Behavior-Based Control: Main Properties and Implications

Maja J Mataric
MIT Artificial Intelligence Laboratory
545 Technology Square #721
Cambridge, MA 02139
maja@ai.mit.edu

Abstract

In addition to its computational and sensory hardware, the architecture of an autonomous robot determines how that robot will behave by imposing key constraints on the way it can sense, reason, and act in order to achieve its task. In this paper we define the basic types of control architectures: purely reactive, behavior-based, and planner-based. We clarify the fundamental differences between the first two approaches, which are commonly confused in the literature. Next we describe the properties of behavior-based systems in detail and compare them to the two alternative approaches. Finally, we describe the application of the behavior-based paradigm in two very different domains.

1 Introduction

An architecture provides a principled way of organizing a control system. However, in addition to providing structure, it imposes constraints on the way the control problem can be solved. In this paper we will explore the constraints and consequences of the behavior-based approach to control and compare it to its two alternatives: purely reactive strategies and planner-based architectures.

2 The Basic Types of Control

At one extreme of the control spectrum lie traditional *planner-based strategies* which employ a centralized world model for verifying sensory information and generating actions in the world (e.g. [**Giralt et al 84**], [**Chatila and Laumond 85**], [**Moravec and Cho 89**]). The information in the world model is used by the planner in order to produce the most appropriate sequence of actions, i.e. a plan.

At the other extreme lie *purely reactive approaches* which implement a control strategy as a collection of condition-action pairs (e.g. [**Brooks and Connell 87**], [**Agre and Chapman 87**], [**Connell 90**]). The systems consists of a collection of purely reactive rules, containing minimal internal state. These systems maintain no internal models, perform no search, and merely lookup and command the appropriate action for each set of sensor readings. They rely on a direct coupling between the sensors and actions, and a fast feedback loop through the world.

Recently, a *hybrid* architecture has gained popularity (e.g. [**Arkin 89**], [**Payton 91**], [**Connell 91**]). It offers a compromise between purely reactive and planner-based approaches by employing a reactive system for low level control, and a planner for higher level decision making. These approaches separate the control system into two, or more, communicating but basically independent parts. In most cases, the low level reactive process takes care of the immediate safety of the robot, while the higher level uses the planner to select action sequences. For the purposes of this paper, hybrid architectures fall within the category of planner-based approaches.

Behavior-based strategies, also referred to as reactive subsumption-style approaches, fall between the above two extremes. Unlike planning systems, the designer of a behavior-based system, rather than the system itself, performs the mapping between the conditions and actions. Although often confused in the literature, purely reactive and behavior-based strategies differ fundamentally. While behavior-based systems embody some of the properties of reactive systems, and may contain reactive component, their computation is not limited to lookup. These systems may use different forms of internal representations and perform computations on them in order to decide what effector action to take. Additionally, a common property of behavior-based systems is their distributed nature; they consist of a collection of parallel, concurrently executing behaviors devoid of a centralized arbiter or reasoner. The behaviors are more powerful than purely reactive rules.

3 Capability Tradeoffs

Although it is commonly believed that purely reactive systems are strictly less powerful than behavior-based and deliberative, planner-based systems, it is difficult to prove this analytically. Specifically, the proof depends entirely on the particular task, environment, and the robot's computational and sensory capabilities. For a class of situations in which the tasks is well defined, the environment sufficiently well known, and the robot equipped with sensors which allow it to perceive the necessary information, purely reactive solutions can be used to solve the control problem [**Schoppers 87**]. In all such cases, the designer must be able to predict and elaborate all possible relevant world states that can be sensed, and the appropriate corresponding actions relative to the goals of the task.

If the task involves achieving goals which are not defined at design-time, then a purely reactive strategy will not suffice, unless it is extremely (and thus prohibitively) complex. For instance, a robot which is required to navigate to various locations specified by the user at run-time cannot perform this task reactively unless its state-action policy includes states for each of the possible goal locations. Purely reactive approaches achieve great efficiency because of their minimal computation. However, since they have so little representational power, they lack run-time flexibility.

The reactive versus deliberative dilemma is captured by the tradeoff between the amount of built-in information and the amount of on-line computation. Depending on the task, it may be easier to hard-wire the action rules, or it may be simpler to maintain an internal world model rather than to encode it implicitly in the reactive rules.

