Humans Teaching Learning Machines: Apprentice Systems and Shared Control of Military Vehicles

Bruce L. Digney
Defence Research and Development Canada- Suffield
PO Box 4000, Station Main
Medicine Hat, Alberta, T1A 8K6, CANADA
Tel: 403-544-4123, Fax: 403-544-4704,
Bruce.Digney@drdc-rddc.gc.ca

Abstract

Many military applications will benefit from the use of unmanned vehicles. However communication realities have limited the effectiveness of the current direct teloperated vehicles. One solution is to increase the on-board autonomy of the vehicle. While this does reduce the communication requirements, the level of machine intelligence is not sufficient to insure operation in extreme situations. Proposed is a shared control solutions that is a synergistic partnership of on-board autonomy and the superior cognitive capabilities of the remote human operator. This paper describes work at the *Defence Research and Development Canada- Suffield* (DRDC-Suffield) in teleoperated vehicles, autonomous control systems and discusses the issues and benefits of shared control of military vehicles.

After many years of attempting to program robots to function in the most benign environments performing even the simplest of tasks progress has been limited. What appears as a simple straight forward operation to a human becomes very complex and unconstrained problem when a programmer attempts to enumerate all possible conditions and encode solutions. For unstructured environments and freely forming solutions learning is essential. While the need for hierarchies within learning control systems is clear, it is also clear that such hierarchies should also be learned. Learning both the structure and the component behaviors is a difficult task. The benefit of learning the hierarchical structures of behaviors is that the decomposition of the control structure into smaller transportable chunks allows previously learned knowledge to be applied to new but related tasks. Presented in this paper are improvements to Nested Q-learning (NQL) that allow more realistic learning of control hierarchies in reinforcement environments.

Learning robots have been researched for many years. Results have been limited due to the excessive time and effort required to learn all knowledge from a blank start. By observing how humans learn it is obvious that they do not learn everything from scratch. But, benefit from instinctive behaviors, in place at birth, years of protected nurturing through infancy, many more years at school and extending all the way to reading manuals and scientific journals. Humans are taught these knowledge building blocks and discover ways decompose them and to combine them in novel ways to solve new problems quickly. This paper also discusses DRDC's efforts that build upon hierarchical learning methods in which robots learn how to decompose learned knowledge into reusable chunks and exploit those chunks in future situations. This is possible because hierarchical learning results in a hierarchical model structure rather than a monolithic structure from which behaviors can be isolated. This makes hierarchical learning control systems very amenable to being taught by humans. DRDC-Suffield is developing such a system called the *Operator's Apprentice*. It is envisioned that the Apprentice would observe and learn from a human operator until it reaches some level of proficiency at which point the apprentice would assume control. Once the human is removed the apprentice's abilities would continue to improve perhaps surpassing the human operator's performance level.

Paper presented at the RTO HFM Symposium on "The Role of Humans in Intelligent and Automated Systems", held in Warsaw, Poland, 7-9 October 2002, and published in RTO-MP-088.

Contents

List of Figures

List of Tables

1 Shared Control

1.1 Introduction

Finite communication bandwidth has limited the situational awareness of the remote operator and transmission delays have restricted the unmanned system's rate of change to be much slower than the communication latencies the closed control loop. These limitations have led many robotics researchers to pursue higher autonomy in vehicles allowing control to reside on board eliminating the need for constant communication to the remote operator. However, subsequent limitations in the cognitive capabilities of autonomous control systems have further led researchers to concede that a form of shared control is required. This is true for military applications that can benefit from unmanned vehicles with higher autonomy, but still require human level cognitive capabilities in extreme situations. Indeed, in military applications where lethal force is controlled or a machine error will result in the loss of human life fully removing the human from the loop is not an option. Presented in this section is an overview of the Defence Research and Development Canada - Suffield (DRDC-Suffield) tactical vehicle program including current teleoperated military vehicles and the *Autonomous Land Systems Program* (ALS) and a description of the work on shared control.

1.2 Unmanned Military Land Vehicles

Research into military land vehicles at DRDC-Suffield has yielded many novel locomotion concept vehicles, teleoperation systems and fielded systems. To provide background these projects will be briefly described.

1.2.1 Telematic Control with Anceaus

The *Anceaus* telematic control system has been developed by DRDC-Suffield and is in current use in on many of the vehicles described in this paper (Brosinsky, 2001*a*). Video from the teleoperated vehicle is transmitted to the operator console. This transmission uses a commercial line of sight video communication system that has difficulties dealing with signal drop out in urban or wooded areas. Additional state information is sent using a separate radio link. This link is capable of two way communication and is further used to send the operator commands to the vehicle. These commands are of a generic form (turn right, stop, etc) with the personality module on each particular vehicle translating the generic commands into specific actuator signals.

1.2.2 Scout Vehicle

The scout of Figure 1 is a gasoline powered, hydraulic driven, skid steered vehicle used to demonstrate unmanned reconnaissance operations. Configured for teleoperation the scout has a sensor mast that supports a pan-tilt platform used for pointing colour as well as infrared imagery. Its world position is determined using differential GPS and it is steered by deferential control of the speed of the wheels on each side of the vehicle. Telematic control is implemented using the anceaus system. Configured for autonomous operation (as pictured in Figure 1) the scout vehicle has two forward range sensors, laser scanner and stereo vision.

1.2.3 Caterpillar D7 Bulldozer

The Caterpillar D7 shown in Figure 2 is capable of both on-board human operation and telematic control. The remote operator controls the vehicle and the blade depth and angle by viewing video transmitted from forward looking cameras. A differential GPS determines the D7's position and orientation which is displayed on the operators console map. Blade control and earth working in general require a skilled operator even for on-board operation. When the control loop latencies are

Figure 1: Unmanned Scout Vehicle

introduced blade control on teleoperated D7 is further complicated. Current work at DRDC-Suffield is striving to automate blade control as well as the repetitive movements common in earth working.

Figure 2: Caterpillar D7 Bulldozer

1.2.4 Articulated Navigation Testbed (ANT)

Shown in Figure 3 is the *Articulated Navigation Testbed* (ANT) concept vehicle. The ANT moves using driven wheels mounted on the ends of actuated legs. Each leg has a single controllable degree of freedom revolute joint at the shoulder that can rotate approximately 400 degrees. By lifting the wheels over obstacles the ANT can traverse extreme terrain. Each body module of the ANT has two legs attached to it with each modules connected using a powered articulated joint. The high number of degrees of freedom associated with locomotion alone preclude manual control for manned or telematic operation. A straight forward example of shared control would have have all time critical locomotion processes controlled by the on-board autonomy system with the operator specifying the desired vehicle pose and ground clearance to be maintained. Under extreme conditions the leg control automatic control may be inadequate. In such situations the shared control system relaxes the real time criteria and allows for explicit control of the legs by the remote operator. This will allow the ANT to progress through the extreme conditions albeit at a slower rate.

Figure 3: Articulated Navigation Testbed (ANT)

1.2.5 Improved Landmine Detection Program (ILDP)

The *Improved Landmine Detection Program* (ILDP) shown in Figure 4 represents the culmination of five years of research into vehicles, telematic control and landmine detection systems at DRDC-Suffield. The ILDP vehicle rides on a bogey suspension with flotation tires that maintain a ground pressure below the level that would detonate anti tank mines. It detects landmines by looking for disturbed soil indications in the infrared images, metal content of the ground using electromagnetic signatures and changes in ground dialectic characteristics using ground penetrating radar. When the fusion of these sensors indicates to some threshold probability the presence of a landmine, then thermal neuron activation (TNA) is used as a confirmatory sensor. This requires that the vehicle stop and the TNA be placed over the suspect location. Once the TNA has confirmed the existence of a mine, the area is marked for later remediation. The vehicle is teleoperated using the *anceaus* system. Fusion of the sensor data is done automatically, but the analysis of the infrared imagery for disturbed soil is still performed by human operators. This is an example of a life critical situation where the automatic system is inadequate to analysis the images and a human operator must remain in the loop.

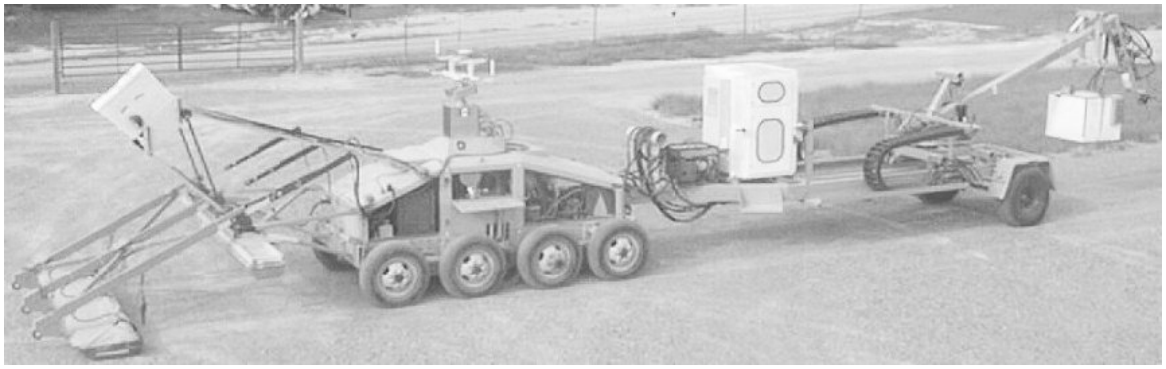


Figure 4: Improved Landmine Detection Program

1.2.6 Cognitive Colonies

Four robots of the Cognitive Colonies project (currently 10 robots in total) are shown in Figure 5. Cognitive Colonies is a joint project in distributed robotics between DRDC-Suffield, DARPA/ITO and Carnegie Mellon University (Scott Thayer and Digney, 2000). The concept is to deploy 100s or

Figure 5: Cognitive Colonies

1000s of small robots and have them cooperate to perform large-scale tasks. Distributed robotics provides greater flexibility, robustness and redundancy than a single large robot. The shear number of robots with their communication and manpower requirements make direct telematic control of all robots impossible. However, a remote operator must be able to assume full control of any individual robot periodically.

