Mathware & Soft Computing 3 (1996) 403-413

Contributions to the Symbolic Processing of Segments in Computer Vision*

J. Cabrera† F.M. Hernández, A. Falcón & J. Méndez
Grupo de Inteligencia Artificial y Sistemas
Universidad de Las Palmas de Gran Canaria
Spain

Abstract

In this paper a processing methodology is introduced for the segment or intermediate level in the context of knowledge-based computer vision systems. The proposed methodology demonstrates how using simple Fuzzy Logic concepts it is possible to associate symbolic descriptions to the entities of this level. It provides with the basic mechanisms for performing symbolic computation, evidence combination, uncertainty management and spatial reasoning at the segment level.

Keywords: knowledge-based image segmentation, fuzzy logic, spatial relations.

1 Introduction

Computer Vision is aimed at studying and developing theories and algorithms to be used in artificial visual systems. During the last years there have been important changes in the design and development methodologies of such systems. This has been due, mainly, to the incorporation of methods and technology from the Knowledge Engineering field; and, also to a change in the conception of a Computer Vision system not as a passive element that senses information and processes it in some way, but as an element that is tightly coupled with its environment in a looping cycle of perception and action [1] [2]. Whilst most areas of Science and Technology deal with problems that can be solved on the frame of well-developed models and theories, in Computer Vision there is a lack of formal models that could be applied with confidence under general circumstances. The solution of such problems usually requires using procedures with multiple heuristics that produce results of partial validity with different degrees of uncertainty. The integration of these uncertain results to produce a result can be achieved using methods from Artificial Intelligence and Fuzzy Sets theory.

*This work has been partially supported by the Consejería de Educación del Gobierno de Canarias through project 93/069

†To whom correspondence should be addressed. E-mail: jcabrera@dis.ulpgc.es

Computer Vision Systems involve processes that act over elements of different typology at various levels of abstraction. The nature of these elements defines the grain of information to be processed, and can also be used to establish the three levels that normally integrate such systems, that is the pixel or low level, the segment or medium level, and the object or high level. At each of these levels it is possible to define symbolic categories or classes with associated degrees of uncertainty [3]. Specifically, the segmentation of images can be conceived as a process aimed at building a symbolic description of an image or scene in terms of segments or aggregates of pixels. The development of this conception produces a description level that uses the segment as the unit of information. In this framework, a segment in a scene is first described numerically. The measures obtained in the numerical domain are then used to derive the segment's degree of membership to symbolic classes [4]. These classes are normally fuzzy, so the symbolic descriptions must have an associated uncertainty factor [5]. The achievement of a final segmentation of an image is not a one-stop process, but it is better formulated as a refinement process guided by the current state of the segmentation. A control strategy for this process can be implemented using fuzzy control rules that are evaluated over a segment and its neighbors. Along this paper, we will describe first the architecture of an implemented segment processor that is capable of producing stable partitions, i.e. segmentations, using mechanisms of Knowledge Engineering and Fuzzy Logic [6]. Then we will concentrate on the analysis of the elements used to implement the control of this level and an application example will be described.

2 The segment level

A layered approach is an intellectual resource commonly used in the analysis and synthesis of complex systems. From the analytic point of view, the level concept helps in ordering as a hierarchy a reality made of facts. Whilst for the synthesis, it constitutes a way of simplifying the process using layers of reduced complexity. In a non-formal sense, a level can be defined as a set of symbols and a series of operators that act over the symbol set. In essence, what defines a level is the election of a symbol set that represents the entities of the domain, and an algebra or set of formalisms or operators that transform the symbols in the set.

At the segment level, the objective is to produce the partition of the image in a set of segments or pixel aggregations along with their respective degrees of membership to defined segment classes like $red - square$ or $straight - line$. The definition of these classes is established using a set of transformations acting over the partition of segments and their associated symbols. We shall consider a segment as a connected aggregate of pixels that is defined by some shape or property uniformness criteria. To clarify the terminology, we will name partition the list of segments that constitutes a complete tessellation of the image. The partition, as a data structure, includes not only the spatial definition of each segment but also the symbolic assignment of segments to classes and the spatial relations among segments.

Applying procedures of different types it is possible to make explicit certain

visual categories or classes related with pixels or aggregations of them. In general, the definition of these classes is not crisp but fuzzy [7] [5]. The fuzziness of their definition may arise from the geometric definition of the segment class, i.e. how a 'triangle' is defined, from the fuzzy nature of the property, e.g. to be 'light green', or even from the spatial relations among segments, e.g. to be 'above' or at the 'right'. Other symbolic diagnostics or classes can also be derived from logic predicates of previously defined classes.

