IMA Conference on Mathematics of Robotics
9 – 11 September 2015, St Anne's College, University of Oxford

Dynamic Control of Synchronised Automata using Networked Q-Learning Techniques

By Christopher Deeks
Centre for Complexity Science, The University of Warwick, Coventry, UK

Abstract

This paper describes a novel approach to the control of a number of autonomous robots operating concurrently but without direct communication or coordination between them. Each automaton is a part of each of the others' dynamic local environments, and control actions need to be selected in response to observations of this local environment. A variation on Q-learning is introduced where the set of actions available to an automaton can be artificially constrained in response to observations and so with learned policies being recalculated online. The model to which this approach has been applied is a representation of a set of production line automata, where there may be a desire for a greater degree of concurrent operation to make the whole production line more efficient, but where there is also the desire to avoid long ramp-up or configuration times. It is shown that robust behaviour within set geometric constraints can be observed, even when the problem is set so that the automata cannot avoid obstructing each other. Training not just one automaton with a machine learning technique, but training a family of them concurrently, is therefore shown to be a feasible candidate for more widespread application in coordinated control tasks in dynamic environments.

1. Introduction

Reinforcement learning techniques are an established family of machine learning techniques that enable machine systems to update their patterns of behaviour from signals or measurements of the environment. The principle is simple, in that a machine has to be instructed to recognise particular occurrences or observations as relatively positive or negative according to a pre-determined reward scheme, but once this scheme and a finite set of available actions have been defined, reinforcement learning algorithms provide a mechanism for a machine to choose future actions to best maximise expected future reward.

Such techniques were first introduced by Watkins [1] as a generalisable approach to computational learning motivated by behavioural learning in the animal kingdom. Their suitability for problems in robotics followed quickly however, with a large body of extant literature on a wide range of algorithmic approaches and techniques for improving convergence to the intrinsically optimal control policy, improving learning times, and researching the effects of several different parameters and decision choices on the performance of such algorithms. Sutton and Barto provided a comprehensive overview of the field and the many options and approaches available [2].

While not the only means to control robotic systems, reinforcement learning algorithms are naturally suited to many types of problem because their defining characteristic is that the robotic system in question does not have a tightly defined series of actions to carry out at particular times. In simple applications, the programmer of the robot may not see the complications and additional overhead of having to choose state and action spaces and a reward scheme that together incentivise the desired behaviour as being worthwhile compared to the alternative of simply programming the robotic system with exactly what it is expected to

do. The power of reinforcement learning is for problems where this approach is difficult or impossible – typically because the environment the robotic system is operating in is dynamic, and responsiveness to external signals is required.

A number of learning systems operating in close proximity to each other can interact, thereby becoming a part of each other's local environment and potentially affecting each other's decision making process. The obvious example is in robot navigation, where the presence of another robot driving across the intended path necessitates a change of plans. Clearly the more robots are present the more dynamic the local environment may become. The robustness of reinforcement learning techniques in situations where a number of systems are co-located and potentially interfering with each other is the subject of the work reported on in this paper. Specifically, we consider whether co-located robots that individually have simple tasks to complete that can be relatively easily solved using reinforcement learning techniques remain capable of satisfactorily completing their tasks when their tasks compel them to interfere with each other. Note that, throughout this work, no communication or negotiation or joint planning between robotic systems is assumed or allowed – the objective of this work is to investigate the robustness of solution techniques that do not require communication, in recognition of the fact that in practice this may not always be reliable or even feasible.

2. Synchronous Automata

The technical challenge that motivated this research is that of synchronous co-located production line robots. This is not the only domain in which problems with synchronising robotic operations might be anticipated, but it does provide perhaps the most widespread use of simple robotic automata for which robust learning techniques could have significant future benefit. In particular, this work was motivated by the work of Doltsinis, Ferreira and Lohse into the potential application of a particular type of reinforcement learning technique known as Q-learning to improve the ramp-up period of automated production lines [3]. That work considered the ramp-up period for workstations on a production line, and the performance that Q-learners could deliver as a substitute for the human decision making process during ramp-up (that is, the process of adopting a pattern of behaviour with sufficient confidence that cyclical or episodic repetition of that behaviour will deliver the desired performance against the relevant criteria, specifically in that case functionality, quality and performance criteria).

The promising results of that research motivated investigation of whether the technique could be generalised to concurrent co-located workstations. Doltsinis et al. considered a single workstation and the ramp up time associated with learning the optimal behaviour for a particular automaton – not the ramp-up time or subsequent performance associated with a number of them working simultaneously and interacting (or interfering) with each other.