1.3 Autonomous Land Systems Program

Given the limitations of current communication and teleoperation systems and real need to remove humans from the hazards of many military scenarios, DRDC has began the *Autonomous Land Systems Program* (ALS). The ALS program strives to increase on-board intelligence and allow for greater independence from the remote operator and hence lessen manpower and communication requirements. This section describes the research that is contributing to the ALS program.

1.3.1 Reinforcement Learning of Hierarchical Control Structures

The use of externally imposed hierarchical structures to reduce the complexity of a learning control system is well established. Within this imposed hierarchy, sequences of actions are abstracted, by hand, into skills and the robot is restricted to fine tuning the prespecified skills. It is clear that having the machine learn the hierarchical structure by itself is an important step to learning more broadly applicable behaviors. In this research (Digney, 1996*a*) (Digney, 1998), a Nested Q-learning technique has been developed that generates a hierarchical control structure as the robot interacts with its world. The Nested Q-Learning algorithm is presented in detail in Section 2.

1.3.2 Apprentice Systems

Given the frailties of real machines and the long learning times required to achieve autonomous operation, it is clear that fully unassisted learning for robots is unrealistic (Digney, 1996*b*). Physical realities make the acquisition of enough training experience prohibitive. In biological agents information is passed through genetically hardwired vital initial behaviors (instincts) and the genetic predisposition to develop useful behaviors. Biological agents also benefit from long periods of infancy in which they are guided and protected before they become self-sustaining, let alone productive. These concepts are referred to as shaping. In this research, methods for pretraining and supplying initial guidance to prepare machines for future endeavours are being developed. The concepts and issues of apprentice systems are discussed in Section 3

1.3.3 Outdoor Navigation

The first step to autonomous vehicles is have the vehicles move about in their world under their own control. This requires that the vehicles sense their surroundings, build models and then plan routes. DRDC is developing obstacle avoidance and path planning algorithms based upon 2.5-D geometric models derived from forward-looking range sensor data (laser range finders and stereo vision). These models can represent positive and negative obstacles as well as terrain gradients, from which path planning algorithms can determine safe and optimal paths.

1.3.4 Learned Trafficability

While clearly necessary, geometric information is not sufficient to insure successful navigation in outdoor environments. Many barriers to navigation cannot be represented in a geometric model alone. Barriers such as soft ground, snow, ice, mud, loose sand, compliant vegetation, debris hidden in vegetation and annoyances such as small ruts and washboard effects do not appear in geometric representations (Digney, 2001), (Manduchi, 2000). Detection of these features and conditions will rely upon sensors such as colour vision, texture, IR imaging and instrumented bumpers. Moreover, many of these terrain conditions change their cues from region to region, season to season and even hour to hour. For instance, what image characteristics indicate soft ground in one region may, in another region, mean something different or be entirely meaningless. This changing nature of trafficability makes the need for learning control clear. DRDC and its contractors are developing the *Learned Trafficability System* (LTS) that will learn the trafficability characteristics and then adapt as terrain conditions change.

1.3.5 Control of Compliant Legs

Control the many degree of freedom of the ANT's legs and body segments is further complicated by the fact that the wheels form a compliant link with the terrain (Brosinsky, 2001*b*). While at first inspection this may seem like a complication, but on further examination the compliance affords some tolerance to control imperfections and uncertain terrain models. Compliance is also a very important component of biological legs. It allows for the storage of energy during the parts of a gait cycle where energy is abundant and the releasing of that energy during the part of the cycle that energy is in demand. Control algorithms are being developed that allow for energy efficient locomotion.

1.4 Shared Control at DRDC-Suffield

To attain higher autonomy and yet achieve the performance level required by military applications DRDC is developing shared control systems. Shared control is a partnership between the on-board autonomous control system and the remote operator. The autonomy system controls the vehicle during routine operations and the remote operator assumes control during extreme situations. The on-board autonomy system supervises its operation and when it becomes uncertain of its next action it notifies the operator whom assesses the situation and assists the vehicle. The autonomy system then resumes control of the vehicle. Communication and operator attention is kept to a minimum as the autonomy system supervises itself and communicates out only when necessary. Communicated information needs only be sufficient for the operator to determine how to assist the vehicle. For instance, still imagery and state information would be enough to assist a vehicle with a road hazard, but short video clips may be required to understand the intentions of other moving vehicles.

Shared control provides a synergy between operator and autonomous control that utilizes the strengths of both that is well suited to military applications. The benefits and issues involved with telematic, autonomous and shared control of military vehicles are now discussed as well as the research currently pursuing is summarized in the following sections.

1.4.1 Military Benefits and Issues

1. **Hazardous and Hostile Environment:** Clearly the removal of soldiers from harms way while maintaining the full functionality of the vehicle is desirable. Additionally, remote operators and autonomous control are more likely to remain calm and make better decisions when not in immediate danger.

2. **Robot Must Win:** It must be remembered that conflicts are a competition. There is an hostile opponent, who is actively trying to destroy the vehicle and whatever telematic, shared control or autonomous system is fielded it must be able to win. If it cannot prevail, having an unmanned vehicle is of little benefit.

3. **Hiding Complexity:** Unmanned vehicles such as the ANT in Figure 3 are too complex and non-intuitive to be controlled by humans. Autonomous control is responsible for low level control while the human controllers supply high level and intuitive directives.

4. **Amplified Use of Manpower:** If a single human controller can effectively control a greater number of vehicles then that force will have advantages over a force that requires one person or more per vehicle. By amplifying manpower a force would be able to field more vehicles or deploy the freed personal to other vital roles.

5. **Persistent Attention:** Many military operations involve persistent observations that fatigue humans, quickly leading to inattentiveness and errors. In a clear application of shared control the machine would untiringly look for scene changes and then enlist human assistance to classify those changes.

6. **Lethal Force Control:** Current ethical considerations do not permit automatic control of lethal force. Whenever lethal force is to be applied from an unmanned vehicle a human operator must be in direct control.

7. **Life Critical Operations:** In contrast to lethal force situations there are other situations in which friendly forces are in danger from machine errors. Operations such as the infrared image classification used in the ILDP land mine detection vehicle is done by a human because the confidence of the automatic classification is not sufficient to balance risking human lives on a machine error. However other processes such as the fusion algorithms are sufficiently capable and allowed to remain under automatic control.

8. **Sacrificial Vehicles:** A grim reality of armed conflicts is that losses are acceptable and that sometimes vehicles and personal will be sacrificed for the benefit of remaining force. Clearly it would be desirable to sacrifice an unmanned vehicle. It must be understood in mind that the vehicle is not simply just sacrificed but destroyed while attempting some task. It is required that the unmanned vehicle must be at least as proficient at that task as an on board human.

9. **Communication Silence and Jamming:** In practice reliable communication links are difficult to insure. This problem is even getting more difficult as forces move into urban areas, into building and underground installations. It is common for the enemy to jam or otherwise disrupt communications. In covert operations communication will give positions or intent away, so communication is kept to a minimum and performed through undetectable means.

10. **Acceptable Path to Higher Autonomy:** The military community is unwilling to support large leaps in autonomy unless there is an incremental verifiable and demonstrable safe path. In shared control as automation technology matures it can be added and the vehicle gains a little autonomy. Progress is observed by the operator and the autonomy addition proves itself reliable or unreliable. Through such a incremental route levels of autonomy will be accepted that would never be if proposed in a single large step.

1.4.2 Shared Control Research

1. **Active Hazards:** One of the main assumptions of shared control is that the world is sufficiently static and no harm will come to the vehicle while it waits for assistance. This assumption may hold in civilian applications but it is common that military applications are actively hostile. Evasive actions are often required faster than communication rates. The vehicle must now weight its action uncertainty against the uncertainty in receiving timely human assistance. Furthermore the vehicle must also weigh the relative dangers of inaction with the danger of those uncertain actions. This requires that the vehicle acquire some knowledge of the relative dangers of its world and the hostile entities within it.

2. **Reverse Shared Control:** The usual flow of assistance if from the human operator to the autonomy system. The defining concept is that that the machine handles the routine and the human supplies the extra cognitive capabilities when required. While this will hold true in most cases there are cases where the machine can lend higher level assistance to a novice operator. For instance when the learned trafficability system has been trained to predict the trafficablity of a region the LTS can assist a novice on-board operator who is not yet familiar with the region.

3. **Asynchronous Assistance and Multi-tasking:** Given the realities of communication a shared control vehicle may be out of contact for long periods of time and latencies in assistance may be large. Assuming that a vehicle does not experience an actively hostile and fatal situation there are any number of non fatal situations that the vehicle can be in and in need of assistance. Instead of waiting for assistances which can be an arbitrary long period of time, it would be desirable for the vehicle to switch to another task and return to the problem task when assistance arrives. This capability to asynchronously interweave tasks and assistance will minimize time spent inactive while waiting for assistance.

4. **Shared Control of Distributed Robots:** Control of many autonomous robots by a far fewer number of operators (in ratios of 100 robots to 1 operator) presents problems for conventional control interfaces. In the cognitive colonies project swarms of robots autonomously form and dissolve cooperating teams as required. As it is impossible for a single operator to direct at the individual robot level, the colonies control system autonomously directs the individual and robot teams. Current work on cognitive colonies at CMU is developing interfaces that impart the operators intentions to the colony and report back the current colony structure and progress. A shared control system must be able to assist the robots at both the individual and team levels. This requires that teams collectively understand the certainty with which they act and agree on when they are sufficiently uncertain to require assistance. As the tactics are generated by the colony the operator may suggest entirely new tactics or assist the team using its current self generated tactics.

5. **Multiple Operators:** It is expected that during operation vehicles under shared control will be directed to perform tasks from many operators as well as receive assistance from many operators. Prioritizing operator requests and ranking quality of assistance provided will be essential.

1.5 Discussions

The benefits listed in this section make it clear that autonomous vehicles under a shared control system will play a major role in fielding unmanned military vehicles. Many research issues still need to be resolved but the synergistic arrangement of man and machine is sound. Research is required in autonomous control, perception, uncertainty modeling and human factors. Of particular importance is human factors in which the man machine interface will evolve as the the relationship between man and machine becomes more of a partnership and less direct master-slave.

