An interactive constraint-based system for selective attention in visual search

R. Cucchiara[2], E. Lamma[1], P. Mello[2], M. Milano[1]

[1] DEIS, Univ. Bologna, Viale Risorgimento 2, 40136 Bologna, Italy
email: {mmilano, elamma}@deis.unibo.it
[2] Dip. Ingegneria, Univ. Ferrara, Via Saragat, 41100 Ferrara, Italy
email: {pmello, rcucchiara}@ing.unife.it

Abstract. We face in this paper the problem of model-based object recognition in a scene. Computer vision techniques usually separate the extraction of visual information from the scene from the reasoning on the symbolic data. We propose to interactively intertwine the two parts: the reasoning task on visual information is based on constraint satisfaction techniques. Objects are modeled by means of constraints and constraint propagation recognizes an object in the scene. We extend the classical Constraint Satisfaction Problem (CSP) approach which is not suitable for coping with undefined information. We thus propose an *Interactive CSP* model for reasoning on partially defined data, generating new constraints which can be used to guide the search and to incrementally process newly acquired knowledge.
Keywords: Machine Vision, Constraint Satisfaction.

1 Introduction

Visual search is a very popular paradigm of computer vision, relying the location and identification of a known target in a visual scene. Many studies of human and computer-based visual systems state that this process is both *data-driven* and *model-driven*. In accordance with recent researches on attention in psychology, the process is mainly bottom-up in the earlier stages, named *pre-attentive* processes, which have the aim at extracting several basic visual features, and generating feature maps [1]. From feature maps, object features (or suitable grouping of features) are matched to target models in order to provide the correct visual search. This process is obviously model-driven and can be carried out in many different ways according to the techniques used for describing the knowledge encoded in the model.

Since the whole visual process should deal with enormous quantities of data extracted by images, in contrast with the limited processing capabilities of the visual system, attention is a powerful mechanism endowed in the human visual system for gathering computation only in semantically significant information subsets. In particular, selective attention in psychology is the paradigm of guiding the visual process in the search of known target object on the basis of a target model.

Historically, computer vision techniques separate items for the two steps of first extracting visual information from the scene, reconstructing complex symbolic objects description and, then, providing the decision process by reasoning on symbolic data [21]. Recently, many proposals of goal-directed vision systems integrate the knowledge of the target in the initial stages of feature selection and computation. Eventually concepts of focus of attention and attentive visual tasks are introduced in computer vision in order to deal with the previously mentioned attentive paradigm [1].

This paper aims at proposing an integrated paradigm for visual search, which exploits model-based vision techniques and concepts of selective attention for target location and identification. The paradigm is based on an original *Interactive Constraint Satisfaction Model.*

Constraint Satisfaction systems provide simple but powerful framework for solving a variety of Artificial Intelligence (AI) problems. A survey on these techniques can be found in [14]. Constraint Satisfaction Problems (CSP, for short in the following) are defined on a finite set of variables each ranging on a finite domain (numerical or symbolic domain) and a set of constraints. A solution to a CSP is an assignment of values to variables which satisfies the constraints.

Constraints are an expressive tool for describing objects in terms of object parts and relations between them. Therefore, we propose to model visual objects by means of constraint satisfaction problems whose variables represent parts of the object or its suitable visual features and constraints represent spatial relations among them. Visual search is carried out by solving the CSP: in particular the localization of at least an object in the scene is provided by a solution for the constraints in the CSP.

A CSP can be solved by using tree search techniques where variables are instantiated sequentially. If a partial instantiation violates any of the constraints, backtracking mechanism is performed leading to a depth-first search of the solution space. However, backtracking is very inefficient since it uses constraints *a-posteriori* after having performed a guess on the value of some variables. Therefore, many consistency techniques (e.g., node, arc, path consistency) have been proposed [15, 16, 17] based on the active use of constraints during the search process. The idea is to remove during the search, by means of constraint propagation, combination of assignments which cannot appear in any consistent solution.

The main drawback of classical CSP if used as a general purpose tool for selective attention in visual search is that variable domains have to be completely defined before the constraint propagation process starts. In our model this requirement corresponds to first acquire all the information from the scene and then to elaborate this information by means of constraint processing. This leads to a functional separation between the perceptive stage for computing the entire set of visual primitives present in the scene in a fully data-driven way, and the further decision process. It carries out many difficulties, especially in terms of efficiency of the visual system, since many, often very complex, visual tasks must be performed also for computing useless information.