It can be assumed that a manufacturing director is unlikely to choose to co-locate automata when there is sufficient space or time available to conduct tasks separately and there is no pressing driver for concurrency. Nevertheless, there are foreseeable circumstances in which there might be benefit or necessity in doing do. These include, but are not limited to:

1) Space-constrained factories, where new projects require new or additional equipment but there is a desire not to locate away from the existing premises. Reversing this logic, were speedy and effective synchronous operation of multiple automata sufficiently robust, there is the possibility that some organisations might actively seek to locate to smaller premises for commercial reasons.
2) Fast turn-around manufacturing runs, where relatively small numbers of items are required. The time taken to physically reconfigure the facility, moving machinery around and then reconfiguring the necessary units for the new assignment can be a

significant overhead. The ability to leave them in place and reliably work around them would enable much speedier turn-around.

3) Mass production runs, where on some tasks there may be scope for greater concurrency on the production line to increase through-put. One obvious approach to take to support this is to enable multiple automata to operate on a particular production element simultaneously.

3. Modelling Interacting Learning Agents

The work reported on in this paper is based on a variant of the third problem. If, for example, a number of automata are all trying to make cuts with a laser, the production line will be able to produce more units more quickly if two or more lasers were able to operate on the same piece of metal concurrently, without colliding or cutting across each other.

Each automaton in a coordinated activity like this could be configured and programmed separately, but then once each one is finished and tested, there is a risk that when the next automaton is added to the workstation then not only will that new one require ramp-up time for configuration and testing, but the existing automaton may require reconfiguration to ensure the control policy it was going to follow is still appropriate in the presence of the second. Adding a third or a fourth could further compound the difficulties.

In theory, each automaton could be programmed separately with a particular sequence of control actions, with each sequence timed and synchronised to ensure it will not get in the way of what the other automata will be doing. However, the initial configuration will be much harder and the ramp-up time will potentially be significantly longer, even if there are no errors. In general, it should not be assumed that multiple automata configured individually can be left to operate side by side and still display acceptable patterns of behaviour; there is a risk of emergent behaviours, as well as the more obvious risk of collisions.

A simplified model of multiple interacting automata has been developed and implemented in a MATLAB-based simulation test-bed. The first research question must be whether acceptable patterns of behaviour can speedily and effectively emerge from a co-located agents using networked reinforcement learning agents. Successful demonstration of this proposition would then enable the possibility of more extensive deployment of networked learning agents in more complex problems, with larger state spaces, greater numbers of automata and different state space models/geometries.

4. Example Problem

We suppose that an industrial automaton is required to guide a laser along a predefined arc on a piece of metal. We further suppose that the factory owner would like to have multiple automata conducting several such tasks simultaneously if possible (for reasons of factory space and/or through-put). The arcs that each individual laser has to trace need not be the same shape, but to ensure regular interactions all automata in the initial experiments will attempt to trace the same arc – a regular circle. However, they are assigned tasks that take different lengths of time to complete at aperiodic intervals, ensuring that there is no trivial solution where all automata can simply trace the same arc one after the other.

The laser head is fitted to a rotating gimbal and the natural geometry to use is polar coordinate geometry. For initial simplicity, we reduce to two dimensions, where r is the distance between the laser head and the target point, and θ is the rotational angle (the third argument, which would be the downward pointing angle from the suspended laser head down to the surface, is neglected for simplicity).

The gimbal rotates – without loss of generality we assume in the anti-clockwise direction. It has the ability to adjust its cutting angle and orientation but only in the direction it is rotating - projected into two dimensions, or looking down directly from above, this means that r and θ vary as the gimbal rotates.

The gimbal can adjust the speed of its rotation – for initial simplicity we assume fast, slow, and still. Whether the laser is on or off is likely to be a decision available to the controlling agent – this is not explicitly introduced in the first version of the model, where it is assumed that when each automaton is stationary the laser is probably off (controlled by a separate agent outside the boundary of the current problem).

Once the workstation is configured, tasks should be a series of repetitive episodic tasks, that is learned patterns of behaviour repeated accordingly over a defined period. However, each individual automaton has different task parameters and these may not take the same amount of time to complete, requiring slack time for some of the automata. Furthermore, the amount of time each individual task takes is aperiodic with respect to at least one of the others.

