A Proposed Framework for Characterization of Robotic Systems

by
Jane Mulligan

Technical Report 92-29
November 1992

Department of Computer Science
University of British Columbia
Rm 333 - 6356 Agricultural Road
Vancouver, B.C.
CANADA V6T 1Z2

A Proposed Framework for Characterization of Robotic Systems

Jane Mulligan
Department of Computer Science
University of British Columbia
Vancouver, B.C.
V6T 1Z2
e-mail: mulligan@cs.ubc.ca

November 2, 1992

Abstract

The plethora of approaches to planning and action in robotic systems calls for a unified framework onto which we can map various systems and compare their structure and performance. We propose such a thinking framework which allows us to examine these systems in a uniform way and discuss their adequacy for robotic problems. The inclusion of an environment specification in problem specifications is proposed as a means of clarifying a robot's abilities and creating a notion of context. Robotic systems are described as a set of <*information-source; computation*> strategies combined with a set of actuators.

1 Motivation

The need for a general framework for discussing a broad class of robotic systems may seem obscure to many readers. Why write down a scheme which seems obvious? One reason is that when we survey the field of robotics and planning we see a huge variety of proposed and implemented systems, all making different claims about their structure and performance [2] [8] [9]. How can we compare such systems and their claims without at least a common language for describing them? For example how do we compare the navigational abilities of a mail delivery robot, which follows a tape on the floor, and a small, autonomous mobile robot which navigates to arbitrary positions via arbitrary paths?

One can only speculate why such a scheme for description and comparison of robotic systems has not been promoted in the past. Perhaps recent improvements in the performance and availability of both computing devices and actuators has allowed us to build more sophisticated integrated systems and hence made comparisons more critical. Perhaps as a natural course our methods have evolved to the point where we are ready, and indeed need, to discuss the overall performance of a system, rather than particular algorithms within it. There have been attempts recently to set up design criteria for intelligent robotic systems [1]. These may help us to build future systems but it is not clear that they help us explain and compare existing systems.

Defining what constitutes an "intelligent system" is at best subjective. For the purposes of our discussion we will refer to intelligent robotic systems in the very general sense of a system which autonomously achieves specific goals in a specific environment. One might argue that a more flexible system, which can achieve a greater number goals under a greater variety of environmental conditions, is more "intelligent" than say, a simple dedicated controller, but this sort of philosophical argument is not our purpose here.

What then are the requirements for such a framework? In the first place the framework must be general enough to include the multitude of devices and architectures which already exist in the robotics community. Our goal is to look at what is fundamental about robotic problems by using common terms to discuss all systems from the most simple to the most complex. After all, as we have seen in recent years, sometimes the simplest systems are the most effective [2].

Another quality we want to be able to capture in our description is that of parallelism, or at least modularity. Distributed systems seem inevitable in the computing world, but more specifically for robotic problems we can think of action as a mechanical computation distributed over compute devices, sensors and actuators. There seems to be an inherently distributed quality to robotic systems.

Finally we conjecture that context is a crucial aspect of tractable reasoning and action. We intend to define our framework in such a way that the context of a problem or computation is made explicit.

1.1 Classes of Robotic Systems

We will lump the vast number of robotic systems that exist into three broad categories: traditional planners, reactive or behavioral systems and robotics and control applications. For our purposes these are the disparate sorts of systems we want to discuss and compare, and hence those which our framework should accommodate.

In the realm of intelligent robotics there currently exist two general approaches to generating action sequences: "traditional planning" techniques and reactive or behavioral systems. Most current *planning* (top down) systems stop short of connecting their *primitives* to devices acting in the world. When they do so they often suffer from the effects of uncertainty in the real world, not accounted for by their models [2]. On the other hand, those who work on *behavioral* (bottom up) approaches to activity have often avoided complex tasks, which necessarily entail "reasoning", in favour of enumerating all possible eventualities and predefining responses. The result is that any relatively complex state space is prohibitive in terms of storage and precomputation or design time [12] [4].

In the robotics and control community various systems have been developed in a more conservative, incremental way than many of the AI systems mentioned above. Goals such as robot design, kinematics, dynamics and more recently obstacle avoidance, grasping, manipulation and assembly planning have been built up in turn to produce more and more sophisticated systems. Generally speaking the robotics community has done a better job at modelling and overcoming the uncertainty inherent in real action systems, than the AI community. Still there is no single task planning methodology which is accepted here, and approaches abound for more restricted goals such as obstacle avoidance and grasping.

