THE FEATURES OF BEHAVIOURAL MODULES IN ROBOTIC ASSEMBLY

The purpose of this paper is to clarify what kind of thing a "behavioural module" or "behaviour" is, and to compile a list of the kind of features a behaviour may be supposed to possess. Since this is a research topic, this is in part a speculative exercise.

Introduction

The notion of behaviours as an important architectural feature of robots has grown out of one particular research approach in artificial intelligence. Most AI researchers would agree with Pat Hayes [Hayes, 1987]:

> The issue is whether one feels that the best approach to understanding human cognition is to approximate it by looking at parts of it, as in 'classical AI', or by looking at the behaviour of a flatworm or a sea slug. Personally, I put my money on the former.

This 'classical approach' has given us expert systems, undisputably the first major commercial fruit of AI research. What worries those who espouse the opposite (sea slug) view is whether these components of cognition can be bolted together into a whole mind. If we were simply talking about a collection of software components, then this would be a legitimate worry simply from the systems integration point of view, i.e., have the processes of decomposition and abstraction which were followed in identifying these components of cognition been such that they can be expected to be synthesisable into a whole?

But we are talking about more than just a collection of software; we are talking about some kind of artificial mind which has "intensionality" [Searle 1980], "grounded symbols" [Harnad 1987], or, as I like to call it, "inherent semantics". The implications of this are still a matter of considerable dispute, but few would quarrel with the general assertion that there are difficult and fundamental problems in this area, which may well turn out to have definite implications for implementors.

The 'sea slug' approach is expressed nicely by Moravec, [Moravec 1984], who says:

> I argue that the most fruitful direction for this track [the development of artificial intelligence] is along the oxtrail forged by natural evolution before stumbling on us. ... It is my view that developing a responsive mobile entity is the surest way to approach the problem of general intelligence in machines.
>
> The argument hinges on the observation made earlier that instinctive skills are much better developed [and took a lot longer to develop] in humans than high level thinking, and are thus the difficult part of the human emulation problem.

These "instinctive skills" are seen as a subcognitive substrate which is the necessary supporting ground for the higher level symbolic functions we usually identify with cognition. It is perhaps a question of degree. The 'classical approach' supposes that these lower subcognitive functions, while perhaps tedious in detail, are nevertheless purely a question of engineering, of implementation detail, something one can bolt on at the end; whereas the 'sea slug' approach supposes that these lower subcognitive functions will certainly impose serious restrictions on the kinds of higher level symbolic functions they will support, and may even turn out to be far more difficult, both theoretically, and in terms of practical implementation, than the higher level functions. Harnad argues this position from a philosophical point of view [Harnad 1987b]:

> I think a lot of what we consider cognition will be going on in the nonsymbolic iconic and categorical systems (discrimination, categorization, sensory learning and generalization) and that symbol manipulation will be constrained in ways that don't leave it in any way analogous to the notion of an independent functional module, operating on its own terms (as in standard AI), but connected at some critical point with the nonsymbolic modules. When I spoke earlier of the "connections" of the atomic symbols I had in mind something much more complexly interdigitated and interdependent than can be captured by anything that remotely resembles ..... AI's pious hope that a pure "top-down" approach can expect to meet up with a bottom-up one somewhere in between.

If these are indeed serious problems, then they can be most easily studied in the simplest systems which have solved them. On the one hand we might be able to construct such systems (Moravec's "responsive mobile entities"), or we could study such simple creatures as insects - or indeed sea slugs.

It is the job of cognitive ethologists to find a well-articulated method of describing animal behaviour, with particular reference to goal-achieving and problem-solving behaviours. The problem of finding a coherent, structured, and not overly complex way of describing purposeful behaviour in an animal turns out (not surprisingly) to be related to the problem of finding a coherent, structured, and not overly complex way of implementing purposeful behaviour in a robot. Dennett suggests for these reasons that "intentional system theory", developed as a descriptive tool by cognitive ethology, is also of interest to AI [Dennett 1983], and advises AI researchers to look at this kind of insect research:

> Several years ago, in "Why not the Whole Iguana?" [Dennett 1978, BBS], I suggested that people in AI could make better progress by switching from the modelling of human microcompetences (playing chess, answering questions about baseball, writing nursery stories, etc.) to the whole competences of much simpler animals. At the time I suggested that it might be wise for people in AI just to INVENT imaginary simple creatures and solve the whole mind problem for them. I am now tempted to think that truth is apt to be both more fruitful, and, surprisingly, more tractable, than fiction. I suspect that if some of the bee and spider people were to join forces with some of the AI people, it would be a mutually enriching partnership.