With more than one automaton each deploying a separate laser head, the requirement is for them not to collide with each other. That is, when they would cross each other's path, they should be able to adjust their position and orientation to maintain the path of the laser along the required arc.

The mathematical construction of this problem is therefore as follows:

- State space: An annulus in (r,θ), where r ranges from $r_{\min}$ to $r_{\max}$ based on the orientation of the laser head, and where θ range from 0 to 2π radians
- Action space: 7 actions: rotate slow, rotate in slow, rotate out slow, rotate fast, rotate in fast, rotate out fast, and stay stationary
- Task: To trace out a parametrised curve in the $x-y$ plane, parametrised by t with a completion time of period T; parameter T represents the time a perfectly trained learning agent would be tracing that portion of the curve

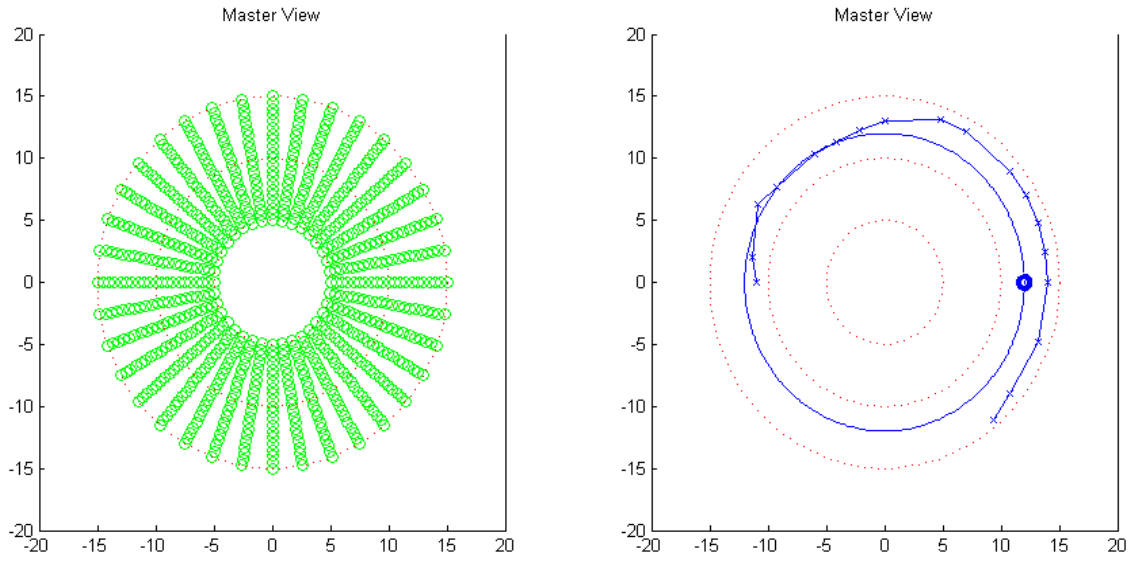

FIGURE 1A. Discretised state space;
FIGURE 1B Single learning agent during learning phase learning to follow the specified arc

Each agent will have a different task, including a different period, and optionally may use a different state space model (although this has not been used in the initial experiments). The action space remains the same throughout this current work.

This model needs to be discretised in order to be suitable for solution by a reinforcement learning agent. Therefore a lattice of (r,θ) points around the annulus is the spatial state space [Fig. 1A]. The density of the state space is a variable to be determined, trading off the increasingly accurate reflection of the true physical state space versus the exponentially increasing computational burden associated with solving the reinforcement learning problem. A number of state space densities have been experimented with during the course of this work.

For time, the natural method for discretising time intervals is by defining them as a series of unitary time steps. That is, is a task has period T then the natural discretisation for initial studies is to have T units of 1 time unit. The exact scaling and consequently the representation of time as a state depends on the task itself in more general application.

Action space is already discretised, therefore the model for this problem is (r,θ,t) space, where t is for the epoch within the current episode, and the extended Q-space for Q-learning is (r,θ,t,a), where a denotes the action.