Therefore, we define an *Interactive Constraint Satisfaction* model which is able to:

- cope with incomplete domain definition;
- request information when needed thus interacting with the vision system;
- guide the knowledge acquisition by means of constraints;
- incrementally process new information without restarting a constraint propagation process from scratch each time new information is available.

This approach results particularly satisfactory in case of visual search conceived as a self-terminating process, in which the search, guided by the CSP, stops as soon as the target is located. In this case performing visual tasks only on-demand, and thus focusing attention only in part of the scene, makes it possible to limit the search only in those part of images where potential target may be located.

Another significant advantage concerns the generality of our approach. While other model-based approaches adopt special-purpose algorithms based on specific models for focusing attention, our approach is independent from the model since constraint propagation is a general-purpose process.

The paper is organized as follows: in section 2 we recall some ideas on CSP and propagation algorithms. In section 3, we describe how to model visual objects in terms of CSPs and present some limitation of the classical CSP approach. Therefore, in section 4, we propose an interactive CSP model and we explain how to use the model for guiding visual knowledge acquisition. An example is presented in section 5. Related works are described in section 6. A discussion and future works conclude the paper.

2 Preliminaries on CSP

A CSP can be defined on a set of variables $X_1, X_2, \ldots, X_n$ ranging respectively on finite domains $D_1, D_2, \ldots, D_n$. A constraint $c(X_1, \ldots X_k)$ defines a subset of the cartesian product of $D_1, \ldots, D_k$, i.e., a set of configuration of assignments which can appear in a consistent solution (if it exists). A CSP can be represented by means of a *constraint network* where each node is a variable and arcs are constraints.

A CSP can be solved by means of constraint propagation. Many consistency techniques (node, arc, path consistency) have been proposed [15, 16, 17] based on the active use of constraints during the search process.

We will refer in the paper to the *arc-consistency* algorithm. An arc in the CSP representing a binary constraint $c(X_i, X_j)$ is arc-consistent if and only if for each value in the domain D_i for variable X_i there exists at least one value in the domain D_j for variable X_j which is consistent with the constraint $c(X_i, X_j)$. A constraint network is arc-consistent if and only if each arc is arc-consistent. For example, consider the constraint $X < Y$ where[3] $X :: [1..10]$ and $Y :: [1..10]$. Arc-consistency removes the value 1 from the domain of Y because there does not exists in the domain of X a value which is less than 1. In the same way, value 10 is deleted from the domain of X. Arc-consistency is not complete since an arc-consistent network can be unsatisfiable. For example, consider variables X, Y and Z all ranging on the domain $[1, 2]$. The constraints between them are $X \neq Y$, $Y \neq Z$ and $Z \neq X$. The constraints are clearly inconsistent even if a solution does not exists. However, arc-consistency is a good tradeoff between propagation and computational cost.

A solution to a CSP is an assignment of values to variables which is consistent with constraints. A solution can be found by intertwining an arc-consistency propagation algorithm with a *labeling* strategy which performs a guess on a variable value thus exploring the solution space.

3 Modeling visual objects through constraints

In computer-based visual systems objects are described by means of their visual features which can be defined at different levels, starting from low-level pixel based features (e.g., luminance, color) up to symbolic visual primitives; examples are lines and curves in 2D and 3D space and surfaces and conics in 3D space [4]. Therefore, objects can be modeled as structured assemblies of these very general geometric elements. This model may not be exhaustive enough when natural shapes must be recognized, but it results an affordable and efficient approach in case of visual search for many structured shapes, such as man-made (especially CAD-made) objects. This framework is currently an active research field since is related to many potential industrial and robotic applications.

In a general approach to visual search, the model of the object should be reliable and general, in order to result possibly invariant to roto-translation in

[3] The notation :: associates a domain to the corresponding variable.

3D space, to distance object-camera and other environment factors: therefore the object model cannot be limited to a matching of measured features to absolute values previously defined, but should be based on geometric and topologic relationships between features.

Within this extent, objects can be modeled by means of constraints and recognized by means of constraint satisfaction. Each object we want to recognize can be represented by means of a constraint graph where each object part or characterizing primitive feature is modeled by a node (variable) of the corresponding CSP and spatial or shape relations among object parts can be represented by arcs (constraints).

Specific aspects of the single primitives may be modeled as unary constraints, such as the minimum length of an object part, its color, the planarity of a surface and so on, while geometric and topologic relationships between them can be represented by binary constraints (examples of geometric constraints are angular relationships between contours, lines, or surfaces, topologic constraints are spatial relationships, such as is *connected to*, *touch*, *is contained* or others).

