A constraint satisfaction framework with Bayesian inference for model-based object recognition

Włodzimierz Kasprzak, Łukasz Czajka, and Artur Wilkowski

Institute of Control and Computation Eng.
Warsaw University of Technology
W.Kasprzak@ia.pw.edu.pl,
WWW home page: http://www.ia.pw.edu.pl/

Abstract. A general (application independent) framework for the recognition of partially hidden 3-D objects in images is presented. It views the model-to-image matching as a constraint satisfaction problem (CSP) supported by Bayesian net-based evaluation of partial variable assignments. A modified incremental search for CSP is designed that allows partial solutions and calls for stochastic inference in order to provide judgments of partial states. Hence the detection of partial occlusion of objects is handled consistently with Bayesian inference over evidence and hidden variables. A particular problem of passing different objects to a machine by a human hand is solved while applying the general framework. The conducted experiments deal with the recognition of three objects: a simple cube, a Rubik cube and a tea cup.

1 Introduction

The recognition of 3D objects in images has been studied for many decades [1], [2]. One aspect of these works is the design of computational frameworks for model-based object recognition, which reflect the structure and uncertainty of the data and provide appropriate tools for knowledge representation, learning and inference. Using a declarative language for model description in such frameworks allows large parts of the recognition system to be application-independent. Many different frameworks have been proposed, e.g. attributed graphs [3], semantic networks [4], relation structure grammars [5]. One of the main problems of such approaches is how to manage and evaluate partial model-to-data matches. It is known that uncertain and vague knowledge can be modelled efficiently in terms of probabilistic theory [6] or fuzzy logic [7]. In this paper we propose a framework that views the 3-D object recognition problem as a modified constraint satisfaction problem ([6], [8]), combined with the Bayesian approach to statistical inference [6], [9]. Both parts are dominantly of declarative nature and additionally there exists well-known ML learning procedures for Bayesian probability distributions [9].

In section 2 the particular problem of hand-holded objects is presented and the segmentation stage of image analysis is explained. Our model-based search and hypothesis judgment approach is explained in section 3. Then the approach is illustrated in section 4 as applied to the recognition of 2 different object shapes: boxes with texture and objects with elliptic boundaries.

2 The problem

2.1 Objects in human hand

The goal of low-level image processing is to separate the human hand and the background from the other objects. In our system this is done based on color processing and moment-based contour filtering, as explained in the earlier paper [10]. First, the input image is subject to color processing in the YUV space. This allows to detect the human hand and (optionally) to focus on its neighborhood only. Morphological operations, edge and contour detection steps follow. Specific contour classes, like small "rectangular" chains, are labelled on base of features extracted by geometric moment functions. Next, the sufficiently long chains are approximated by linear segments. Consecutive segments are then approximated by arcs (fig. 1). Finally, a model-based detection of objects allows to assign model instances to segments groups detected in the image (fig. 2).

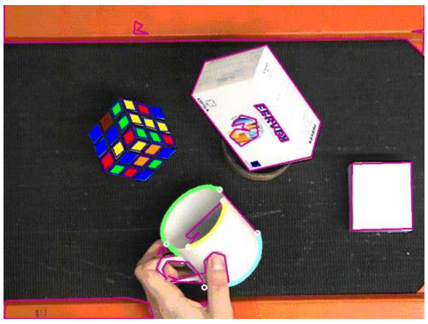
Fig. 1. After segment detection

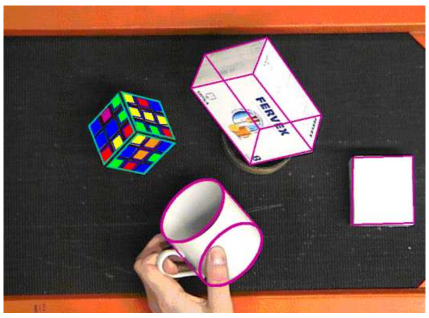
Fig. 2. Object recognition results

2.2 Elliptic arc detection

Straight line segments are detected by a conventional method of edge chain approximation. Here a more detailed description of our approach to elliptic arc detection follows. This arc detection algorithm consists of 4 steps:

1. Detection of pairs of consecutive line segments, to be approximated by arcs (figures 3 and 4);
2. Extending existing arcs by including neighbor line segments (fig. 5) if they together can be approximated by an arc;

3. Connecting consecutive arcs together if they form a larger arc of the same type, i.e. convex or concave (e.g. ACBJ and JFE in fig. 6 are not connected);
4. Verifying the shape of an arc sequence. If the consecutive arcs form a closed contour then it is tried to approximate such contour by a circle or ellipse.

