MEANING AND EFFICIENCY IN THE GESTALT-SYSTEM
An Approach to Automatic Image Description

E. Michaelsen, M. Arens, L. J. Doktorski, P. Ding
FGAN-FOM, Gutleuthausstrasse 1, 76275 Ettlingen, Germany[1]
mich@ fom.fgan.de

U. Stilla
Photogrammetry and Remote Sensing, Technical University of Munich, Arcisstr. 21, 80280 Munich, Germany
Uwe.Stilla@bv.tu-muenchen.de

Keywords: Image understanding, pattern recognition, structural and syntactical approach, production systems.

Abstract: Knowledge-based image recognition and description systems have to balance soundness versus efficiency. For mining purposes in particular a theory of meaning – i.e. formal semantic or ontology – has to be given. However, efficiency is also most important for practicability. This contribution uses production systems. The semantics are analysed by means of confluence. Productions are used in an accumulating way instead of reductive. The interpretation scheme given allows breaking reasoning at any time with an approximate interpretation as result. Two example systems using this interpreter are discussed.

1 INTRODUCTION

Image mining – if not performed with simple features such as colour histograms – needs meaning, i.e. semantics. In contrast to other recognition tasks on pictorial data not only a class label is to be assigned to an image. Instead, pictures need to be described in a structural way and the nodes of this structure need to revere to concepts about facts, objects, etc. of the world. Thus automatic reasoning is prerequisite to real image mining.

One classical syntactic way of automatic generation and reduction of meaningful structure is given through formal languages. A moderately sized system of production rules can analyse huge input data sets and reduce them to meaningful structures (reduction trees). However, such systems though being fast on strings tend to lead to unpredictable computational effort on images or image sequences.

A common way to meet efficiency constraints and still derive the structure of the input image is to restrict the search space during production application by means of an assessment measure for the utility of single productions with respect to the desired goal.

This contribution explores the possibility of using production systems in an approximate way: given assessment measures for the objects and productions, the question arises to which point production rules have to be applied to a given input image in order to meet efficiency constraints and still derive *the meaning* of that image.

2 RELATED WORK

Semantic networks have been proposed for automatic image or video content analysis by Niemann (1989) and his school. This work has led to the creation of the ERNEST system (Sagerer, 1982). The latter presumably ventured the deepest investigation concerning the semantics. The meaning possibilities for the productions used in our work (e.g. in sections 3.2 and 4) have been investigated by Sagerer with much more rigor as admissible meanings of links in semantic nets. His work culminated in proven soundness from the logic level down to the epistemological level.

Matsuyama & Hwang (1990) constructed a production system named SIGMA featuring sophisticated control mechanisms – e.g. including meta rules. Examples are given from automatic

[1] www.fom.fgan.de

understanding of aerial images. Work on and with the similar SCHEMA system of Hanson and Riseman (Draper et al., 1998) has been pursued for decades by many researchers. Though being conceptually closely related to our scheme these systems were not intended to work efficiently albeit still approximately correct on large data sets.

Recently, the field of image and image sequence understanding has gained renewed attention under the term *Cognitive Vision* as discussed, e.g., in the Dagstuhl workshop (Christensen & Nagel, 2006).

3 STATE OF THE THEORY

In section 3.1 a brief overview on confluence in transition operations on configurations of objects is given. This is not a sufficient condition for sound semantics, but it is a necessary one. Explicitly listing admissible transitions is practically impossible. Instead production systems are used. These are introduced in section 3.2. Only productions of two specific normal forms are permitted which also helps to keep track of the meaning. Possible standard interpretations are listed. In Section 3.3 an interpreter system is given that can perform efficient inference based on a given production system. Usually, complete search is not feasible. So the system has to be capable of providing approximate solutions. Large parts of the theory are still missing which is discussed in section 5.

3.1 Confluence on Sets of Objects

Let *S* denote a finite set of *classes* such as *{***line***,* **longline***,* **angle***, ...,***building***}*. Similar to object oriented programming, each such class has a number of methods containing at least one method (constructor) that can construct objects of that class. Usually, a method to present or draw the object is compulsory, too.