2 Learning Hierarchical Control Structures

2.1 Introduction

The need for hierarchical structures within learning control systems is clear. Without some form of hierarchy, a learning system would be bogged down in the innumerable details of the lowest level of control. Many researchers have recognized this and have imposed handcrafted hierarchies. Although the imposed hierarchies result in more tractable problems, they also impose the designers preconceived notions on the control system. It is desirable that the control system able to generate its own hierarchical structure. Work on Nested Q-learning (NQL) (Digney, 1996*a*) has shown that a hierarchical structure can be learned in a reinforcement learning environment. Although this method generated hierarchical structures, it was considered to be seriously handicapped by the need to classify every distinct sensory state as a feature. With each distinct feature becoming the termination point for a new behavior the control system quickly became overwhelmed with choices of behaviors, of which the vase majority were irrelevant. In this paper, extensions able to remove that handicap are introduced. The solution is to have only useful features emerge. To do this two emergence criteria are introduced; high frequency of state occurrence and high reinforcement gradients. As only features with a high probability of relevance emerge learning will be much faster. A hierarchical learning control system is developed and tested against a non-hierarchical learning system.

2.2 Background

An issue closely linked to robot learning is the architectures in which the control strategies are implemented (Tyrrel, 1992). The two main ideologies are flat and hierarchical. Figure 6 shows the flat and the hierarchical architectures schematically. Flat architectures have direct connections from sensors to actions, through a single level of control. When some situation is perceived all behaviors compete for control with the strongest response winning. Hierarchical architectures have an indirect coupling of perceptions to actions through a hierarchical control structure. When a situation is perceived a high level behavior becomes active and issues commands to one or more lower level behavior(s). In turn, these lower level behaviors control yet lower level behaviors until primitive actions are activated and some physical action(s) is performed. Both architectures have benefits and drawbacks and have been used to implement learning techniques. One of the major benefits of hierarchical structures in robot learning is that learning difficult tasks can be made more tractable by clever design at the hierarchical structure. By abstracting away many details of a complex world to lower level behaviors, learning can more easily be implemented in the higher levels.

Many researchers have recognized the need for hierarchical structures in learning control systems (Maes and Brooks, 1990) (Long-Ji, 1993) (Dayan and Hinton, 1993) (Singh, 1992). These approaches are all similar in the respect that the structures are hand designed and individual components are learned or vise versa. Previous work in Nested Q-learning (NQL) (Digney, 1996*a*) demonstrated the autonomous construction of a control hierarchy able to learn both the structure and the behaviors using a reinforcement learning technique based on Q-learning (Barto A.G. and C.H, 1989). When the agent initially started out, no information about structure or useful behavior was given to it. The control system discovered distinct recognizable features in its world and learned the relationship between different features, its environment and its tasks. These relationships formed behaviors which were usually a hierarchical assembly of other behaviors and primitive actions. The structure that emerged was of as many levels as the task required. Using NQL, the control system also learned bottom-up reactive/opportunistic functions for each behavior which served to 1) provide a high level command source at the top of the hierarchy and 2) facilitate the invocation of previously learned beneficial behaviors in new situations without the necessity of relearning them.

One of the simplifying assumptions made during the initial work on NQL was that all perceivable states in the environment were valid termination points for possible behaviors. Useful behaviors

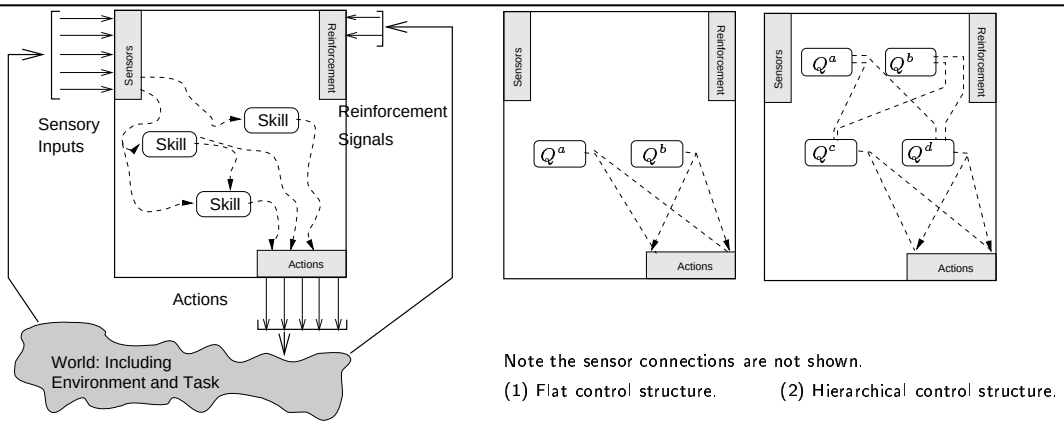


Figure 6: Schematic of control architectures: (a) sensory, action and reinforcement signal configuration for a learning control system and (b) control architectures: (1) flat structure and (2) hierarchical structure. Note: $Q^?$ represents behaviors and the sensory connections in (b) have been removed for clarity.

would then emerge out of this pool of candidate behaviors. This assumption works for only very small problem spaces. When the robot is faced with problems of any realistic size, the pool of candidate behaviors grows too large very quickly. In this paper, a method for removing this assumption and allowing only behaviors with a high probability of relevance to emerge, leaving all other behaviors undiscovered is presented.

2.3 Emergent Behaviors in Nested Q-Learning

Consider an animat placed in some world and expected to perform some task defined by delayed rewards. As the animat moves about in its world, it experiences reinforcements from various sources. High negative reinforcements result from damaging actions, nominally negative reinforcements represent energy expended and positive reinforcement signal represent successful performance of some task(s). It is also able to perceives changes in its state as it moves around. A typical reinforcement learning problem is to learn the state, x, action, u, evaluation function $Q(x,u)$ such that the total positive reinforcement received by the animat is maximized. This results in a single monolithic evaluation function $Q(x,u)$ that, although able to solve the current problem, the information gained, albeit likely gained at less cost, will be useless in new tasks.

Nested Q-learning decomposes the monolithic evaluation function into a hierarchical structure of smaller evaluation functions, each representing a behavior which can later be invoked and reused in a new task or environment. To achieve this, the NQL algorithm allows the selection of behaviors in addition to primitive actions. These behaviors are defined as learned sequences that bring about some distinct termination state. Originally, these states were defined as any distinct state. This was a severely limiting and unnatural assumption. In this paper, only behaviors that prove useful emerge while all other possible behaviors remain undiscovered. These emergent behaviors will be selected based on two criteria. First, high changes in reinforcement gradient and secondly, a high state occurrence rate or frequency of visits to a particular state.

As the animat moves around, it perceives itself in various states and entering most of these states results in nothing more than normally occurring reinforcement signals. However, some states will represent dramatic changes in reinforcement. It is reasonable to assume that high reinforcement signal gradients might represent a non-typical and likely useful location in the state space. The high reinforcement signal gradients will be used to differentiate useful features from irrelevant ones. The animat's state space, as perceived by sensors will be discrete and represented by x, with

$x \epsilon \left\{ 0, \quad 1, \quad \cdots \quad x \quad \cdots \quad x_{max} \right\}$ where x_{max} is the maximum number of distinct states currently existing but can increase as new states are discovered and x is the animat's current state. As the agent arrives at state x it experiences some total reinforcement signal, R_{TOTAL}. This reinforcement signal can be high negative in situations such as collisions with obstacles, but is usually some nominal value indicating the energy expended in reaching the current state. In addition to driving the adaptive mechanism, this signal can be used to learn a squared external reinforcement signal gradient mapping function $m(x)$ which learns the areas of non-typical reinforcement in the state space. This mapping will be used to generate potentially useful features in the state space and is determined as follows:

$$e_m = m(x) - \left(\frac{\partial R_{TOTAL}}{\partial t} \right)^2 \tag{1}$$

or

$$e_m = m(x) - \left(\frac{\Delta R_{TOTAL}}{\Delta t} \right)^2 \tag{2}$$

and

$$m(x) \longleftarrow m(x) - \eta e_m \tag{3}$$

where e_B is the environment, x is the current state, η is the learning rate and $\longleftarrow$ represents learning by some type of incremental parametric storage device such as a neural network. High changes in the gradient of the reinforcement signal during the transition from state to state, learned as a high value in $m(x)$, indicates a non-typical location in state space and a potentially useful feature. Eventually, as $m(x)$ converges, potentially useful features emerge using a simple threshold operator

$$x = \begin{cases} \text{feature} & \text{if } m(x) > T_1 \\ \text{non-feature} & \text{otherwise.} \end{cases} \tag{4}$$

where T_1 is the minimal squared external reinforcement gradient for a state to be considered as a feature.

Another criteria for useful behavior emergence is the occurrence or frequency of visits to particular state space locations. During operation, important states will be visited more often than less important ones. However, the animat must already be successfully performing one or more tasks for these features to emerge. At first, the reason for doing this might seem a bit unclear. Why generate new behaviors once the animat is operating successfully? The reason is that features useful in one or more tasks are likely to be useful in other, yet to be learned tasks. By learning the decomposition of the currently well learned tasks, portions of what has been learned can be used in new tasks. To learn this decomposition, a occurrence function $c(x)$ is defined. The strength of this function is increased by some small amount upon each visit to each state,

$$c(x) \longleftarrow c(x) + \epsilon \tag{5}$$

where ϵ is some incremental amount. This function is also periodically decayed by some decay factor,

$$c(x) \longleftarrow \lambda c(x) \text{for all } x \tag{6}$$

where λ is the decay factor, $0 < \lambda < 1$ and x denotes all currently discovered states. The result is the emergence of states that are more commonly visited or high occurrence. These commonly visited states can then be extracted as potentially useful features and used to generate behaviors. For the animat to extract such useful features from the occurred function $c(x)$, a threshold function similar to Equation 4 is used

$$x = \begin{cases} \text{feature} & \text{if } c(x) > T_2 \\ \text{non-feature} & \text{otherwise.} \end{cases} \tag{7}$$

where T_2 is the criteria for determining whether or not a state is visited often enough to warrant it emerging as a feature. When a single task is being performed, key states from the state trajectories

are extracted. That is, the key states of the action state sequence which must be visited for successful outcomes. When more than one task is being performed, these equations decompose the action sequences of the tasks into sub-trajectories (sub-behaviors) that the tasks have in common.