Example 1. If we want to model a rectangle, as shown in figure 1, we can identify four nodes corresponding to the four edges composing the rectangle (numbered respectively X_1, X_2, X_3 and X_4) and we impose the following symmetric constraints:

$touch(X_1, X_2)$, $touch(X_2, X_3)$, $touch(X_3, X_4)$,
$touch(X_4, X_1)$, $no\_touch(X_2, X_4)$, $no\_touch(X_1, X_3)$,
$same\_lenght(X_2, X_4)$, $same\_lenght(X_1, X_3)$, $perpendicular(X_1, X_2)$,
$perpendicular(X_2, X_3)$, $perpendicular(X_3, X_4)$, $perpendicular(X_4, X_1)$,
$parallel(X_2, X_4)$, $parallel(X_1, X_3)$.

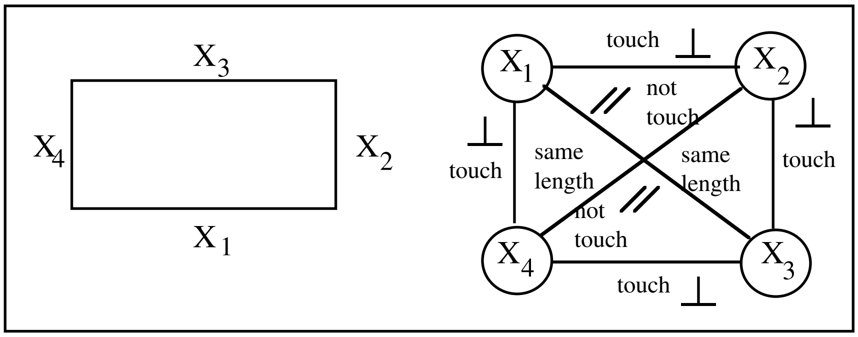

Fig. 1. Constraint-based model of a rectangle

Some of these constraints are redundant. This redundantly is useful when constraints are propagated by means of an incomplete algorithm such as for example arc-consistency.

Suppose the vision system provides all the information about the segments in the scene. Therefore, each variable ranges on a domain containing all the segments. We can perform a straightforward but inefficient backtracking algorithm in order to find a solution. Since we model objects by means of constraints, we can exploit constraint propagation in order to find a solution or at least prune the search space. First an arc-consistency can be performed that removes combinations of assignments which cannot appear in any consistent solution, i.e., removes combination of assignments which cannot form a rectangle. A labeling strategy then assigns to each variable a value (a possible segment). Again an arc-consistency starts and further reduces domains accordingly. Finally, when a solution is found (if it exists), we have identified a rectangle in the scene, i.e., four segments which are consistent with constraints. Note that in this specific example there are symmetries in the problem. Following the definition given in [18], a symmetrical problem is a problem where some permutation of the variables maps a solution onto another solution.

The problem with this kind of representation concerns the fact that we have to acquire from the vision system all the information about segments in the scene before processing them. The segment extraction is a computationally expensive task if compared with the constraint satisfaction process. In this paper we propose an alternative approach. We intertwine the vision process with the constraint satisfaction process. The advantages of this proposal are mainly two: first, we acquire information from the low level vision system *on-demand*, i.e., only when needed. Therefore, we limit as much as possible the computationally expensive queries to the system. Second, we are able to guide the feature extraction by means of constraints, thus focusing attention on most significant parts of the image. In fact, when we find one segment which is possibly part of a rectangle, we guide the system in order to find another segment which satisfies the constraints with the first.

4 Interactive CSP

A standard CSP approach needs all the information and the knowledge on the problem at the beginning of the computation. Then it propagates constraints by removing assignments which cannot appear in any consistent solution. The interaction with a low level system and the consequent propagation requires a partial acquisition of knowledge which lasts during all the computation process. Therefore, we have to change the CSP model we use and allow the propagation algorithm to work on partially defined domains.

To this purpose, we define an *Interactive CSP* (*ICSP*) model which has to:

- cope with incomplete domain definition. Therefore, domains can be partially defined in the sense that some domain element can be already at disposal for propagation, while other domain elements have to be acquired from the vision system in the future;
- request information when needed, thus interacting with the vision system and performing a knowledge acquisition on demand;

- guide feature extraction by means of constraints. Therefore, the CSP subsystem should generate at each computational step new constraints on the basis of already acquired knowledge;
- incrementally process new information without restarting a constraint propagation process from scratch each time new information is available. Therefore, the constraint satisfaction model should maintain a representation of constraints on undefined values and check them against newly available information.