A class is called *primitive* – and such is any object of that class – if none of its constructors uses information from any other object. Primitive objects in the context of computer vision are usually constructed by operations working directly on the input data, e.g., by segmentation of images or image sequences.

An object **o** of one of the classes is denoted as a pair *(s,d)* in the following, where *s* is a class name and *d* is a structure of numerical attributes containing the specific features of that object.

A *configuration* is a finite set *C* of objects. It represents a state of semantic analysis. The set of all possible configurations is called Ω. A *transition* $\rightarrow \in \Omega \times \Omega$ is a binary relation on Ω and defines transition chains of the form

$$C_0 \xrightarrow{*} C_n := C_0 \rightarrow C_1 \rightarrow \ldots \rightarrow C_n . \qquad (1)$$

Such a transition is called *confluent* iff

$$\forall A,B,C \in \Omega : \\ (A \xrightarrow{*} B \wedge A \xrightarrow{*} C) \Rightarrow \qquad (2) \\ \exists D \in \Omega : B \xrightarrow{*} D \wedge C \xrightarrow{*} D \ .$$

If the transition is confluent and terminating – i.e. all transition chains have finite length – this guarantees that for any (start-)configuration C_0 a *unique* final configuration $\hat{C}$ exists and can be reached after a finite number of transition steps. Without these properties, a transition cannot have semantic meaning. The final configuration $\hat{C}$ of a confluent and terminating transition contains all its possible meaning. For further reading on confluent systems we refer to (Janzen, 1988).

3.2 Productions

For practical applications transitions are given by productions. We restrict the formulation of productions to the two forms given below. A *production* p can be defined as

$$p : (o_1, o_2) \xrightarrow[\varphi]{\pi} r \\ \text{with } o_1 = (s_1, d_1) ,\ o_2 = (s_2, d_2) \qquad (3) \\ \text{and } r = (s_r, d_r) \text{ where } s_1, s_2, s_r \in S$$

The other possible formulation of a production p is given by

$$p : (o_1, \ldots, o_n) \xrightarrow[\varphi]{\pi} r \\ \text{with } \forall i = 1,\ldots,n : o_i = (s, d_i) , \qquad (4) \\ \text{and } r = (s_r, d_r) \text{ where } s, s_r \in S \ .$$

In both cases, π is a constraint formulated on the attributes of the objects $o_1, \ldots$ involved on the left hand side of the production and *r* denotes some (resulting) object of a class within *S*, which is created using the constructor method φ. Note that in (3), the two left hand side objects can be of different class, whereas the arbitrary number of left hand side objects in (4) all must be of the same class. It follows, too, that *r* cannot be an object of a primitive class in either case.

The meaning of productions of the form defined in (3) can be threefold:

1) the two left hand side objects are *parts of* the object *r* such as the two **longline** objects in production $p_{1,3}$ are the two legs of the object **angle** (see section 4.1),

2) one of the left hand side objects is interpreted as being an object *r* provided that the other object is given as *context* – such as in production $p_{1,6}$ the **longline** object close to an object **spot** leads to the assumption of an object **row** with one element and

3) the two left hand side objects are *concretizations of* the object *r*. An example is the construction of a 3D-object from two 2D-view objects. Such productions have been used in (Stilla & Michaelsen, 1997). In this case π resembles an epipolar constraint and φ implements a triangulation. Concretization relies critically on external knowledge, e.g., about the conditions while taking the pictures – such as in the stereo case the internal and external camera parameters.

The role of productions of the form defined in (4) is that of combining evidence from multiple measurements into a single hypothesis. Examples are given by productions $p_{1,1}$, $p_{1,2}$, $p_{1,5}$, $p_{2,2}$, $p_{2,4}$, $p_{2,6}$. Often the required linear and metric structure can only be given in a local tangential linear space, where π is based on, e.g., Euclidean distance and φ performs averaging or other forms of least squares optimizations. Such an example is given in $p_{2,4}$, $p_{2,6}$. Instead of averaging over the attribute values of the direct predecessors it is often more exact to perform the minimization on the attribute values of the primitive predecessors. Sometimes these kinds of productions resemble Hough-techniques or other accumulator based methods – such as in $p_{1,1}$.