Now that the two criteria for useful feature emergence have been developed, the extension to NQL can be discussed. The core of the NQL algorithm remains very much as previously described (Digney, 1996*a*). However, instead of having all possible behaviors discovered and available for learning and use very soon after the animat begins operation, features emerge and are learned and used continuously throughout the life of the animat. This continual emergence of features causes added difficulties with learning and must be taken into consideration. This is because behaviors that emerge early will not be able to benefit from behaviors that emerge later. Again, consider an animat initially placed within a world without any form of previous experience or knowledge of its environment or task. The animat has some primitive actions, a, from which it can choose to physically act within its world. These are $a \in \left\{ a_0, \quad a_1, \quad \cdots \quad a_j \quad \cdots \quad a_J \right\}$ where a_j is a primitive action and J is the total number of primitive actions. Initially, without any behaviors having emerged, the possible actions available to the animat, u, are only primitive actions, $u \in \left\{ Q^{a_0}, \quad Q^{a_1}, \quad \cdots \quad Q^{a_j} \quad \cdots \quad Q^{a_J} \right\}$ where Q^{a_j} is a non-adaptive behavior representing the performance of primitive action a_j and should not be confused with the adaptive behaviors Q^{f_i} and u is the action to be taken by the animat. As the agent gains experience within its task and environment, potentially useful features will emerge according to the previously described criteria. These features, f_i, are labeled $f \in \left\{ f_0, \quad f_1, \quad \cdots \quad f_i \quad \cdots \quad f_I \right\}$ where f_i is a discovered feature and f_I is currently the last discovered feature. The total number of action (both primitive actions and behaviors) that may be taken by the agent now includes both the primitive actions and the emergent behaviors, $u \in \left\{ Q^{a_0}, \quad \cdots \quad Q^{a_j} \quad \cdots \quad Q^{a_J}, \quad Q^{f_0}, \quad \cdots \quad Q^{f_i}, \quad \cdots \quad Q^{f_I} \right\}$. The total number of possible actions is the sum of all primitive actions and all currently possible behaviors, $u_{total} = J + I$. The state of the agent is established by the state of all the incoming sensors and is represented by a discrete value x_l which ranges from x_1 through x_{max}. The evaluation function for each feature is a function of the robot's current state and all possible actions, $Q^{f_i} = f(x, u)$. The evaluation function for each behavior now becomes, $Q^{f_i} = f(x_1, \cdots x_{max}, Q^{a_1} \cdots Q^{a_J}, Q^{f_1}, \cdots Q^{f_I})$, where both the number of states, x_{max}, and the number of behaviors, f_I, are open ended and subject to initial discovery and then to increases or decreases due to ongoing changes. It is seen that the learning algorithm described above becomes nested and possibly recursive. That is, the evaluation function, $Q^{f_{desired}}$, can invoke other behaviors including itself while attempting to reach the feature, $f_{desired}$. It is this nested nature that will allow hierarchical control structures to emerge.

As the agent interacts with the environment it receives an external reinforcement signal(s), r_{EXT}. It is through this signal that the agent is driven to perform tasks of external benefit. The reinforcement signal is defined as

$$r_{EXT} = \begin{cases} 0 & \text{if external task is achieved} \\ -R_{EXT} & \text{otherwise} \end{cases} \tag{8}$$

where r_{EXT} is an external reinforcement signal and R_{EXT} is a positive constant. In addition, there are various internal reinforcement signals, r_{INT}. These drive the agent to perform tasks of internal benefit such as *avoid danger* and *find fuel*. In the nested Q-learning algorithm there is also a reinforcement signal that effects only the currently active behavior(s). The reinforcement signal drives the action of the agent to reach the desired feature

$$r_{FEAT} = \begin{cases} 0 & \text{if } x = f_{desired} \\ -R_{FEAT} & \text{if } x \neq f_{desired} \end{cases} \tag{9}$$

where r_{FEAT} is the feature's reinforcement signal and R_{FEAT} is a positive constant. Upon performing a particular selected action, u^*, be it a primitive action or a behavior, the robot advances from state x_v to the next state x_w and incurs a total reinforcement signal, r_{TOTAL}. Included in this total reinforcement signal is the cost of performing the selected action. This cost includes the physical cost

of performing the action and possibly the mental (computational) cost of choosing the action. These costs are designated as r_{LOW}, with

$$r_{LOW} = \begin{cases} -C & \text{if } u^* \text{ is a primitive action} \\ \sum_{k=0}^{K} r_{TOTAL}^{u*}(k) & \text{if } u^* \text{ is an adaptive behavior} \end{cases} \tag{10}$$

where $\sum_{k=0}^{K} r_{TOTAL}^{u*}(k)$ is the total reinforcement signal from the invoked behavior summed over the number of steps, K, required to perform the behavior, u^*, and C is a constant that reflects the cost of performing the primitive action. Of course, if many lower levels of behaviors are employed, this process becomes nested at r_{LOW} which represents the experiences (physical and mental) of all lower levels. The total reinforcement signal for the invoking behavior becomes

$$r_{TOTAL} = r_{EXT} + r_{INT} + r_{FEAT} + r_{LOW} \tag{11}$$

where r_{TOTAL} is the sum of all reinforcement signal sources. This total reinforcement is used to construct the Q functions of expected reinforcement, from which useful top-down control strategies will emerge. The error, e_Q, is defined to be,

$$e_Q = \gamma \cdot \max_u \{Q_{x_w,u}\} - Q_{x_v,u^*} + r_{TOTAL} \tag{12}$$

where γ is the temporal discount factor $0 < \gamma < 1$ and $\max_u\{Q_{x_w,u}\}$ is the current prediction of the maximum total future reinforcement remaining when the agent leaves state x_w. This error is used to adapt the evaluation functions,

$$Q_{x_v,u=u^*}(k+1) = Q_{x_v,u=u^*}(k) + \eta_Q \cdot e_Q \tag{13}$$

$$Q_{x_v,u\neq u^*}(k+1) = Q_{x_v,u\neq u^*}(k) \tag{14}$$

where η_Q is the rate of adaptation and k is the index of adaptation.

Features may emerge at any time during operation and their corresponding behaviors are added to the evolving behavior pool. Incorporating these newly introduced behaviors into the action selection mechanism is required. This makes action selection more difficult than in the action selection mechanism described in previous work (Digney, 1996*a*). For the top-down goal directed action/behavior selection, a random based exploration policy is used. Although this is not the most effective exploration policy, it is easily implemented and other more efficient forms of exploration have been studied in depth elsewhere (Thurn, 1992). For the currently invoked behavior, $Q^{invoked}$, action/behavior selection is determined according to

$$u^* = \begin{cases} argmax_u\{Q^{ALL} + E_{invoked}\} & \text{if } Q^{invoked} = Q^{f_i}(\text{a behavior}) \\ a_j & \text{if } Q^{invoked} = Q^{a_j}(\text{a primitive action}) \end{cases} \tag{15}$$

where argmax_u is a maximum function taken over all primitive actions and existing behaviors, $E_{invoked}$ is the exploration policy and Q^{ALL} is all possible choices including both other behaviors and primitive actions. Note, that there can be more that one behavior invoked at any time. In fact, it is usual that many behaviors will be active at the same time in response to either top-down or bottom-up directives. The nested nature is seen again in Equation 15 where the currently active behavior is able to select another behavior or a primitive action. If another behavior is selected that behavior may also go on and select yet another behavior and so on. If a primitive action is selected, Q^{a_j}, then the physical actuator action a_j is performed. The exploration policy for behavior $Q^{invoked}$ that is discovered at time d_i is

$$E_{inv} = RAND(k_{inv}\delta(t^{inv} - d_{dis}^{inv})e^{-\tau(t^{inv}-d_{dis}^{inv})} + k_{other}\sum_{i=0,\ i\neq inv}^{I}\delta(t^i - d_{dis}^i)e^{-\tau(t^i-d_{dis}^i)} \tag{16}$$

where t_i is the behavior local time measured relative to each behavior (and is effectively proportional to the number of times each behavior has been invoked), d_{dis}^{inv} is the time of discovery for the invoked behavior, d_{dis}^{i} is the discovery time for behavior, i, and t^{inv} and t^{i} are the behavior local time for the invoked, $Q^{invoked}$, and the other behaviors Q^{i}. The time constants, τ, control the rate at which the exploration contribution to action selection is reduced. The constants k_{inv} and k_{other} controls how much exploration is due to the invoked behavior and how much is due to other behaviors that may have been discovered later. The $\delta(w)$ is the Dirac delta function,

$$\delta(w) = \begin{cases} 1 & \text{if } w < 0. \\ o & \text{if } w > 0. \end{cases} \tag{17}$$

The first term directly following k_{inv} in Equation 16 represents the contribution to exploration due to the discovery of the currently invoked behavior itself. That is, the behavior will have an initially high exploration component, decaying toward pure exploitation. The second term of Equation 16 represents the contribution to exploration due to the discovery of other behaviors. If this second term were zero then behaviors that are discovered after earlier behaviors have converged would never be tried by those converged behaviors. Without partially refreshing the exploration component of the earlier behaviors, the earlier learned behaviors may never learn to benefit from the new behaviors as they emerge.

The preceding derivation describes a NQL technique through which a top-down action selection mechanism will generate a hierarchical control structure and cascade goal seeking commands downward through the structure as behaviors are discovered continuously throughout the life of the animat. Each behavior, whenever invoked, will in turn invoke other behaviors and/or primitive actions in an attempt to fulfill the desired goals of higher behaviors. The bottom-up or sensory based action selection mechanism and how it fits with the top-down action selection to result in a flexible reactive control hierarchy is described elsewhere (Digney, 1996*a*) and is not explored in this paper.

