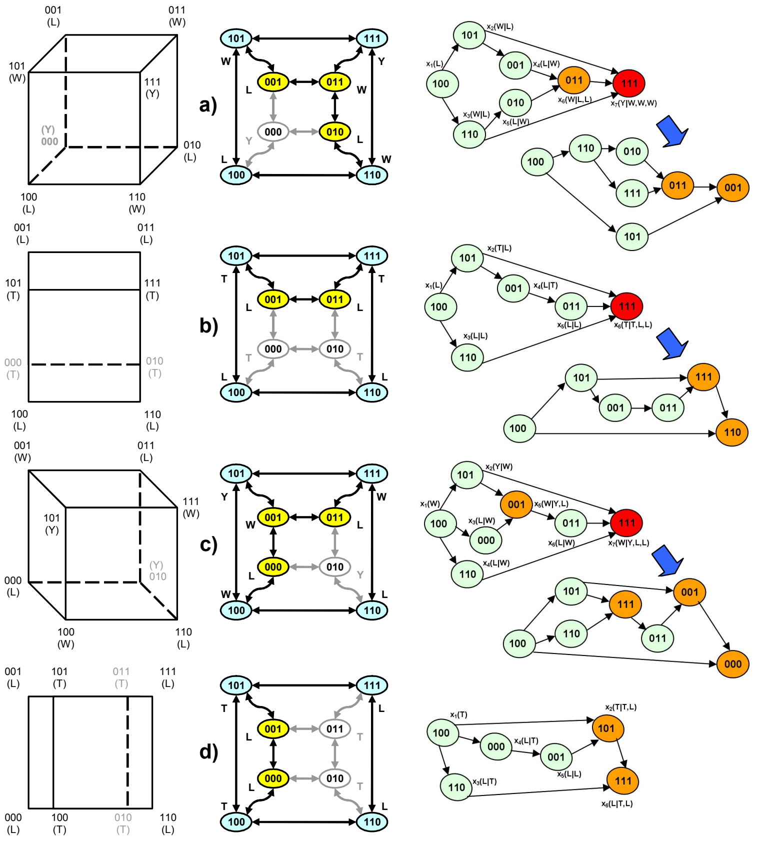

Figure 5: A viewpoint sequence a)..d) of a training simulation. From left to right: a) the projective views are given; b) the planar graphs; c) the Bayes' belief networks, showing trigram-realizations and their bigram replacements.

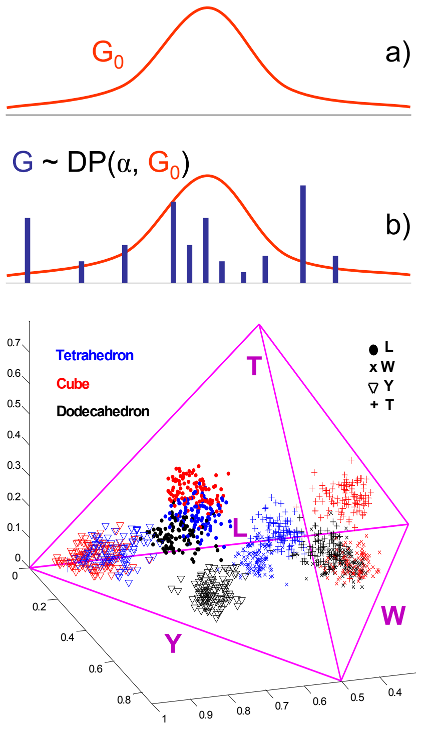

Figure 6: Defining a Dirichlet process. Top: a) using a Gaussian prior for the base measure G_0; b) sampling from G_0 yields the discrete G. Bottom: the 4-simplex shows the n-gram partitioning of the DP process.

4.2 Defining a Dirichlet Process

The n-gram modeling of the polytopes results in a bag of distributions, hence, with one for every polytope type observed. Thus inference, such as given by (6), can be generalized by modeling the resulting bag of distributions by a distribution of distributions.

Hence, given a space Ω, a σ-field of subsets $\mathfrak{B}$, forming a measurable space $(\Omega, \mathfrak{B})$ – then a stochastic process P, partitioned by elements $A_i \in \mathfrak{B}$ is termed a Dirichlet process (DP) on $(\Omega, \mathfrak{B})$ [7]. It is the distribution of a random probability measure $G_{(\Omega,\mathfrak{B})}$ such that for any finite partition $(A_1, ..., A_K) \subseteq \Omega$ we write $G \sim DP(\alpha, G_0)$, where G_0 is the base measure of G and α is a scaling parameter.