A Production *p* can modify a given configuration *C* in two different ways: a *Reduction* replaces left hand side objects with the inferred right hand side object within the configuration *C*:

$$C \xrightarrow{p,reduce} D \text{ with } D = C \setminus \{o_1,\ldots,o_n\} \cup \{r\} \tag{5}$$

where $o_1,\ldots,o_n$ denote the left hand side objects of the production *p* and *r* being the right hand side object of *p. Accumulation* instead leaves the left hand side objects in *C* and just adds the right hand side object to that configuration:

$$C \xrightarrow{p,accumulate} D \text{ with } D = C \cup \{r\}. \tag{6}$$

The term 'reduction' has as antagonist 'generation'. Both directions can be used in a syntactic manner either to parse configurations or to construct them – i.e. for pattern recognition and for computer graphics. Most structural recognition approaches – such as the production systems SIGMA or SCHEMA or contemporary work with PROLOG rules or the semantic net methods – can be mapped to reducing transitions. However, examples can be given that have two different final reductive interpretations – the well known picture puzzles. Therefore confluence is not given in such systems. There may be more than one final consistent assignments of meaning, which jeopardizes the meaning of such production system. Reduction has obviously good halting properties – because with each step the configuration is becoming smaller.

With *Accumulation* only possible hypotheses are piled up. Decisions are avoided. Due to the properties of set union transitions defined as accumulation are confluent (Lütjen 2001). However, this only gains sound meaning if the interpretation rests on probability calculus. For each object **o** a special attribute is required that gives an assessment for its probability. This assessment must be sound with respect to the expectations of a statistical model. So instead of speaking of objects it makes more sense to speak of hypotheses or cues here. There is uncertainty in this way of dealing with a production system, but it leads to a unique final interpretation where, e.g., the two different meanings of a picture puzzle can coexist with their probability attributes summing up to at most one.

3.3 A Production System Interpreter

Listing all admissible productions for a given configuration *C* and a set of productions *P* is not feasible in most cases. Instead hypotheses are formed for each object $\mathbf{o} \in C$. Such a hypothesis has the form of a triple $h = (h_a, h_o, h_p)$ where $0 \le h_a \le 1$ is an assessment of **o**, h_o is an index or pointer allowing access to **o** and $h_p \in P \cup \{nil\}$ denotes an index or pointer to a production (initially set to *nil*). Based on this notation, the following interpretation cycle can be formulated:

1) The hypotheses are ordered according to the assessment h_a in a queue. The best *n* hypotheses are chosen. For each of these 2) is performed.

2a) If h_p = *nil* then all those $p \in P$ are chosen, which contain an object of the same class as h_o = **o** on their left hand side. *h* is then replaced by clones of it with the corresponding indices h_p, where the assessment of those newly introduced hypotheses may be altered according to different *importance* of different productions. h_o is then called *triggering object* of *h*.

2b) Else there is $h_p = p$ with a corresponding right hand side object r. With this object there is a construction method φ which takes the triggering object h_o as fixed. It queries the configuration C for the missing parts of the left hand side of the production h_p which fulfil the constraint π. Thus, the construction method φ may result in none, one or even many possible objects r. For all of these objects, new initial hypotheses are added to the queue. Thus – while the number of objects in C is always rising – the number of hypotheses in the queue may rise or fall.

3) While the queue is not empty, one may continue with step 1) or stop and return the current configuration as result. This result will be identical to the final unique solution, if the queue is empty. Otherwise, the current configuration will only resemble an approximate solution.