2 A Framework

2.1 Problems and Tasks

If we want to characterize autonomous robotics it is important to clarify somewhat what exactly the issues are. One persistent difficulty in the field has been that although many systems have been built we have no way to compare their performance or methods. A possible solution is to compare systems by their sets of *abilities*. In other words compare what tasks they can achieve given their particular assumptions about the world.

For the purposes of this discussion, we will assume that tasks have an inherent precision and complexity. Generally a task requires certain physical differences between the goal state and the world state to be resolved to a level within some specified tolerance. Specified or constrained techniques or operations for resolution of these differences may also form part of the task requirements, and the whole process is limited by the available world effectors and environmental factors. We will refer to this as a ***robotic problem*** and describe it as a ***<task;environment>*** pair.

The inclusion of the environment in the problem specification is an important extension to the usual discussion of robot tasks. There are two important reasons for this inclusion: first the difficulty and nature of a task are greatly affected by the constraints imposed by the environment in which it is to be performed. For example navigation in the open countryside is vastly different from navigation within a robotics lab. Second, by associating task and environment we generate a notion of *context*.

2.2 Information

In recent years there has been hot dispute between proponents of traditional and reactive systems, often in regard to where and how models or sensor observations of the world are used. In fact planning and reactive control are both computations on simplified models and they both work as long as the model and computation are adequate to produce or approximate the desired results.

Let us propose sensing and modelling as a continuum where if we had a perfect suite of sensors, **or** a perfect model of the world, we would be able to plan and act successfully in that world. Naturally in most instances robotic systems fall somewhere along this continuum where imperfect learned or programmed models are compensated for by observations from some imperfect set of sensors, or vice versa. In other words sensing and modelling are in some sense complementary, so that the computational time and space required for each can be traded off. Thus, for a particular task requiring information about particular aspects of the environment, those aspects may be sensed, or predicted via a model, or in the usual case, some combination of the two. For the purposes of this document we will refer to combined sensing and modelling resources as an ***information source***. This promotes the view, similar to the control and behaviorist communities, that instead of inserting sensor operations, we should make them integral to our computations. In other words, representation cannot usefully be separated from computation.

In reality even the simplest controller uses a model of the system it controls, for example the threshold value determining whether an automatic street lamp goes on or off. Models are in a sense **more** powerful than sensor information because they allow prediction or 'future sensing'. Further a *pure sensor* generates only raw measurements from the world, without the interpretation possible from models or memory retaining a *history* of events. The above tradeoff does not reflect this aspect of information extending into the past or future. The issue of prediction is particularly important in more complicated goal directed sequences of robot actions where **prediction of sensory consequences** becomes essential in order to recognize progress, success or failure for a particular task. Prediction also allows the agent to select among courses of action based on aspects of their outcome.

We conjecture that there is information inherent for a robotic problem, in other words to successfully solve a robotic problem certain information is required to specific precision in order to adequately compute the result. Which information sources provide this fundamental information is a separate design issue. Mackworth [10] discusses adequacy criteria for visual knowledge representation, but seeks more generality than our goal of having information sources dedicated to particular computations. Some issues worth noting however are soundness and completeness, with respect to the task in our case, and efficiency and acquisition issues.

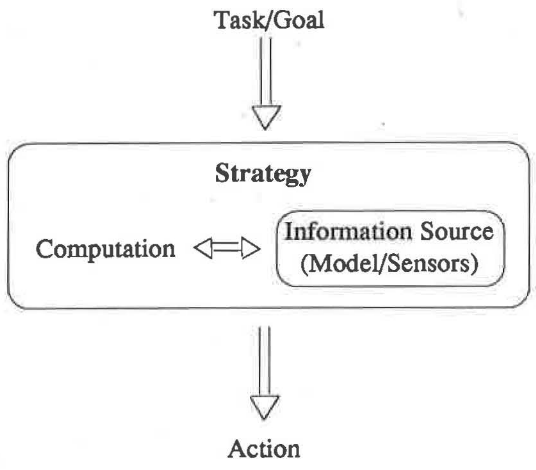

Figure 1: Strategy module

2.3 Strategies

Knowledge representation, despite the fact that it is often referred to as a separate discipline, is inherently tied to the types of computations performed by the system of which it is a part. Thus the nature of its information sources determines the types of computations or reasoning an agent can perform and hence the sequences of actions it generates. For the most part only a fraction of the information present in the environment will be sensed or modelled, for example our mail delivery robot may only sense a tape on the floor, but since this model is sufficient to the robot's navigation task, it can be the entire "world model".