Let us consider again example 1 and suppose we do not acquire all the segments from the scene but we start a constraint propagation relying on an incomplete knowledge (only few segments, at least one, are known). These segments are tested for consistency and used in order to generate new constraints which can be used in order to guide the search.

On the basis of these requirements, we define the Interactive CSP model as follows:

Definition 1. An interactive CSP (ICSP) is defined on a finite set of variables $\{X_1, X_2, \dots, X_n\}$ each ranging on a partially defined domain $\{D_1, D_2, \dots, D_n\}$ where each $D_i = [Def_i \cup UnDef_i]$. Def_i represents the defined part, while $UnDef_i$ is a domain variable itself representing information which is not yet available. Both Def_i and $Undef_i$ can possibly be empty[4]. Also, for each i, $Def_i \cap Undef_i = \emptyset$ A constraint among variables defines a possibly partially defined subset of the cartesian product of variable domains.

A solution to the ICSP is, as in the case of the standard CSP, an assignments of values to variables which is consistent with constraints.

On the contrary, the constraint propagation is quite different from the standard case. Consider for the sake of clarity only binary constraints $c(X_i, X_j)$. In the more general case, in order to propagate the constraint $c(X_i, X_j)$ we have to propagate four kinds of constraints and collect the propagation results[5].

$c(Def_i, Def_j) \cup c(Def_i, Undef_j) \cup c(UnDef_i, Def_j) \cup c(UnDef_i, UnDef_j)$

The propagation between definite domains is the usual constraint propagation, while propagation on undefined domains produces new constraints. These new constraints can be used in order to guide the search and to incrementally process newly acquired knowledge from the vision system.

Let us consider an example on integer numbers in order to clarify the behavior of the model. Then in the next section, we will apply the model for the recognition of an object in a vision system. Consider the constraint $X < Y$ where X ranges on the finite domain $[1, X1]$, and Y ranges on $[-3, 2, Y1]$ where $X1$ and $Y1$ are domain variables. Suppose we are using an arc-consistency propagation algorithm. The propagation of constraints on definite parts is the following. Value $1 \in Def_X$ is supported from the value 2 in the domain of Y. Viceversa, while the

[4] When both are empty an inconsistency arises.

[5] We refer, with an abuse of notation, to domains instead of variables. However, we meaning is straightforward.

value $2 \in Def_Y$ is supported by the value 1 in the domain of X, the value -3 is not supported and can be deleted from the domain of Y. The resulting domains from this first propagation are $X :: [1]$, $Y :: [2]$. It is worth noting that this first result of the propagation should be considered together with other propagations, i.e., $c(Def_i, Undef_j) \cup c(UnDef_i, Def_j) \cup c(UnDef_i, UnDef_j)$. In particular, the propagation between the defined part of X and the undefined part of Y leads to the generation of the following constraints: $Y1 > 1, Y1 \neq 2$ and $Y1 \neq -3$ (these last constraints derive from the definition of ICSP: $Def_i \cap UnDef_i = \emptyset$). The propagation between the defined part of Y and the undefined part of X leads to the generation of the following constraints: $X1 < 2$, $X \neq 1$ and a conditional constraint $Y \neq -3 \; if \; X1 < -3$. This last constraint represents in some sense the support for the value -3 for Y. This constraint together with the the first propagation between defined domain results in the final domain for Y which contains also the value -3 since we are not yet sure that only values greater than -3 are provided for $X1$. Finally, the last propagation gives the constraint between the undefined parts which is exactly the same as the constraint between the original variables, i.e., $X1 < Y1$.

Therefore, the result of the propagation will be:
$X :: [1, X1], Y :: [-3, 2, Y1]$ such that
$X \neq 1, \; Y1 \neq 2 \; Y1 \neq -3$
$((Y1 > 1, X1 < 2) \vee X1 < Y1), Y \neq -3 \; if \; X1 < -3$

Let us now consider two possible knowledge acquisitions from the underlying system.

- Suppose that a new value, -7, for X is now available. This value can be checked only against newly introduced constraints. In fact, the value -7 becomes part of variable $X1$'s domain which is $X1 :: [-7, X2]$. Therefore, the constraint $X1 < 2$ is satisfied provided that $X2 < 2$. The conditional constraint $Y \neq -3 \; if \; X1 < -3$ is now satisfied and can be removed. The constraint $X1 < Y1$ becomes $X2 < Y1$.
- Suppose that a new value 7 for X is now available. The value 7 becomes part of the variable $X1$ domain which is $X1 :: [7, X2]$. The constraint $X1 < 2$ is not satisfied but the other constraint in the disjunction $X1 < Y1$ can be satisfied if $7 < Y1$. This constraint is a conditional constraint for the support of value 7: $X \neq 7 \; if \; Y1 > 7$.