While it is relatively simple to illustrate examples of tasks for which purely reactive strategies are not optimal, it is difficult to formalize a definite set of criteria for deciding for or against them, as the decision depends entirely on the task of the system. More interestingly, while the drawbacks of pure reactivity are obvious, the tradeoffs between behavior-based and planner-based systems is much more complex. In the next part of this paper we will describe the key properties of behavior-based systems and the way they relate to the alternatives.

4 Modularity

The organization and interaction methodology of behavior-based systems differs drastically from the classical hierarchical system approach. Their capabilities are not implemented with a typical set of distinct ¡modules, such as "navigation", "learning", "planning", etc. In place of a top-down decomposition, they progress bottom-up, starting with a collection of reactive rules implementing survival behaviors. New, more complex capabilities are introduced by adding task-specific modules which, when combined, result in the desired global behavior. The approach takes advantage of properties emerging from the interaction of the various system components, thus allowing for implicit rather than explicit integration of the different parts of the system. Incremental design, punctuated by testing and debugging, produces the desired side effects in a controlled fashion.

Although the control is distributed throughout the system, the resulting complexity is not unmanageable due to minimized inter-module dependence and communication. It is important to note that most of the interaction between the modules happens through the world rather than within the system. The exchange of knowledge within the system is mostly done implicitly, by observing, and relying on, the effects of the other components. Instead of depending on the correctness and consistency of information exchanged between modules, most information is obtained directly by the modules themselves, through sensing the world and the robot's current state and goals. Thus, the components of the system depend on each other implicitly. The coherence of the resulting external behavior is dependent on and maintained through the interaction with the world.

In contrast to classical modularity which favors the development of each of the modules independently, new modules for behavior-based systems must be designed while keeping the entire system in mind. Since there is no centralized controller, the control of the robot is distributed over the many different basic modules, acting in parallel, whose interaction results in meaningful system-level integration.

The task decomposition used in designing behavior-based systems may, for many tasks, be orthogonal to the intuitive software engineering style. In the next section we offer some heuristics which clarify the process.

5 Design Strategies

The decomposition and construction of a system begins with the identification of the basic set of reflexes which provide guaranteed safe "survival" in dynamic, unstructured environments. For example, in the case of a mobile robot, they guarantee collision-free navigation. The reflexes are a collection of simple rules which receive inputs from the sensors, and send commands directly to the effectors. The design proceeds bottom-up, but it also incorporates top-down constraints. Consequently, even the basic reflexes of the robot are geared toward the purposes of the rest of the system.

The following heuristics describe the process of using the task-specific constraints to generate the behaviors.

1) *Specify the desired behavior(s) in qualitative terms.*
This defines the behavior in observer space. The behaviors are the goals of the system.

2) *Specify the behavior in terms of the actions in the observer space.*
This is the process of breaking down the behavior into observable, disjoint actions, which will serve as subgoals in the process of reducing the difference to the goal. For instance, for collision avoidance the subgoal is moving away from the obstacle. For boundary following, the subgoals include moving towards the boundary, moving away from the boundary, and remaining aligned with the boundary.

3) *Specify the actions in terms of the robot's effectors.*
This is the process of selecting the action set, and the point at which the decision about the granularity of control is made. To maximize sensory awareness, make motions small and incremental rather than ballistic. The required precision of the task is the lower bound of the action granularity, i.e. the action set must be at least as fine-grained as the precision required to perform the task.

The strength of reactive as well as behavior-based control is based on probabilistic convergence, resulting from constant sensing. Consequently, rather than reducing a difference to a subgoal based on a single sensory input, the reduction is an incremental process, allowing for dynamic error correction. This minimizes the probability of the world changing while the robot is "busy."

6 Sensing and Action Selection

The choice of the architecture affects how the robot will perform its sensing operations and how the sensory information will be processed. In planner-based systems, all of the available sensory information is combined into a global world model. The problem of how to appropriately perform the combining is referred to as sensor fusion (e.g. [**Arkin 90**]). Methods have ranged from simple potential field computations (e.g. [**Khatib 86**], [**Brooks 86**]) to Kalman filters (e.g. [**Ayache and Faugeras 88**], [**Crowley 85**]). However, the more complex the global integration, the more likely it is to result in constraints on the sensing resources.

