Many hands make light work? An investigation into behaviourally controlled co-operant autonomous mobile robots

D P Barnes, R A Ghanea-Hercock, R S Aylett, A M Coddington
UK Robotics Ltd.,
Derwent House, Clarence Avenue,
Trafford Park, Manchester, M17 1QS, UK.
dave.barnes@robotics.co.uk

Abstract

The past ten years has seen a flurry of research activity into the behavioural control of autonomous mobile robots. Yet despite this effort, many researchers are of the opinion that behavioural robots are incapable of achieving tasks more complex than simple can collecting, box pushing, herding or moving in formation. If such robots are to gain industrial credibility, these criticisms must be addressed. To focus the research we have studied the application of multiple mobile robots to a complex nuclear plant decommissioning problem. We argue that it is possible for multiple mobile robots to co-operatively perform a complex task provided that solutions to a number of key issues are incorporated into a behavioural control architecture. These include: behaviour conflict resolution, behaviour adaptation and behaviour scheduling. We have designed behavioural control methods to address these issues and our work has resulted in the creation of a behaviour synthesis architecture (BSA) which has been implemented on two real mobile robots. The application of the BSA to our complex industrial task is detailed and the results from the work are presented.

Introduction

UK Robotics Ltd. in collaboration with the University of Salford, have for some time been investigating the area of multiple autonomous co-operant mobile robots. Initial work (Barnes 1993) laid the foundations for our current project: Multiple Automata for Complex Task Achievement (MACTA)(Barnes & Aylett 1994). The previous project focused upon a material handling application involving two mobile robots known as Fred and Ginger. The research resulted in the design and implementation of a behavioural robot control architecture (Barnes 1996) capable of providing autonomous co-operant control of our two mobile robots. The application of this architecture, known as the behaviour synthesis architecture (BSA), was successfully demonstrated in our research laboratory. The task executed by the two robots was the co-operant relocation of an object (a perspex 'tray'), while avoiding obstacles en route. The research showed that co-operation could be achieved between autonomous robots and that behavioural control was a powerful facilitor in such situations. Since this early work, we have concentrated on increasing substantially the complexity of the task presented to our robots and have begun to address the new research problems that this has generated. Our current research is focused upon a nuclear plant decommissioning problem that requires complex task execution by multiple co-operant mobile robots. This paper elaborates upon our industrial application problem and provides an overview of the foundations for the research.

Industrial focus

There are hundreds of nuclear reactors in service and under construction around the world and as many of these reactors were built prior to 1980, it means that a major decommissioning bill will fall due shortly after the turn of the century. With current technology, plant dismantling is still a labour-intensive process and the design of equipment for dismantling, especially remote equipment, is still in its infancy. Currently, critical decommissioning activities involve the use of multiple manually controlled robotic devices. One scenario requires two mobile robots to manipulate objects from one to the other, while being visually monitored by an operator with the aid of several other mobile robots. This is a multi-robot, multi-degree-of-freedom problem and requires significant concentration and skill on the part of the human operator. Understandably, dismantling productivity is well below that of a comparable non-radioactive environment and safety is of paramount importance. It is upon this nuclear plant decommissioning problem that our research has focused and to investigate this application area, we have devised a laboratory based robot task involving our Fred and Ginger robots. This task is described as follows:

1. Fred navigates through our laboratory to a docking beacon. Ginger tracks Fred.
2. He then docks with this beacon and acquires an object (a pipe section).
3. After acquiring the object he then navigates with the object around the laboratory. Ginger moves nearer to Fred.
4. He transfers the object to Ginger while they are both still moving. Ginger acquires the object and finally navigates across the laboratory to a target location where the object is deposited.

