

# Robotic Target Tracking with Approximation Space-Based Feedback During Reinforcement Learning<sup>\*</sup>

Daniel Lockery and James F. Peters

Department of Electrical and Computer Engineering,  
University of Manitoba  
Winnipeg, Manitoba R3T 5V6 Canada  
dlockery@ee.umanitoba.ca, jfpeters@ee.umanitoba.ca

**Abstract.** This paper presents a method of target tracking for a robotic vision system employing reinforcement learning with feedback based on average rough coverage performance values. The application is for a line-crawling inspection robot (ALiCE II, the second revision of Automated Line Crawling Equipment) designed to automate the inspection of hydro electric transmission lines and related equipment. The problem considered in this paper is how to train the vision system to track targets of interest and acquire useful images for further analysis. To train the system, two versions of Watkins' Q-learning were implemented, the classical single-step version and a modified strain using an approximation space-based form of what we term *rough feedback*. The robot is briefly described along with experimental results for the two forms of the Q-learning control algorithm. The contribution of this article is an introduction to a modified version of Q-learning control with rough feedback to monitor and adjust the learning rate during target tracking.

*Keywords:* Approximation space, target tracking, monocular vision, reinforcement learning, rough sets, Q-learning.

## 1 Introduction

The problem considered in this paper is how to influence a system with a control algorithm that learns from past experience of actions for any given situation. The system consists of a robotic platform with a monocular vision apparatus used to track and acquire images of desired targets. Control of the vision system is accomplished via the Q-learning algorithm [16, 18]. Two versions of this method

---

<sup>\*</sup> Best Paper Award in Joint Rough Sets Conf. (JRS2007), Lecture Notes in Artificial Intelligence 4482, Springer-Verlag, Heidelberg, 2007, 483-490. ISBN 978-3-540-72529-9. Acknowledgements: The author gratefully acknowledges the suggestions and insights by Andrzej Skowron, David Gunderson, Maciej Borkowski, Christopher Henry concerning topics in this paper. This research has been supported by Manitoba Hydro grants T247, T260, T270 and Natural Sciences and Engineering Research Council of Canada (NSERC) grant 185986.

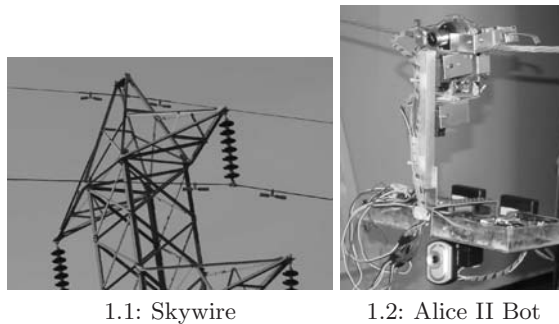
are included, Watkins' single-step approach and a modified version of Watkins' algorithm that employs a rough set approach.

Several examples of recent work have been reported for the target tracking control problem, including [1, 2, 6, 11–13]. The robotic platform as well as reinforcement learning are a common thread. The contribution of this article is the implementation and comparison of a rough coverage feedback value for adjusting the learning rate of a control algorithm versus the classical implementation.

This article is organized as follows. A brief introduction to the robotic platform is provided in Section 2. The rough set and approximation space theory employed in the control algorithm are briefly discussed in Section 3. Section 4 includes a look at the reinforcement learning control algorithms, followed by the experimental results in Section 5

## 2 The ALiCE II Robot

The ALiCE II device is an inspection robot designed to crawl along a 9mm sky wire that exists at the uppermost part of large transmission and distribution hydro power lines (see Fig. 1). The bot must function in a harsh environment buffeted by wind, hampered by electromagnetic fields, rain, and huge swings in temperature.



**Fig. 1.** Environment for Target Tracking System

A camera system is attached to the base of ALiCE II to allow for a complete view of all conductors and the tower structure beneath it partially shown in Fig. 1.1. Through low-level control, the robot is able to navigate back and forth along the sky wire and pause as required to acquire images of interest from the camera. Fig. 1.2 provides a front overhead view and a rear view of the experimental setup demonstrating the line crawling robot. There are a number of different types of targets of interest including insulators, pins, conductors, and tower structures. Once a target has been sighted, tracking is used to help gather the best images by maximizing the surface area and compensating for external influences like wind speed. The camera mounted on the underside of the robot's

platform consists of two servo motors that pan and tilt the camera as required for positioning.

### 3 Rough Sets and an Example Approximation Space

This section includes a brief introduction to rough sets followed by a description of an approximation space and an associated example. Representation of objects and features occur in the form of data tables to simplify the processing steps [12]. Let  $U$  denote a non-empty finite set called a *universe* and let  $\mathcal{P}(U)$  be the power set of  $U$  (*i.e.* the family of all subsets of  $U$ ). In this paper, elements of  $U$  correspond to observed behaviors. A *feature*  $\mathcal{F}$  of elements in  $U$  is measured by an associated probe function  $f = f_{\mathcal{F}}$  whose range is denoted by  $\mathcal{V}_f$ , called the *value set* of  $f$ ; that is,  $f : U \rightarrow \mathcal{V}_f$ . There may be more than one probe function for each feature. For example, a feature of a behavior may be the reward obtained for performing an action. The similarity or equivalence of objects can be investigated quantitatively by comparing a sufficient number of object features by means of probes [10]. For present purposes, we identify the set of features with the set of associated probe functions, and hence we use  $f$  rather than  $f_{\mathcal{F}}$  and call  $\mathcal{V}_f = \mathcal{V}_{\mathcal{F}}$  a set of feature values. If  $F$  is a finite set of probe functions for features of elements in  $U$ , the pair  $(U, F)$  is called a *data table*, or *information system* (IS).

