

Learning from the Schema Learning System

Bruce A. Draper*

Dept. of Computer Science
University of Massachusetts
Box 34610
Amherst, MA., 01003-4610
bdraper@cs.umass.edu

Abstract

A major problem that confronts developers of knowledge-based vision systems is how the information in the knowledge base (both object-specific knowledge and the control strategies for applying the knowledge) is acquired. Hand construction of knowledge bases is an onerous, time-consuming and errorful process, and is inadequate for systems requiring more than a few object models, especially if the domain within which the system operates changes. Over the past four years we have addressed this problem by developing the Schema Learning System (SLS) to learn strategies for applying object-specific knowledge in complex domains. The goal is build a system that can learn (under supervision) to recognize a new object or object class without any direct human intervention.

This paper will not describe SLS in any detail; that has been done elsewhere. (See Draper et al. [9] for a preliminary presentation, or Draper [10] for a more recent and complete description). Instead, this paper summarizes some of our conclusions from four years of machine learning and computer vision research, emphasizing our representation of recognition strategies, our syntactic approach to integrating visual modules, and the implications of SLS for goal-directed, as opposed to reconstructionist, theories of vision.

Introduction

Computer vision research often involves searching for sequences of visual operators that can achieve a desired goal. For example, McGlone and Shufelt [12] recently presented a paper in which they sequenced four simple operators – line extraction, corner detection, linking and matching – to find roofs in aerial images. Their work is typical of

applications-driven research in that they adopt a performance task, in this case roof recognition, which they solve using existing computer vision techniques. Their approach is to build a program that manipulates a series of increasingly abstract representations, starting with line segments and then progressing to corners, chains of lines and corners, and finally parallelepipeds. At each level of representation, data instances are filtered, eliminating to the extent possible those that are not (parts of) roofs. Parallelepipeds that are not filtered out are returned as hypothesized rooftops.

We will use McGlone and Shufelt's work as a running example because it is typical, in both the problem statement and the approach, of so much work in knowledge-directed vision. Nearly every performance task in computer vision seems to require a slightly different combination of visual operators and representations to satisfy it. In part, this springs from differences between object classes: some objects have distinct colors, while others have identifiable shapes, textures, or other defining properties. In part it arises from having different visual goals: identifying an object, for example, is a different problem from determining its position and orientation relative to the camera. Whatever its sources, the integration problem is difficult because of the lack of a general theory of how the components of a vision system should be combined.

For the past four years, we have hypothesized that the key to visual integration is learning. As a first step toward testing this hypothesis, we have developed the Schema Learning System (SLS), a supervised learning system for developing object- and goal-specific recognition strategies. SLS is given a problem statement in the form of a goal, such as to recover the (3D) pose and orientation of a building, or the (2D) image location of a tree. From the problem statements, a set of training images, and a library of visual operators, it learns executable recognition strategies that satisfy the goal by invoking controlled sequences of known operators [10]. Figures 1 and 2, for example, show the result of looking for the pose of a specific building

*This work is supported in part by Rome Labs under contract #F30602-91-C-0037 and by ARPA (via TACOM) under contract #DAAE07-91-C-RO35

and the image location of a tree by reprojecting the hypotheses returned by learned strategies back onto a test image. These recognition strategies replace those that computer vision researchers such as McGlone and Shufelt currently build (at great expense) by hand.

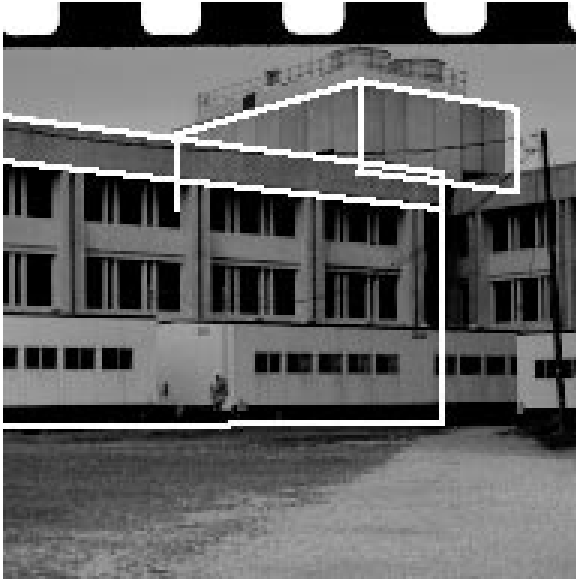


Figure 1: The pose of the Lederle Graduate Research Center (LGRC), reprojected onto the image it was recovered from. The pose was recovered by a special-purpose strategy learned by SLS for determining the position and orientation of the LGRC.