It should be noted that using the segment as the information grain at this level poses two problems that do not exist at the pixel level. First, it is necessary to obtain an initial partition of the image to define the spatial extension of the segments. This operation is based on one or several maps of pixel diagnostics provided by lower layers of the system. The second question arises from the fact that it is normally not possible to obtain a 'correct' partition of an image in one step. Instead, it is normally more robust to allow the definition of the partition evolve as more information is discovered about the segments. This evolution or refinement of the partition can be based on operations of fusion and division of segments.

At this level, we consider the following elements as necessary to define a symbolic processing methodology:

a) **Features**. A feature, $f_i(s)$, is a numeric property of a segment s, obtained by means of Procedures, which are a type of Operator of numeric nature. A set of features defines a numerical domain where each segment can be described numerically using measures related to its morphology or tonal properties (color, texture, ...) of the pixels that constitute it.

b) **Operators**. There are three types of operators. Procedures and Classifiers operate in the numeric domain defined by the features with different objectives. As stated above, Procedures are used to compute segment features, whilst Classifiers allow to define symbolic classes from features. They project the numeric domain of features into a symbolic domain where the classes of segments are defined. The Rules constitute the third type of operator and permit the definition of new classes imposing logical conditions in terms of other defined classes.

c) **Classes of segments**. They represent a visual category in a symbolic domain. The degree of membership of each segment to a certain class as being 'Round' or 'Window' is expressed by a value in the interval [0-100].

d) **Transformations**. This special type of operators acts in the spatial domain to perform operations of fusion or divisions of segments. They accomplish for the necessity of modifying the spatial definition of the segments when certain conditions that control the state of the partition are verified. Functionally, they are conceived as Control Rules and Control Actions. When the conditions, stated in the fuzzy domain, of a Control Rule are verified an attached Control Action of some type is performed with the objective of modifying in some sense the spatial definition of the partition.

The structure of segment classes and their evaluation are defined from pixel maps of symbolic content. In a pixel map, the value assigned to each pixel is its degree of membership to a typology or class of pixels [3]. A detailed description of a processing architecture for the pixel level capable of producing maps of pixel diagnostics has been described in [10][13]. This processing architecture for the pixel level will be referred hereafter as pixel processor. Pixel maps are obtained by symbolization processes that use feature maps according to a methodology that is very similar to that utilized at the segment level. With these elements, it is possible to do symbolic computation, both at the pixel level and the segment level, as the representation is based on types and classes that can be considered as symbols. As a consequence, it is possible to utilize operations relaying in hierarchies of classes or conditions among them that can be formulated as rules or productions. The utilization of membership degrees to classes allows the usage of fuzzy classifiers and rules with certainty factors.

Segments' features are obtained from pixel maps or other features by means of procedures that operate in a numerical domain. Basically, the classifiers are discriminant and decision functions that close the gap between the numerical domain of features and the symbolic domain of classes. They use features as inputs to produce a symbolic output in the interval [0-100]. In essence this operator implements a classification procedure followed by a fuzzy decision function, as it is detailed below [8]. Let $f_i(s), i = 1, ..., n$, be a set of features evaluated for a segment s, $F(s)$ a functional computed using the feature set, and $C(s)$ the result output by the classifier. The computation done by the classifier can be summarized as:

$$C(s) = D_M(F(s))$$

Where D_M is a decision function and the functional $F(s)$ may be of any type (unitary, lineal, quadratic,...). For example, a quadratic functional would take the form:

$$F(s) = \sum_{i=1}^{n} \sum_{j=1}^{n} [f_i(s) - a_i] b_{ij} [f_j(s) - a_j]$$

Where $\{a_i\}$ and $\{b_{ij}\}$ are set of numeric coefficients. It is possible to use different types of decision functions that can take the form $D_M(u; p_1, ..., p_m)$, where M denotes the type of decision function and $p_1, ..., p_m$ are the parameters required for that function. Consider, for example, the sigmoid or S-decision function, $S(u; \alpha, \beta)$, [9] that is often used in fuzzy systems. The decision function $D_{SS}(u; \alpha, \beta)$ is defined as:

$$D_{SS}(u; \alpha, \beta) = 100 S(u; \alpha, \beta)$$

Where the α and β parameters define the interval of fuzziness between the full rejection (0) and total acceptance (100). Many other types of decision functions can be used within this simple scheme [10].