In contrast, behavior-based systems have no centralized controller, so there is no place in which all of the different sensory inputs are merged into a single model. However, various subsets of the sensors may be combined in task-specific behaviors for the purpose of generating the appropriate action. For example, arm position and hand contents together can determine if a robot should head home [**Connell 90**], or lateral sonar data and a compass reading can produce the direction to turn in [**Mataric 90**]. In general, a small number of perceptual filters are combined and coupled directly to an action. The hard problem lies neither at the sensor nor at the integration level, but at the level of arbitrating among potentially many actions and selecting one [**Maes 89**], [**Brooks 91**].

In the majority of behavior-based systems the solution is a built-in, fixed control hierarchy imposing a priority ordering on the behaviors. It ensures that only one of the behaviors can have control over the actuators, and that control is always appropriated to the behavior with the highest priority within the given context. In contrast to such hard-wired control methods, more flexible solutions have been suggested. Many of those are based on selecting an action by computing a multivariable function implicitly encoded in behavior activation levels. By spreading activation throughout the behavior networks and using carefully chosen thresholds for triggering actions, action selection can be tuned and even learned [**Maes 91**], [**Brooks 91b**]. Yet another set of approaches favors voting schemes for behavior arbitration (e.g. [**Payton et al 92**]). The key insights into the real problems and solutions will come only after many systems with rich sensor suites and large numbers of constituent behaviors have been implemented and tested.

7 Synthesizing "Correct" Behavior

The behavior space of a behavior-based system is defined by the different action sequences commanded by the various modules in the system. These action sequences emerge over time, in contrast to planner-based systems in which they are computed at run-time. The time required for deliberation at run-time is traded off for deliberation at design-time. Consequently, designing the modules of a behavior-based system requires anticipating the relevant action sequences which will reduce the difference to the appropriate goals. Since each reactive action only provides the first step toward the goal, the designer must make sure that it will be consistent with both the environment and with actions generated by other behaviors.

[**Rosenschein and Kaelbling 86**] and [**Kaelbling 88**] propose programming languages for generating provably correct behavior. However, automated behavior synthesis is still a distant goal. [**Lozano-Pérez, Mason, and Taylor 84**] describe a system for automated fine motion planning, a subset of the general problem, in itself of forbidding complexity.

Planner-based systems might appear more amenable to analysis and correctness proofs in that the interaction of their centralized, explicit world models and the reasoner seems more controlled. However, the existence of unpredictable conditions in the world make these systems as difficult to validate as behavior-based ones. Regardless of the architecture, the step which loops the system through the real world introduces uncertainty which makes detailed analysis very difficult.

8 Experiments

In discussing the key features of behavior-based systems, we have stressed the importance of properly distributing the task over the modules of the system, and effectively controlling the interactions among those modules. In this section we will describe some past and current work which has addressed those issues.

Our previous work has been on integrating a number of capabilities into a real-time control system. Specifically, our goal was to construct a physical system capable of performing both higher-level reasoning and robust real-time action without introducing a hybrid solution consisting of a reactive low-level, and a deliberative high-level.

We demonstrated a fully integrated control system on a mobile robot Toto which performed navigation, landmark detection, map learning, and planning, without explicit integration [**Mataric 90, 92b**]. Inspired by biology and the subsumption architecture, our approach proceeded bottom-up, building increasingly more sophisticated capabilities on top of simpler ones. For instance, the robot's basic behavior was that of moving safely through its environment, avoiding obstacles, and following boundaries of objects. Landmark detection was added by utilizing the combination of way the robot was (already) moving and the sensory information it was perceiving (i.e. a combination of proprioception and external perception).

Our system maximized the use of direct information from the world, and minimizes communication within the system. While levels of control depended on each other's performance implicitly, they exchanged very little direct information. The control is fully distributed over all of the components, and does not use any manipulable data structures or symbolic representations. Instead, the representation of the entire system is homogeneous; it consists of simple behaviors which encode both the control strategy and the knowledge of the system [**Mataric 91**]. Integration is accomplished through the components' interaction with the environment. The homogeneous, behavior-based nature of the representation does not make a distinction between the reactive and deliberating parts of the system.