Examples of image segmentation results are provided in figures 7 (detection of line segments) and 8 (approximation of linear segment chains by arcs, circles and ellipses).

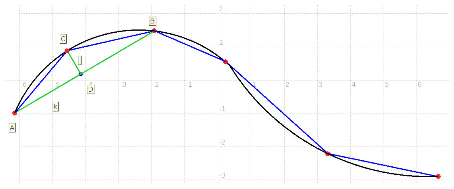

Fig. 3. The first stage of arc detection - arc ACB is found

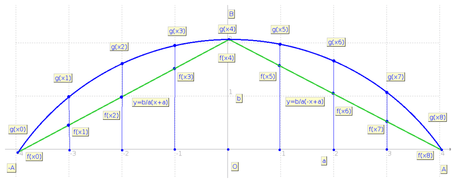

Fig. 4. Approximating two line segments by an arc

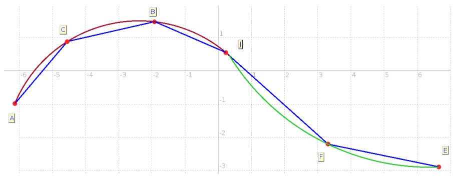

Fig. 5. Extending an arc by including a next line segment - arc ACB is extended to ACBJ

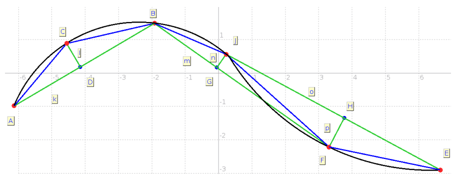

Fig. 6. There is no arc BJF but there is a second arc JFE

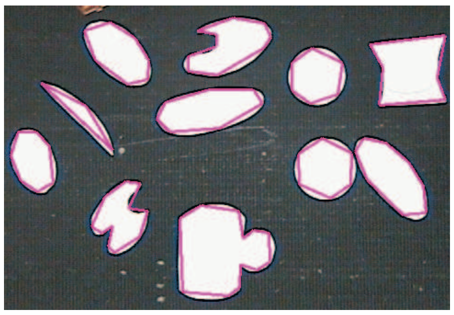

Fig. 7. Detected line segments that approximate contour chains

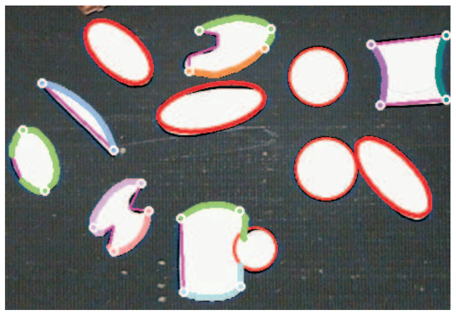

Fig. 8. Line segment chains approximated by arcs, circles and ellipses

3 The framework for object modelling and recognition

3.1 CSP

The search space definition in a **discrete Constraint Satisfaction Problem** consists of following elements:

– A state set S, where a particular **state**, $\mathbf{s} = (d_1, d_2, ..., d_n)$, is defined by assignments to its variables, $X = x_1, x_2, ..., x_n$, where each x_i, $(i = 1, ..., n)$, can take values from a (finite) domain D_i.
– **Actions**, $a \in A$, mean transitions between states: $a_k : s_i \rightarrow s_j$.
– The **goal test** checks a set of constraints, $C(X)$, which induces allowed combinations of assignment values for subsets of state variables.
– A **solution state** is every state that satisfies the goal test, i.e. the sequence of actions is not relevant, but the final state only. The state $\mathbf{s} = (d_1, d_2, ..., d_n)$, satisfies the goal test if: $C(d_1, d_2, ..., d_n) = True$.

