Self Adaptive Software

A White Paper for the Workshop on New Visions for Software Design and Productivity

By

Paul Robertson

Dynamic Object Language Labs, Inc.
9 Bartlet Street #334
Andover, MA 01810

978 372-7635

probertson@doll.com

We have proposed self-adaptive architectures as an ideal approach for solving problems of aerial image understanding [1]. It is clear that these architectures are useful for a broad collection of problems, including problems of highly volatile dynamic embedded applications. In order to demonstrate such capability, we are expanding and extending the GRAVA architecture to new image understanding domains (face recognition) and extending it for real-time use. In the following sections we briefly describe GRAVA and the extensions currently underway.

We have proposed self-adaptive architectures as an ideal approach for solving problems of aerial image understanding [1]. It is clear that these architectures are useful for a broad collection of problems, including problems of highly volatile dynamic embedded applications. In order to demonstrate such capability, we are expanding and extending the GRAVA architecture to new image understanding domains (face recognition) and extending it for real-time use. In the following sections we briefly describe GRAVA and the extensions currently underway.

GRAVA Architecture

The basic approach is to treat a vision program as a "circuit" of interconnected special purpose agents. This leads to a model in which agents negotiate for inclusion in the current "circuit". This is similar to the way that "methods" negotiate with the method combination algorithm in CLOS to produce a generic function. The combination that takes place in CLOS utilizes the specification or signature of the individual methods and an open method combination mechanism. In the case of visual interpretation agents specifications can not usually be known precisely but may be approximated by evaluating the effectiveness of various agents on a training set of visual images. Agents have explicit representations of their capabilities and on that basis are able to construct a good circuit for the problem domain in the current environment. Because the agents are not blindly executing their tasks but are involved in a feedback loop with the environment, they are able to renegotiate their roles when the environmental conditions change.

We employ a probabilistic and information theoretic framework for agent negotiation developed at Oxford University that allows circuit synthesis to be viewed as a process of maximizing mutual information between specified behavior and actual behavior and for an active feedback mechanism to be involved in continually trying to maintain that maximum in the face of a changing environment.

The implementation consists of a C++ library of image analysis primitives and an agent architecture implemented in Yolambda (a dialect of Lisp). The agent architecture deals with several aspects of the problem solution:

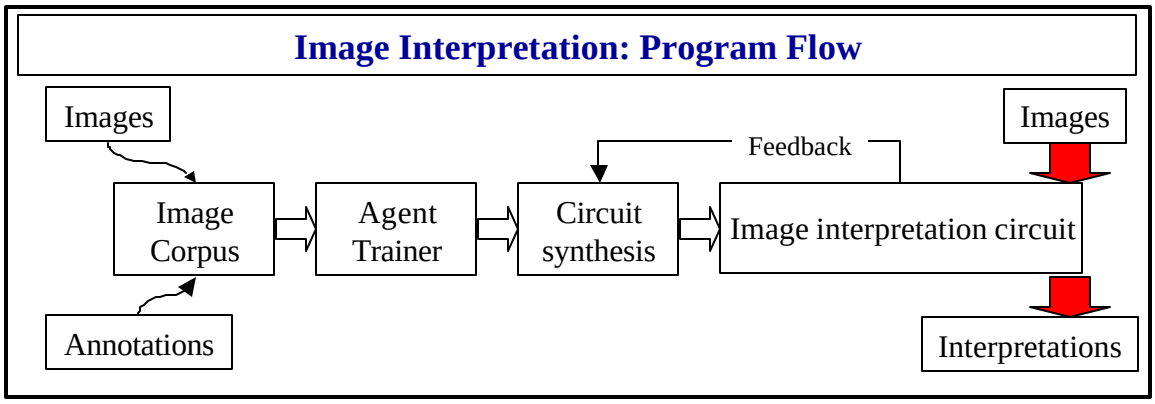
1. It allows the agents to be trained on a corpus of representative images so as to learn the relationship between parameters of the base tools with relevant image features.

2. It allows meta-agents to learn dependencies among the agents and to build a representation of the utility of each agent to particular image analysis requirements.