Our current work is concerned with testing the limits of distributedness by employing the behavior-based strategy for controlling a collection of twenty mobile robots. In this research we are removing the abstraction barrier between individual agent control and the control of a collective. Specifically, we are exploring strategies for controlling groups of mobile robots using local behavior-based strategies. Thus, the collection of robots is akin to a collection of behaviors controlling a single robot, and the issues of behavior interactions (both within individual robots and among them), interference, and emergent behaviors are the focus of our experiments.

[**Mataric 92a**] describes a series of experiments for utilizing local strategies in order to minimize the complexity of controlling a group of robots in a homing task. In our work on collective robotics we are defining a taxonomy of tasks and the appropriate control strategies and behavior primitives for achieving them. Specifically, we are classifying a class of local behavior strategies which allow for complex and robust global behaviors to result without centralized top-down control. These strategies, all based on local sensory information, include behavior imitation, simple avoidance protocols, and the use of perceived population density.

Traditional planner-based architectures for control do not appear well suited for problems involving multiple robots. The global state space of a system containing multiple robots grows exponentially, since all of the robots may interact with each other, and their interactions may have extended temporal consequences. Additionally, the uncertainty in perceiving the state becomes even more unreasonable with the increased complexity of the environment. Consequently, classical planning is intractable in this domain. As an alternative, we are considering various local strategies which lend themselves well to the distributed nature of the behavior-based control paradigm.

9 Conclusions

The control architecture constrains the way an autonomous robot senses, reasons, and acts, thus affecting its task performance. We have described the properties of the behavior-based approach to control, and its influence on those key components of robot control.

Unlike purely reactive systems, whose limitations can be identified, behavior-based systems in principle share the representational power and flexibility of planner-based system. Consequently, it is difficult to compare these approaches theoretically. Even when given a well defined task, the resulting solutions may appear equally effective. The best way to evaluate the performance of the different architectures is to test them on a variety of tasks on a variety of physical implementations.

With robotics in its infancy, it is premature to limit ourselves to a particular architecture choice. Much experimenting and analysis is necessary to produce a body of algorithms for different tasks, and apply those to the different types of architectures in order to begin to gain an understanding of the issues.

Acknowledgements

I am grateful to Rodney Brooks for useful discussions and comments on an earlier version of this paper. The research reported here was done at the MIT Artificial Intelligence Laboratory. Support for this research was provided in part by the Mazda Corporation, in part by the University Research Initiative under Office of Naval Research contract N00014–86–K–0685, and in part by the Advanced Research Projects Agency under Office of Naval Research contract N00014–85–K–0124.

References

[**Agre and Chapman 87**] "Pengi: An implementation of a theory of activity", P. Agre and D. Chapman, *Proceeding of Sixth National Conference on Artificial Intelligence, AAAI-87*, Seattle, WA, 1987, 268-272.

[**Arkin 90**] "Integrating Behavioral, Perceptual and World Knowledge in Reactive Navigation", Ronald C. Arkin, *in Designing Autonomous Agents, P. Maes, ed.*, 1990, 105–122.

[**Arkin 89**] "Towards the Unification of Navigational Planning and Reactive Control", Ronald C. Arkin, *AAAI Spring Symposium on Robot Navigation Working Notes*, March 1989, 1-5.

[**Ayache and Faugeras 88**] "Building, Registrating, and Fusing Noisy Visual Maps", N. Ayache and O. Faugeras, *International Journal of Robotics Research*, Vol. 7, No. 6, December 1988, 45-65.

[**Brooks 91a**] "Intelligence Without Reason", Rodney A. Brooks, *Proceedings, IJCAI-91*, August 1991, 569-595.

[**Brooks 91b**] "Integrated Systems Based on Behaviors", Rodney A. Brooks, *special issue of SIGART on Integrated Intelligent Systems*, July, 1991.

[**Brooks and Connell 87**] "Asynchronous Distributed Control System for a Mobile Robot", Rodney A. Brooks and Jonathan H. Connell, *SPIE's Cambridge Symposium on Optical and Opto-Electronic Engineering Proceedings*, Vol. 727, October, 1986, 77-84.