Brooks's behaviour-based mobile robots

I think Brooks was the first roboticist to investigate this problem from the point of view of experimental implementation [Brooks 1986], though several writers with neurophysiological leanings have elaborated various theoretical schemes from a similar viewpoint [Albus 1981], [Powers 1973], [Henderson 1984], [Lyons 1985]. One thing to which all these authors have given attention is the importance of servo-like behaviours, in which action and its associated sensing are entwined within a module, which seems to higher levels to be simply a goal-seeking operator. These can be much more complex than the servomechanisms of control theory, since the arbitrarily complex computations of state-machines referring to large bodies of knowledge can be interposed between the input and output. For example, Albus defines something he calls a "generalised servo" which he claims would be capable of such complex tasks as recognising its mother [ibid].

Brooks adopts this basic kind of notion, calling it a "behaviour", but refuses to embed it in the classical hierarchical structure, which he criticises for its fragility and computational complexity. Instead he prefers to experiment with what he calls a "subsumption architecture", in which the behaviours are implemented on separate circuit cards with simple four-wire interconnections which are cross-coupled with inhibitory and hallucinatory links in a manner loosely suggested by neurons in biological systems. He explicitly suggests that AI has the necessary level of technology, and theoretical sophistication, for an achievable research goal to be the construction of whole animals with the level of competence of insects [ibid]. The informal and satirical style of his paper, in which he makes some fierce criticisms of most current AI, mean that this paper has never been published outside of MIT, and perhaps it never will be:

> I suspect much current AI work will in hindsight appear to have been chasing details of epicycles. Current models for AI are based on concepts of knowledge and belief and symbol processing systems. I claim that all such things are concepts invented by observers of intelligent systems to explain them. The intelligent systems themselves do not work like that.

Nevertheless, this paper has achieved considerable "underground" notoriety. It is probably true to say that most of those with existing large research investments in the 'classical approach' have adopted an attitude of scepticism, or at any rate a cautious "not proven", with respect to Brooks's assertions, but there is an enthusiastic minority who have been waiting for someone substantial to say this kind of thing for years. Note that Brooks's considerable reputation in robotics derives from the fact that he was one of the major architects of the 'classical approach' in assembly robotics. It was while collaborating with Lozano-Perez on a paper defining the next major phase of MIT assembly robotics research [Lozano-Perez and Brooks 1985] that he decided that the whole approach was irretrievably damned, and switched to experimenting with behaviour-based architectures in mobile robots.

The use of behaviours in assembly robots

In mobile robots a number of behaviours must be active at once, and a problem of immediate practical importance is how collaboration is achieved, and conflicts resolved, between these behaviours - Brooks proposes his "subsumption architecture". We have taken the same basic notion of a behaviour, and used it in a proposal for the architecture of assembly robots [Smithers and Malcolm 1987]. The world of the assembly robot is far more highly structured, and the task it has to perform (assembling something) is of much greater inherent complexity; but on the other hand, a great deal more about it is known in advance. For these reasons the problems of collaboration and conflict between behaviours are much reduced, and the problems of how to specify and construct appropriate individual behaviours for a certain kind of task become the difficult ones. Thus these two experimental arenas highlight different aspects of the problems of behaviour-based robot architectures.

Of course, one day, mobile robots will be clever creatures with dexterous manipulative capabilities, and then these two experimental arenas will become one; but today, when considering simple systems, mobile robots emphasize the problems of parallel behaviours and general reactive competence, whereas assembly robots emphasize the problems of the decomposition of the assembly task into behaviours which are largely executed serially.

