Proceedings of the 1994 Second Australian and New Zeleand Conference on Intelligent Information Systems IEEE Publ. 94TH8019, Piscataway, NJ, pp. 312–316

Consecutive Tree Search for Dynamic Road Scene Analysis

Włodzimierz Kasprzak and Dirk Wetzel [1]

Abstract— **A general analysis strategy is expressed in terms of state space search. The advantages of optimal tree search and true maintenance systems in the context of dynamic analysis are discussed. A consecutive tree search algorithm is proposed for the control mechanism in image sequence analysis. It combines the focusing strength of tree search with locally parallel tracking of competitive interpretations. An example of a road object tracking system under ego-motion is particularly considered.**

Keywords— **tree search, image sequence analysis, inference management, matching, road scene analysis, object tracking, true maintenance systems**

I. Introduction

A robust control for dynamic analysis means usually a tradeoff between computational complexity and the quality of results. This leads to the requirements of *incremental* and *refreshing* analysis modes. The incremental mode means to focus on important model and data parts first and subsequently to extend the temporary interpretation to a full one. The refreshing mode induces two recognition phases – hypothesis initialization and tracking. The key problem is a proper selection of the initial interpretation among the set of competitive interpretations. This requires the existence of a robust *judgment* scheme of partial results. For the complex outdoor scene analysis this task has not been solved yet. Opposite to single image analysis in the case of dynamic analysis the *stability* of a hypothesis over the time allows the creation of reliable judgment schemes. But this requires a parallel tracking of competitive interpretations.

In this paper a control paradigm for image sequence analysis is discussed that satisfies the requirements of incremental mode and stability of generated results. A general analysis strategy is presented in section II in terms of a partial interpretation space (*state space*) traversal [1]. Two classes of approaches to the management of *competitive* analysis results are distinguished: a *tree search* control ([2]) and a *truth maintenance* system ([3], [4]). A related design problem discussed also in section II concerns the problem of *matching* classes between the model and data set ([5]). The *consecutive tree search* algorithm is described in section III. The road scene analysis under ego-motion ([6]) constitutes the application field of the algorithm (section IV).

[1]The authors are with the Bavarian Research Center For Knowledge Based Systems: FORWISS, Am Weichselgarten 7, D–91058 Erlangen, Germany, e-mail: *name*@forwiss.uni-erlangen.de

II. General Design Problems

A. Incremental Analysis Strategy

In the case of a complex hierarchical scene model a general strategy of model and data subset selections should be performed by each analysis control. Such a strategy tries to avoid an exhaustive model– or data–dependent matching while focusing on important model concepts or data items ([7]). Let us shortly explain a general strategy similar to the one presented by Niemann et al. in [8]. In **Fig.** 1 examples of a hierarchical model and a data set are provided (top drawings). The ultimate analysis goal is to create the best instance of a top level concept in the model and to explain the largest part of the low level data. At least two *inference* classes (concept modification and instantiation) are possible. The modification can be performed in two ways: from top to down or from bottom to up along the part hierarchy. The first modification type performs a data–driven selection of a (temporary) goal concept in the model, whereas the second one determines the sub-model for instantiation (matching) and at the same time, by restricting the domains of concepts, it induces restrictions on the data items and inference results. For example during the analysis in **Fig.** 1 the bottom–up selection of concept A, the top–down modification of concepts (B, C, D) and the bottom–up instantiation of modified concepts $(Q_1(D), Q(B), Q_2(D), Q(C), Q(A))$ takes place.

The control strategy consists of repeated application of inferences to specified model concepts and data items. We explain the strategy by means of by the well known *state space* paradigm [1]. The *inferences* are here abstracted to movements in the state space of partial interpretations, that is generated over the data and model sets (**Fig.** 1 (bottom)). In the first step the concept A is selected and a modified concept $Q(A)$ is generated – this is equivalent to a *hypothesis generation* step $H1$ in state space. A *testing* step for $H1$ means a local analysis for the modified concept $Q(A)$. A modification is equivalent to the *backward chaining* between (implicit) regions in state space (the steps $R1, R2, R3, R4$) and an instantiation means the *forward chaining* between states (the steps $V1$, $V2$, $V3, V4, V5$).