Figure 6-top a) shows an example for choosing a base distribution; b) sampling from the distribution; and at the bottom the m=4 simplex, where the DP yields a clustering effect on $(\Omega, \mathfrak{B})$ with $G \ \sim DP(\alpha, G_0)$, a resulting n-grams clustering of the experimental setup.

5 Conclusions and Future Work

We have shown the aggregation of structural information by perceptual grouping; we defined a n-gram graphical model and used belief networks to model object reconstruction recipes with Monte-Carlo simulation training and real data. Thus, we have developed further our past approaches [3, 2], which have used so far predefined recipes, by implementing statistical learning of object recipes, and a Dirichlet process was defined to cover the bag of distributions learned by one construct.

Moreover, from this work, it follows that the combination of both, the planar graph representation proposed in [2], and the statistical approach of grammatical inference by the n-gram model of this work, performs better than the subgraph matching approach proposed therein, when they are compared in terms of runtime complexity and learning efficiency. However, as simple as humans may investigate an unknown object in order to understand its topology, the approach supports such an investigation by combining a sequence of images from different viewpoints for learning of implicit object topology.

The next step will be to feed a self-organizing map by the outcomes of the belief net in order to implement associative neural categorical object learning.

References

[1] F. Ackermann, A. Maßmann, S. Posch, G. Sageder, and D. Schlüter. Perceptual grouping of contour segments using markov random fields. *Patt. Rec. and I. A.*, 7(1):11–17, 1997.

[2] P. M. Goebel and M. Vincze. A cognitive modeling approach for the semantic aggregation of object prototypes from geometric primitives. In *Proc. ACIVS, Delft Univ. (NL)*, 2007.

[3] P. M. Goebel and M. Vincze. Vision for cognitive systems: A new compound concept connecting natural scenes with cogn. models. In *LNCS, INDIN 07, (A)*, 2007.

[4] P. M. Gruber and J. M. Wills. *Handbook of Convex Geometry.* Elsevier, 1993.

[5] C. Higuera. *Data Complex. in Pattern Recogn., Eds. M. Basu and Tin Kam Ho.*, chapter Data Complex. Issues in Gramm. Inference. Prentice Hall., 2006.

[6] F. Jelinek and R. L. Mercer. *Interpol. estimation of Markov source param. from sparse data. In Pattern Recognition in Practice, E. S. Geltsema and L. N. Kanal, eds.* North Holland, 1980.

[7] N. L. Johnson, S. Kotz, and N. Balakrishnan. *Discrete Multivariate Distributions.* Wiley, 1997.

[8] D. Jurafsky and J. H. Martin. *Speech and Lang. Proc.: An Intro. to Natural Lang. Proc., Comp. Linguistics and Speech Recognition.* Prentice Hall., 2003.

[9] C. Kermorvant and P. Dupont. Stochastic grammatical inference with multinomial tests. In *ICGI '02: Proc. on Grammatical Inf.* Springer, 2002.

[10] K. Koffka. *Principles of Gestalt Psychology.* Hartcourt, NY, 1935.

[11] A. Levinshtein, C. Sminchisescu, and S. Dickinson. Learn. hierarch. shape models from examples. In *CVPR*, 2005.

[12] D. G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence.*, 31(3):335–395, 1987.

[13] D. Marr. *Vision: A Computational Approach.* Freeman & Co., 1982.

[14] S. Procter. *Model-Based Polyhedral Object Recognition Using Edge-Triple Features.* PhD thesis, University of Surrey, UK., 1998.

[15] S. Sarkar. and K. L. Boyer. Perceptual org. in comp. vision: A review and a proposal for a class. structure. *IEEE Trans. on Systems, Man, and Cybernetics.*, 23(2):382–399, 1993.

[16] W. Stöcher and G. Biegelbauer. Auto simultanous calib. of a multi-view laser stripe profiler. In *ICRA '05: Proc. of IEEE Int. Conf. on Rob. and Aut.*, 2005.

[17] A. Witkin and J. Tenenbaum. On the role of structure in vision, in human and machine vision. In *Proc. Conf. on Human Vision.*, pages 481–543. API, 1983.

[18] M. Zillich. *Making sense of images: Parameter-free perceptual grouping.* PhD thesis, ACIN, Automation & Control Institute, TU-Vienna., 2007.

[19] S. W. Zucker. *Vision, Brain, and Cooperative Computation (M.A. Arbib and A.R. Hanson, Eds.)*, pages 231–262. Cambridge, MA: MIT Press., 1987.