For these reasons a behaviour in an MIT mobile robot may not look much like a behaviour in an Edinburgh assembly robot. This is a superficial difference caused by the difference in domain. The important underlying architectural functions of behaviours in both systems are the same. Some of these are pretty clear, some of them are reasonable suspicions, and some are open questions. This paper is an exploration of just what it is (and might be) that is special about these behaviours.

The architectural implications of assembly behaviours

The first important point about a behaviour is that it is (so I assert) a necessary fundamental component of an intelligent system. As I noted in the introduction to the report of my practical experiment with assembly behaviours [Malcolm 1987]:

> That is one of the important points about behaviours: they are not only modular units of functional capability; they are abstraction devices which enable the existence of a tractable representation of the world, and by means of their practical capability connect that representation to the world.
>
> This is quite a different view of the proper relationship between a representation and the thing represented. It is based upon the view that intelligence, which can only be manifested as the knowledge-based behaviour of an agent, is hosted upon a sub-cognitive substrate of capabilities, to use Hofstadter's terminology. Brooks would say, expressing the same idea, that it is epistemologically offensive to give a robot concepts beyond its real world behavioural capabilities.
>
> In a small and humble way, these behaviours can be likened to

> the sub-cognitive substrate which enables the world of information with which the planner deals. This Janus-faced concept of "enabling a world" covers both the idea of an abstraction device from the world to the representation which is enabled, and the connection of that abstracted representation to the world by the practical capability.

Since a behaviour will (most likely) be implemented as some kind of software algorithm, and should be modular, then it follows that a behaviour, as software, should be well-structured and modular in the usual computer science meanings of those terms. I shall pay no more attention to this requirement, since the ideas are already well understood, and their wisdom unlikely to be disputed.

The behaviour also has an effect upon things in the world. With respect to the purposes of the robot of which it is a component, this real physical effect should also be modular. This poses us a theoretical problem, because while we well understand, informally, what is required in an assembly task, the languages in which we communicate assembly tasks to one another are highly context dependent, and make use of such awkward (for a robot) human capabilities in interpreting them as common-sense and ingenuity. We do not have a formal language with which to specify the assembly task, and constructing one will not be easy. Consequently, the best we can say at the moment, with respect to this modularity of assembly function, is that it should seem to be nicely modular to an experienced human. It is unfortunate that this can't (at the moment) be specified more formally; but that should not divert us from its importance.

The effect upon the real world is mediated through the robot manipulator. This is a construction of considerable complexity, and this second requirement of behaviours - modular functionality with respect to the task - places some requirements upon robot design which are not met by current assembly robots.

The current state of assembly robots

It is an accident of history that the assembly robots we have today are programmed in a computer programming language (often like Pascal or BASIC) with robotic extras, where the robot is driven as an output peripheral by motion commands which specify the Cartesian position which the end-effector should go to, with default speeds, accelerations, and trajectories; and that any sensors it has (beyond the joint-sensors used in the motor servos) are connected as input peripherals. There is a great deal of compromise and assumption already built into this fait-accompli. It was a combination of the best we could do with the computers, mathematics, and engineering at our disposal, and things that looked obvious at the time. Before considering the requirements of the functional modularity of behaviours upon this, I want to consider in detail the effectiveness of current assembly robots for assembly work.

An essential and important component of the reliability of the robot is the fact that its joints are controlled by position servos. In software architectural terms these are goal-seeking operators which reliably achieve the commanded positions, despite perturbations due to friction, load, etc., and so permit the robot's motion to be specified as a sequence of positions in joint space.

Modern assembly robots usually permit the specification of positions in terms of the Cartesian position of the end-effector. This is achieved by a computational translation process via solution of the inverse kinematics of the robot. Since the accuracy of this is not guaranteed by means of a feedback loop in the way that joint positions are, then the accuracy with which the robot will achieve a Cartesian position will depend upon the accuracy of the kinematic model, the solution techniques, and the engineering of the robot so as to minimise its deviations from this model, e.g., by being stiff so as not to deflect much under load. Consequently current assembly robots are stiffer and heavier than they would need to be if the position of the end-effector were sensed directly, and controlled via a feedback loop. Current research projects are addressing various ways of removing this deficiency.