This is a complex task that requires a range of robot capabilities. Navigation, obstacle avoidance, beacon docking, tracking, object acquisition/release and co-operant object transfer are all essential. Yet these activities must be correctly sequenced and an operator clearly does not want to be involved in having to control manually every single motion of both robots. Rather it is far more appropriate to allow the robots to solve behaviourally their local problems of obstacle avoidance, beacon navigation, tracking e.t.c. However, to apply a behavioural control régime to a complex task such as this, is far from trivial and a number of key issues were identified. Firstly, the number of behaviours required for the task means that many of these are potentially competing with one another at any given point in time. Our robots however, can only perform one activity at a time, hence some form of behaviour conflict resolution was required. Secondly, given that the task requires a large number of interacting behaviours, 'hand-crafting' these so as to generate the desired task achieving behaviour, is extremely difficult. While small behaviour numbers are manageable, as more are added, honing these by hand becomes impractical and hence some form of automated behaviour adaptation was required for this purpose. Finally, behavioural control architectures have been used extensively in the context of enabling an autonomous robot to perform a single task, not a sequence of tasks. Therefore, for a behavioural control approach to be adopted that requires each robot to perform a sequence of operations, some form of behaviour scheduling was required. The following sections detail our research into these key issues.

Behaviour conflict resolution

The activity of co-operating behavioural robots can be regarded as a continuum between two basic types of diverse behaviour. At one extreme, the behaviour can be regarded as being essentially egotistic, where a robot is concerned purely with self directed behaviour, e.g. obstacle avoidance and energy conservation. At the other extreme their behaviour can be regarded as being essentially altruistic, e.g. when a group of robots need to work together to perform some common task. Multiple robots co-operatively relocating an object is an example of such behaviour. However, these diverse types of behaviour are essentially in conflict! The first would cause a robot to remain stationary and to stay well away from all other objects within its environment, including other robots, while the other type of behaviour would drive a robot to team up in close proximity with its fellows to perform an activity of work. Given the different behaviours that can be found in single robot, *monad* and multi-robot, *polyad* scenarios, the research focused upon the design of a control architecture that could accommodate such diverse and conflicting behaviour types. What emerged was the BSA, see figure 1, and it constitutes an important addition to the mobile robot control architectures of Arkin (Arkin 1989) and Brooks (Brooks 1986). For purely conceptual convenience, four different behaviour levels in the architecture were identified:

- A **self** level contains those behaviours concerned with the maximisation and replenishment of internal resources, e.g. remaining stationary to conserve battery power.
- An **environment** level contains those motion behaviours associated with activities involving other objects within the robot's environment, e.g. collision avoidance.
- A **species** level contains those behaviours associated with co-operant activities e.g. maintaining a correct position and orientation with respect to an object while co-operatively relocating this object.
- A **task** level contains those behaviours specific to a particular task, e.g. navigating to the initial location of an object to be relocated, then subsequent navigation to the desired goal location.

Sensory stimuli, from our developed robot sensor systems, provide the appropriate internal and external state information needed for the various behaviour

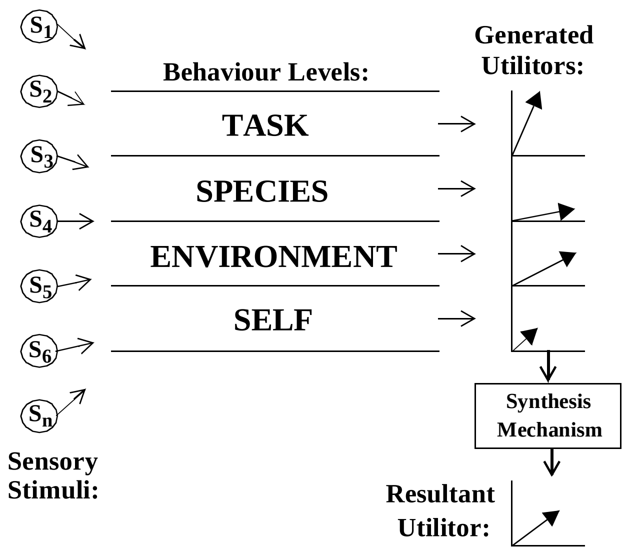

Figure 1: The behaviour synthesis architecture (BSA).