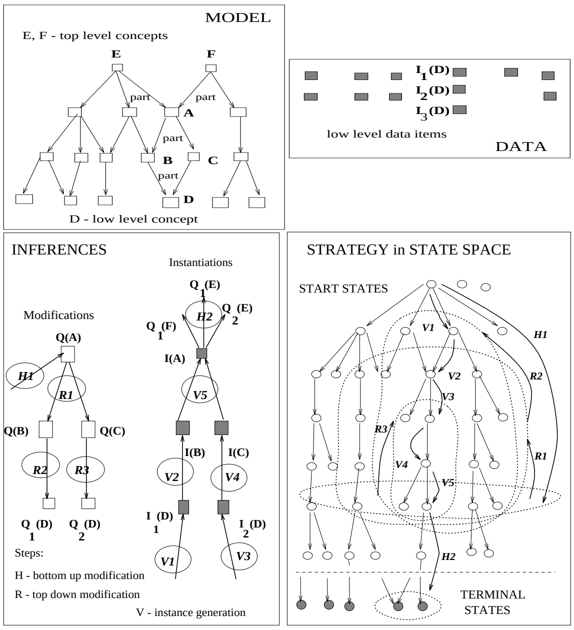


Fig. 1 The inferences and state space traversal

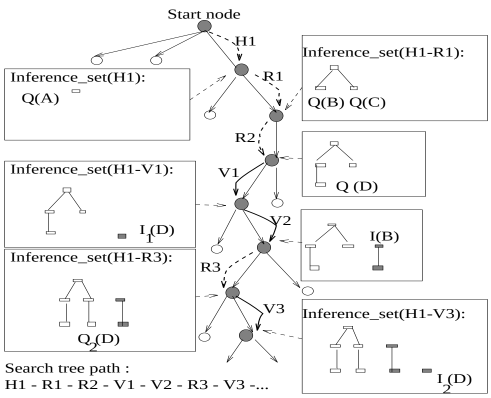


Fig. 2 The decision tree search

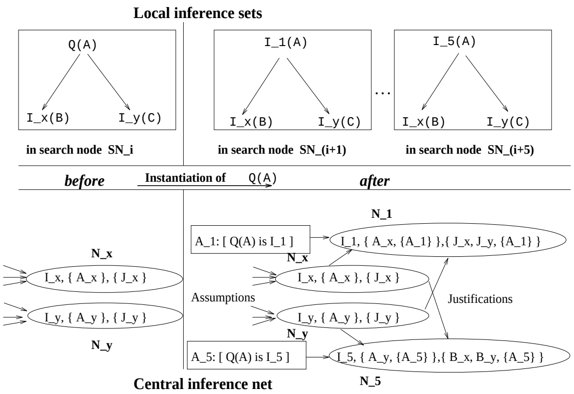


Fig. 3 The central inference net

B. Consistency Maintenance

Now the question is how the decisions taken during the analysis are represented and how a consistent interpretation is searched for. Two different approaches are considered: (a) *decision tree* search [2] and (b) *central inference net* maintenance [4].

In the case (a) the processing steps $H1, R1, R2$, $V1, V2$, $R3, V3$, $V4$, $V5, H2$ correspond to one search path in the decision tree (**Fig.** 2). Each graph node represents one consistent inference subset. An optimal tree search is assumed, that is guided by judgments of search tree nodes. This algorithm allows to focus on important hypotheses without a time consuming check of all the possible interpretations.

In the case (b) only one central inference net is given that may contain many inconsistencies. In the example in **Fig.** 3 five competitive instances of the modified concept $Q(A)$ are generated. In the search paradigm this instantiation causes the generation of five competitive search tree nodes, where each of them contains one instance of $Q(A)$. In the second paradigm five assumption nodes $A_1, ..., A_5$ and five justification nodes $J_1, ..., J_5$ are added to the central inference net. An assumption is contained in the net as long as one of its conclusions (justification nodes) is not proved to be wrong. All the justification nodes containing this wrong assumption can be immediately removed from the inference net.