Lütjen (1986) published this interpreter 20 years ago. Since then, work in this field has shown that the success of the production system in terms of reaching a meaningful configuration close to the unique solution heavily depends on proper assessment criteria. However, these are needed anyway in the accumulating association scheme to have a meaning at all – as discussed in section 3.2. Moreover, the assessments have to be related to probabilities. There has been work on this issue as well (Michaelsen & Stilla, 2002).

There have been different implementation projects – assembler-based VMS code supporting special hardware, UNIX-based KBV derivates (an offspring of the SCHEMA working group), and simple machine independent MATLAB code. Currently a new object oriented implementation using Microsoft `.NET` and `C#` is under way. For this endeavour we use the acronym **GESTALT** (**G**rouping **E**vidence **S**ystem for **T**reatment of **A**lternatives in **L**ayered **T**ask Solvers) which has been mainly motivated by the example given in section 4.1.

4 TWO EXAMPLE SYSTEMS

In order to show the scope of applicability of the theory and system outlined in Section 3 we briefly describe two quite different example systems. The first example consists of a production system for the detection of buildings in SAR-images. The second system was designed to determine the mutual geometry of image pairs, which is needed, e.g., for image-based navigation of unmanned aerial vehicles.

4.1 Building Recognition in SAR Images

This production system has been constructed to recognize man-made structures such as buildings in high resolution SAR images and has been published in (Michaelsen et al., 2006a). It consists of the following seven productions:

$$\begin{aligned}
&p_{1,1} : (\text{line},...,\text{line}) \xrightarrow[\textit{regression}]{\textit{colinear}} \text{longline}\\
&p_{1,2} : (\text{pixel},...,\text{pixel}) \xrightarrow[\textit{average}]{\textit{adjacent}} \text{spot}\\
&p_{1,3} : (\text{longline},\text{longline}) \xrightarrow[\textit{inter sec t}]{\textit{adjacent}} \text{angle}\\
&p_{1,4} : (\text{angle},\text{angle}) \xrightarrow[\textit{axis}]{\textit{symmetric}} \text{symmetry}\\
&p_{1,5} : (\text{symmetry},...,\text{symmetry}) \xrightarrow[\textit{average}]{\textit{adjacent}} \text{cluster}\\
&p_{1,6} : (\text{longline},\text{spot}) \xrightarrow[\textit{initialize}]{\textit{adjacent}} \text{row}\\
&p_{1,7} : (\text{row},\text{spot}) \xrightarrow[\textit{append}]{\textit{fitting}} \text{row}
\end{aligned} \tag{7}$$

$p_{1,3}$ is the standard example for the normal form given in (3). For a triggering object **longline** search areas around the end points of that line are constructed and queried for other objects **longline** ending there which must have a different orientation. Objects **angle** inherit most attributes from the predecessors, however, one new location is computed by intersecting the two constituting lines. $p_{1,2}$ is the standard example for the normal form given in (4). For a triggering object **pixel** a search area around it is constructed and searched for other objects **pixel**. The resulting object **spot** is positioned by averaging and thus has sub-pixel accuracy.

Note that this system contains a cyclic part-of structure – production $p_{1,7}$. A row can be a part of a larger row. Such recursive structure is common for string rewriting grammars. It allows generation or reduction trees of arbitrary depth, and thus the definition of an infinite language by finite means. Here the production $p_{1,2}$ was the straight forward way to capture the appearance of bright spots arranged in long straight rows of equal spacing where building facades are seen in high-resolution SAR images.

$p_{1,4}$ captures another important property of buildings – they tend to have some symmetric structure. Often this is a non-local property – so that building parts quite far apart from each other still give the same symmetry axis. This is an important cue for man-made architecture. The clustering of symmetry axis by production $p_{1,5}$ requires an appropriate setting of the coordinate system in which the axis equation attributes and the metric for it are given. Locality has to be balanced against orientation.

It is obvious that **pixel** and **line** are the primitive classes of this system. While **line** objects can be extracted using filter operations designed for enhancement of contours or line-like image structures, **pixel** objects are found using a bright spot filter based on inequality operations. The decision whether such an object is present in a given image location is based on the filtered image and a threshold.