2.4 Simulation

To evaluate the NQL algorithm and it's capabilities, (the emergent features and the subsequent learning of a hierarchical structure of behaviors), an animat and the two dimensional environment of Figure 7 were used. The animat was placed in a grid world 6 by 10 units in size as shown in Figure 7(b). This grid world was bounded by an impassable barrier and additional barriers could be erected within the world in any configuration desired. Different from related work (Digney, 1996*a*) in which the animat had a number of different sensors, the animat in this study was only capable of perceiving its location within the grid world. In this study, the animat's primitive actions were capable of moving it *left*, *right*, *up*, *down* or *no motion* shown in Figure 7(a). These actions were non-deterministic and when invoked they only achieved their desired outcome with a probability of 0.75. Otherwise another action was taken randomly. Figure 7(b) also shows the initial configuration of the grid world. The animat always started from the indicated starting region located on the left most side. There were three goal locations labelled A, B and C, placed as indicated in Figure 7(b). Although their positions are indicated, the goals were not necessarily always active or visible to the animat. The goals were activated or made visible to the animat as desired to test the capabilities of the control system. An internal barrier was placed as shown in Figure 7(b), forcing the animat to pass through a narrow passage en route to the goals. In previous work, the animat learned to associate various goals with their appropriate external stimuli (i.e. blue overhead light caused animat to move toward blue floor panel). In this paper, such associations were not studied and the animat was instructed to seek all goals. This was implemented as a random selection over all goals that had been discovered up to that time.

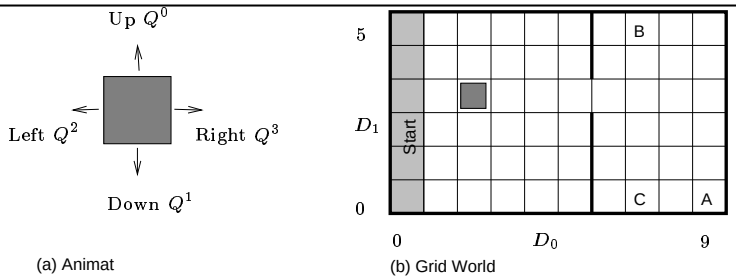


Figure 7: The animat and its gridworld. Note the five primitive actions (up down, left, right and stay), the start locations, the initial barrier and the location of the three possible goals.

2.4.1 Emergent Features, Behaviors and Structures

The world was initially configured with barriers as shown in Figure 7(b). Two goals, A and B, were activated and goal C was left undetectable. The control system initially had no behaviors available to it as shown in Figure 8(a) and its actions were thoroughly random and undirected. This can be contrasted to previous work (Digney, 1996*a*) in which the control system was quickly flooded with potential useful behaviors and then had to sort out the useful from the many irrelevant. Once placed in its world, the animat randomly moved about undirected. After a short period of time it learned that a high reinforcement signal gradient occurred near active goal locations (the animat first discovered goal A before goal by chance). This constituted the discovery of a feature, f_0, and the behavior Q^5 emerged. The subsequent expansion of the control system at the emergence of Q^5 is shown in Figure 8(b). Next, the second goal location (goal B) feature, f_1, was discovered and the control system expanded to include behavior Q^6 as shown in Figure 8(c).

Note that at this point only two goals are currently active and two behaviors in existence. These have resulted from two reinforcement gradient based features emerging from $m(x)$. As the animat performed these two behaviors, two other features, based on high occurrence peaks in function $c(x)$ were discovered: f_2 at spatial location $D_0 = 5$ and $D_1 = 3$ and f_3 at $D_0 = 6$ and $D_1 = 3$. They emerged as the locations adjacent to the mouth of the barrier; clearly important locations in the state space. The hierarchical control system expanded to include behaviors Q^7 and Q^8, as shown in Figure 8(d). The two existing behaviors quickly integrated the new behaviors into their action sequences. The hierarchical structure of the control system is further shown in the new action sequence when behavior behavior Q^6 was activated is shown in Table 2 and Figure 9. This shows that the sub-behaviors, Q^7 and Q^8 are evident in the state-action-behavior trajectories of the higher level behaviors Q^5 and Q^6 (not shown). For convenience, the behaviors, features and locations discussed are summarized in Table 1.

At time ≈ 150, the third goal (goal C) was activated and made detectable to the animat. Shortly thereafter, its squared reinforcement gradient was learned and another feature, f_4 emerged at location $D_0 = 9$ and $D_1 = 0$. The control system expanded to incorporate the new behavior, Q^9 as shown in Figure 8(e). From the behavior sequence of Table 3 and Figure 10, it is seen that behavior Q^9 quickly exploited the existing behaviors Q^5, Q^6 and to a greater extent Q^7 and Q^8. The performance of the hierarchical control system and its component behaviors are shown in Figure 13(b). The primitive actions have a constant performance and are not shown. The events previously described are shown between time $= 0$ and time ≈ 150. For graphing purposes, any undiscovered behaviors were given a dummy performance value of -200 until their time of discovery. From this plot, it is seen that the learning of behavior Q^9 was considerably quicker than the learning of the two previous reinforcement gradient based features, Q^5 and Q^6. This new behavior, Q^9, was learned more quickly because it was

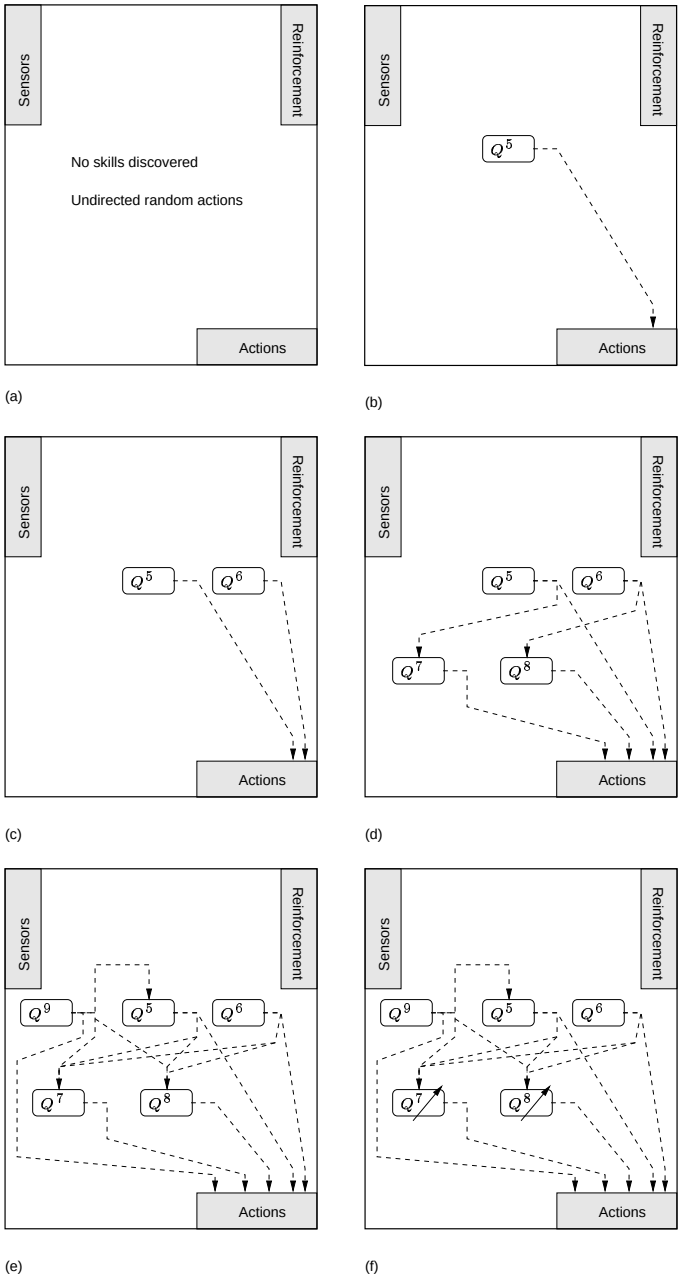


Figure 8: Emergent behaviors and structures.

Feature	Behavior	Location		Discovery	Comments
		D_0	D_1	Time	
NA	Q^0	NA	NA	NA	Primitive action
NA	Q^1	NA	NA	NA	Primitive action
NA	Q^2	NA	NA	NA	Primitive action
NA	Q^3	NA	NA	NA	Primitive action
NA	Q^4	NA	NA	NA	Primitive action
f_0	Q^5	9	0	≈ 50	First feature
f_1	Q^6	7	5	≈ 50	Second feature
f_2	Q^7	5	3	≈ 100	First feature
f_3	Q^8	6	3	≈ 110	Second feature
f_4	Q^9	7	0	≈ 150	Third feature

Table 1: Behaviors, Features and Locations for the Hierarchical Control System.

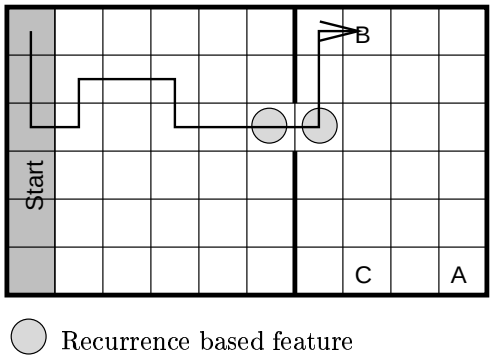


Figure 9: Animats's path for behavior Q^6.

Depth	Behavior	D0	D1	Comments
1	$- > Q^6$	0	5	Start
2	$-- > Q^8$	0	5	
3	$--- > Q^1$	0	5	
3	$--- > Q^3$	0	4	
3	$--- > Q^3$	0	3	
3	$--- > Q^0$	1	3	
3	$--- > Q^7$	1	4	
4	$---- > Q^3$	1	4	
4	$---- > Q^3$	1	4	
4	$---- > Q^3$	2	4	
4	$---- > Q^1$	3	4	
4	$---- > Q^3$	3	3	
4	$---- > Q^3$	4	3	
3	$--- > Q^3$	5	3	
2	$-- > Q^0$	6	3	
2	$-- > Q^3$	6	4	
2	$-- > Q^0$	7	4	At goal

Table 2: Behavior Q^6 before the new barrier. See Figure 9 for path. Note the arrow length represents the depth into the structure.