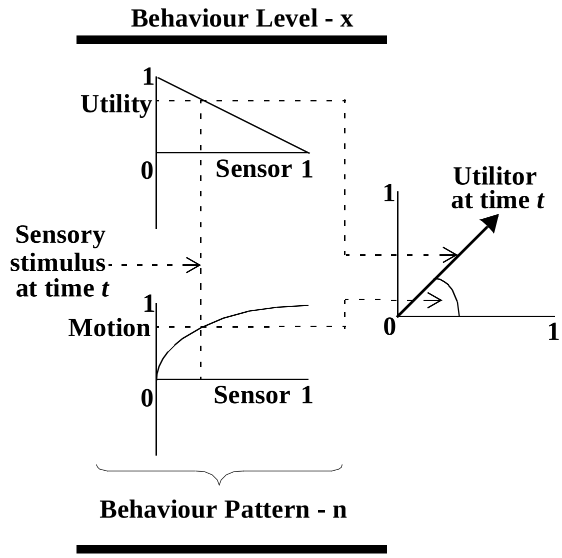

Figure 2: Behaviour pattern example.

levels and from each relevant level, appropriate motion responses are generated that relate to the desired actuation. Any level can contain a number of *behaviour patterns*, **bp**′**s**, where

$$\mathbf{bp} = \begin{bmatrix} r \\ u \end{bmatrix} \tag{1}$$

and

$$r = f_r(s) \tag{2}$$
$$u = f_u(s) \tag{3}$$

r is the desired motion response and is a function, f_r, of a given sensory stimulus, s. Associated to every response is a measure of its utility or importance, u. This quantity is a function, f_u, of the same sensory stimulus. The use of utility originated from our early research into formalisms for modelling co-operant behaviour. Game theoretic studies (Von Neumann & Morgenstern 1953) showed that single and n-player games could be used to model monad versus nature and monad1 versus monad2 competitive and co-operant scenarios. Strategy selection in these games is dependent upon the information a player may have regarding their opponent's or partner's move and the relative utility (or pay-off) of any counter or co-operative move. As this information is analogous to the sensory stimuli available to a robot and utility is used to great effect in the selection of an appropriate strategy from a set of possible strategies, it was realised that such a concept could be incorporated within our control architecture. Hence a **bp** defines not only what a robot's motion response should be for a given sensor input, but it also provides a measure as to how the relative importance of this response varies with respect to the same sensor input. The values of r and u constitute a vector known as a *utilitor*. Figure 2 shows an example of a simple **bp** that might exist at a given level. Consider the situation where the sensory stimulus relates to a robot's forward facing distance to obstacle measuring sensor and the associated motion response relates to the forward translate velocity for that robot. From figure 2 it can be seen that as the robot gets nearer to the object then its forward translate velocity will be reduced to zero. At the same time, the associated utility for this motion response increases. Thus as the robot gets nearer to an object in its path, the more important it becomes for the robot to slow down. At any point in time, t, multiple conflicting motion responses are typically generated. For example, a robot may be navigating towards a goal location while co-operatively relocating an object when an obstacle unexpectedly appears in its path and at the same time it senses that it must recharge its batteries. In such a situation, what should it do? In the BSA, conflicting motion responses are resolved by a behaviour synthesis mechanism to produce a resultant motion response. Competing utilitors are resolved by a process of linear superposition which generates a resultant utilitor, UX_t where:

$$UX_t = \sum_{n=1}^{m} u_{(t,n)} \cdot e^{j \cdot r_{(t,n)}} \tag{4}$$

and m equals the total number of related utilitors generated from the different behaviour levels, e.g. all those concerned with translation motion or those concerned

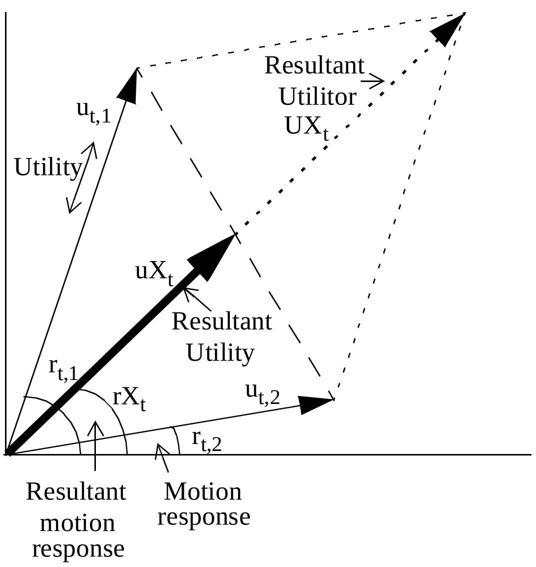

Figure 3: Generating a resultant utility and motion response from two constituent utilitors.