The question then becomes, can we classify information sources and computations according to the types of tasks they can achieve? Such a classification might clarify the capabilities of particular robotic systems and indicate how such capabilities could be extended to match some desired task. We will call these ***<information-source, computation>*** pairs ***strategies***. Grouping information and computation in this way potentially gives us a notion of a "procedural primitive" which helps us address the "how" of describing robot activity, as opposed to state representations [7, p. 451].

2.4 Robotic Systems

A further issue in the representation and control of action are the actuators and other physical elements available to the system. Obviously sophisticated information and computation can still only generate the actions of which the actuators are capable and hence all tasks will be fundamentally limited by the physical properties of these devices (speed, dexterity, accuracy).

A set of strategies within a system addressing various aspects of a problem or set of problems must all share real system resources such as sensors and actuators. We shall refer to the pair of sets ***<strategies,actuators>*** as a ***robotic system***. This separation is useful since it clarifies the need for infrastructure and resource allocation among strategies at the level of robotic systems. It also allows us to discuss systems such as reasoners which do not use actuators (null actuators), within the same framework.

It is always important to keep in mind the **abilities of the robot**, for example in a simple robotic hand-eye system we must keep in mind that our system is disabled by human and animal standards in having one eye, one arm, one finger and a thumb, and doubtless a poorly organized brain. Our framework allows us to speak about a robotic system's abilities in terms of classes of environments in which it can perform particular tasks, and classes of tasks it can perform in particular environments. On this basis we will seek to compare such systems.

3 Discussion

3.1 Uncertainty and Action

One of the key criticisms of traditional planning systems is their adoption of the *Omniscient Fortune Teller Assumption* [11], basically the assumption that every event in the world can be predicted and accounted for. In the real world robotic systems spend most of their time simply trying to overcome deleterious of **uncertainty**, both regarding their own knowledge and action and the behavior of the environment. In fact in many low level task planning systems there are two aspects to action primitives - those motions necessary to achieve the task given perfect information, robot and computation, and those computations necessary to reduce uncertainty in these aspects of the system to within task tolerances.

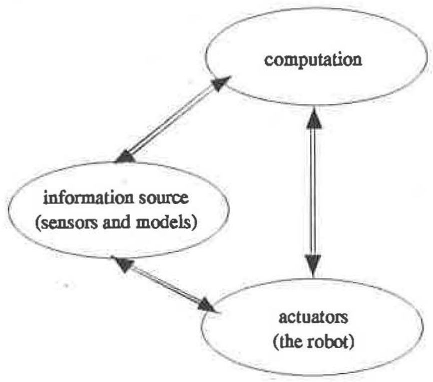

Figure 2: The three sources of uncertainty which most robotic systems use most of their resources to overcome: the robot (actuators), the environmental information and the algorithms or computations applied.

Naturally these uncertainty issues have not gone unnoticed in the robotics community[5][2][3, p. 524]. For our discussion we will assume three potential error sources:

1. uncertainty in actuation
2. uncertainty in information
3. uncertainty in computation

In other words: What I do may be inaccurate, What I know may be inaccurate and How I do it may be inaccurate. In our framework the uncertainty in information and computation arise from the set of strategies and the infrastructure which connects them. As illustrated in figure 2 these three systems compensate for and complement each other. For example Erdmann's [5] randomized search strategy (computation) compensates for inadequacies in sensors and actuators in the peg-in-hole problem. Similarly, a robot assembling an object in a highly engineered environment has sufficient actuator accuracy and virtually complete knowledge of the world and hence can use almost no computation - ie. replay a recorded assembly sequence.