In particular, in our problem: the variables in X correspond to line segments of the object model, the values in D represent the current image segments and an action is assigning a value to some variable in given state. The variables and the set of constraints, $C(X)$, can be represented as a graph, $G(X, C(X))$ where nodes X represent variables and arcs $C(X)$ represent constraints between particular variables.

The structure of our CSP search is presented in table 1. While starting from an empty assignment the goal is to match (assign) eligible image segments (values) with model entities (variables). We introduced two **modifications** to the basic CSP search. The first modification is due to the definition of a **Bayesian network** for every problem. The subfunction Score calculates probability value of a partial solution, that consists of eligible assignments to variables. This score is due ti stochastic inference process performed in a dedicated Bayesian net, created for current CSP problem. An empty assignment to a variable is also possible.

The basic algorithm for CSP is a depth-first tree search with a backtracking step when the path is not consistent with given constraints. The second modification of a typical CSP is that now partial paths can be potential solutions. The backtrack step is performed now when currently selected (extended) path does not satisfy the constraints of given problem or its score is lower than the score of predecessor path. In our view this is not a general failure but a situation where the previous state corresponds to a partial solution. The current path is conditionally stored as a possible partial solution when it is of higher score than the previous best one.

Still, if we succeed to find a complete path (with assignments for all variables) then we immediately stop the search with this final solution.

3.2 Bayesian net

This is a simple, graphical notation for conditional independence assertions and hence for compact specification of full joint distributions. Syntax:

Table 1. Modified backtracking search for the framework of CSP and Bayesian net

```
function BacktrackingSearch( csp ) returns Solution
    static solution = { } ;
    path = { }
    solution = RecursiveBacktracking( solution, path, csp)
    return solution

function RecursiveBacktracking( solution, path, csp) returns solution
    IF path is complete (Stop test)
    THEN return solution
    var ← SelectUnassignedVariable( csp.variables, path)
    valueList ← GetDomainValues( var, path, csp)
    FOR EACH value ∈ valueList
        IF (path∪ {var ← value } ) are consistent with csp.constraints
           AND Score(path∪ { var ← value })> Score(path)
        THEN add { var ← value } to path
             IF Score(path) > Score(solution)
             THEN solution = path
             result = RecursiveBacktracking(solution, path, csp)
             IF result ≠ failure
             THEN return result
             remove { var ← value } from path
    return failure
```

- a set of nodes, one per variable;
- a directed, acyclic graph (link means that "direct influence");
- incoming links of given node represent a conditional distribution for this node given its parents, $P(X_i|Parents(X_i))$.

In the simplest discrete case, conditional distribution is represented as a conditional probability table (CPT), giving the distribution over X_i for each combination of parent values.

3.3 Example: a cube model

Let the hierarchic structure of a generic cube structure is given, i.e. the concept "cube" consists of 12 "edges" numbered as (0, 1, 2, ..., 11). In our framework this object has two other corresponding models. First, there is a "planar" graph of constraints (fig. 9), where line segments correspond to vertices and arcs to constraining predicates. For this particular object these constraints may be as follows: A = *line segments are connected*; B = *line segments are parallel*; C = *line segments are of similar length*. Second corresponding model is a Bayesian network that represents stochastic dependencies between the "high-level" concept "cube", intermediate-level concepts "views" and low-level "edges" (that can be observed in the image) (fig. 10).