Experience shows that current industrial assembly robots, although costly, because of the engineering necessary to resist deviation from the internal kinematic model, can nevertheless achieve sufficient accuracy of end-effector location for assembly purposes. Direct closed-loop control of end-effector location will therefore be used primarily to improve cost-effectiveness.

There is also the question of force control, necessary for many types of assembly work, but restricted and awkward with current position-controlled robots. These extra competences in controlling the robot itself are definitely of great importance, and there are some types of assembly which will remain impractical for robots until these further refinements of robot control have been solved. These extra competences, however, while improving the general competence of the robot, and adding to the variety and complexity of possible behaviours, will not affect the fundamental requirements that behaviours pose upon the architecture of the robot system.

Controlling part motions in terms of robot motions

The purpose of the assembly task is to bring the parts together in the spatial relationship to one another defined by the final assembly. This process can be simply decomposed into operations of acquiring and fitting individual parts, and precedence relationships between these operations. In general the acquisition of a part is relatively simple compared to fitting it into the assembly. This fitting often requires the achievement of a number of spatial relationships between the part and the assemblage, and in such cases, some kind of strategy for achieving these is necessary [Koutsou 1985]. For example, in the case of placing a block carefully down upon a table, in the presence of uncertainty, a possible strategy would be to move the block down until a corner contacts the table, then to rotate the block until an edge contacted the table, and finally to rotate the block about the edge so as to bring the bottom of the block into full contact with the table surface. This may be achieved by the use of constrained motion, such as by dropping the object from a height above the table, and letting gravity do the work; or it may be achieved by pushing the object firmly down onto the table, making use of gripping surfaces which have been designed to slip under that load; or it may be achieved by a guarded motion until the first contact is made, followed by two guarded and force-servoed rotations. These are extreme cases, and intermediate versions may be devised.

Having illustrated the complexities lurking in even so simple a case as placing a brick upon a table, I now wish to consider the question of how such motions of the part are to be programmed; and how they are to be achieved by the robot. The simple obvious answer to the second question is that they should be achieved by means of the geometric motions in terms of which our robot is so conveniently programmable. Those who have attempted to contrive robot programs which attempt to achieve reliable motions of the parts of the assembly by means of a program written in terms of motions of the robot know what a curiously difficult and frustrating task this is; indeed, it is the major reason why assembly robots are not more used in assembly work. The consequent slowdown in the expansion of the robot market has already bankrupted a number of robot suppliers, and is worrying them all.

It is of course hardly surprising that this is a frustrating and difficult task. No matter how precisely the motions of the robot are controlled, it is not actually precise motions of the robot we wish to achieve, but precise motions of the parts. Consequently every little imprecision of form or location of a part is a hostage to fortune, which threatens to propagate and multiply through the process of the assembly until some intended fitting operation simply fails to occur, possibly involving damage.

Add sensors and more problems

The solution to the problem is obvious: we must add sensors to the robot, so as to be able to sense the positions of the part, and thus adjust the motions of the robot to cope with these little perturbations from the intended motions of the parts. The programming of the taking of sensor readings, and the consequent adjustments of the motions of the robot, create the requirement that the programming language of the robot should now have the full power and generality of a computer systems programming language. A robot so equipped with appropriate sensors and programming language is theoretically capable of controlling the motions of the parts and so achieving the intended assembly reliably, but practice proves that this is a tedious and highly skilled process if performed directly by a human programmer. Indeed, it is sufficiently expensive that it is a moot point whether a robot programmed in such a way is worth all the bother.

For these reasons Artificial Intelligence research bent itself to the task of compiling these tedious low level robot programs from higher level specifications. At Edinburgh we produced the language RAPT, which enables the motions of the robot to be specified in terms of the spatial relationships required to be achieved between the parts [Popplestone et al 1978]. Other laboratories addressed themselves to the problems of planning collision-free motions, of planning grasps, of planning constrained motion, and other useful components of the assembly planning task [Lozano-Perez 1982]. When all these components were brought together under the aegis of an automated (or human) assembly planner, the process of programming robots for assembly tasks would finally have been simplified to the point where it was a marketable solution. There was one particularly nasty problem which most people had been saving up for last, that of representing knowledge about the various kinds of uncertainty, and of being able to analyse the effects of the motion and sensing strategies intended to control these uncertainties.