The definition of a segment class can be obtained from numerical features by means of classifiers, as the primary way for generating symbols that can be associated to the segments. However, it is also possible to built new symbols, i.e. segment classes, using rules. As Operators, rules are used to build the definition

of classes using logic constructions similar to Horn's clauses. A rule of this type contributes to the definition of a segment class and comprises the evaluation of a series of conditions that can accept linguistic qualifiers as 'very' and 'moreless' [9] [11]. The uncertainty is transformed and assigned from the classes, classifiers and rules that are involved in the definition of the new class. This process comprises two main stages. First, a primary diagnostic or *intrinsic value* is computed from the data sources, i.e. classifiers and rules. If the class being defined is an instance of a more general class, the intrinsic value is normalized to obtain a final value. This normalization is based on considering the instances as fuzzy subsets of the more general class. The process followed to combine the evidences produced by the classifiers, rules and instances involved in the definition of a class is summarized below.

Step 1: Compute all the classifiers and instances. Compute also all the classes that are used in the conditions of the rules.

Step 2: Compute all predicates included in the conditions of the rules.

Step 3: Compute the result of each rule by first evaluating the predicates of the rule and using the certainty or confidence of the rule. If $C_{rp}(s), p = 1, ..., h$, represent the evaluation of the predicate p of rule r and CR_r is the certainty of this rule in the interval [0-100], the result of the rule, $C_r(s)$ is given by:

$$C_r(s) = \frac{CR_r}{100} \min_{p=1,...,h} [C_{rp}(s)]$$

Step 4: Combine the results achieved by classifiers and rules to compute the intrinsic value. The intrinsic value, $IC(s)$, is obtained from the evidences returned by the set of m classifiers and rules, $C_{ck}(s), k = 1, ..., m$. This evaluation is performed incrementally as follows:

IC(s) = 0
for k=1 to m
    IC(s) = COM(IC(s), C_ck(s))

The $COM()$ function represents in a generic way some method of evidence combination. Our system incorporates actually two alternative methods, termed 'Logic' or 'Evidence', that implement two simple and often used evidence combination laws:

$$COM(u, v) = \begin{cases} u + v - \frac{uv}{100} & \text{if 'Logic'} \\ 50[EV(\frac{u}{50} - 1, \frac{v}{50} - 1) + 1] & \text{if 'Evidence'} \end{cases}$$

where $EV(a, b)$ is,

$$EV(a, b) = \begin{cases} a + b - ab & \text{if } a \geq 0 \wedge b \geq 0 \\ a + b + ab & \text{if } a \leq 0 \wedge b \leq 0 \\ \frac{a+b}{1-min(|a|,|b|)} & \text{if } (a > 0 \wedge b < 0) \vee (a < 0 \wedge b > 0) \end{cases}$$

Figure 1: Architecture of the segment processor and domains at this level

The 'Evidence' law uses the domain [-1,1] and is based in the Theory of Evidence as is utilized by some knowledge-based systems [12].

Step 5: Normalize the intrinsic value using the values achieved by classes that are instances of the class being computed. Let $C(s)$ be the class value, and $C_{ik}(s), k = 1, ..., n$, the diagnostics of the n instances, then the normalization is performed as:

$$C(s) = \max(IC(s), \max_{k=1,...,n}[C_{ik}(s)])$$

In next section, we will briefly introduce the concept of Segment Processor [13] to provide with an overview of the software or processing architecture at the segment level.

2.1 Architecture of the Segment Processor

From the point of view of functionality, the Segment Processor's architecture (Figure 1) is organized in three blocks, each one addressing a specific objective. The first of these blocks is charged with the obtainment of the initial partition. This is the objective of the presegmenter module that is part of the so-called Bottom-Up (BU) processor. Different methods can be used to achieve this goal so the presegmenter module has been conceived as an interchangeable part within the Segment Processor. The second functional block has the goal of computing the segment diagnoses. This block is structured by the distinction between a numerical domain, where the segments are described by features, and another symbolic domain based on classes. The third functional block is located at the Top-Down (TD) processor and manages the diagnostics requests and controls the Segment Processor. This control includes evaluating the partition's state from a set of rules that may trigger actions to modify the spatial definition of the segments.

The BU processor receives as input data pixel diagnostic maps, or pixel classes, previously requested by the TD processor from the Pixel Processor; and produces symbolic diagnostics for the Partition's segments as output data. Its action has two different phases: acquisition of the entities through the definition of the initial partition, and the symbolic description of the segments obtained, in terms of segment classes. The TD processor receives diagnostic requests about segments, and transforms these requests into sequences of commands for the BU processor to compute the diagnoses, and in requests for pixel diagnoses directed to the Pixel Processor. The TD processor can be programmed using control rules to execute actions that modify the image partition. The premises of these rules are established in terms of segment diagnoses computed by the BU processor and may include conditions about spatial relations among neighbor segments as explained in the next section.