When the knowledge acquisition is finished, constraints can be solved and the constraint propagation is the same as in usual CSP models.

Another requirement of the interactive CSP concerns the possibility of guiding the search by means of constraints. In fact, the constraints
$X :: [1, X1], Y :: [-3, 2, Y1]$ such that
$X \neq 1, \; Y1 \neq 2 \; Y1 \neq -3$
$((Y1 > 1, X1 < 2) \vee X1 < Y1), Y \neq -3 \; if \; X1 < -3$
can be used in order to guide the acquisition of variable values. Suppose we are looking for a value for Y. We can collect all the constraints involving $Y1$ and guide the search in order to find consistent values with the constraints. For

example we can use the constraint $Y1 > 1$, $Y1 \neq 2$, and $Y1 \neq -3$ in order to collect values which are already consistent with the definite parts of X and Y. The constraints between undefined parts cannot be used in general except for the case that the underlying system is able to provide couples of values already consistent with this constraint.

5 An Example on Visual Recognition

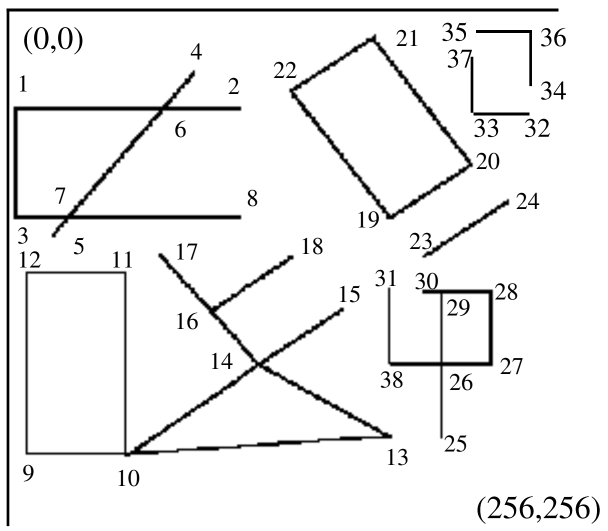

Fig. 2. Example of a visual scene

In this section, we consider an example for the recognition of (one or more) rectangles in the scene depicted in figure 2. It is an image of 256 x 256 binary valued pixel, obtained after an edge detection of a grey-level image. The rectangle model is given by means of symmetric constraints in section 3. We first describe how to solve the problem by means of a classical CSP approach. Then, we present an alternative solution based on the ICSP model proposed in section 4.

In a classical CSP model all the segments should be collected as a first step. All the segments in figure 2 can be collected in the following form: they have an associated name, the coordinates (x1, y1, x2, y2 for the two staring and ending points), the length and the angle with the horizontal axes. These characteristics are sufficient for checking the constraints on the rectangle.

In the following we will refer to a segment from a to b as (a, b). The 41 segments of figure 2 represent the domain of variables X_1, X_2, X_3 and X_4 (corresponding to the four edges of the rectangle in figure 1). Let us consider the result of an arc-consistency propagation method. As described in section 2, a constraint network is arc-consistent if and only if for each value of a variable domain, there exist at least one value in each other variable domain which is consistent with constraints. We will refer to one domain propagation only since, in this case, all constraints are symmetric. Therefore, the four variables have always the same domain during the computation and before the labeling step.

Consider segment $(1,3)$. This segment is not consistent with constraints since there does not exists in other domains a segment with the same length. Therefore, it is deleted from domain D_1. The segment $(3,8)$ is, instead, consistent since it satisfies all the constraints. Note that both constraints $touch(X_1, X_2)$ and $touch(X_4, X_1)$ are satisfied by the same segment $((1,2))$ in the domain of X_2 and X_4. The arc-consistency is an iterative algorithm which stops when the network has reached the *quiescence*, i.e., no more propagation is possible. After the first arc-consistency iteration, the variable domains contain only segments $(3,8)$, $(1,2)$, $(9,10)$, $(10,11)$, $(11,12)$, $(9,12)$, $(16,18)$, $(14,15)$, $(31,38)$, $(26,29)$, $(27,28)$, $(28,29)$, $(26,27)$, $(33,37)$, $(33,32)$, $(34,36)$, $(35,36)$, $(26,25)$. At a second iteration, segments $(16,18)$, $(14,15)$ are no longer supported by $(14,16)$ which has been deleted at the first step, segments $(3,8)$, $(1,2)$ are no longer supported by $(1,3)$ and segment $(31,38)$ is no longer supported by $(26,38)$. Remaining segments are $(9,10)$, $(10,11)$, $(11,12)$, $(9,12)$, $(26,29)$, $(27,28)$, $(28,29)$, $(26,27)$, $(33,37)$, $(33,32)$, $(34,36)$, $(35,36)$, $(26,25)$.