These three basic components, given a $< task; environment >$ pair, are the basic sources of error or inaccuracy in the the performance of the *robotic system*. The proficiency of each can be traded off against the others to achieve some level of system adequacy or ability, which can then be compared to the task adequacy requirements. Thus, to include a task in the abilities of a robotic system :

$$\textit{system adequacy} \geq \textit{task adequacy}$$

3.1.1 Learning How

A robotic system can only deal with a genuinely *unknown/unpredictable* environment by exploring and *learning* about it [6]. With this in mind, the robots Brooks presents as operating in unstructured and unpredictable environments actually have very strong assumptions about the presence of walls etc because these features of the environment are highly predictable [2]. To be successful, every agent must "learn" from observation of the environment, whether this takes the form of statistically setting simple thresholds, as with adaptive controllers, or some more sophisticated mechanism.

Our proposed framework allows us to discuss learning by reversing the question we asked above. Instead of attempting to design a system which meets the adequacy requirements of some robotic problem, we ask, given a particular robotic system, what robotic problems can it learn to solve?

3.2 Describing Systems

How do the three classes of robotic systems we have discussed so far, traditional planning, reactive systems and various autonomous control applications, map to the framework described above? For traditional AI systems, a relatively high level activity such as planning is assumed to have more primitive layers beneath it [6], in our paradigm these might be control strategies, sensing strategies and so forth. If a planner were to interact directly with actuators however, it would be a single monolithic strategy, performing a single type of computation on a single (large) model information source, which is assumed to be perfect.

Reactive systems fall naturally onto our discussion framework since they are, by design, modular. In the simplest case each finite state machine or simple controller in the system is a strategy using a set of encoded assumptions plus some sensors as its information source.

Robot applications are difficult to describe in general terms because they take so many forms, although individually they can be mapped to our framework. Typically modern robotic systems have standard inverse kinematic and dynamic systems which could be considered strategies, using perhaps a geometric or dynamic robot arm model and joint angle encoders as their information sources. Other aspects of these systems, such as obstacle avoidance, navigation, peg in hole assembly etc, could be considered other strategies within the system, each using some form of model and some subset of the available sensors.

3.2.1 Comparison

Let us return to our example of the mail robot and the autonomous mobile robot (augmented toy truck) from section 1. We will assume the mail robot has a single strategy with a sensor which locates the magnetic tape and a controller which then computes the forward motion of the cart. For the truck we will assume an external camera determines its position and internal computations allow it to determine the desired direction of travel based on this current location. Internal controllers, along with motor speed and wheel angle sensors are required to generate the desired motion. Thus the truck has two strategies one (navigation) computing where to go based on the external camera plus some model, and one (control) computing how to go there based on internal sensors and models.

If our problem is to navigate along the tape path of the mail robot in a lighted lab, both robots are perfectly capable of performing the task. If we now change the environmental specification so that the lab is in darkness, the camera can no longer be used for navigation by the truck, and the truck system fails. The mail robot can still sense its tape in the dark, and so will succeed where the truck failed. Of course if the goal is to navigate, in the lighted lab, to any location off the tape the truck will succeed and the mail cart will fail. If the lights are off, both systems fail. For this simplified example then we have outlined the ability sets of the two systems. They overlap only in the case where they follow the tape in the lighted lab, but interestingly they each possess abilities that the other lacks despite the fact that the truck may seem more sophisticated than the mail cart.

3.3 Robotic Problems and Systems

Naturally if we want to examine the adequacy requirements of robotic problems versus robotic systems we will have to look at how we can describe tasks and environments and thus clarify their requirements. We must distinguish problem specification and system specification. Are problem specifications necessarily system dependent in the sense that we must be *given* particular physical devices such as sensors and actuators before we can describe the physical transformations we are interested in? No doubt the engineers designing manufacturing systems, and using various sequences of devices to automate tasks designed for humans, can verify that there are many ways to achieve the same physical goals. For problem specifications then we have to find some way of describing tasks without presuming some specific physical devices.

For designing a robotic system for a particular problem, the question becomes what sensors and actuators, in combination with which models and computations, generate the desired action? We have proposed strategies as <*information-source; computation*> pairs which work together to compute some aspect of a task, but as yet we have no insight into how these aspects are determined.

Typically when designing algorithms to solve problems the computing community assumes a serial computing machine, or more recently a limited variety of MIMD and SIMD parallel computers [2]. With robotic tasks using sensors and actuators we have barely begun to evaluate how to distribute essentially probabilistic mechanical computations over the bewildering range of physical devices available.