C. Matching Classes

In the instantiation example in **Fig.** 1 exactly one instance for each modified concept has been generated. In general the number of modified concepts and instances and the ordering of inference steps may be flexible. This flexibility results from the existence of two *matching* classes: (1) a *model dependent* matching means a search for data items (i.e. image segments, previous inference results) that can fit given model entity, whereas the (2) *data dependent* matching looks for model entities, that explain a given data item. In the former case the number of matching pairs *(concept, data item)* is determined by the model, whereas in the later case the number of matching pairs *(data item, concept)* is determined by the available data set.

D. Problem Definition

In this paper a control algorithm for dynamic image analysis is defined that satisfies two (to some extent excluding) conditions: (1) it avoids an exhaustive search, (2) a proper initialization of consistent interpretations and a stable tracking of instances are provided.

In the context of dynamic analysis the consistency maintenance approach has at first view an advantage over the tree search approach if stabilization power is considered. As the decision about competitive instances is postponed to a later time (if necessary after several images) a parallel tracking of competitive hypotheses is always possible. Thus the tracked hypotheses are stabilized independently all the time. The selection decision made after such a long initialization phase is more proper than an immediate decision.

APPL: *Parameter_Init*
Search_tree_Init;
WHILE APPL: *End* is not satisfied
  WHILE search nodes exist in OPEN
    v = *Select_and_eliminate_node_from_*OPEN;
    IF APPL: *Single_End* is satisfied
    THEN GOTO NEXT
    APPL: S = *Goals(v)*; bottom–up modification
    IF S is not empty
    THEN *Init_Subspaces(S)*;
    ELSE IF an Entity $o_l \in$ DATA(v) can be refreshed
         THEN *Refresh(* o_l, v*)*;
         ELSE IF an Entity $o_l \in$ DATA(v) can be instantiated
              THEN *Instantiate(* o_l, v*)*;
              ELSE IF an Entity $o_l \in$ DATA(v) can be modified
                   THEN *Modify(* o_l, v*)*;
  NEXT: *New_Init()*

Fig. 4 The consecutive tree search algorithm

The use of data dependent matching for concepts, whose instance number can only be approximately specified in advance, enlarge the stabilization power of the analysis. This matching case is also useful for the process of combining data items, which have different judgments, into a more abstract unit ([9]) and at the beginning of the analysis.

At other side the selectivity of analysis is an important requirement for image sequence analysis. Here an optimal tree search approach would be prefered as its provides inconsistency–free solution paths with a judgment controlled generation of partial interpretations. The selectivity is increased if a model dependent matching is made for every concept.

III. Consecutive Tree Search

A tree search approach with model dependent matching constitutes the basis of proposed control paradigm. Due to the specific stability requirements of image sequence analysis, the approach includes elements of data dependent matching and a parallel propagation of inconsistent sub-interpretations.

A. The Algorithm

The general *consecutive* tree search algorithm (image sequence analysis with explicit search space representation) is described in **Fig.** 4. The functions activated by the control module are always search tree operators. They select and expand the current best search tree node, according to the local inference set of the node. The inner WHILE–cycle corresponds to single image processing. There are three types of inference operators that generate, modify or *refresh* the instances and perform appropriate search space expansion. The function *New_Init* selects some search space nodes and propagates them to the next image search tree.

The *refreshing* operator is equivalent to a repeated instantiation of instances, that are referred by a propagated search tree node. In this way it means an update of previous instances on the basis of new data items. In order to refresh an instance all of its part instances should be already refreshed.

The application functions (APPL) determine the processing parameters, the termination conditions and the selection of temporary goal concepts in the model.

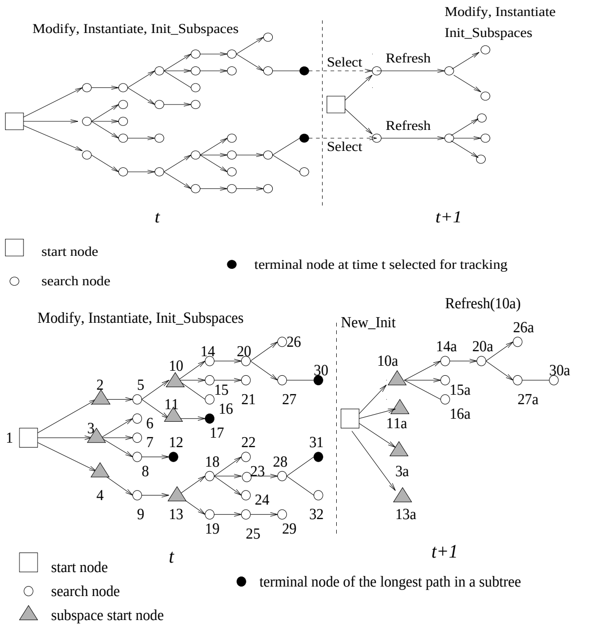


Fig. 5 Two solutions to search tree propagation.