with rotation motion. Given a resultant utilitor, a resultant utility, uX_t, and a resultant motion response, rX_t are simply obtained from

$$uX_t = \frac{|UX_t|}{m} \tag{5}$$

and

$$rX_t = \arg(UX_t) \tag{6}$$

X identifies the relevant degree of freedom, e.g. translate or rotate, and the resultant motion response, rX_t, is then executed by the monad. From equation 4, it can be seen that generating a resultant utilitor from different behaviour levels within the architecture constitutes a process of additive synthesis, see figure 3.

Behaviour adaptation

While behaviour based methods, such as subsumption (Brooks 1986) and BSA offer a direct mapping of sensory stimuli to response, they suffer from an inability to adapt to environments which demand a structured response, such as escaping a dead end corridor. To solve this problem we have been investigating the application of fuzzy logic (Zadeh 1965), as fuzzy rule systems can easily cope with such situations through linguistic reasoning (Surmann, Peters, & Huser 1995). A fuzzy controller is also normally very robust and can tolerate major degradation of its rule structure, (Kosko 1992) and insensitivity to noise or uncertainties in the control inputs, which make it ideally suited to mobile robot control (Watanabe & Pin 1993).

A Fuzzy control system works by encoding an experts knowledge into a set of rules which are smoothly interpolated and the resultant is defuzzified to give a crisp actuation output. Each rule is specified as either a triangular, trapezoid or some other function e.g. gaussian and assigned to some range of input variable. Trapezoid functions were selected as these are computationally efficient, which is a priority when used for real-time control systems. The fuzzy rules normally take the form

$$\text{if } (x \text{ is } A) \text{ and } (y \text{ is } B) \text{ then } (z \text{ is } C) \tag{7}$$

where x, y, z are linguistic variables representing inputs and outputs of the fuzzy controller, and A, B and C are the terms for the variables in the universes of discourse X, Y, and Z. Fuzzy rules can be represented by a fuzzy associative memory matrix, (FAM) (Kosko 1992). In this system the base dimensions represent the input variables and each FAM entry represents an output fuzzy set. There are typically 3 to 5 output membership sets, e.g. negative large (NL), negative small (NS), zero (ZE), positive small (PS), and positive large (PL). Using a FAM representation, the weight for the i_{th} FAM entry was calculated using the minimum rule

$$w_i = \min\{F_i(x), F_i(y)\} \tag{8}$$

where x and y represent the input dimensions of the FAM. The total defuzzified response for n output membership sets is then

$$F_T = \frac{\sum_i (w_i \cdot B_i)}{\sum_{i=1}^{n} w_i} \tag{9}$$

where B_i is the output fuzzy set. As the BSA uses a utility function for each **bp**, it was realised that these functions could provide a direct means of adapting the relative utility of each behaviour from a higher level of control. Our developed fuzzy control layer takes direct sensory input and applies negative feedback, F_T, on the utility of selected behaviours. The contribution of each behaviour is therefore adjusted dynamically to suit the current set of environmental stimuli. The net effect is to allow the robot to focus on its current response while maintaining some awareness of a more general goal. An example of this is when the robot enters a 'potential well' while navigating towards a beacon, see figure 4. As the robot approaches an obstacle, the importance of avoiding it increases due to an active obstacle avoid f_u function within the BSA, mean while the fuzzy rule base responds by turning down the utility of moving towards the beacon,

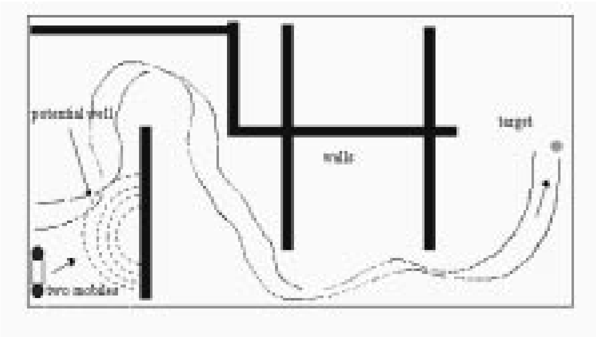

Figure 4: Typical path of simulated coupled robots using the BSA with behaviour adaptation.