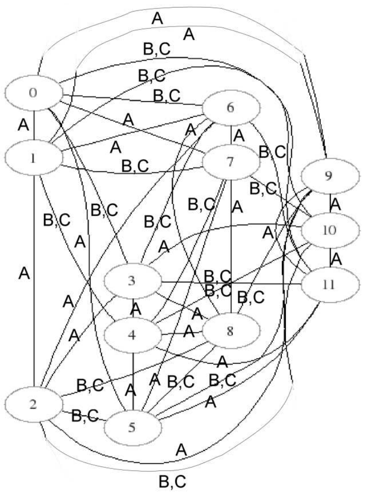

Fig. 9. Graph of constraints for a cube

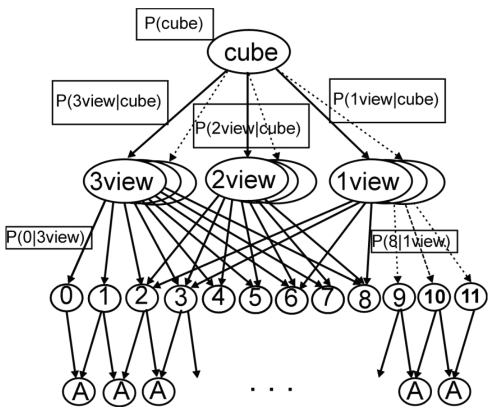

Fig. 10. Bayesian net for a cube

The score of a partial state, in which some number of variables X_i have already been assigned image segments l_k , but not all of them, is obtained due to stochastic inference in Bayesian net, i.e. the computation of posterior probability of "cube" cause, given evidences. For example if segments are assigned to X_0 and X_2 then one computes the probability: $P(cube|X_0 = l_1, X_1 = l_2)$. This leads to a summation of pdf over all domain values for remaining (non-evidence) variables, $X_2, ..., X_l$. Thus, scores of partial matches or a complete match, between image segments and model entities, are naturally obtained by the same evaluation method.

4 Results

4.1 A Rubik cube model

Although a Rubik cube seems to be a straightforward extension of a simple cube but it adds an important feature, a well-defined texture of its faces. Hence instead of a wire-frame model we need to create a surface model for it. First we detect image segments that are rectangular polygons filled by some color and assign them to the single-face model. Polygons satisfying face constraints are grouped into faces. Thus the graph of constraints (fig. 11) contains variables at "two abstraction levels", i.e. the CSP variables correspond to polygons or to faces. The polygon constraints have the following meaning: A = *two polygons are neighbors*, B = *polygons have parallel edges*, C = *polygons are not neighbors*, D = *polygons are of similar size.* The constraints between face variables are: fA = *faces are neighbors*, fB = *the edges of two faces are pair-wise parallel.*

The corresponding Bayesian net has a similar structure in the simple cube case, although now the "face" nodes does not share their "polygon" children

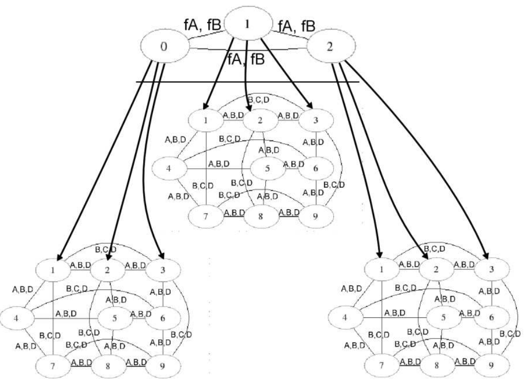
Fig. 11. Graph of constraints for "face" and "polygon" variables of the Rubik cube

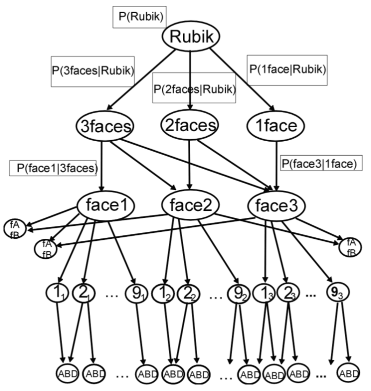
Fig. 12. A Bayesian net for the Rubik cube

(fig. 12). Some results of model-based detection of a Rubik cube are shown in fig. 13-16.