One important kind of uncertainty is geometric. This interacts with the shapes of the parts, and other geometric uncertainties, in ways determined by the ways the parts are brought into contact with one another, which are in turn part of the strategies for achieving the part fitting relationships. Thus this business of geometric uncertainty, and how it propagates through the assembly, is a decoration of complexity on top of, and in terms of, the assembly process itself. In other words, it is inherently more complex than the assembly process itself, considered without it. And the assembly process, in the absence of uncertainty, is sufficiently complex that for many of the component sub-problems, we still only have partial solutions to simplified versions of the problem. As if this were not enough, this is only the problem of geometric uncertainty; how to deal with some of the other kinds of uncertainty is still an open research topic. Looked at in the light of this argument, it is hardly surprising that such experimental implementations of geometric uncertainty analysis as have so far been tried have proved unpleasantly hungry for computer power [Fleming 1985], [Durrant-Whyte 1987], [Erdmann 1985].

Using behaviours to control parts motion

This is where the notion of behaviours comes in. This suggests that the problems of controlling part motions should be handled by modular goal-seeking operations. The uncertainties of part form and location should be treated as the perturbations which these goal-seeking operations should control. Consequently the uncertainties of part motion are swallowed as securely by these behaviours as the uncertainties of robot motion are swallowed by the joint servos. To the higher levels we merely seem to have reliable operators for achieving the required kinds of part motions. The decomposition of these operators - the goals of the behaviours - into the various motions and sensings required to control the perturbations due to the uncertainties is something that can be relied upon to happen automatically, and is not the concern of higher levels. Thus at the level of programming the robot for the assembly task, we do not need to consider uncertainties at all, since the effect of the behaviours is to contrive that the imperfect and uncertain world seems to be ideal. Similarly, the need for the assembly programming level to consider the use of sensors is now restricted to those uses of sensors which are not for the purpose of controlling uncertainty. These other uses of sensors, such as in deciding what to do depending on what kind of parts arrive, are inherently less complex than using sensors to control uncertainty [Malcolm and Fothergill 1986].

Put like this, it seems so simple and obvious - add sensors to sense where the parts are, and add goal-seeking operations to ensure that the parts end up where they are meant to be. This is just a simple generalisation of the process already used with such success to control the motion of the robot with joint servos. The bad news is that this would be an extremely expensive step. We were able, when controlling the motions of the robot, to arrange the internal instrumentation of the robot's joints with purpose-designed sensors. This is not practical with the parts of the assembly we wish the robot to put together, so more general kinds of external sensing are required. Sensors capable of that degree of generality and precision are either very expensive, or don't even exist yet, and in many cases, e.g. vision, will require far more computational resources than running the robot.

The good news, however, is that it is possible to make use of prior knowledge about the assembly task to simplify the sensing task very considerably. It is even possible in some cases to take advantage of naturally occurring constraints on motion to provide the feedback to control errors in part motion without requiring sensing at all, such as by pushing or dropping actions. It is also the case that design-for-assembly can take advantage of the cheapest capabilities of the available robots and sensors. By these kinds of means the level of programming of the robot control system can be raised to the level of part control, i.e., the operations required by the assembly task, such as acquire-peg, and put-peg-in-hole.

Because we are still a long way from developing a formal language with which to specify assembly tasks, and for the same reasons, this stage of devising assembly behaviours is inevitably ad hoc and ingenious. Here we are dealing with the complexities of shape fitting strategies within the six dimensions of configuration space, and their relationships with friction, weight, centres of inertia, surface finish, elastic deformation, and other physical properties. Some of these are difficult to predict in advance in sufficient detail, and require a certain amount of practical experiment. This is the same world as the world of engineering design and development, which is also characterised by the use of ingenuity in problem solving. In other words, it is not a failure in the implementation of part motion control behaviours that they are unprincipled and ingenious compared to the simple generality of the robot motion control servos. This relatively unprincipled and ingenious nature is simply a natural consequence of the detailed complexity of the things now being controlled, the impossibility of direct instrumentation, and the consequent need to improvise strategies of knowledge-based indirect sensing and control. Nevertheless, by these kinds of means, which are already familiar to practical engineers, we can extend the level of control, and consequently the level of programming, to goal-seeking control of part motions, i.e., behaviours. "Behaviour" is the natural English term with which to refer to patterns of action and sensing described in terms of presumed goals.