B. Search Tree Propagation

We distinguish two versions of the *New_Init* procedure. In the first solution a selection of best nodes from the current terminal set is done first. These nodes are the starting nodes of the new search tree. An immediate correspondence of data items and previous instances to be refreshed is assumed, on the basis of equal indices in the set of competitive instances. This solution would correspond to an immediate refreshing of all instances of selected search node in *Refresh*. Only one successor node of the "refreshed" node in the search tree would be generated. This solution would work well only then if the proper interpretation is contained in one of the propagated search nodes and if the same number of hypotheses is generated in subsequent images.

As this requirement is seldom satisfied an other solution to *New_Init* was implemented. Subspaces that correspond to different goal concepts are recognized. The root node of a subtree was generated by a bottom–up modification of a goal concept. Instead of propagating some terminal nodes all the root nodes of subtrees, that contain a node in the OPEN set, are selected for propagation. On the basis of the instances from selected subtree an *appended* subtree root node is propagated to the next

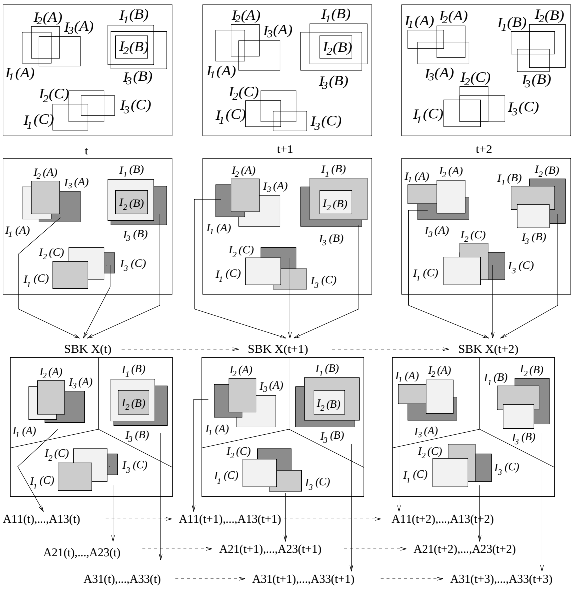


Fig. 6 Example of two tracking modes

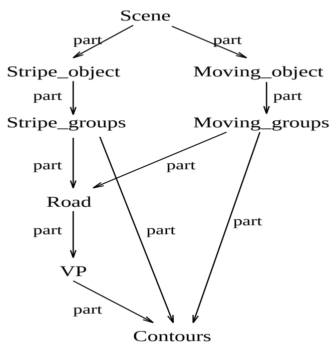


Fig. 7 The model structure for road object tracking

search tree. While the original node contains modified concepts an appended root node refers all the instances, that have been generated in its subtree (called a *local instance* net).

C. The Refreshing Step

If the *Refresh* procedure is applied to an appended node the longest path from the previous subtree is repeatedly generated (**Fig.** 5 (bottom)). As the previously performed instantiations are now repeated, the generated hypothesis numbers need not to be equal in both images. All competitive instances along the refreshed search path are updated (i.e. recursive estimation of instance attributes) and not only the currently best instances. In this solution it is no longer required that the same number of competitive instances (structure) is generated in subsequent images. Now after several images the refreshed longest "search tree path" should be most of the time the best solution in this subtree. If this path is proved to be wrong the analysis for current image can still be continued from other search paths with the information contained in the local instance net of the appended start node.