4.2 Search for Mutual Image Geometry

This production system for estimating the mutual geometry of an image pair has been published in (Michaelsen et al. 2006b). The internal camera parameters are assumed to be known. It consists of the following six productions:

$$\begin{aligned}
&p_{2,1}:(\text{corresp},\text{corresp})\xrightarrow[\mathit{scale}\wedge\mathit{rot}]{\neg\mathit{adjacent}}\text{pair}\\
&p_{2,2}:(\text{pair},\ldots,\text{pair})\xrightarrow[\mathit{average}]{\mathit{adjacent}}\text{similarity}\\
&p_{2,3}:(\text{pair},\text{pair})\xrightarrow[\mathit{inhomogenous}]{\neg\text{colinear}}\text{quad}\\
&p_{2,4}:(\text{quad},\ldots,\text{quad})\xrightarrow[\mathit{svd}]{\mathit{adjacent}}\text{homography}\\
&p_{2,5}:(\text{corresp},\text{quad})\xrightarrow[\mathit{Nister\,5\,point}]{\neg\text{colinear}}\text{quint}\\
&p_{2,6}:(\text{quint},\ldots,\text{quint})\xrightarrow[\mathit{Nister\,5\,point}]{\mathit{adjacent}}\text{ematrix}
\end{aligned}\tag{8}$$

$p_{2,1}$ would be a standard example if the relation *adjacency* were not negated. For high accuracy of a similarity transform estimate the two objects **corresp** should be far apart from each other. If the expected number of objects fulfilling the relation *adjacency* with respect to the triggering object is one out of *n* the expectation for the opposite is *n-1* out of *n*. The same holds for productions $p_{2,3}$ and $p_{2,5}$.This production system implements a combinatorial enumeration. If run to the final confluent interpretation it will have constructed $O(n^5)$ objects **quint** where *n* is the number of primitive objects. This is usually not feasible. The process has to be terminated before - with an only approximate solution. From experience this solution is good enough and often better than a RASNAC solution obtained with comparable effort.

Productions p_{23} to $p_{2,6}$ contain state-of-the-art computer vision geometry – namely the inhomogeneous solution for a planar homography from four point correspondences, singular value decomposition, and the 5 point algorithm for minimal and non-minimal samples of correspondences as introduced by Nister (2004). Note that *adjacency* in the space of homographies as well as in the space of essential matrices requires appropriate metrics – both are 3x3 matrices, but none is a vector space. Homographies have eight degrees of freedom, while essential matrices have five. Note also that Nister's algorithm solves a polynomial of order ten – giving between zero and ten solutions. Thus, production $p_{2,6}$ usually produces more than one new object.

It is obvious that **corresp** is the only primitive class of this system. Such a pointwise correspondence cue between image structures in an image pair can be acquired by many different methods – including various trackers and simple other production systems. Correspondence only makes sense at image locations where no aperture problem occurs which is usually tested by squared averaged gradient filters. A particular image correspondence object can always be assessed by the *similarity* of the appearance in the frames (i.e. some correlation), the *precision* of its localization and *a priori knowledge* coming e.g. from a Kalman filter prediction.

5 DISCUSSION AND CONCLUSIONS

A proper mathematical theory often consists of the following parts: 1) Define the space of solutions and prove the existence of a solution; 2) prove uniqueness of the solution; 3) give a metric in the space of solutions; 4) give an algorithm iterating to the solution and prove convergence; 5) prove bounds on the distance to the solution at a given iteration. The theory of 'piling up meaning' for pictures or videos outlined in this contribution is very preliminary yet. There are some results concerning steps 1) and 2). Most work has been spent on step 4). Several examples of systems behaving well in practice were built. However, with step 3) not being completed we cannot expect that the work on the applications and on step 4) rests on secure ground – and step 5) though being very important for applicability is in far distance.