We will not describe the learning algorithms used by SLS here; interested readers are referred to my dissertation [10]. Instead, we will present informal conclusions drawn from our experience with SLS, focusing on items that may be of interest to other researchers in the area of machine learning and vision.

Representing Recognition Strategies

To the extent that SLS is successful, it is successful because of its representation of recognition strategies. SLS represents strategies as multi-level decision trees called *recognition graphs* that are generalized to direct hypothesis formation as well as hypothesis verification (see Figure 3). The premise behind the formalism is that object recognition is a series of representational transformations interleaved with small verification tasks. Strategies begin by extracting low-level hypotheses, such as points, lines, or regions, from an image. Then recognition strategies verify these (low-level) hypotheses, separating to the extent possible hypotheses that are reliable from those that are not.



Figure 2: A tree located by a special-purpose strategy learned by SLS. The goal of the strategy is to find the (2D) image location of trees.

Verified hypotheses (or, more accurately, hypotheses that have not been rejected) are then transformed to a more abstract level of representation, where a new verification process takes place. The cycle of transformation followed by verification continues until hypotheses that satisfy the goal (i.e. the problem statement) have been generated and verified.

The structure of the recognition graph reflects the verification/transformation cycle. Each level of a recognition graph is a decision tree that controls hypothesis verification at one level of representation by computing selected features and determining if the corresponding hypothesis is reliable. When a hypothesis is verified it is transformed (by a transformational operator) it to another level of representation, where the process is repeated.

That each level of a recognition graph is a univariate decision tree is not particularly important. Any two-class classifier can be used, and we are currently developing recognition graphs that use artificial neural networks and multivariate decision trees instead. (One could also build a hybrid system that selected the best classification induction algorithm for each level of abstraction, *a la* Brodley [7].) It is important, however, that the classifier have a very low false negative rate. At all but the highest level of abstraction, if a classifier verifies a false instance the mistake will most likely be corrected when the instance is transformed to the next level of representation. If a classifier rejects a correct instance, on the other hand, it is a mistake that the system

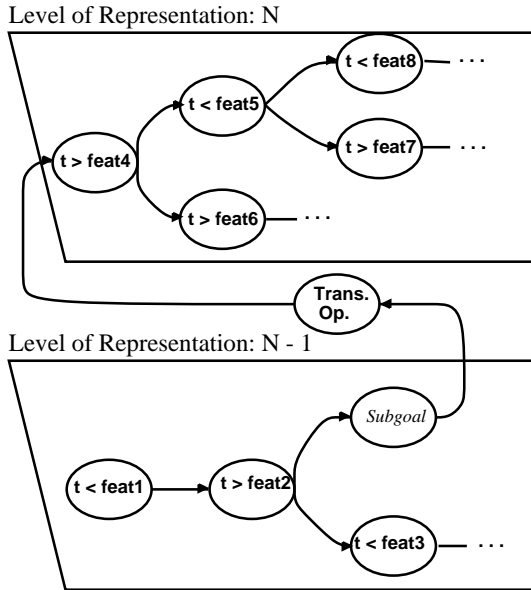


Figure 3: A recognition graph. Levels of the graph are decision trees that verify hypotheses by measuring their features. Hypotheses that reach subgoals are transformed to the next level of representation by transformation operators.

cannot recover from. Consequently, the classifiers in SLS are trained to produce many more false positives than false negatives in order to maximize the overall performance of its strategies.