D. Local Instance Net

Instances/Nodes	Time 0	5	20	50
I(VP)	3	3	3	2
I(Road)	4	4	2	2
I(Stripe_groups)	2	2	1	1
I(Moving_groups)	2	2	1	1
I(Moving_object)	35	21	3	3
I(Stripe_object)	12	6	6	6
I(Scene)	2	2	1	1
Total I	60	45	17	16
Nodes	138	107	58	54
Inconsistency	of all	small	no	no

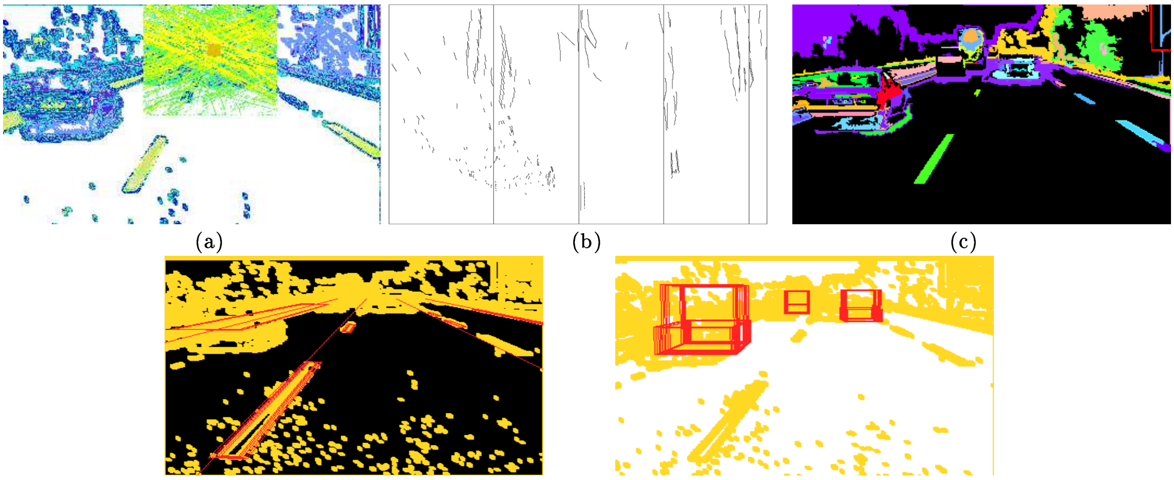
Tab. 1 Example of consecutive search complexity

Let us explain the local tracking of competitive instances by the example in **Fig.** 6. In the image at time t a subspace for tracking three objects in the image has been expanded. The longest path in the subspace contains the instantiation of three object concepts A, B, C. In each instantiation step three competitive instances have been generated. Assuming that the number of generated instances in two consecutive images is the same an immediate selection of the best instance leads to a node with three instances $\{I_3(A), I_3(B), I_3(C)\}$ in image (t), $\{I_1(A), I_3(B), I_2(C)\}$ in image $(t+1)$ and $\{I_3(A), I_2(B), I_3(C)\}$ in image $(t+2)$. If at the time $(t+2)$ it is verified that the instances tracked with the node X_{ijk} are wrong, there is no possibility to go back to competitive solutions as long as the other search tree nodes have not also been continuously refreshed. Due to the tracking of one tree node with three instances alone the image description will not be stable. There is a high probability that in the initialization phase 9 search paths of length 3 with total number of $3 \times 3 \times 3$ instances (27 search tree nodes) will be generated.

The bottom drawing in **Fig.** 6 represents the local inference net propagation. In this case one search path of length 3 with 3×3 competitive instances (9 search tree nodes) will be generated only. After several images a better stability of some instances should be achieved and the selection of the consistent instance set will be much easier than the single image based selection. The number of competitive instances in the net is reduced step by step to one instance for each object. Thus in the tracking phase the local instance net will be reduced to 3 instances only.

IV. Example

An example of a road scene model for road object tracking under ego-motion is presented in **Fig.** 7. Competitive instances of following concepts are usually generated during the analysis (**Fig.** 8): Vanishing point (VP), Road, Groups, Stripe_object and Moving_object. Competitive instances of the concept Scene represent different object numbers if a model dependent matching for the concept Moving_object is applied.