There is still a severe problem with the meaning of the accumulative use of productions: As pointed out in section 3.2. we have to give evidence attributes with each of our cue objects. They should in fact be probabilities. This, however, requires normalization to one. But the objects in a configuration accumulated so far cannot contain or reach information from other objects in this set that are not preceding them. Giving up this principle would jeopardise the efficiency and thus practicability.

We have presented two example systems solving quite different tasks and operating in a different way. This demonstrates exemplarily that the GESTALT interpreter is a useful tool for a broad range of recognition applications. However, it cannot solve the problems *alone*.

Preliminary to its application iconic image processing tools and segmentation procedures for the primitives are required. It is hard to mend on the interpretation level what has been missed and spoiled in the processing chain before.

Moreover, for many tasks, a final or approximate interpretation configuration is not sufficient as result. Further processing and decisions are needed after the interpretation has been terminated. This may be quite simple, such as giving the *best* object **ematrix** as result in the system presented in section 4.2. Or a little more complicated – such as giving a threshold for the quality of the objects **ematrix** and in case there is no better one take the best object **homography**. Such system may also interact with a Kalman filter for flight control of an unmanned aircraft in both directions – getting prior information from it and handing measurements of epipole and rotation to it.

The decision system following on top of the saliency recognition system outlined in section 4.1. may either be an AI-reasoning system or also a human interpreter. Both require a sophisticated interface.

REFERENCES

Christensen, H. I., Nagel, H.-H. (eds), 2006.*Cognitive Vision Systems*, Springer, Berlin LNCS 3948, pp. 183-198

Draper, B., Collins, R., Brolio, J., Hanson, A., Riseman, E., 1989. *The Schema System,* IJCV, Vol. 2 pp. 209-250.

Janzen, M., 1988. *confluent String Rewriting,* Springer, Berlin.

Lütjen, K., 1986. *Ein Blackboard-basiertes Produktionssystem für die automatische Bildauswertung.* In: Hartmann, G. (ed.) *Mustererkennung 1986,* DAGM 1986, Springer, Berlin, Informatik Fachberichte 125, pp. 164-168.

Lütjen, K., 2001. *Systeme und Verfahren für strukturelle Musteranalysen mit Produktionsnetzen,* Diss., Univ. of Karlsruhe, institute for communications engineering, ISSN: 1433-3821.

Matsuyama, T., Hwang, V. S.-S., 1990. *Sigma a Knowledge-based Image Understanding System*, Plenum Press, New York.

Michaelsen E., Stilla U., 2002. *Probabilistic Decisions in Production Nets: An Example from Vehicle Recognition.* In: Caelli T., Amin A., Duin R. P. W., Kamel M., Ridder D. de (eds). *Structural, Syntactic and Statistical Pattern Recognition SSPR/SPR 2002,* Springer, Berlin, LNCS 2396, pp. 225-233.

Michaelsen E., Soergel U., Thoennessen U., 2006a. *Perceptual Grouping in Automatic Detection of Man-Made Structure in high resolution SAR data.* Pattern Recognition Letters, Vol. 27, No. 4, pp. 218-225.

Michaelsen, E., von Hansen, W., Kirchhof, M., Meidow, J., Stilla U., 2006b. *Estimating the Essential Matrix: GOODSAC versus RANSAC.* ISPRS Symposium on Photogrammetric Computer Vision (PCV 2006).

Niemann, H., 1989. *Pattern Analysis and Understanding,* Springer, Berlin.

Nister, D., 2004. *An Efficient Solution to the Five Point Relative Pose Problem,* IEEE PAMI, vol. 26, no. 6, pp. 756–769.

Sagerer, G., 1982. *Darstellung und Nutzung von Expertenwissen für ein Bildanalysesystem,* Diss., Univ. of Erlangen-Nürnberg, Springer, Inormatik Fachberichte 104, Berlin.

Stilla U., Michaelsen E., 1997. *Semantic modelling of man-made objects by production nets.* In: Gruen A., Baltsavias EP., Henricsson O. (eds). *Automatic extraction of man-made objects fromaerial and space images (II).* Birkhäuser, Basel, pp. 43-52.