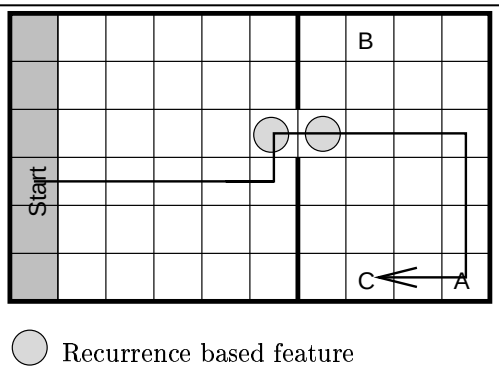


Figure 10: Animats's path for behavior Q^9.

Depth	Behavior	D0	D1	Comments
1	$-> Q^9$	0	2	Start
2	$--> Q^3$	0	2	
2	$--> Q^7$	1	2	
3	$---> Q^3$	1	2	
3	$---> Q^3$	2	2	
3	$---> Q^3$	3	2	
3	$---> Q^3$	4	2	
3	$---> Q^0$	5	2	
3	$---> Q^3$	4	2	
3	$---> Q^3$	4	2	
3	$---> Q^0$	5	2	
2	$--> Q^5$	5	3	
3	$---> Q^3$	5	3	
3	$---> Q^3$	6	3	
3	$---> Q^3$	7	3	
3	$---> Q^3$	8	3	
3	$---> Q^1$	9	3	
3	$---> Q^1$	9	2	
3	$---> Q^1$	9	1	
2	$--> Q^2$	9	0	
2	$--> Q^2$	8	0	At goal

Table 3: Behavior Q^9 before the new barrier. See Figure 10 for path. Note the arrow length represents the depth into the structure.

able to exploit the previously learned behaviors, Q^7 and Q^8, and to a lesser extent Q^5 and Q^6. The initial effort expended in learning to decompose the two behaviors Q^5 and Q^6 into Q^7 and Q^8 paid off by speeding the learning of the new behavior Q^9.

2.4.2 Recovery From Changes

Next the capabilities of the Nested Q-learning technique to adapt to changes were evaluated. After the third goal was introduced an additional barrier was introduced at and the animat was required to recover. At time ≈ 200, the configuration of the world was changed to the configuration shown in Figure 11. This configuration was almost identical to the initial one, except for the addition of a new barrier at $D_0 = 2$. This barrier effectively forced the animat to move to the lower edge of its world when traveling from the start location toward the goal locations on the right side. Behavior performance as displayed in Figure 13 (b) shows that after the introduction of the new barrier, the

animat recovered and was able to relearn its control strategies.

From the resulting state-action-behavior trajectories for the relearned behavior Q^6, it is seen that most of the adaptation was confined to the lower level behaviors. This trajectory is shown in Figure 12 and Table 5. Figure 8(f) shows the final hierarchical structure with the lower levels adapted to the new barrier.

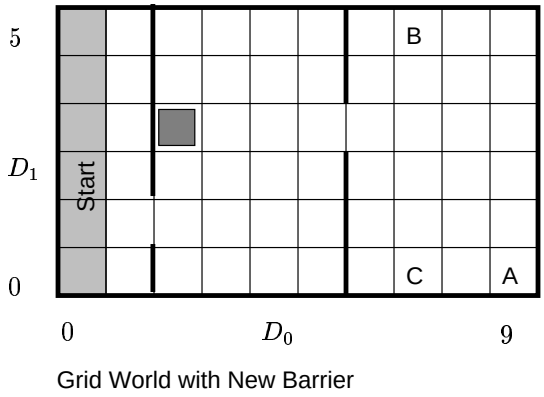


Figure 11: Animat's gridworld with new barrier introduced.

Table 4: Behaviors, Features and Locations for the Non-Hierarchical Control System.

Feature	Behavior	Location		Discovery	Comments
		D_0	D_1	Time	
NA	Q^0	NA	NA	NA	Primitive action
NA	Q^1	NA	NA	NA	Primitive action
NA	Q^2	NA	NA	NA	Primitive action
NA	Q^3	NA	NA	NA	Primitive action
NA	Q^4	NA	NA	NA	Primitive action
f_0	Q^5	9	0	≈ 50	First feature
f_1	Q^6	7	5	≈ 50	Second feature
f_2	Q^7	7	0	≈ 150	Third feature

2.4.3 Comparison with Non-hierarchical Control Systems

An obvious criticism of this approach is that the complexities added by introducing intermediate behaviors in addition to the primitive actions will increase the number of choices available to the animat, thus making the learning more difficult. For a single task this is indeed correct, but it is the improvement in the animat's ability to learn many tasks over its lifetime that is argued to be worth the poorer performance at learning a single, isolated task. To compare the NQL techniques of emergent hierarchical control with non-hierarchical control, an identical set of tasks was learned by a non-hierarchical control system. The non-hierarchical control system is presented with the same animat/gridworld/barrier/multi-task scenario as was described and presented to the hierarchical control system in the previous sections.

The non-hierarchical control system had identical learning parameters to the hierarchical system, except that it was not allowed to generate hierarchies. In fact, the identical simulation code was used, with the hierarchy generating capabilities disabled. The non-hierarchical control system was presented with an identical sequence of events. That is, initially only two goals were detectable, then, later, a third goal and then a new barricade were introduced. The performance of all these tasks, for

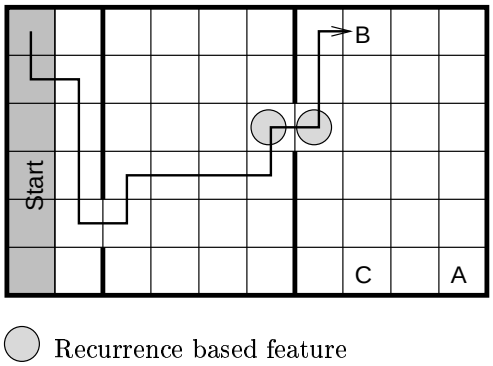
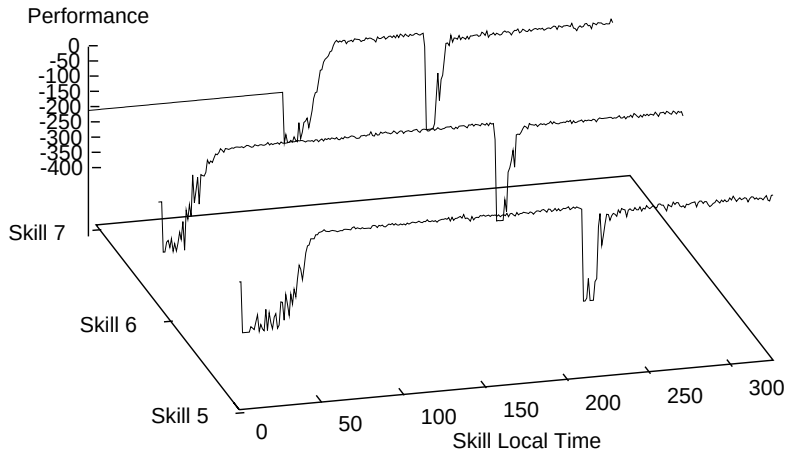


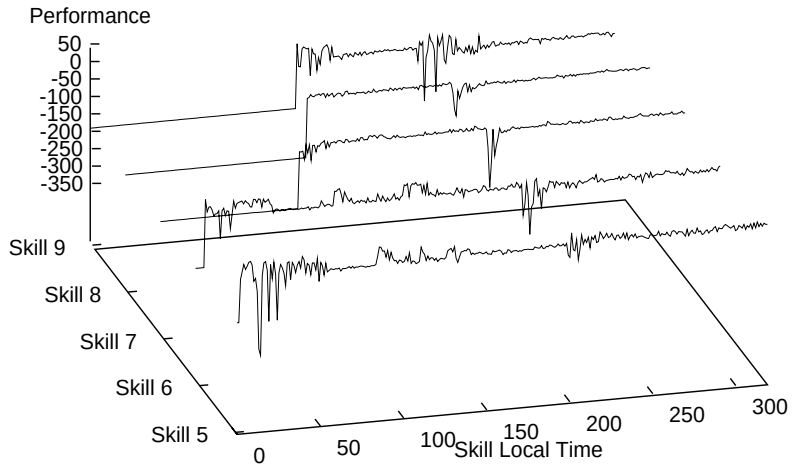
Figure 12: Animats's path for behavior Q^6 after new barrier.

Depth	Behavior	D0	D1	Comments
1	$- > Q^6$	0	5	Start
2	$-- > Q^8$	0	5	
3	$--- > Q^1$	0	5	
3	$--- > Q^1$	0	4	
3	$--- > Q^1$	1	4	
3	$--- > Q^1$	1	3	
3	$--- > Q^1$	1	2	
3	$--- > Q^7$	1	1	
4	$---- > Q^3$	1	1	
4	$---- > Q^0$	2	1	
4	$---- > Q^3$	2	2	
4	$---- > Q^3$	3	2	
4	$---- > Q^3$	4	2	
4	$---- > Q^0$	5	2	
3	$--- > Q^3$	5	3	
2	$-- > Q^0$	6	3	
2	$-- > Q^3$	6	4	
2	$-- > Q^3$	6	5	At goal

Table 5: Behavior Q^6 after the new barrier. See Figure 12 for path. Note the arrow length represents the depth into the structure.