The significance of the recognition graph formalism is that it biases the learning algorithms toward the types of strategies that human knowledge engineers have found to be successful. McGlone and Shufelt’s system, for example, maps directly onto the recognition graph formalism: they use four transformation operators (image to line segments, line segments to corners, corners and line segments to chains thereof, and chains of lines and corners to parallepipeds), and after each transformation they filter out unreliable hypotheses. The recognition graph is also significant because it cast the learning problem as one of both classification and symbolic (transformational) inference. To our knowledge, members of the machine learning community have looked at both classification and symbolic inference, but not at the combination of the two.

Syntactic Approach to Integration

In terms of the integration of visual modules, SLS is on the opposite end of a spectrum from the approach taken by Aloimonos and Shulman [1]. Aloimonos and Shulman recommend analyzing the semantics of each visual operator closely, and integrating them by merging their mathematical con-

straints. Although such an approach is very powerful in theory, it is difficult, if not impossible, in practice.

SLS, on the other hand, takes an essentially syntactic approach to visual integration. It starts with a library of visual representations and transformational operators. For each representation, the library contains a list of measurable features and routines for measuring them. For example, the library entry for line segments specifies that they have endpoints, length and contrast, and gives the datatype of each. For transformational operators, the library specifies the type of representation(s) required as arguments and the type of representation produced, so that a line extraction operator, for example, takes an image as input and produces line segments as output. The library also contains default values for any compile-time parameters a transformational operator might have. The library does not contain estimates of the cost or reliability of an operator, since we discovered that our estimates of such values tended to be highly inaccurate.

Obviously, the purely syntactic approach has some disadvantages. SLS operates at the level of an automatic programming system, with no special knowledge about physics or optics or geometry. The advantage of the syntactic approach, however, is that it allows SLS to integrate widely disparate operators and representations, so that a single recognition strategy can exploit many different types of information, including but not limited to color, texture, shape, size, context and function. It also allows new operators and representations to be added to the system as they are developed.

A Theory of Vision

According to the *Encyclopedia of Artificial Intelligence*, “the goal of an image understanding system is to transform two dimensional data into a description of the three dimensional spatiotemporal world.” [13, pg. 389]¹ Such definitions reflect the influence of Marr and the *reconstructionist* school of computer vision [11], which holds that vision is the process of reconstructing the three dimensional geometry of a scene from two dimensional images, essentially by inverting the geometry and physics of optical perspective projection. Symbolic recognition is viewed as a secondary process that follows and is dependent upon geometric reconstruction.

There is, however, an alternate definition of computer vision, in which the goal is to enable actions, generally by applying concurrent, special-purpose strategies. This *goal-directed* approach has roots in the psychology literature of the 1960’s and

¹Although the citation is to the original source, we discovered this quotation in the introductory paragraph of Dana Ballard’s article on animate vision [6].

the cybernetics literature of the 1940's and '50s. (See Arbib [5] for a review.) What is particularly compelling about this approach, however, is that it keeps resurfacing in the computer vision literature with different motivations. It first appeared in the early 1970's in the work of Arbib, who modeled perception in terms of action-oriented schemas [3, 4]. Several years later, Brooks proposed a layered approach with each layer integrating perception and action as a solution to problems of real-time robotics [8]. Similar ideas surfaced again in the work of Aloimonos, who was trying to circumvent the practical difficulties of reconstructionist vision [2], and Ballard, who, like Arbib, was modeling biological vision [6]. Partly as a result of this repeated convergence, the notion of vision as a collection of concurrent, special-purpose strategies is once again gaining popularity.

We believe SLS gives a boost to theories of goal-directed (a.k.a. purposive) vision. Such theories have been criticized by researchers of the reconstructionist school who argue that the goal of computer vision research is not just to create object recognition systems, but to put forth a coherent and parsimonious theory of vision. These researchers claim that by modeling vision as a loose (to be critical, *ad hoc*) collection of special-purpose recognition systems, proponents of goal-directed vision abandon that goal. SLS puts forth a counterclaim by example, however; a claim that special-purpose recognition strategies do not have to be *ad hoc* or unstructured, that they can arise through predictable and scientific mechanisms in response to a viewer's environment. Indeed, the criticism can be turned around: given that special-purpose strategies can be acquired through experience, it seems unnecessary and unjustified to assume that all visual goals must be met by a single general-purpose mechanism.