These "behaviours" are the natural generalisation into the arena of part motions of the servos with which reliable control of the spatial motions of the robot have been achieved, and share the same two characteristics of improving reliability in the face of perturbations, and reducing the total computational complexity of getting the robot to perform the task. There is no magic to this business of the reduction in computational complexity. It is naturally much simpler to accomplish corrections in view of sensed errors, than to try to predict them in advance from a finer-grained and more complex model of the world - as is done in the 'classical approach' to the problem of uncertainty in assembly robot programming.

A point of particular importance is that part control via behavioural modules provides an interface to a task planning system in terms of ideal operations in an ideal world, and provides for reliable operation of plans contrived by such a planner. Unlike planners which need to concern themselves with details of uncertainty, and the use of sensing and motion strategies to control it, working prototypes of this kind of planner already exist.

This also makes it possible to use current types of CAD system for simulation of robot programs which can be expected to execute reliably. This solves the uncertainty problem currently plaguing CAD robot simulation, which naturally arises when the CAD systems are intended to simulate not the goal-seeking part control operations of behavioural modules, but the simple position controlled motions of the robot, with the part motions uncontrolled, and consequently subject to uncertainty.

What progress can be made with today's technology?

Observations of the computational requirements of planners, and the facilities offered by the best of today's robot controllers, suggest that part control by means of behavioural modules should be implementable in the more advanced type of robot control systems commercially available today, but that even the simpler forms of planning will have to remain off-line until an order or two of magnitude extra power is economically available.

In other words, the implementation of part control using behavioural modules is a technique that could be applied to today's assembly robot systems, using today's technology.

Experimental status

Using our behavioural modules approach, we have succeeded in constructing a complete working model of an assembly task planner, which generates assembly plans which are reliably executed by a real robot, despite the presence of deliberately large amounts of uncertainty. Shakey, STRIPS, and ABSTRIPS made plans and executed them reliably in the real world, and Shakey did use an informal behaviour-like approach to reliable execution of the plan operators, but Shakey's planner did not attempt plans of the complexity of assembly plans, with their shape-matching fitting problems, but restricted itself to simple relationships of abutment in its "assemblies". Our system is, to the best of our knowledge, the first successful implementation of a complete ASSEMBLY planner interfaced to a reliable ASSEMBLY execution system.

The major problems with such systems are well recognised to be the achievement of reliable execution in the real world, i.e., the uncertainty problem, and the detailed complexity involved in planning the assembly down to the level of geometric motions of the robot. Using the behavioural module approach we had little trouble with reliability of execution of the plans, and were able to use simple well-known planning techniques in a relatively simple planner. The planner is a backtracking hierarchical planner, which repeatedly refines the level of detail of the plans, often encountering failures in the lower levels, and using failure-directed backtracking to prune the search [Malcolm 1987].

Inherent Semantics

Searle has explained by his Chinese Room argument [Searle 1980] that it is impossible for computers, which are purely formal or syntactic machines, to have intensionality. It is of course often asserted that such-and-such a computer system has the following semantics, but this is human-attributed semantics, rather than the inherent kind which Searle refers to as "intensionality". Without intensionality no thinking is possible. Searle is often mistakenly supposed to have gone as far as asserting that machines of any kind could not think. On the contrary, while he asserts that computers, or any other purely syntactic machine cannot think, he is quite willing to suppose that a machine could think which was not purely syntactic, i.e., which had a semantic or intensional component deriving from its "causal powers".

Harnad refers to this as the "symbol grounding" problem [Harnad 1987], and this is a more convenient term for our argument, since we can refer to symbols being grounded or ungrounded. Harnad refers to the 'classical AI' approach as being 'symbolic functionalism', and thinks it makes the mistake of very seriously underestimating the difficulties of grounding symbols, and the constraints upon the symbolic components of a cognitive system which groundedness of its symbols entails. Let me quote the last paragraph of the definition of behaviours with which I began [Malcolm 1987].