3.3.1 Intelligence is Infrastructure

Once we have various primitives and build them into strategies, what then? Some form of infrastructure is required to allow the robotic system to work together as a coordinated whole.

The structure of the robotic systems we have proposed here is designed as a distributed set of strategies calling on a limited set of hardware (computer, actuator, sensor) resources. We have paid very little attention to *how* various parts of a whole intelligent system interact. Ad-hoc approaches abound but their limitations only serve to strengthen the argument that not only *what* but **how** we put together components of our system, unquestionably affects their performance. Obviously no module is an island and we are obliged to consider how best to combine these aspects of intelligent activity.

There seems to be a general assumption in AI that there are low level action/reflex layers and high level reasoning/planning layers. The problem is that these layers have rarely been implemented in a single unified system. Planning systems stop at the top and reactive systems stop at the bottom. What is needed is an organized, coherent infrastructure which is not only comprehensible, but will scale up.

Developing a loose characterization of the components of a robotic system does not answer the many questions that arise regarding how strategies are designed and composed into an overall system or how, when and what they communicate with each other. The framework we present is only a rough first stab at discussing these issues.

4 Conclusions?

In the preceding notes we have proposed the need for a uniform description for robotic systems in order to clarify and compare the performance of such systems. We have also proposed the need to explicitly specify the environment along with the task for a robotic problem, in order clarify its context.

The framework put forward combines sensor and modelling resources into a unit called an information source. These units are then combined in strategies, or <*information-source; computation*> units with the claim that representation and computation are interdependent. Finally a robotic system is composed of a set of strategies combined with a set of actuators.

We present three basic sources of uncertainty in robotic systems: information, actuation and computation. The uncertainty arising from these sources defines the competence level of the system. If we then examine certain tasks to be performed in environments of various complexities, we can evaluate the adequacy of the robotic system for a particular robotic problem or $< task; environment >$ pair. The issues of how to precisely describe robotic systems and problems and enumerate their adequacy criteria are under investigation.

Acknowledgement: This research was funded by the IRIS Centres of Excellence program, under the supervision of Alan Mackworth in the Laboratory for

Computational Intelligence, UBC.

References

[1] James S. Albus. A Theory of Intelligent Systems. In *Fifth IEEE International Symposium on Intelligent Control*, volume 2, pages 866–875, Philadelphia, Penn., 1990.

[2] Rodney A. Brooks. Intelligence without reason. Technical Report AI Memo 1293, MIT AI Lab, April 1991.

[3] E. Charniak and D. McDermott. *Introduction to Artificial Intelligence.* Addison-Wesley Publishing Company, Reading, Mass., 1985.

[4] Jonathan H. Connell. A Behavior-based Arm Controller. Technical Report AI Memo 1025, MIT Artificial Intelligence Laboratory, Cambridge,Mass., June 1988.

[5] Michael A. Erdmann. *On Probabilistic Strategies for Robot Tasks.* PhD thesis, Massachusetts Institute of Technology, Cambridge, Mass, August 1989.

[6] K.S. Fu. Learning Control Systems And Intelligent Control Systems: An Intersection of Artificial Intelligence and Automatic Control. *IEEE Transactions on Automatic Control*, pages 70–72, February 1971.

[7] K.S. Fu, R.C. Gonzalez, and C.S.G. Lee. *Robotics: Control, Sensing, Vision and Intelligence.* McGraw-Hill Book Company, New York, N.Y., 1987.

[8] Michael P. Georgeff. Planning. In *Annual Review of Computer Science*, volume 2, pages 359–400. Annual Reviews Inc., 1987.

[9] Jean-Claude Latombe. *Robot Motion Planning.* Kluwer Academic Publishers, Boston, Mass, 1991.

[10] A.K. Mackworth. Adequacy Criteria for Visual Knowledge Representation. Technical Report 87-4, Dept. of Computer Science, U.B.C., Vancouver, B.C. Canada, January 1987.

[11] David L. Poole, Alan K. Mackworth, and Randolph G. Goebel. Computational intelligence: A logical approach. Copies generated by authors, University of British Columbia, Vancouver, B.C., October 1991.

[12] M.J. Schoppers. Universal Plans for Reactive Robots in Unpredictable Environments. In *IJCAI'87 Proceedings of the Tenth International Conference on Artificial Intelligence*, volume II, pages 1039–1046, 1987.