Note that not every segment in the remaining domain is part of a consistent rectangle since arc-consistency is not a complete propagation algorithm. In particular, segments $(33,37)$, $(33,32)$, $(34,36)$, $(35,36)$, $(26,25)$ which are not part of any rectangle could be deleted by a more powerful constraint propagation algorithm (e.g., path-consistency). Now a labeling step starts instantiating variable X_1 to $(9,10)$. Constraint propagation now reduces domains of other variables to $X_2 :: [(9,12),(10,11)]$, $X_4 :: [(9,12),(10,11)]$ and X_3 can be instantiated to $(12,11)$.

Consider now the approach based on the ICSP. The main advantage is that we do not need to have all the information about segments in order to start the propagation. Therefore, we suppose that at the beginning all variable domains are undefined, i.e., the defined part is empty. Now we can ask the visual system to give us the first information about segments. This first query to the visual system can be guided in some sense if we have unary constraints on variables. For example suppose we have a color constraint on one of the segments, e.g., $red(X_1)$, we can guide the visual system to acquire first a red segment thus concentrating on "red zones". In general, the first acquisitions can be guided by means of heuristics on the problem, say first we acquire segments belonging to closed figures. In this case, we avoid this guide since we do not have neither unary constraints nor heuristics. Therefore, the system gives us the first segment for X_1, say $(1,3)$. Therefore, the domain of X_1 is $D_1 = [(1,3), X_1']$ where X_1' represents the undefined part of X_1. Now, since all other variables have only an undefined domain, $(1,3)$ is consistent with constraints. However, new constraints are generated in order to check the consistency of future acquired information and to guide the search. In particular, all the constraints containing variable

X_1 are "splitted" into two parts: the first results from the propagation of the undefined part of all other variables with the defined part of X_1:

$$touch((1,3),X_2), touch(X_4,(1,3)), no\_touch((1,3),X_3), same\_length((1,3),X_3),$$
$$perpendicular((1,3),X_2),\ perpendicular(X_4,(1,3)),\ parallel((1,3),X_3)$$

the second is derived from the propagation of the undefined part of all other variables with the undefined part of X_1:

$$touch(X_1',X_2),\ touch(X_4,X_1'),\ no\_touch(X_1',X_3),\ same\_length(X_1',X_3),$$
$$perpendicular(X_1',X_2),\ perpendicular(X_4,X_1'),\ parallel(X_1',X_3).$$

The first part is used in order to guide the knowledge acquisition from the visual system. In fact, for variable X_2, we are looking for a segment that touches and is perpendicular to $(1,3)$. Such a segment exists and is for example $(1,2)$. Variable X_2 ranges now on the following domain: $D_2 = [(1,2), X_2']$. Again all the constraints involving X_2 can be "splitted" by constraint propagation in two parts. Note that constraints involving the defined part of X_1 and the defined part of X_2 are solved and can be ruled out. The new constraints derived from the defined part of X_2 and the undefined part of other variables are the following:

$$touch((1,2),X_3),\ no\_touch((1,2),X_4),\ same\_length((1,2),X_4),$$
$$perpendicular((1,2),X_3),\ parallel((1,2),X_4).$$

Constraints involving only definite parts are:

$$touch(X_1',X_2'),\ touch(X_2',X_3),\ no\_touch(X_2',X_4),\ same\_length(X_2',X_4),$$
$$perpendicular(X_1',X_2'),\ perpendicular(X_2',X_3),\ parallel(X_2',X_4).$$

Now, knowledge acquisition for X_3 can be driven from constraints and fails since there does not exists any segment for X_3 which touches and is perpendicular to $(1,2)$ and does not touch and is parallel to $(1,3)$.

A successful computation can be started when the system finds one of the segments belonging to a rectangle, such as for example $(12,9)$. The computation can be guided by means of constraints and the attention focuses only on the part of the scene containing the rectangle.