$f_u(s)Nav := f_u(s)Nav - F_T$. The emergent response is for the robot to follow the perimeter of the obstacle until a free path towards the beacon is found when it returns to navigating in that direction. One advantage of this method is the modularity of the system. The utilities of multiple conflicting behaviours can be dynamically modified prior to generating a resultant utilitor, by increasing the number of input/output dimensions of a fuzzy rule base, without modifying the set of $\mathbf{bp's}$ themselves. Alternatively to avoid creating a FAM with large numbers of input-output dimensions, which are difficult in themselves to design, small groups of behaviours can be assigned to separate FAM's. The principle difficulty in using a fuzzy system is the rapid increase in possible combinations of rules along with the number of input and output dimensions. This has led many researchers to use global search algorithms and artificial neural networks to find an optimum set of rules. We have applied a standard genetic algorithm (Holland 1992), to the process of selecting the FAM entries, with manually selected fuzzy input sets and functions. The encoding scheme for genetic optimisation takes each FAM entry and assigns it to a fixed length binary vector, these are then strung together to form a 'chromosome' vector. Each complete FAM is then evaluated against a specified fitness function and the normal process of re-combination applied to select the fittest matrix. In our experiments we have used a fitness function that is a combination of avoiding collisions with obstacles, minimising distance from a goal beacon, and achieving a minimum displacement of our coupling *x*-*y* table (described later).

Behaviour scheduling

Despite the success of the BSA, it did initially suffer from a problem common to all behavioural architectures. Effectively, $\mathbf{bp's}$ may interact in ways which are not useful to the robot. While utility functions

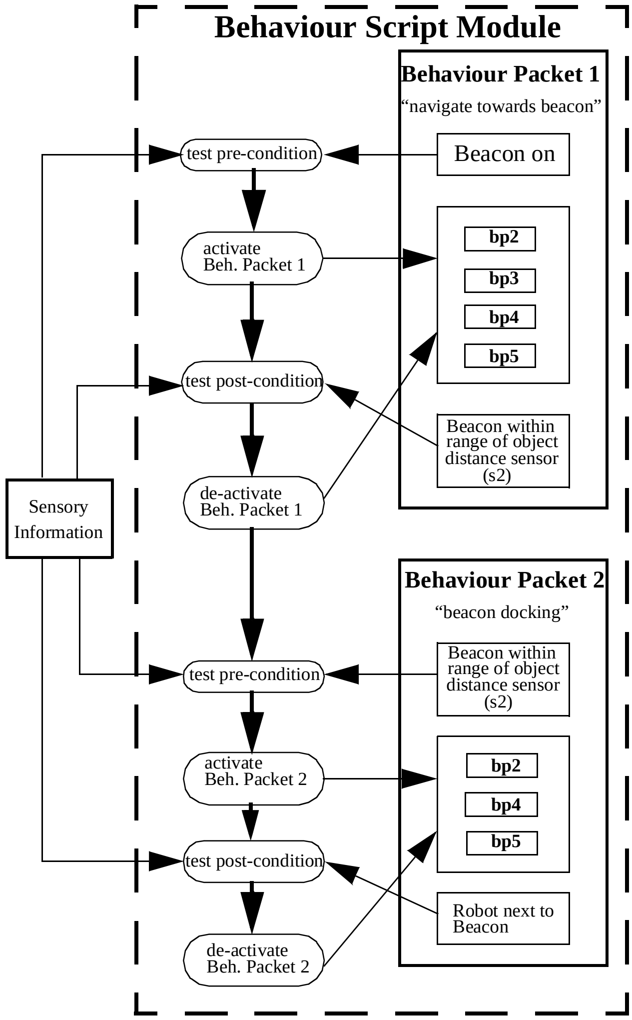

Figure 5: Behaviour script example. **bp2** - **bp5** represent appropriate behaviour patterns while **s2** is a robot to beacon distance measuring sensor.