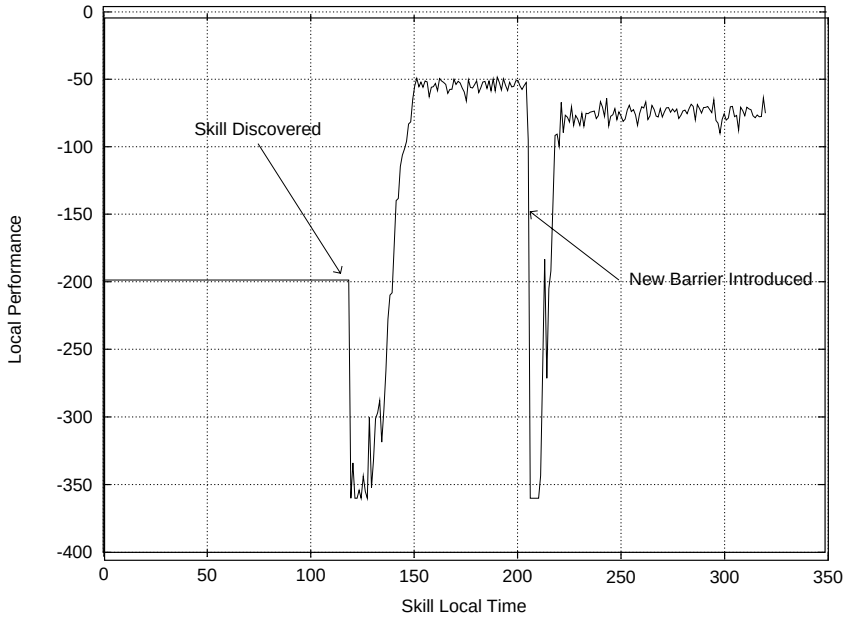


(a) Non-Hierarchical Structure

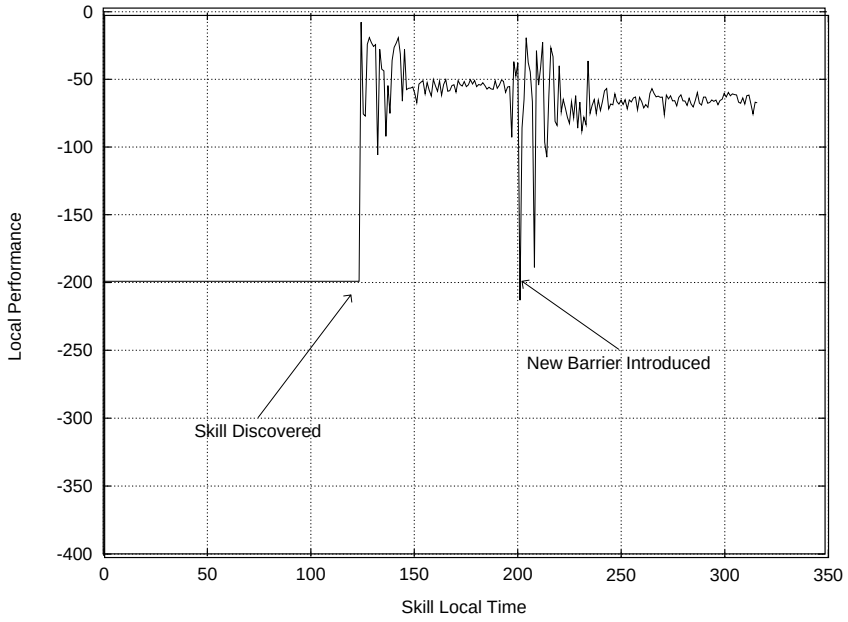


(b) Hierarchical Structure

Figure 13: Comparison of Performance: (a) Non-Hierarchical (b) Hierarchical.



(a) Non-Hierarchical Structure



(b) Hierarchical Structure

Figure 14: Performance of a new behavior for (a)non-hierarchical(Q^7) and (b)hierarchical (Q^9) control systems. Note that the time of discovery and barrier addition are indicated.

both the non-hierarchical and hierarchical control systems are shown in Figure 13. It was seen that both hierarchical and non-hierarchical control systems performed poorly at learning their first tasks, namely Q^5 and Q^6. However, at a later time when a new task was introduced, the non-hierarchical system had to learn it from scratch whereas the hierarchical system could use previously learned behaviors whenever beneficial. Similarly, whenever changes occurred (e.g. the new barrier) the non-hierarchical control system had to adapt each behavior individually while in the hierarchical adaptation was confined to the the elemental behaviors effected and minimized the effect on the other behaviors. The performance at the third task (locating goal C) for the hierarchical and non-hierarchical control systems is shown in Figure 14(a) and (b) respectively. Note that the third task in the hierarchical control system is Q^9 and in the non-hierarchical control system it is Q^7. This encapsulation of control strategies into behaviors allowed the hierarchical control system to clearly out-perform the non-hierarchical control system, especially when new tasks were introduced and changes in the world occurred.

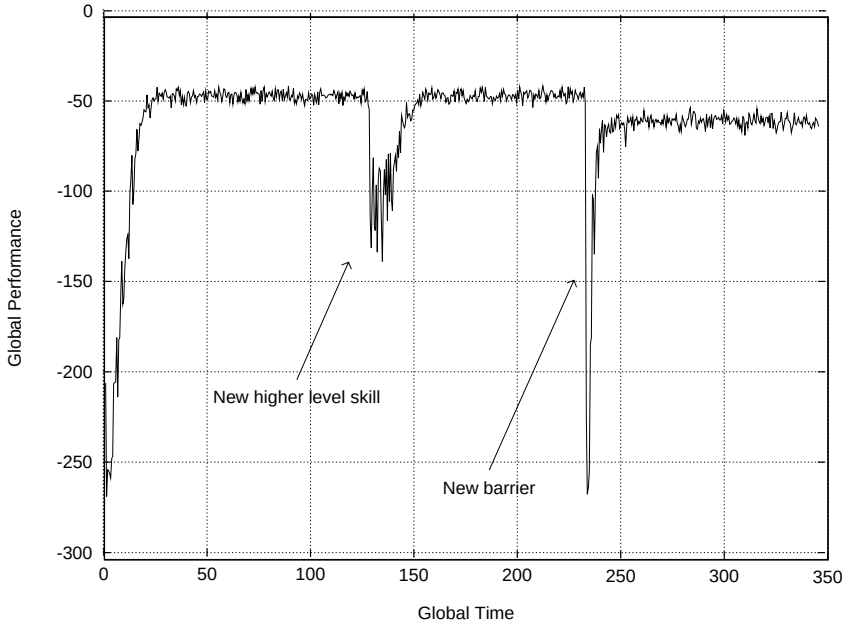
To further quantify this improvement, a measure of the overall performance was introduced. This measure, is the usage weighted average performance at any given time of all the behaviors that currently existed within both the non-hierarchical and hierarchical control systems and is plotted in Figure 15(a) and (b), respectively. The rate of learning is similar on the first task, but as new tasks and changes are introduced, the non-hierarchical control system is clearly out-performed by the self-generating hierarchical control system.

2.5 Discussion

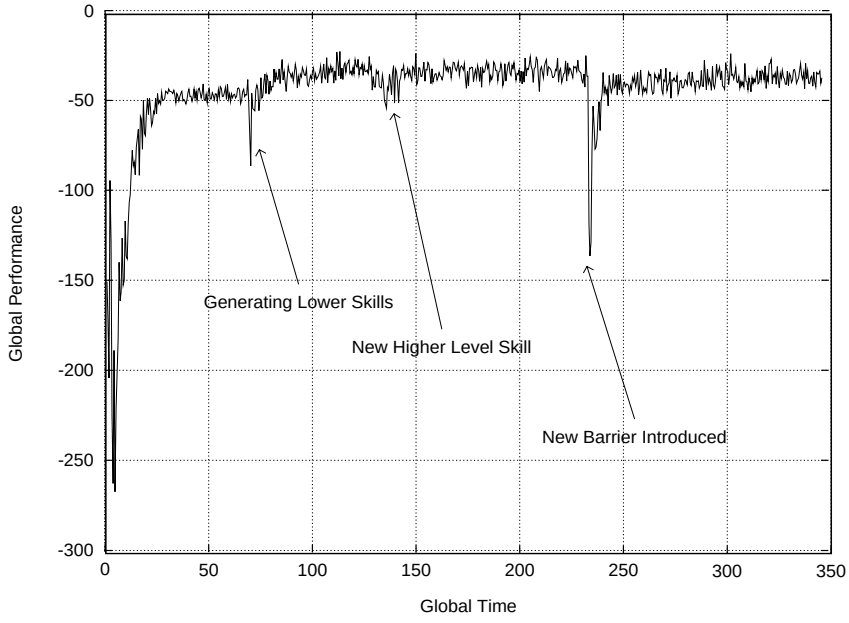
In the preceding simulation studies two criteria were used to discover useful features. The squared external reinforcement signal gradient and the recurrence of states. The squared external reinforcement gradient criteria led to the discovery of goal locations within the state space. Although not explicitly simulated in this paper, had there existed areas of high negative reinforcement signals, these too would have been discovered as useful features. Although they would not be directly useful they would be indirectly practical, representing areas of the state space to be avoided (behaviors that are learned to **NOT** be active). The occurrence criteria led to the discovery of important locations of the state space that were used during task performance. These common locations became the point for decomposition of the tasks into smaller sub-tasks (or sub-behaviors).

By decomposing its behaviors into smaller sub-behaviors, the animat's control system demonstrated a capability to reuse existing knowledge and improve its performance at learning new but related tasks. This was evident when a new behavior (Q^9) was requested once the control system had learned the first two tasks (Q^5 and Q^6) and decomposed them into their elemental components (behaviors Q^7 and Q^8). As the animat began to learn behavior Q^9, it invoked behavior Q^7 to move the animat to the mouth of the barrier in one step. At this time behavior Q^7 was operating very well and this allowed a single command from the infant behavior Q^9 to move all the way from the barrier mouth. Behavior Q^5 was then invoked to bring the animat to within two steps of its goal. The invocation of Q^5 resulted in a somewhat inefficient overall path to the goal, but when one thinks about this, it is how animals and humans often behave. They often prefer to use established behaviors and routines at the expense of efficiency.

Another benefit of the decomposition of behaviors into a hierarchical structure is the resistance to change. High level behaviors and any new behaviors that are to be learned are ultimately composed of some number of lower level behaviors. When changes occur will often be confined to the lower level behaviors that are affected and the higher level behaviors will remain intact. In these simulations, when a new barrier was placed in between the start location and the initial barrier, most of the disruption was confined to the behaviors that moved the animat from the start to the initial barrier's mouth, namely behaviors Q^7 and Q^8. Some disturbance was shown in all behaviors, but this was to be expected because lower level behaviors during their period of recovery effected the higher level behaviors by briefly feeding back higher negative reinforcements. Once the brief disturbance settled,



(a) Non-Hierarchical Structure



(b) Hierarchical Structure

Figure 15: Overall behavior performance for (a)non-hierarchical and (b) hierarchical control systems.

the lower level behavior had learned how to navigate through the additional barrier and the higher level behaviors had remained the same. If the simulation were to be continued, additional recurrence based features would be discovered near the doorway of the new barrier.