> In a small and humble way, these behaviours can be likened to the sub-cognitive substrate which enables the world of information with which the planner deals. This Janus-faced concept of "enabling a world" covers both the idea of an abstraction device from the world to the representation which is enabled, and the connection of that abstracted representation to the world by the practical capability.

In other words, the behaviours are grounding the symbols employed by the planner. Thus "behaviours" as defined here are in fact the magic component required by a computer system in order to escape Searle's Chinese room argument, and to have intensionality, grounded symbols, or inherent semantics. Note that a behaviour-based computer system can't just be a computer, since the behaviours entail connection to the real world through the sensing and action capabilities of a robot. Note also that the simple connection of a robot to a computer system does not necessarily entail grounding of the symbols.

For example, suppose the robot is connected to the computer system in the 'classical' manner, wherein the robot is controlled via geometric motion primitives, mapped onto the output commands of the language, and the sensors are mapped onto the input commands; and lastly, that this robot is supposed to be doing an assembly task. The required operations of the assembly are "compiled" down into atomic robot motions and sensings. This is a one-way uncontrolled process, which is the reason why it is both complex and unreliable; and this uncontrolled unreliability is in turn the symptom which declares the symbols of the assembly operation to be ungrounded. The symbols which are grounded are the atomic robot motions, but there does not exist a secure coding-decoding between the robot motions and the assembly operations (part motions).

The secure coding-decoding required to connect a symbol to its real world referent can only be accomplished by the kind of goal-seeking operators exemplified on a simple level by servos, and on a more complex (knowledge-based) level by behaviours. These operators are two-faced: their sensory components connect the real world to the symbols in the symbolic level system; and the fact that they are goal-seeking connects the symbolic manipulations to effective causal action in the world, and ensures the continued secure binding of the symbol to its referent.

The problem of senseless behaviour

The story told so far begins to seem a little strained on the realisation that the assembly planning and assembly execution systems may be two separate systems, with only a simple one way file-transfer connection between them. How does symbol grounding survive that kind of connection? It suddenly falls to pieces on encountering senseless behaviours. Senseless behaviours are behaviours which involve no sensory component, but contrive their goal-seeking control of part motion by taking advantage of constrained motion. A simple example is dropping a chamfered peg into a chamfered hole. The action of the robot is confined to opening its gripper. The reliable result within the domain of part motion is that the peg ends up securely fitted into the hole (within the competence of the chamfers, provided the robot is situated within a certain radius of the hole, etc.). How can senseless behaviours ever ground a symbol? Without senses nothing can be known by the system except what it is "told", i.e., semantic attribution by a human being?

It is first necessary to realise that there is no problem about a simple unvarying action having a varying and goal-seeking effect upon the motion of a part. Indeed, it was the lack of necessary correlation between robot motion and part motion which created many of the problems behaviours are endeavouring to solve.

To solve this conundrum it is necessary to introduce the notion of a semantic loan. This intermediate category (between groundedness and ungroundedness) is the one which is used by AI experimenters who are aware of the symbol grounding problem (i.e. you can't just get the engineering dept to ground any old set of symbols for you), and who wish to do experimental development work on the component parts of what they hope will one day be a working behavioural system with inherent semantics. What they do is to take out a semantic loan - they build an ungrounded test system, but one in which the symbols have been restricted to those they reasonably suspect of being groundable, one day.

In this way it is possible to construct experimental models of parts of a mind, using only a computer, and consequently being restricted to ungrounded symbols, but groundable ungrounded symbols. Of course, it is a risk that you take, that the semantic loan will be repayable; risks of that sort are part of research. Looked at in terms of this metaphor, the typical "symbolic functionalist" ungrounded system becomes a system based on semantic loans that "behaviouralists" suppose unrepayable.