are ideal in the context of generating a resultant robot motion, they are sensor dependent not sub-task dependent. Hence situations can arise when the associated utility for a particular **bp** needs to be forced to zero, irrespective of its input sensor value. This action effectively producing a **bp** of zero importance and hence one which does not contribute to the resultant motion response. We argue that the root of the problem is in allowing all $\mathbf{bp's}$ to be active at all times rather then restricting active behaviours to those most useful for the achievement of a particular sub-task. What was required was a means of allowing the task structure to create a context in which only appropriate $\mathbf{bp's}$ would be activated. In the BSA, a structure known as a *behaviour script* was designed precisely for this purpose. A behaviour script consists of *behaviour* *packets*, each of which contain a triplet: [sensor pre-condition(s), **bp′s**, sensor post-condition(s)]. Sensor pre- and post-conditions are a combination of a particular sensor and either an initiating or terminating condition. These are similar to the continuous action model implemented by Gat (Gat 1992) in which activities are initiated and terminated by conditions, while Zelinsky's 'graphical sketches' (Zelinsky, Kuniyoshi, & Tsunkune 1994) represent a more specialised form of this approach to navigation only. As each behaviour packet within the behaviour script is carried out, the pre-condition for the next is encountered so that finally, the whole script is executed. Hence this process constitutes an ideal mechanism for scheduling behaviours. Figure 5 illustrates the process of executing a fragment of a behaviour script.

Results obtained

Our research prior to the MACTA project resulted in the creation of an early version of BSA, **bp′s** to accomplish the co-operant object relocation task and a simple behaviour packet to demonstrate the behaviour script concept. To accomplish our new robot task, additional hardware has been developed for our two mobile robots together with new **bp′s**, a behaviour adaptation method and a full implementation of the behaviour script mechanism. The results from this work are presented as follows:

Our behavioural co-operating robots

To perform our original co-operant object relocation task, Fred and Ginger were equipped with ultra-sonic sensors for obstacle detection and an instrumented self-centering x-y table (historically known as the capture-head) upon which the object to be relocated could be placed. The capture-heads served the purpose of providing each robot with information concerning the cartesian position of the object relative to each robot's central axis. When an object was in contact with each capture-head, the two robots were effectively mechanically coupled (just as two people are when jointly carrying an object) and this meant that the relative motion of one robot could be transmitted to the other robot and vice versa. Distance data from an array of ultra-sonic sensors and the x-y data from a capture-head formed the sensory input to each robot's BSA. Work on the project to date has resulted in additional beacon sensors and manipulators being incorporated into both Fred and Ginger. Figure 6 shows Fred and Ginger with this added hardware. Our infra-red beacon design has been extended to function as an inter-robot tracking system as well as being used for fixed beacon location. The new manipulators each use a high torque d.c. motor (600 mNm continuous at 20 rpm) to drive two gripping fingers. At present there is no force feedback as only a simple on/off action is required. The payload capacity is approximately 500 gm with a reach of 20 cm and the maximum object diameter is 11 cm. We have found that when the two robots are coupled, when co-operatively transporting an object, they act as an articulated unit. This is due to the 3 DOF passive compliant capture-heads and an induced rotation motion about the gripper vertical axes due to gripper slip on the pipe section. This increases the complexity of the control problem and effectively reduces their operating velocity while coupled. However, the advantage of being able to acquire and release an object automatically, as compared to performing this task manually on our previous project, is a major benefit.

Figure 6: Fred and Ginger with capture-heads, manipulators, ultra-sonic and beacon sensors.

Behaviour adaptation and software simulator

A dynamic simulation of our coupled mobile robots has been created which runs the BSA and fuzzy control system. Each simulated robot has an equivalent set of sensors to the physical robot, with floating point representation. The simulation was implemented on a 486 PC and typical times to evolve a working FAM are 10 to 20 hrs., depending on the population size and the complexity of the environment. During trials of the simulated robots, the use of a hierarchical control structure enabled the robots to adapt the relative importance of their behaviours which significantly improved their ability to cope with difficult environments. In particular they were able to successfully negotiate local minima in the 'potential fields' caused by attraction to the goal and repulsion from the walls. The evolved matrix was then transferred to our real robots, which were directed to move through a similar real environment with walls positioned relative to