[**Brooks 86**] "A Robust Layered Control System for a Mobile Robot", Rodney A. Brooks, *IEEE Journal of Robotics and Automation, RA-2*, April, 1986, 14–23.

[**Chatila and Laumond 85**] "Position Referencing and Consistent World Modeling for Mobile Robots", R. Chatila and J. Laumond, *IEEE International Conference on Robotics and Automation*, March, 1985, 138-145.

[**Connell 91**] "SSS: A Hybrid Architecture Applied to Robot Navigation", Jonathan H. Connell, *IBM Research Report*, 1991.

[**Connell 90**] "A Colony Architecture for an Artificial Creature", Jonathan H. Connell, *MIT A.I. Lab Technical Report* 1151, June, 1990.

[**Crowley 85**] "Dynamic World Modeling for an Intelligent Mobile Robot Using a Rotating Ultra-Sonic Ranging Device", J. L. Crowley, *Proceedings of the*

1985 IEEE International Conference on Robotics and Automation, St. Louis, MO, March 1985.

[Giralt et al 84] "An Integrated Navigation and Motion Control System for Autonomous Multisensory Mobile Robots", G. Giralt, R. Chatila and M. Vaisset, *First International Symposium on Robotics Research*, M. Brady and R. Paul (eds.), MIT Press, Cambridge, MA, 1984, 191-214.

[Kaelbling 88] "Goals as Parallel Program Specifications", Leslie P. Kaelbling, *Proceedings, AAAI-88*, Saint Paul, MN, August 21-26, 60-65.

[Khatib 86] "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots", O. Khatib, *International Journal of Robotics Research*, Vol. 5 No. 1, 1986, 90-98.

[Lozano-Pérez, Mason, and Taylor 84] "Automatic Synthesis of Fine-Motion Strategies for Robots", Tomás Lozano-Pérez, Matthew W. Mason, and Russell H. Taylor, *International Journal of Robotics Research*, Vol. 3(1):3-24.

[Maes 91] "Learning Behavior Networks from Experience", P. Maes, *Proceedings of the first European Artificial Life Conference*, Paris, Dec. 1991.

[Maes 89] "The Dynamics of Action Selection", Pattie Maes, *IJCAI-89*, Detroit, MI, 991–997.

[Mataric 92a] "Minimizing Complexity in Controlling a Mobile Robot Population", Maja J Mataric, *Proceedings, IEEE International Conference on Robotics and Automation*, Nice, France, May 1992.

[Mataric 92b] "Integration of Representation Into Goal-Driven Behavior-Based Robots", Maja J Mataric, *IEEE Transactions on Robotics and Automation*, to appear, 1992.

[Mataric 91] "Behavior Synergy Without Explicit Integration", Maja J Mataric, *Proceedings of the AAAI Spring Symposium on Integrated Intelligent Architectures, Stanford University*, Sigart Bulletin, Vol. 2, No. 4, August 1991, 130-133.

[Mataric 90] "A Distributed Model for Mobile Robot Environment–Learning and Navigation", Maja J Mataric, *MIT A.I. Lab Technical Report* 1228, May, 1990.

[Moravec and Cho 89] "A Bayesian Method for Certainty Grids", Hans Moravec and D. Cho, *Proceedings, AAAI Spring Symposium on Robot Navigation*, March 1989, 57-60.

[Payton et al 92] "Do Whatever Works: A robust approach to fault-tolerant autonomous control", D. Payton, D. Keirsey, D. Kimble, J. Krozel, K. Rosenblatt, *in preparation*, 1992.

[Payton 91] "Internalized Plans: a representation for action resources", D. Payton, *Designing Autonomous Agents, Maes, Ed.*, MIT Press, 1991, 89-103.

[Rosenschein and Kaelbling 86] "The synthesis of digital machines with provable epistemic properties", S. Rosenschein and L. Kaelbling, *Proceedings of the Conference on Theoretical Aspects of Reasoning About Knowledge*, Joseph Halpern, ed. Morgan Kaufmann, Los Altos, CA, 1986, 83-98.

[Schoppers 87] "Universal Plans for Reactive Robots in Unpredicatable Domains", Marcel J. Schoppers, *Proceedings, IJCAI-87*, Menlo Park, CA, 1987, 1039-1046.