3. It allows for meta information represented in the meta-agents to monitor the performance of the systems as it operates on real data.

4. It allows the program to self-adapt in response to changes in the environment.

The current architecture is depicted below in the program flow diagram:



The feedback loop allows the performance of the program on real images to cause the circuit synthesis module to adapt the image interpretation circuit so as to ensure the best match between the produced interpretations and the specifications provided in the form of annotations.

The original system needs significant extra work in order to meet more general and real-time objectives. We intend to pursue the following tasks in the proposed extension of the contract in order to carry this technology to a mature demonstration of the self-adaptive program paradigm.

Optimization of key capabilities and tools

The system described constitutes a very different kind of design problem to engineers than traditional programming environments. In fact debugging systems like the one described here is much more like debugging a control system than debugging a traditional program (of course the programming of the individual agents is still a traditional kind of programming problem). In traditional programming problems we are concerned that our program performs correctly but in the system described, we are interested in what happens at key moments when agents use their self knowledge to first detect a discrepancy and then produce a 'corrective force' that results in adaptation from the circuit synthesis capability. In debugging such systems we are interested in smooth transitions between re-synthesized circuits. Adaptive software is highly non-linear making a thorough mathematical approach to guarantees of stability problematic. We need tools that are intimately linked to the adaptive architecture that allow debugging and monitoring of adaptation transitions.

Face recognition is a well-studied field and numerous algorithms have been developed that do various aspects of face tracking and identification. Existing algorithms vary in robustness, speed, and versatility. For example many algorithms require faces to be presented from a canonical view. DOLL is building a face understanding capability for the MIT intelligent room [2] that can meet a variety of needs and be flexible enough to incorporate a variety of algorithms and change over time as needs change.

Among the capabilities that are useful in the context of the intelligent room are:

- The ability to track the positions of faces within the room as people move about.
- The ability to identify and track multiple faces in the room.
- The ability to usefully integrate information from multiple cameras in multiple locations in the room to achieve identification and tracking of faces.
- The ability to keep track of how many people are in the room at any point in time.

- The ability to compare faces found in the room against a database of faces in order to search for a match.

The architecture will have to rationalize requirements of the room at a point in time including priorities and the recourse requirements of various combinations of algorithms and the availability of computing resources and sensors. Tracking may occur in real time whereas identification of faces may be performed over a number of frames. In some cases algorithms may be composed in order to achieve the required ends. For example in identifying faces certain algorithms require a canonical pose. Given two or more camera angles there are a number of algorithms for producing 3D models from which canonical poses can be synthesized. Such algorithms when composed with a canonical pose type face identification algorithm may allow face recognition to occur in unconstrained contexts. The existing self-adaptive image understanding architecture allows for the composition of "tools" in the way described above.

The individual recognition and tracking algorithms determine how well the intelligent room can potentially "see" but the ability to dynamically reconfigure which algorithms are being utilized based on factors such as position of faces within the room, the number of cameras providing usable poses etc. determines the robustness of interpretation. The hard part of this problem is not how to track a face, or how to recognize a face but how to reconcile the capabilities that the system has, the requirements that the system has on face identification, the resources available and required, and the fusion of data between multiple cameras. A set of good implementations of basic algorithms is a basic requirement but robust performance is achieved by addressing the broader set of issues.

The approach to designing software systems as described above is novel and very different from traditional software development approaches. The approach offers some significant advantages over traditional methods, most notably in robustness. This new approach is particularly suitable for systems that involve interaction with an external environment that we are unable to model accurately. In the past that category of computation has been very limited. The future of embedded computation in systems that interact with the real world may well represent the majority in the near future. It is clear that this new paradigm for software development and other of a similar nature demand much more attention as we move forward.

[1] Paul Robertson, A Self-adaptive architecture for image understanding, DPhil thesis, University of Oxford, 2001

[2] Michael Coen, Brenton Phillips, Nimrod Warshawsky, Luke Weisman, Stephen Peters, and Peter Finin. Meeting the Computational Needs of Intelligent Environments: The Metaglue System. In Proceedings of MANSE'99. Dublin, Ireland. 1999