The main advantage of the approach is that in the average case, we acquire from the visual system a fewer number of segments since we guide the extraction by means of constraints. Note also that this kind of acquisition corresponds to an a-priori application of consistency techniques since the visual system provides only consistent values with constraints. Moreover, the applied consistency is stronger than arc-consistency since we query the visual system with different constraints on many variables. For example, in the above mentioned example, when we query the system for the acquisition of a value for X_3, we ask for a segment which both touches and is perpendicular to $(1,2)$ and does not touch and is parallel to $(1,3)$. This corresponds to a path-consistency since the consistency check involves three variables. In fact, in the case of segments $(33,37)$, $(33,32)$, $(34,36)$, $(35,36)$, suppose we have already acquired value $(33,37)$ for X_1 and $(33,32)$ for X_2. The acquisition of a value for X_3 fails since there does not exists any segment that does not touch $(33,37)$ but touches $(33,32)$.

6 Related Works

Several works are related to visual search which makes use of symbolic approaches in order to describe the *a-priori* knowledge on the target. Most of them, in the context of model-based vision, describe a framework of model matching, with model description and matching operators suitable for verifying similarity between visual features extracted by images and their symbolic description in the models. Model matching is provided at different abstraction levels: examples of low-level primitive matching is given by Rao and Ballard, [2]; more often it is performed at level of high-level feature maps [3], and in other cases it is distributed in a hierarchical way at different abstraction levels [4].

Different paradigms are used for representing models and providing the decision making process, in part with neural network techniques, in part with symbolic methods, such as rules [7], interpretation tree search [5] or constraint satisfaction networks [6]. This last work is the one which is closer to our work for the use of constraint satisfaction. However, the approach is quite different since in [6] the authors suggest the use of constraints on real number and a bound propagation, while we use finite domain constraints and an enumerative propagation.

While most of these proposals concern only the final recognition task, new trends address definition of a model which embraces all the visual process, starting from the primitive selection, their parameters tuning and the final decision. In this context, particular focus is given to the control strategies for the knowledge-directed computer vision [8]. A good example is offered by Bolle et al. proposal, [4]: it describes the use of an integrated constraint network, which is used in all levels of the visual process, starting from the extraction of local features, and assembly feature computation up to hypothesis verification. In [4] the constraint network is a hierarchical network where each node is a feature hypothesis and links between nodes represent (with continue-valued weights) conflict or agreement between hypothesis. The final scene identification is furnished by propagating the constraint network into a single globally consistent interpretation.

Our approach shares with that proposal the model of constraint network, and, consequently, some advantages: in particular a constraint network approach avoids some drawbacks due to a strictly sequential control strategy, such as, for instance, that given by a tree-search; moreover it provides a unified paradigm which is general-purpose, potentially extendable also to visual task as long as more complex than the one described by the above example. However, our ICPS-based approach exhibits some substantial differences: firstly, the object model is encoded in the constraint graph, where arcs represent constraints, symbolically coded in terms of geometric, topologic or metric relationships between features, represented by nodes. Secondly, while the control strategy in [4] is mainly data-driven, our control strategy is strongly goal-directed and model-based, in accordance with the goal of visual search. We suggest the use of a bottom-up data-driven process only for the earlier stages of image processing in order to extract some salient features of the target: this is in accordance with the current trends of active vision and perception [9] of computer vision community, that deal with some analysis performed in perception psychology [1]. In addition to these pre-attentive tasks, we propose an approach which deals with an attentive-based control with an effective and strong interaction between the visual system and the supervisor control of the CSP-based system. In this model, the catalyst features extracted from images fire to the constraint propagation: the ICSP model, whenever faces with an incompletely defined domain, i.e. whenever needs more information from the visual systems, sends a feedback to the visual system which is able to extract other visual primitives on-demand. Moreover, the CSP controlled process is able to instantiate new constraints and propagate them to the visual system so that the new computed visual features rely only on a limited area of the scene where the focus of attention has been concentrated. This model is general enough and does not lead to specify a fixed set of visual routines: the provided example relies on only simple 2D geometric features (such as straight segments and their geometric relationships), but it can be straightforwardly extended to other 2D and 3D features, such as curves, regions, surfaces and their relationships.

A possible limitation of our model is the need of a full-knowledge of the target object and the problem arisen with cluttered environment, noise, imprecision and occlusion while the noise and imprecise features problem can be overcome by suitably relaxing the constraints or accepting tolerances. Actually this model is not adequate for achieving a correct visual search in case of occluded object, since occlusion may cause the lack of perception of some modeled features.