The other elements needed to define a reinforcement learning problem are the transition model and the reward scheme – that is, the mapping from state-action pair (or a point in Q-space) to a new state, and the mapping from a state-action pair to a real number R, which define what transition occurred following the execution of an action, and the relative value of that transition. Transition models are in general stochastic, however in a tightly constrained

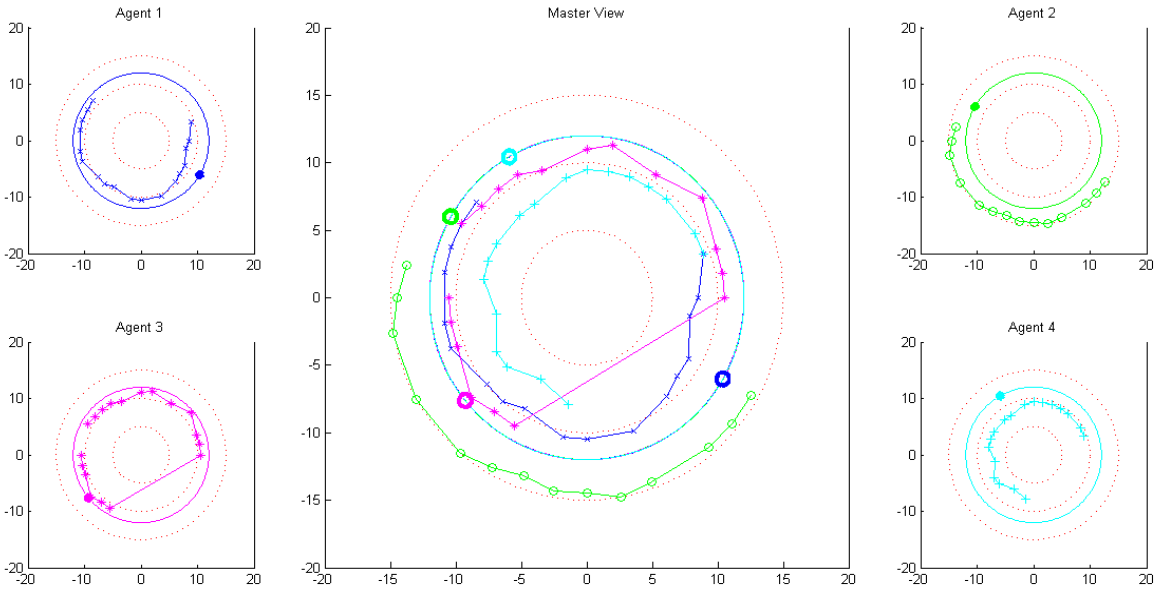

FIGURE 2. Four learning agents learning their assignments, all following the same arc but with aperiodic schedules. Path histories represented by line plots. Larger markers on the reference arc represent where each agent should be at that point in its own cycle. Note that enforced state space exploration is optionally enabled in order to speed convergence, accounting for the large jump in the path of agent 3. This is not an allowed behaviour in a control policy, it is merely for the learning phase.

problem such as the one considering here, where there is little probability of the robot physically malfunctioning and producing entirely the wrong behaviour, it is reasonable to adopt a deterministic transition model. Hence in this initial model transitions are translations in the direction and relative magnitude defined by the chosen action.

The reward scheme is the mechanism by which the agent differentiates good actions (in the sense of moving it towards its goal or objective) from bad ones (in the sense that some actions are counter-productive or ineffective). The reward scheme must therefore encapsulate what the goals and requirements of that system really are. In this case, the objective is to maintain a trace of a defined curve on an episodic schedule. Thus the following reward scheme was chosen: all points in Q-space are initialised with value 0, and all actions that leave the system in a state from which is unable to 'see' the arc at it is supposed to be tracing earn reward 0. Any actions that leave the system in state from which it can see the arc earn reward 1. This reward is then multiplied by a second reward for whether the current position of the learning agent is out of phase with the episodic schedule needed to maintain the trace, with reward 1 if the latest action reduced the phase difference between actual and expected positions, and 0 if it did not. In this manner, reward can only be maximised by closely tracing the arc in phase.

For each agent is isolation, these are relatively straightforward Q-learning problems, the primary novelty being the use of time epochs and treating episodic time as an argument of state. It is allowing for and modelling interactions that introduces more complex behaviours.

In this initial model, interactions consisted of a sense-and-avoid routine where, before each action selection by the learning agent, the set of allowed actions was pruned to remove actions that could bring the automaton into a collision with any other entities detected within the locally neighbouring states. There is no communication between agents, rather a redefinition of the internal logic. Thus, when there is a potential collision, which is guaranteed to happen at regular aperiodic intervals with the model designed, the agents will find they do not have the action they would choose in the absence of the other entity and must conduct online recalculation of the best alternative action to take instead.