To compare the benefits of the self-generating hierarchical structure with non-hierarchical control systems, an identical animat was used in an identical situation. From the results, it is seen that the non-hierarchical structure was forced to learn everything from scratch whenever a new task was attempted. Also, when changes occurred, each behavior had to be adapted in its entirety, resulting in much poorer performance.

It is clear that for this example the self-generating hierarchical control structure of NQL can outperform non-hierarchical control structures. However, there is much more work that must be done before such systems can be used in real applications. Most notably, the technique described in this paper still uses discrete spaces, states, actions and behaviors and generalization within these states, actions and behaviors is not possible. Operation within continuous space and generalization methods for NQL are currently being pursued. This should allow the operation of applications in more realistic situations with real sensors and actuators. Also, the results presented indicated that the animat was beginning to view its world at a rudimentary level of abstraction. This abstraction will be necessary for learning at higher levels where seeing the world with a high level of detail recognition would be unnecessary and impeding. Current work will further concentrate on techniques for generating such abstracted features in realistic sensory systems.

2.6 Summary

A Nested Q-learning technique was developed for generating hierarchical control structures for a simple animat. Unlike previous work, this technique employed some simple selective criteria that allowed only those features with a good probability of becoming useful to emerge. This resulted in faster learning of the hierarchical control structure. The structure itself was shown to represent the abstracted higher level behaviors and their decomposed elemental sub-behaviors. Decomposition resulted in information easily being transferred to new but related tasks and a higher resilience to change as changes were confined to the behavior level which they affected, leaving most other behaviors untouched. The self-generating hierarchical control system was shown to outperform a non-hierarchical control system whenever new tasks were added or when change occurred. Although not yet ready for realistic application, the steps necessary for advancement have been outlined.

3 Apprentice Systems

3.1 Introduction

After many years of attempting to program robots to function in the most benign environments performing even the simplest of tasks progress has been limited. This difficulty is due to the fact that what appears as a simple straight forward operation to a human becomes very complex and un-constrained problem when a programmer attempts to enumerate all possible conditions and encode solutions. For humans performing new tasks is easy because they have many years of experience to draw upon. These experiences are learned in decoupled `action-state-outcome` and `action-state-utility` models that can be adapted to fit related situations and recombined to perform composite tasks correctly the first time. When confronted with learning a new task a human adapts existing skills or learns how to combine existing skills in new ways or at last resort and greatest cost learns from scratch.

It would appear the solution is to develop robots able to learn, especially robots that can not only learn from scratch, but also decompose learned models into reusable chunks and recombine those chunks whenever possible. However, researchers have been developing learning robots for many years that usually fail due to the excessive time and effort required to learn everything from scratch. From looking at how humans function it is clear that they do not learn much from scratch. That is, they

benefit from instinctive behaviors in place at birth, years of protected nurturing through infancy, many more years at school and extending all the way to reading scientific journals. Humans are taught these knowledge building blocks and discover ways decompose them and then to combine them to solve new problems quickly. Learning a solution from scratch must remain possible as it sometimes unavoidable, but more costly as it has a high likelihood of repeated failure. In short, humans combine or adapt knowledge they were taught whenever possible and generate new knowledge only when needed.

3.2 Learning System Requirements

DRDC-Suffield's efforts in developing learning hierarchical control structures are ideally suited for use in apprentice systems. In the hierarchical structure learning algorithm of Section 2 machines learn how to decompose learned knowledge into reusable chunks and exploit those chunks in ways similar to humans. This is possible because hierarchical learning results in a hierarchical model structure rather than a monolithic structure from which behaviors can be isolated. What is more important is that hierarchical learning control systems can be taught by humans and be able to adapt to situations quickly and with high confidence. Such a system called an *Apprentice* is being developed at DRDC-Suffield. It is envisioned that the apprentice would observe and learn from a human operator until it reaches some level of proficiency at which point the apprentice would assume control. Once the human is removed the apprentice's abilities would continue to improve perhaps surpassing the human operator's performance level.

3.3 Training of Apprentices

Issues involved with humans teaching learning machines include:

1. **What types of tasks could be taught to machines?** The level of task complexity that can be taught to a machine will depend upon the sophistication of the learning algorithms used. This level of sophistication can range from simple rote memorization to the learning of hierarchical decompositions. How well any learning method can be applied to real situations is an open question. Current research has proven nothing except learning in unstructured environments and tasks is difficult. The work presented in this paper on the learning of control structures is new and ideally suited to apprentice systems.

2. **What methods can be used by humans to teach machines?** The hierarchical learning method presented in Section 2 has no minimum level of required direction and will function using the vaguest of critical direction and able to fully use the most explicit direct training signal. While the training signal from a human teacher may be of high information content the best that can be expected from the environment will be a critical signal of a binary nature. The learning system must be capable of using training signals of all forms and origins. Specific methods for training machines would include:

   (a) **Imitation:** The machine observes the human operating and follows the actions.

   (b) **Instruction:** The machine is given instructions of what to do in specific situations.

   (c) **Graduated Complexity:** Many irrelevant aspects of the world and task are hidden from the machine during training. The complexity of the real world is then gradually increased.

   (d) **Protected Exploration:** The robot is allowed to explore and find solutions on its own, but is prevented from taking fatal or damaging actions.

   (e) **Graduated Lessons:** Off line lessons are formulated to create a specific capability in a machine. These lessons would need to be sufficiently diverse to facilitate the machine learning the underlying principles and not just memorizing the lessons.

3. **How quickly can a machine learn the information being taught?** Clearly the speed at which the the machine can be taught will depend upon the learning algorithms used. As demonstrated in nested Q-learning it is expected that slower initial learning will result in useful decompositions and abstractions. Simple rote memorization will be very fast, but will result in non-existent generalization capabilities and very brittle end results.

4. **How quickly can a machine adapt and use the information taught to it?** The flexibility of the learning algorithms ability to generalize and reuse previously learned information, but will also depend upon the diversity of the previous experiences. If the machines training experiences are well representative of the expected situations it will generalize quickly. If the previous experiences are non-representative of the expected situations it will generalize more slowly.

5. **Would methods for humans teaching humans be applicable to humans teaching machines?** As machines develop human like learning capabilities it is very possible that similar training methods to humans or at least animals will be applicable.

6. **What is the best method of teaching machines with different sensory modes than that of human?** When teaching learning machines the teacher must be aware of *sensory mismatch*. That is, the human teacher may observe the state of the situation through sensors that are very different from the sensors used by the learning control system. For instance when teaching a vehicle to move through a doorway a human teacher would use vision while the vehicle may use sonar and odometry. The teacher must be aware of this mismatch and structure the instructions and lessons to minimize its affect.

7. **What is the effect of incorrect teaching on machines or teaching poor solutions to machines?** Clearly the unlearning of incorrect or outdated skills is very important. Poor solutions may be the best that can be taught given the incomplete knowledge that will available at the time of training. Machines that can learn will be able to improve or learn different skills should their training be incomplete or misleading. This refinement of poor solutions is a major differentiation of apprentice systems and simple memorization systems.

4 Conclusions

How humans interact with machines will change as machines become more intelligent and are capable of learning and adaptation. The current role of humans as direct controllers of remote machines will quickly become outdated as machines become capable of independent operation, their tasks become more complex and communication realities limit human involvement. The concepts of shared control, hierarchical learning and apprentice systems detailed in this paper are active research areas at DRDC. This research program will see unmanned military vehicles that are capable of being trained, prolonged independent operation in the face of changing situations and able to make the decision when to request human assistance and then learn from that assistance. All these capabilities are required for machines to perform military relevant operations in unstructured and dynamic environments. The work described in this paper will see human roles will progress to controllers providing task level directives, intermittent assistance and ultimately teachers.

References

Barto A.G., Sutton R.S. and Watkins C.H (1989). Learning and sequential decision making. In: *COINS Technical Report.*

Brosinsky, Chris (2001*a*). The application of telematics in the canadian landmine detection capability. In: *IFAC Conference - Telematics Applications in Automation and Robotics.*

Brosinsky, Chris (2001*b*). Articulated navigation test-bed. In: *SPIE-Aerosense.*

Dayan, P. and G.E Hinton (1993). Feudal reinforcement learning. In: *Advances in Neural Information Processing Systems 5.*

Digney, Bruce L. (1996*a*). Emergent hierarchical control structures: Learning reactive hierarchical relationships in reinforcement environments. In: *4th International Conference of Simulation of Adaptive Behavior, SAB 96.* MIT/Bradford Books. pp. 363–.

Digney, Bruce L. (1996*b*). Learning and shaping in emergent hierarchical control systems. In: *In Proceedings of Space'96 and Robots for Challenging Environments II.*

Digney, Bruce L. (1998). Learning hierarchical control structures for multiple tasks and changing environments. In: *Simulation of Adaptive Behavior SAB98.* MIT/Bradford Books.

Digney, Bruce. L. (2001). Learned trafficability models. In: *SPIE-Aerosense.*

Long-Ji, L (1993). Hierarchical learning of robot skills. In: *IEEE International Conference on Neural Networks.* pp. 181–186.

Maes, P. and R.A. Brooks (1990). Learning to coordinate behaviors. In: *Eighth National Conference on Artificial Intelligence.* pp. 796–802.

Manduchi, Roberto (2000). Learning terrain classification for autonomous cross country navigations. In: *Darpa MARS-SDR meeting.*

Scott Thayer, Tony Stenz and Bruce Digney (2000). Cognitive colonies. In: *Darpa MARS-SDR meeting.*

Singh, P.S (1992). Transfer of learning by composing solutions of elemental sequential tasks. *Machine Learning* **8**, 323–339.

Thurn, S.B (1992). Efficient exploration in reinforcement learning. In: *Technical Report CMU-CS-92-102, Carnegie Mellon University.*

Tyrrel, T (1992). The use of hierarchies for action selection. In: *From animals to animats 2: SAB 92.* pp. 138–148.