Future Work

Research on the Schema Learning System is by no means finished. Ongoing efforts include work on 1) integrating systems that learn structural object models (such as structure from motion algorithms), 2) automatically learning parameter settings for visual operators (see Bhandaru, *et al.*, this volume), and 3) combining dynamic and static control of visual operators. Longer term goals include reducing the amount of supervision needed during training, having SLS determine its own object classes, and learning concurrent special-purpose strategies that cooperate to interpret a complete scene.

Conclusion

SLS is an example of using machine learning technology to solve a pressing problem in computer vision. Its success, however, depends less on radical

scientific breakthroughs than on formalizing common knowledge. Although many systems use hand-coded knowledge to integrate visual modules, and although the techniques of writing such systems in terms of reasoning across multiple levels of abstraction and the "generate and test" paradigm are well known, no other system puts them together into a formalism like the recognition graph, or tries to bias learning to produce such strategies. Indeed, the very idea of approaching visual integration at the level of automatic programming was anathema to many who preferred to think in terms of physics, optics or geometry. Despite its pragmatic approach, however, SLS ends up making a contribution to the general theory of computer vision by helping crystallize the alternative to Marr: vision as a collection of special-purpose skills acquired in response to the environment.

At the same time, the machine learning community has tended to study classification and symbolic inference as distinct problems, even though interpretation tasks such as vision seem to require a combination of the two.

References

- [1] J. Aloimonos and D. Shulman. *Integration of Visual Modules: An Extension of the Marr Paradigm*. Academic Press, Inc., Boston, 1989.
- [2] J. Aloimonos. "Purposive and Qualitative Active Vision," *Proc. of the DARPA Image Understanding Workshop*, Pittsburgh, PA., Sept. 1990. pp. 816-828.
- [3] M.A. Arbib. *The Metaphorical Brain: An Introduction to Cybernetics as Artificial Intelligence and Brain Theory*. New York: Wiley Interscience, 1972.
- [4] M.A. Arbib. "Segmentation, Schemas, and Cooperative Computation," *Studies in Mathematical Biology, pt. 1*. S. Levin (ed.). MAA Studies in Mathematics, vol. 15, 1978, pp. 118-155.
- [5] M.A. Arbib. "Schema Theory," in *The Encyclopedia of Artificial Intelligence*, 2nd ed., S.C. Shapiro (ed.) New York: Wiley and Sons, 1992, pp. 1427-1443.
- [6] D.H. Ballard. "Animate Vision," *Artificial Intelligence*, 48:57-86 (1991).
- [7] C.E. Brodley. "Addressing the Selective Superiority Problem: Automatic Algorithm/Model Class Selection," *Proc. of Tenth International Machine Learning Conference*, June 27-29, Amherst, MA., pp. 17-24.
- [8] R.A. Brooks. "Intelligence without Representation." *Proc. of the Workshop on the Foundations of Artificial Intelligence*, Cambridge, MA.: MIT Press, 1987.

- [9] B. Draper, A. Hanson and E. Riseman. "Learning Blackboard-based Scheduling Algorithms for Computer Vision," *International Journal of Pattern Recognition and Artificial Intelligence*, 7(2), 1993.
- [10] B. Draper. *Learning Object Recognition Strategies*. Ph.D. dissertation, Univ. of Massachusetts, 1993. Available as tech. report 93-50 from the Dept. of Computer Science.
- [11] D.C. Marr. *Vision*. San Francisco: W.H. Freeman and Co., 1982.
- [12] C. McGlone and J. Shufelt. "Incorporating Vanishing Point Geometry Into a Building Extraction System," *Proc. of the ARPA Image Understanding Workshop*, Washington, D.C., April 1993. pp. 437-448.
- [13] J.K. Tsotsos. "Image Understanding," in *The Encyclopedia of Artificial Intelligence*, first edition, pp. 389-409. 1987.