To demonstrate the potential efficacy of Q-learners to the synchronised automata problem, it is necessary to demonstrate that (i) the techniques can be applied in this way and still converge to effective control policies despite perturbations caused by interaction, and (ii) that the policies generated still deliver appropriate behaviour within the defined geometric constraints.

5. Results

The rates of convergence to the a priori calculated optimal control policy for the four agents both in isolation and with the ability to detect each other's presence are shown in Figure 3. An example of the behaviour of the synchronised automata is captured in Figure 4.

Regarding convergence, it is clear that introducing the ability to recognise obstacles has not significantly altered the convergence rate for the agents. This is an encouraging result, as it indicates that the novel elements applied to model with the interaction are not affecting the overall performance of the system. Simulation runtimes were also similar, and can surely be improved upon following deeper consideration of the sampling strategies and state space models.

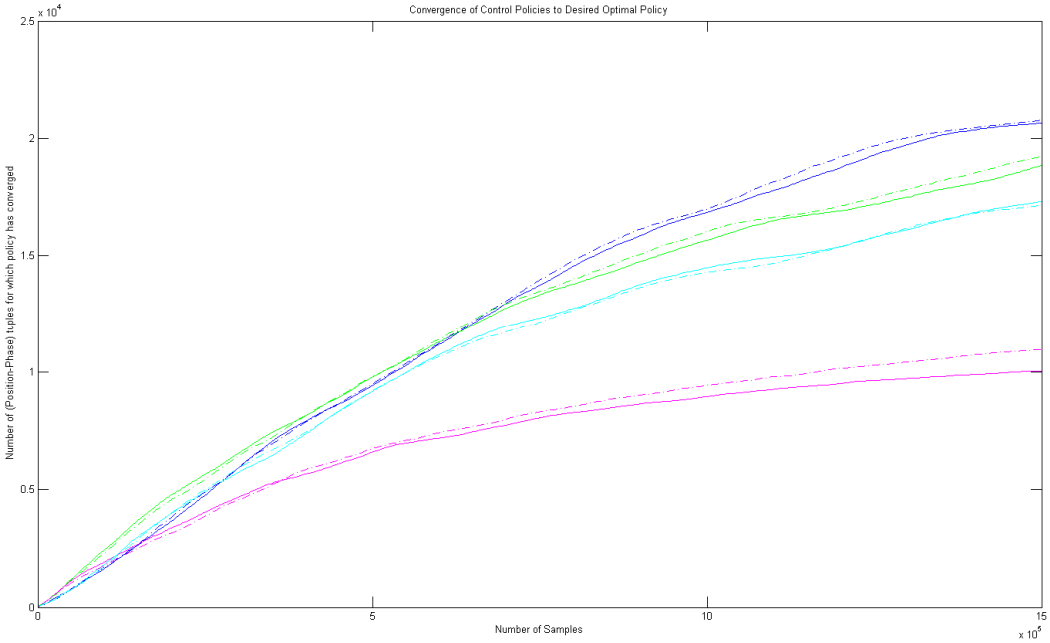

FIGURE 3. Convergence of learned policy to the optimal control policy calculated a priori. Solid lines are the convergence rates of agents in isolation (i.e. ignoring potential interaction) and dashed lines are those allowing for perturbation.