7 Conclusions and Future Works

We have presented a model for interactive CSP which can be applied to object recognition and identification in a visual system. Objects are modeled by means of constraints and constraint propagation is the general-purpose tool for detecting a solution. The propagation has mainly two purposes: first, to reduce the search space by pruning those assignments which cannot appear in any consistent solution. Second, it is used in order to guide the search by generating new constraints at each step. Therefore, information can be acquired on-demand from the visual system thus reducing computational expensive visual tasks. Moreover, newly acquired information does not need to be checked against the whole network but only against newly generated constraints.

A very effective programming paradigm for solving CSP [20] is Constraint Logic Programming on finite domains (CLP(FD)). CLP (see [10, 11]) is a class of programming languages based on the Logic Programming (LP) paradigm [13, 12]. We are currently extending the CLP framework in order to cope with the ICSP model thus leading to an expressive and powerful language for visual applications.

Future works concern the extension of the system to other 2D and 3D features, such as curves, regions, surfaces and their relationships. The second step concerns the definition of a hierarchical architecture where constraint propagation is performed at different levels of abstraction, thus considering spatial relations between objects.

References

1. V. Cantoni, "Attentional Engagement in vision system", in Artificial Vision Academic Press 1997.
2. P. N. Rao, D.H.Ballard, "An active vision architecture based on iconic representations", Artificial Intelligence, 78, pp. 461-505, 1995.
3. J. Lemaire, "Use of a priori descriptions in a high level language and management of the uncertainty in a scene recognition system", Proc. of ICPR96, vol.1, pp 560-564, IEEE Press, 1996.
4. M. Bolle, A. Califano, R. Kjeldsen "A complete and extendible approach to visual recognition", IEEE Trans. on PAMI, 14, 5, 1992.
5. P.Gaston, T. Lozano Peres, "Tactile recognition and localization using object models: the case of polyhedra on a plane", IEEE Trans. on PAMI, 6, 3, 257-265, 1984.
6. I.D. Reid, J. M. Brady, "Recognition of object classes from range data" Artificial Intelligence, 78, pp. 289-365, 1995.
7. P.Pellegretti, F. Roli, S. Serpico, G. Vernazza, "Supervised learning of descriptions for image recognition purposes", IEEE Trans. on PAMI, vol. 16 n. 1, pp. 92-98 (1994).
8. B. Draper, A. Hanson, E.Riseman "Knowledge-directed vision: control, learning and integration", Proc. of IEEE, vol. 84, n. 11, pp. 1625-1681, 1996.
9. R. Baicsy "Active Perception", Proc. of IEEE, vol. 76 n. 8, pp. 996-1005, 1988.
10. J.Jaffar , J.L.Lassez, "Constraint Logic Programming", in *Proceedings of the Conference on Principle of Programming Languages*, Munich 1987.
11. J.Jaffar, M.J.Maher, "Constraint Logic Programming: a Survey", in *Journal of Logic Programming on 10 years of Logic Programming*, 1994.
12. J.W.Lloyd, "Foundation of Logic Programming", Second Extended Edition, Springer-Verlag 1987.
13. R.Kowalski, "Logic for Problem Solving", North-Holland,1979.
14. V. Kumar, "Algorithms for Constraint-Satisfaction Problems: A Survey", in *AI Magazine*, vol. 13, 1992, pp. 32–44.
15. A.K. Mackworth, "Consistency in Networks of Relations", in *Artificial Intelligence*, vol.8, 1977, pp. 99–118.
16. R.Mohr, T.C.Henderson, "Arc and Path Consistency Revisited", in *Artificial Intelligence*, vol.28, 1986, pp. 225–233.
17. A.Montanari, "Networks of Constraints: Fundamental Properties and Applications to Picture Processing", in *Information Sciences*, vol.7, 1974, pp. 95–132.
18. J.F. Puget, "On the Satisfiability of Symmetrical Constrained Satisfaction Problems", Tech. Report ILOG Headquarters, 1993.
19. P.Van Hentenryck, "Constraint Satisfaction in Logic Programming", MIT Press, 1989.
20. P.Van Hentenryck, H.Simonis, M.Dincbas, "Constraint Satisfaction using Constraint Logic Programming", *Tech. Report* CS-91-62, Dept. of Computer Science, Brown University, November 1991.
21. D. Vernon, "Machine Vision: Automated Visual Inspection and Robot Vision" Prentice Hall, 1991.