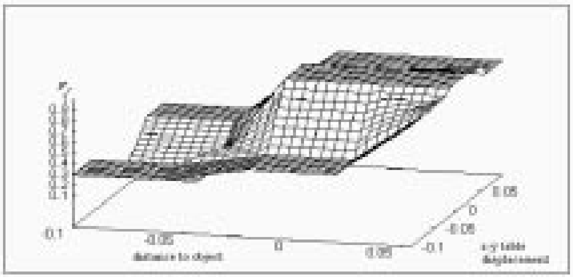

Figure 7: An evolved FAM action surface (population size 22).

the scale of the simulated model. Data from the firing of each fuzzy rule was then stored for later analysis. Runs of the real robots were also made using a hand-crafted FAM. By comparing the weight arrays (i.e. complete record of all rule activity) between an evolved and hand-crafted FAM, we have been able to analyse whether it is viable to use an evolved control structure for a real mobile robot when it forms part of a hierarchical control system. From our observations of the real robots, the use of evolution based methods to assist in the design of a FAM action surface for the purpose of providing behaviour adaptation has been confirmed. The important criteria appears to be limiting the search space to optimising the interaction of existing hand-crafted behaviours, rather then evolving the entire set of behaviours from scratch. Figure 7 shows an evolved FAM action surface which takes two inputs, distance to obstacles and displacement of the x-y table. The output is applied as negative feedback, F_T, to the **bp** controlling attraction to the goal, (an infra-red beacon on the real robot).

Behaviour patterns, packets and scripts

To move towards being able to achieve our laboratory based robot task, we have had to develop new **bp′s**, packets and scripts. Appropriate $f_r(s)$ and $f_u(s)$ functions were initially hand-crafted via experimentation. A FAM action surface for selected **bp** interaction was then generated. When satisfied with the performance of a set of **bp′s**, they were then grouped into a behaviour packet for inclusion into a script. Generally, behaviour packet generation is an iterative process requiring a good deal of **bp** refinement before they can be put into a given task script. So far we have created behaviour packets to co-operatively TRANSPORT an object, NAVIGATE to a beacon, DOCK with a beacon, RELEASE an object and TRACK another robot. Table 1 shows details of our TRANSPORT behaviour packet. Identified are those **bp′s** that are active during this sub-task and table 2 shows their associated

SLOTS	CONTENTS
`name`	`A`
`tag`	`TRANSPORT`

`Active Behaviour Patterns`	`1. translate`
	`2. obstacle avoid`
	`3. centre capture head`

`Sensory Postcondition`	`robot detects beacon`

Table 1: TRANSPORT behaviour packet.

functions. As can be seen, simple functions have been used for each required $f_r(s)$ and $f_u(s)$. As the robots are capable of translate (forward/back-wards) and rotate (clockwise/anti-clockwise) motion, then the **bp′s** have been designed accordingly. Using our current behaviour packets we have been able to manually generate a script for each robot that causes:

1. Fred and Ginger to co-operatively transport a pipe section.
2. When both robots are near a beacon, Fred releases the pipe section. Ginger moves towards this beacon and docks with it while carrying the pipe section.
3. After docking at the beacon, Ginger releases the pipe. Fred tracks Ginger while Ginger executes sub-task 2 above.

The successful execution of this script has demonstrated that we have made significant progress towards achieving the full laboratory based robot task described previously. Correct sub-task execution by each robot requires the complete integration of all our developed hardware and software and we have learnt a great deal concerning behaviour pattern, package and script generation. The lessons from this work have enabled us to investigate how to automate the process of script generation and these results are presented in the literature (Aylett et al. 1997).

Conclusions

Our early work into co-operant object relocation and the resultant design of the BSA incorporating **bp′s**, packets and scripts, laid the foundations for this MACTA project. We have significantly increased our desired robot task achieving capability by focusing on a complex nuclear plant decommissioning problem. To investigate this domain our studies have involved the use of both real and simulated robots. Our Fred and Ginger mobile robots, used on our earlier work, have been considerably enhanced in terms of both hardware and software. They now sport new beacon detection