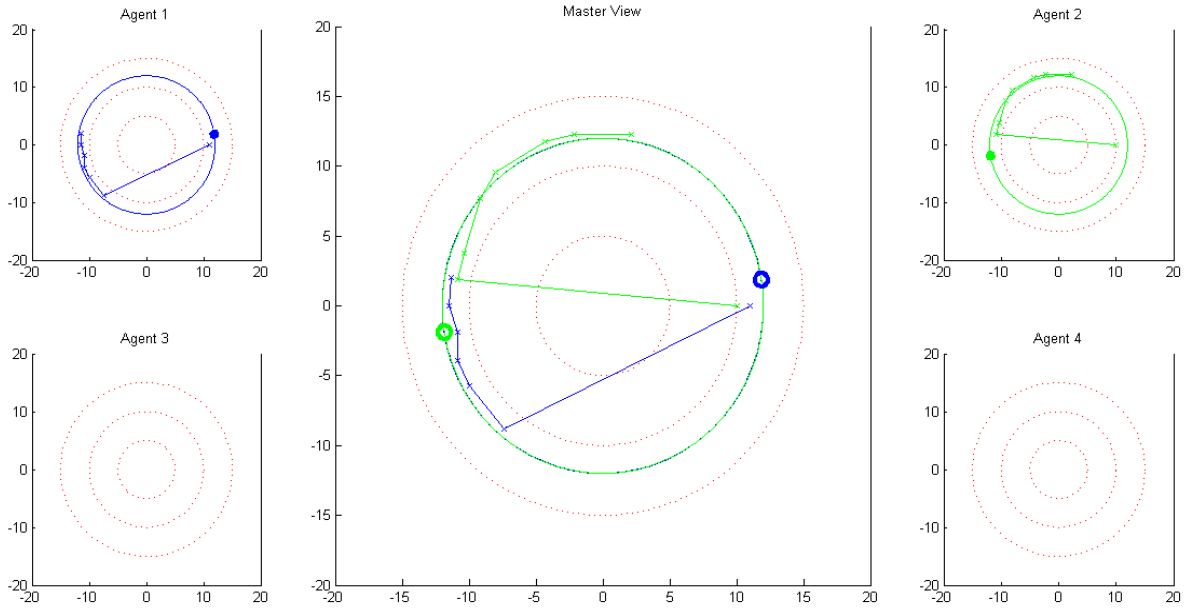

FIGURE 4. Two learning agents learning their assignments with action filtering enabled (agents 3 and 4 were present but omitted here for clarity). The agents are still in their learning period, hence the sudden jump as the agents reinitialise for another trial.

Regarding Figure 4, note the path of agent 1 in the lower left portion of the arc, where it clearly steps aside and bypasses agent 2. Such a perturbation is still within the look radius of the automaton and clearly within the geometric constraints defined by the operating annulus. It demonstrates that the policies that are being learned are enabling the agents to adapt to the presence of other automata with no knowledge other than of their own task and that there is an entity to avoid. This is consistent with the convergence results that shows that, despite these aperiodic perturbations, the agents are still able to learn stable control policies that deliver behaviour slightly perturbed from the optimal policy but still within the relevant constraints. This is encouraging evidence that the desired application of interaction learning agents to the synchronisation of automata can be achieved with relatively simple models, without the more substantial system configuration and extended ramp-up time.

6. Analysis & Next Steps

This paper has reported on mathematical modelling work investigating the introduction and extension of reinforcement learning agents (specifically Q-learning agents as introduced by Watkins) to problems featuring a number of interacting automata each trying to conduct an individual task but facing obstacles caused by the other automata, forcing them to adapt or deviate from their optimal control policy. To that end, a number of novel elements have been implemented within the model, including temporal state indexing and pre-filtering of allowed actions based on local detections. Integrated together into a single test environment, it has been demonstrated that with no direct knowledge or communication with other agents, and working on very simple state space models and simple geometric constraints, it is possible to create complex patterns of irregular behaviour from the agents as they adapt to each other's presence. Particularly encouraging is clear observational evidence, such as presented in Figure 4, that agents are learning to cope with enforced deviations from their trajectories whilst complying with their constraints and task objectives.

The model developed is simple in terms of the task assigned, and in having been reduced to two dimensions. Among the future activities planned are to generalise the model to more-realistic three dimensional space, and work is on-going to generalise the test framework for other task types in other geometries. Other anticipated tasks include investigating the sensitivity of the performance in particular scenarios to the various learning parameters, and the choice of learning methodology (random walks with online learning, but with enforced state space jumps to ensure coverage have been adopted in this work, but there are alternatives). The goal of this work is to be able to parametrise a particular task (be it laser tracing or otherwise) and be able to demonstrate a speedy configuration to an acceptable solution using only simple learning agents, and the results obtained in the work so far indicate that this is feasible.

REFERENCES

[1] WATKINS, C. J. C. H. 1989. Learning from Delayed Rewards, King's College Cambridge PhD Thesis

[2] SUTTON, R. S. & BARTO, A. G. 2012. Reinforcement Learning: An Introduction (2nd Edition)

[3] DOLTSINIS, S., FERREIRA, P. & LOHSE, N. 2014. An MDP Model-Based Reinforcement Learning Approach for Production Station Ramp-Up Optimization: Q-Learning Analysis, *IEEE Transactions on Systems, Man, and Cybernetic Systems*

ACKNOWLEDGEMENTS

The research reported in this paper was funded by the Engineering and Physical Sciences Research Council under the supervision of Prof. R. Mackay of the Centre for Complexity Science and Prof D. Ceglarek of the Warwick Manufacturing Group, University of Warwick.