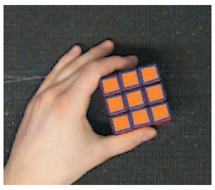
Fig. 13. Polygons grouped to a single face

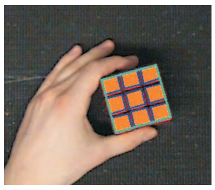
Fig. 14. A single face view is recognized

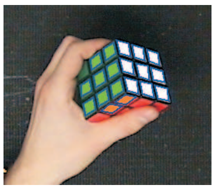
Fig. 15. Polygons grouped to two faces

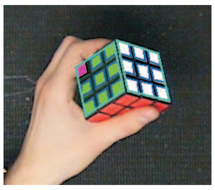
Fig. 16. A two-face view is recognized

4.2 A tea cup model

Let us represent a tea cup again by a wire-frame model. But the edges of a cup, numbered as (0, 1, 2 and 3) (called "variables" in CSP), correspond mostly to elliptic segments and ellipses. The parts 0 and 2 represent arcs that are parts of the top and bottom ellipses in a 2D view of this object. The CSP constraints are: A = *the elliptic arcs are of similar length*, B = *an arc is connected with a linear segment*, C = *the segments are parallel*, D = *the segments are of similar length*. Some results of model-based detection of a tea cup are shown in fig. 17-20.

5 Summary

An application-independent framework has been designed and tested for objects of several types that are passed to the robot by a human. It combines advantages

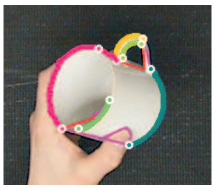
Fig. 17. Example of detected segments

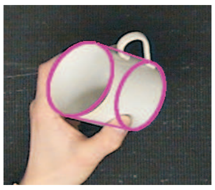
Fig. 18. Object recognition result

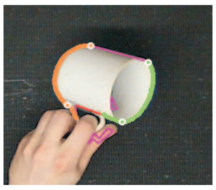
Fig. 19. Example of detected segments

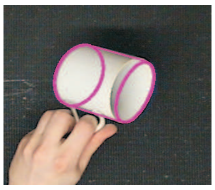
Fig. 20. Object recognition result

of two modelling tools: an easy structure representation and efficient search methods of the CSP model, with well-defined learning and inference methods of a probabilistic Bayesian network model. We demonstrated practically how to handle in this framework wire-frame and textured objects of both linear and curved edges. The aim of future work is to extend the graph of constraints to handle many-level objects, with hierarchies like existing in a semantic network model, and to use continuous pdf in the Bayesian network.

Acknowledgments

The paper has been supported by the Polish Ministry of Science and Higher Education with the grant N N514 1287 33.

References

1. Besl P., Jain R.: Three-Dimensional Object Recognition, Computing Surveys, 17, 75–145 (1985), No. 1
2. Faugeras O.: Three-dimensional computer vision. A geometric viewpoint. The MIT Press. Cambridge, Mass. 1993
3. Tsai W., Fu K.: Subgraph error-correcting isomorphisms for syntactic pattern recognition. IEEE Trans SMC, 13, 48–62 (1983)
4. Niemann H., Sagerer G., Schroder S., Kummert F.: ERNEST: A semantic network system for pattern understanding, IEEE Trans. PAMI 12, 883–905 (1990)
5. Kasprzak, W.: A Linguistic Approach to 3-D Object Recognition. Computers & Graphics 11, 427–443 (1987), No.4, Pergamon Journals, UK
6. Russel S., Norvig P. : Artificial Intelligence. A modern approach. Prentice Hall, second edition, 2002
7. Chan K., Cheung Y.: Fuzzy-attribute graph with application to chinese character recognition. IEEE Trans. SMC 22, 402–410 (1992)
8. Haralick R., Shapiro L.: The Consistent Labeling Problem, Part I. IEEE Trans. PAMI 1, 173–184 (1979), No. 2
9. Duda, R.O., Hart, P.E., Stork D.: Pattern Classification and Scene Analysis. 2nd edition. J. Wiley, New York, 2001
10. Kasprzak, W., Szynkiewicz, W., Czajka, L : Rubik's Cube Reconstruction from Single View for Service Robots. Machine Graphics & Vision 15, 451–460 (2006), No. 2/3, IPI PAN Warsaw