3 The Conditions

In the frame of the Segment Processor a 'condition' is a predicate composed of several premises connected by logic 'And' conjunctions. Conditions are used in the different types of rules, i.e. rules used in the definition of classes and the control rules, used at the Segment Processor. A distinctive feature of the conditions is their expressive capacity to include a large variety of spatial relations as allowed by the premises' syntax. A condition's premise must follow the following syntax:

premise::= prem_atom_true | prem_list
prem_atom_true::= Is (True, [qualifier_1] x)
prem_list::= Is (property { and property }, [qualifier_1] x)
property::= [qualifier_2] [order] class_name
qualifier_1::= [neigh] [qualifier_2] [spatial]
neigh::= All | Any
qualifier_2::= [Not] [linguistic]
linguistic::= Very | MoreLess
spatial::= Above | Below | Left | Right | Beside | InBeside | Neib |
ExNeib | InNeib | Contains | User_defined_localizer
order::= Gt (number) | Eq (number) | Lt (number)

The qualifiers *All* and *Any* refer to the environment and operate together with the spatial localizers that will be analyzed below. The syntax shows that it is also possible to express order relations (Gt, Eq, Lt) that permit to establish comparisons over the degree of the property used in the premise. According to the syntax, the basic premise has the following elemental expression:

Is (property , x)

where x denotes the segment considered as the focus of attention and *property* is the diagnostic that conditions x. For example, the premise:

Is (Green, x)

will be the expression of condition: 'The focus-of-attention segment is green'; and its evaluation will return the degree in which the focus-of-attention segment satisfies the diagnostic associated with the 'Green' class. The sense of the premise can be modified using the modifiers Not, $Very$ and $MoreLess$.

At the segment level it is very interesting to allow for premises that are conditioned over the spatial vicinity of the focus-of-attention segment. This can be accomplished incorporating spatial relations to the premise's syntax according to the basic form:

Is (property, spatial x)

where *spatial* represents a token associated with a basic spatial relation, e.g. $Above, Below, Left$, or other user-defined spatial relations. Using this possibility, the premise:

Is (Green, Below x)

will express the condition: 'Any segment below the focus of attention is green'. Some localizers have a limited ambit of application. For example, localizers as $Above$, $Below$, $Left$, $Right$, $Beside$, $ExNeib$ and $Contains$ are only evaluated over the external neighbors of a given focus-of-attention segment. On the other side, the $InBeside$ and $InNeib$ localizers select exclusively the segments that are localized inside the focus-of-attention segment, that is, its 'internal' neighbors. Finally, localizers as $Neib$ does not distinguish between internal or external neighbors. By default, the evaluation of a spatial condition implies testing the condition over all the neighbors and returning the best result, i.e. an or-operation. However, there may be situations in which a condition must be established over all the neighbors. This can be done using the All modifier as in the following examples:

Is (Not Red, All InNeib x)

Is (Very Round, All Not Below x)

that express, respectively, the conditions: 'All the segments contained by the focus-of-attention are not red' and 'All the segments that are not below the focus-of-attention are very round'. The evaluation of the predicate $A\ Below\ B$ returns a value in the interval [0,100] that quantifies the relative vertical position of A below B, being A and B two neighboring segments. The expression used to compute the spatial relative positions of two segments can be found in [13]. These measures are obtained consistently so that, for example, $A\ Below\ B \equiv B\ Above\ A$.

The utilization of predicates that include spatial relations between segments makes possible also to establish order relations or comparisons between neighboring segments for a given diagnostic. The defined order relations are: Gt or 'greater than', Eq or 'equal to', and Lt or 'less than'. These relations can be evaluated in a fuzzy way as follows. Let us consider two segments, A and B, and let SC represent a diagnostic on which we want to base the comparison $SC_A\ Gt\ SC_B$, where SC_A and SC_B are the values of the diagnostic obtained from A and B respectively. Let u be a variable defined as the difference $SC_A - SC_B$ and let δ be a parameter that measures the half-width of the interval in u that separates the two extreme results: SC_A is 'absolutely' greater than SC_B, so that condition evaluates to 100, or just

the opposite so that the evaluation should return 0. Between these two extreme situations, a fuzzy region may exist in which a value somewhere between 100 and 0 should be returned. The extent of that interval can be adjusted tuning the δ parameter to control the fuzziness of the comparison. In our implementation, the Gt relation is evaluated using a sigmoid function as follows:

$$SC_A \; Gt(\delta) \; SC_B \; = \; 100S(u; -\delta, \delta)$$

In this way, a condition that asserts: 'Some neighbor of the focus-of-attention segment is larger (than the focus-of-attention) with uncertainty 10', can be expressed as:

Is (Gt(10) Large, Neib x)

The order relations Eq and Lt are defined in a similar way. The evaluation of the premises follows the algorithm proposed in [13], while the evaluation of the condition uses a min-operator to return the minimum of the results obtained by the premises included in the condition.