Description	Translate bp's
translate... Generates a constant response with constant utility.	$r1 = 0.5$ $u1 = 0.5$
obstacle avoid... $s1$ - ultra sonic array. Deccelerates robot when object detected. Negates any +ve **bp's** with max. initial utility.	$r2 = f_r(s1) = -s1$ $u2 = f_u(s1) = s1$
centre capture head... $s2$ - optical encoder. 'Over-damped' response to sensor input with 'under-damped' utility.	$r3 = f_r(s2) = \sqrt{s2} + 1$ $u3 = f_u(s2) = \sqrt{s2}$
	Rotate bp's
obstacle avoid... $s1$ - ultra sonic array. Rotate away from object with max. initial utility.	$r4 = f_r(s1) = s1 + 1$ $u4 = f_u(s1) = s1$
centre capture head... $s2$ - optical encoder. 'Under-damped' response to sensor input with 'under-damped' utility.	$r5 = f_r(s2) = \sqrt{s2}$ $u5 = f_u(s2) = \sqrt{s2}$

Table 2: Active $\mathbf{bp's}$ for the TRANSPORT behaviour packet. All r, s and u are normalised.

sensors and manipulators for object acquisition and release. Additional $\mathbf{bp's}$, packets and scripts have been created to utilise this new hardware and a simulator tool has been developed to generate higher level FAM action surfaces that provide on-line feedback to appropriate resident $\mathbf{bp's}$. This facility greatly enhances the performance of our real robots when they are behaviourally executing a sub-task and we have successfully demonstrated Fred and Ginger performing key sub-tasks of the intended final laboratory based decommissioning problem. With respect to the question posed by the title of this paper: Many hands make light work?, this can be answered in two ways. Firstly, behaviourally controlled autonomous mobile robots can be successfully used to co-operatively achieve a complex task, hence yes, they can make light work for each other. But secondly, from the point of making light work for the human operators who have to 'programme' such vehicles, behaviour conflict, adaptation and scheduling are essential features that must be incorporated into a behavioural architecture. Provided these features are present, then yes, getting multiple behavioural autonomous robots to do what you want them to do is achievable!

Acknowledgments

The MACTA research is supported by the UK EPSRC, Control, Design and Production Group.

References

Barnes, D. P. 1993. Multiple co-operant mobile robots for advanced manufacturing applications. EPSRC Grant GR/F71454.

Barnes, D. P., and Aylett, R. S. 1994. ITE:Multiple automata for complex task achievement. EPSRC Grant GR/J49785.

Barnes, D. P. 1996. A behaviour synthesis architecture for co-operant mobile robots. In *Advanced Robotics and Intelligent Machines*, Gray, J. O., and Caldwell, D. G. eds., IEE Control Engineering Series 51, 295-314.

Arkin, R. C. 1989. Motor schema-based mobile robot navigation. *Int. Journal of Robotics Research* 81(4): 92-112.

Brooks, R. A. 1986. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation* RA-2: 14-23.

Von Neumann, and Morgenstern, O. 1953. *Theory of games and economic behaviour.* Princeton University Press.

Zadeh, L. 1965. Fuzzy Sets. *Journal of Information and Control* 8: 338-353.

Surmann, H.; Peters, L.; and Huser, J. 1995. A Fuzzy System for Realtime Navigation of Mobile Robots. In 19th Annual German Conf. on AI, KI- 95, 170-172. Bielefield.

Kosko, B. 1992. *Neural Networks and Fuzzy Systems, A Dynamical Systems Approach to Machine Intelligence.* Prentice Hall Int.

Watanabe, Y., and Pin, F. G. 1993. Sensor based navigation of a mobile robot using automatically constructed fuzzy rules. In Proc. ICAR, 81-87. Tokyo, Japan.

Holland, J. 1992. *Adaptation in Natural and Artificial Systems.* MIT Press.

Gat, E. 1992. Integrating planning and reaction in a heterogenous asynchronous architecture for controlling real-world mobile robots.In Proc. 10th National Conf. on Artificial Intelligence, AAAI-92, 809-815.

Zelinsky, A.; Kuniyoshi, Y.; and Tsunkune, H. 1994. Monitoring and coordinating behaviours for the purpose of robot navigation. In Proc. IROS.

Aylett, R. S.; Coddington, A. M.; Barnes, D. P.; and Ghanea-Hercock, R. A. 1997. Supervising multiple co-operating mobile robots. In Proc. 1st. Int. Conf. on Autonomous Agents. Marina del Rey, CA.