Looked at in this way, the initial assertion of meaning by a human that is necessary to first establish the referents of the symbols of a system based on senseless behaviours can be thought of as a semantic loan. The important question then becomes: is this attribution the kind that could reasonably be supposed to be within the powers of implementable sensory behaviours? Clearly there will be at least some cases where this is disputable, but at the same time, if it is admitted that there will be at least some cases in which experiment will redeem the semantic loan, then it must also be admitted that experiments with senseless behaviours will to that extent be able to offer useful results.

The Features of Assembly Behaviours

The following list of features contains overlapping and redefined features; these draw attention to different aspects of the same feature.

- computational modularity.
- functional modularity of effect in the assembly task.
- complexity reduction.
- goal achieving operators controlling part motions.
- reliability.
- encapsulating uncertainty and uncertainty-reducing sensing.
- simplifying planning and simulation of plans to an ideal world.
- contriving to make the real world seem ideal (to the planner).
- inherent (or borrowed) semantics.

REFERENCES

Albus, J.S., ''Brains, Behaviour, and Robotics'', BYTE Books, McGraw-Hill, 1981.

Brooks, R.A., ''Achieving Artificial Intelligence Through Building Robots'', AI Memo 899, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, May 1986.

Dennett, D., "Intentional Systems in Cognitive Ethology", BBS 6, p343-390, 1983.

Durrant-Whyte, H.F., ''Uncertain Geometry in Robotics'', Proc 1987 IEEE Int Conf on Robotics and Automation, pp 851-857, March 1987.

Erdmann, M., ''Using Backprojections for Fine Motion Planning with Uncertainty'', Proc. IEEE Int. Conf. on Robotics and Automation, 1985.

Fleming, A., ''Analysis of Uncertainties in a Structure of Parts 1'', and ''Analysis of Uncertainties in a Structure of Parts 2'', DAI Research Papers No. 271 and 272, Department of Artificial Intelligence, University of Edinburgh, 1985.

Harnad, S.,''Categorical Perception'', Cambridge University Press, 1987.

Harnad, S., net discussion in comp.ai.digest, 1987b, of his book, above.

Hayes, P., net discussion in comp.ai.digest, 17 Dec 1987.

Henderson, T., and Shilcrat, E., ''Logical Sensor Systems'', Journal of Robotic Systems Vol. 1, No. 2, pp. 169-193, 1984.

Koutsou, A., ''A Geometric Reasoning System for Moving an Object while Maintaining Contact with Others'', DAI Research Paper No. 267, Department of Artificial Intelligence, University of Edinburgh, 1985.

Lozano-Perez, Chapter 6, ''Task Planning'', in ''Robot Motion'', eds M. Brady et al., MIT press, 1982.

Lozano-Perez, T., and Brooks, R., ''An Approach to Automatic Robot Programming'', MIT AI Lab, AIM 842, 1985.

Lyons, D., ''Robot Schemas'', Ph.D. thesis, 1985, University of Massachussetts, Amherst, USA.

Malcolm C.A., and Fothergill P.A.,''Some Architectural Implications of the Use of Sensors'', invited paper at NATO Pisa conference Sep 1-5, 1986, in ''Languages for Sensor Based Control in Robotics'', ISBN 0-387-17665-9, eds Rembold and Hormann, Springer-Verlag, NATO ASI Series F, vol 29, pages 101-124, 1987 (DAI Research Paper 294).

Malcolm, C.,''Planning and Performing the Robotic Assembly of Soma Cube Constructions'', MSc Dissertation, Edinburgh University, September 1987, 87 pages.

Moravec, H.,''Locomotion, Vision, and Intelligence'', in Robotics Research 1, eds Brady and Paul, MIT Press, 1984.

Popplestone, R.J, Ambler, A.P, Bellos, I, ''RAPT: A Language for describing assemblies'', Industrial Robot, Sep 1978, 131-137.

Powers, W.T., ''Behaviour: The Control Of Perception'', Wildwood House, 1973.

Searle, ''Minds, Brains, and Programs'', BBS vol 3, pp 417-457, 1980.

Smithers, T, and Malcolm, C, ''A Behavioural Approach to Robot Task Planning and Off-Line Programming'', DAI Research Paper 306, Advanced Robotics Workshop, Manipulators, Sensors, and Steps towards Mobility, May 1987 Karlsruhe, 14 pages.