4 Applications

Following these ideas, we have implemented a two-level system capable of performing image segmentations [10] [13]. The pixel and segment processors of the system can be programmed using a special purpose declarative language. The application included in this paper has been taken from a real sedimentology problem which is illustrated by Figure 2. Given a back-lighted sample of sand grains (see upper left image in Figure 2) the objective is to detect the particles that are approximately round. To solve this task a two stage process is followed. The first stage is very simple and outputs a first segmentation where the background and the set of particles are differentiated (upper right image in Figure 2). Once the particles have been isolated, the morphologic gradient of the particles area is used by the presegmenter to resolve the particle area into individual particles. In this way, a final partition is obtained (lower left image in Figure 2). In order to qualify each particle as round, the class of 'Round' particles is computed from different features, related to the shape of the particles, for every segment in the final partition. The particles whose degree of membership to the class 'Round' is above certain threshold, i.e. 70, are depicted in the lower left image in Figure 2.

5 Conclusions

The fuzzy nature of many of the tasks that a computer vision system must develop (recognition, evaluation of spatial relations, information fusion, ...) suggests the utilization of fuzzy logic techniques in this field. The objective of this paper has not been to demonstrate the superiority of the proposed scheme over other techniques or algorithms that are used also at the segment level. But to illustrate how a new scheme can be used to solve segmentation problems in different contexts by means

Figure 2: Top) Original image and initial partition. Bottom) Final partition and selected round particles.

of simple mechanisms based on fuzzy logic concepts that fit naturally within the context of knowledge-based vision systems.

References

[1] Aloimonos J., Weis I., Active Vision, *Int. Journal of Computer Vision*, 2, (1988), 333-356.

[2] Bajcsi R., Active Perception, *Proc. of the IEEE*, 76, 8, (1988), 996-1005.

[3] Wilson R., Spann M., *Image Segmentation and Uncertainty*, Research Studies Press Ltd., 1988.

[4] Niemann H., Brünig H., Salzbrunn R., Schröder S., A Knowledge-Based Vision System for Industrial Applications, *Machine Vision and Applications*, 3, (1990), 201-229.

[5] Dubois D., Prade H., Yager R. R., Introduction in *Readings in Fuzzy Sets for Intelligent Systems*, Dubois D., Prade H., Yager R. R. (Eds.), Morgan Kaufmann Pub., (1993), 1-20.

[6] Krisnapuram R., Keller J.M., Fuzzy Set Theoretic Approach to Computer Vision: An Overview, in *Fuzzy Logic Technology and Applications*, IEEE Tech. Activities Board, Marks R.J. (Ed.), (1994), 25-32.

[7] Huntsberger T.L., Rangarajan C., Jayaramamurthy S.N., Representation of Uncertainty in Computer Vision using Fuzzy Sets, *IEEE Trans. Comput.*, 35, 2, (1986), 145-156.

[8] Risemann E.M., Hanson A.R., A Methodology for the Development of General Knowledge-Based Vision Systems, in *Vision, Brain and Cooperative Computation*, Arbib and Hanson (Eds.), MIT Press, Cambridge Mass., (1987), 285-328.

[9] Zadeh L.A., PRUF- a meaning representation language for natural languages, in *Fuzzy Reasoning and its Applications*, Mandani and Gaines (Eds.), Academic Press, London, (1981), 1-39.

[10] Méndez J., Falcón A., Hernández F.M., Cabrera J., A Development Tool for Computer Vision Systems at Pixel Level, *Cybernetics & Systems*, 25, 2, (1994), 289-316.

[11] Zadeh L.A., Commonsense Knowledge Representation based on Fuzzy Logic, *IEEE Computer*, 16, 10, (1983), 61-65.

[12] Gordon J., Shortliffe E.H., The Dempster-Shafer Theory of Evidence, in *Rule-Based Expert Systems*, Buchanan and Shortliffe (Eds.), (1984), 272-292.

[13] Cabrera J., *Sistema Basado en Conocimiento para Segmentación de Imágenes. Desarrollos y Aplicaciones*, Doctoral Dissertation, Universidad de Las Palmas de Gran Canaria, 1994.