(a) (b) (c)
(d) (e)

Fig. 8 Instances: (a) vanishing point, (b) road (top view), (c) groups, (d) 5 stripe objects, (e) 3 moving objects

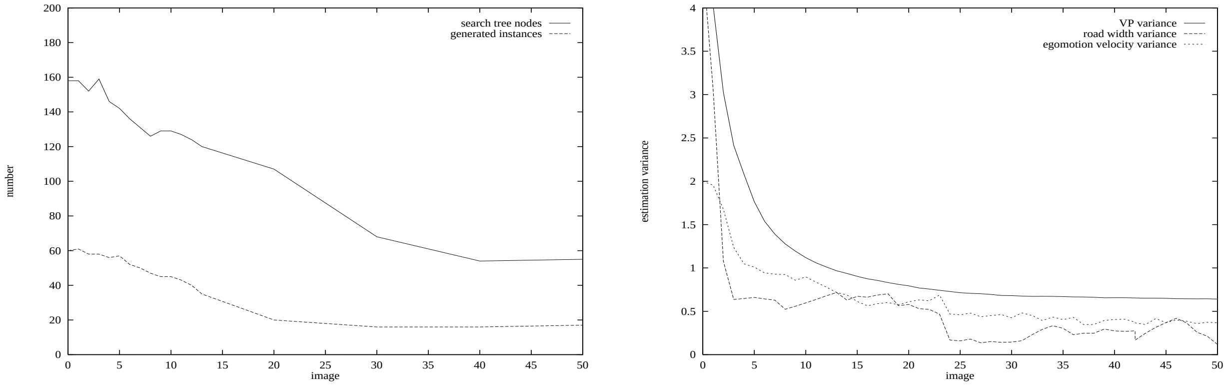


Fig. 9 Results: (left) complexity of search, (right) stability of three instances

Following complexity of the consecutive tree search algorithm has been experimentally verified for given model and image sequence (**Table** 1): during the initialization phase the average number of competitive instances for one concept was reduced from three to two; after 50 images 1.2 competitive instances were stored in average (**Fig.** 9(a)).

The stability of tracked instances was achieved after 10–15 images. In **Fig.** 9(b) estimation variances of three refreshed instances in 50 images are presented. The variance of the vanishing point position is expressed in $100 pixel^2$ units, the road width variance is given in m^2 and the translational velocity of the ego-motion is given in $(m/0.04s)^2$.

V. Summary

Two control paradigms for image analysis – with an explicit search graph and with a central inference net – have been discussed. A consecutive tree search algorithm has been defined which contains a mechanism for local instance net tracking. In this way a flexible tradeoff between low computational complexity and high quality of the image sequence analysis is possible.

Acknowledgment

The work reported here was supported by the *Alexander von Humboldt Foundation*, Bonn, Germany and the *Deutsche Forschungsgemeinschaft, Bonn, Germany* The real image sequence is by courtesy of the *BMW AG., Munich, Germany.* Only the authors are responsible for the content of the paper.

References

[1] Nilsson N.J.: *Principles of Artificial Intelligence*, Berlin etc, Springer, 1982.

[2] Pearl J.: *Heuristics. Intelligent Search Strategies for Computer Problem Solving*, Addison–Wesley, Reading, Mass., 1984.

[3] Doyle J.: *A Truth Maintenance System*, Artificial Intelligence, *12*(1979), 231–272.

[4] De Kleer J.: *An Assumption–Based TMS*, Artificial Intelligence, *28*(1986), 127–162.

[5] Ballard D.H., Brown C.M.: *Computer Vision*, Prentice Hall, Englewood Cliffs, NJ, 1982.

[6] Graefe V., Kuhnert K.D.: *Vision-based autonomous road vehicles.* In: I. Masaki, ed., *Vision-based Vehicle Guidance*, Springer, 1992, 1–29.

[7] Riseman E.M., Hanson A.R.: *The VISIONS Image Understanding System.* In: Brown Ch.(ed.): *Advances in Computer Vision*, Hillsdale N.J., *1988*, 6–103.

[8] Niemann H. et al.: *ERNEST: A Semantic Network System for Pattern Understanding*, IEEE Transactions on Pattern Analysis and Machine Intelligence, *12, 1990*, 883–905.

[9] Grimson W.E.L.: *Object Recognition by Computer: The Role of Geometric Constraints*, MIT Press, Cambridge, Mass., 1990.