For each subset  $B \subseteq F$  of probe functions, define the binary relation  $Ind_B = \{(x, x') \in U \times U : \forall f \in B, f(x) = f(x')\}$ . Since each  $Ind_B$  is an equivalence relation, for  $B \subseteq F$  and  $x \in U$  let  $[x]_B$  denote the equivalence class, or *block*, containing  $x$ , that is,

$$[x]_B = \{x' \in U : \forall f \in B, f(x') = f(x)\} \subseteq U.$$

If  $(x, x') \in [x]_B$ , then  $x$  and  $x'$  are said to be *indiscernible* with respect to all feature probe functions in  $B$ , or simply, *B-indiscernible*. Information about a sample  $X \subseteq U$  can be approximated from information contained in  $B$  by constructing a  $B$ -lower approximation

$$B_*X = \bigcup \{[x]_B \mid [x]_B \subseteq X\},$$

and a  $B$ -upper approximation

$$B^*X = \bigcup \{[x]_B \mid [x]_B \cap X \neq \emptyset\}.$$

The  $B$ -lower approximation  $B_*X$  is a collection of blocks of sample elements that can be classified with full certainty as members of  $X$  using the knowledge represented by features in  $B$ . By contrast, the  $B$ -upper approximation  $B^*X$  is a collection of blocks of sample elements representing both certain and possibly uncertain knowledge about  $X$ . Whenever  $B_*X \subsetneq B^*X$ , the sample  $X$  has been classified imperfectly, and is considered a rough set. In this paper, only  $B$ -lower approximations are used.

### 3.1 Approximation Spaces

The original discussion of approximation spaces was introduced by Pawlak [7] and has since been expanded to a generalized version [14, 15]. Approximation spaces provide the basis for a new form of reinforcement learning based on acceptable patterns of behaviour viewed in the context of a line-crawling robot. This section briefly introduces approximation spaces and the generalized version with rough coverage.

The original definition of an approximation space provided by Pawlak [7] contained the pair  $(U, Ind)$ . Where  $U$  corresponds to a non-empty finite set and  $Ind$  represents an indiscernibility relation on subsets of  $U$  ( $Ind \subset U \times U$ ) [7]. More recently, a generalized approximation space was introduced by Skowron and Stepaniuk [14], [15] represented by a triple,  $(U, I, \nu)$ .

- $U$  is a non-empty set of objects, and  $\mathcal{P}(U)$  is the powerset of  $U$ ,
- $I : U \rightarrow \mathcal{P}(U)$  is such that  $x \in I(x)$  for any  $x \in U$ ,
- $\nu : \mathcal{P}(U) \times \mathcal{P}(U) \rightarrow [0, 1]$  is an overlap function (inclusion or coverage).

Similar to the classical description,  $U$  corresponds to a non-empty, finite set. The uncertainty function  $I$  maps each  $x \in U$  to a neighbourhood such that a given object  $x$  is associated with a set of objects that are similar in some respects. This function can also be used to help define a covering of  $U$  [12] specialized relative to  $B_*X$  and blocks in the  $Ind_B$ -partition of  $U$ . Pertaining to coverage of sets,  $\nu$  is a measure of overlap and is referred to as *inclusion* or *coverage* depending upon the configuration of the expression. In this paper, *rough coverage* is used as a measure of set overlap as the basis for a measure of Q-learning performance.

$$\nu(X, Y) = \begin{cases} \frac{|X \cap Y|}{|Y|}, & \text{if } Y \neq \emptyset, \\ 1, & \text{if } Y = \emptyset. \end{cases} \quad (1)$$

The value of  $\nu$  represents the degree of coverage, ranging from 1, when the sets are equal to one another ( $X = Y$ ) to the minimum value of 0, when there are no common elements in  $X$  and  $Y$  ( $X \cap Y = \emptyset$ ) [15]. Anything in between 0 and 1 represents at least some degree of overlap between the two sets in question.

We used the average rough coverage as a performance metric. Let  $d$  denote a partial function that represents a decision about an object based on evaluation of  $D$  denote the set  $D = \{x \in U \mid d(x) = 1(\text{accept})\}$ .  $\nu$  is computed by substituting  $B_*D$  or known accepted tracking behaviours for  $Y$  and each  $[x]_B$  substituted for  $X$  in (1).  $\bar{\nu}$  (average coverage) is computed by averaging individual  $\nu$  values, and provides the backbone for a new form of Q-learning represented by Alg. 1.

## 4 Reinforcement Learning Control Algorithms

The line crawling robot uses reinforcement learning to discover how to control the movements of a digital camera used to track randomly moving targets. The control algorithm selected was single-step Q-learning and it was modified to

incorporate rough coverage as a feedback performance metric and then compared to the classical algorithm. This section includes a brief overview of Q-learning and the modification made to provide rough feedback followed by the formal algorithm.

The Q-learning algorithm learns based on the action (or  $Q$ ) value associated with each state as opposed to using the value function associated with being in a given state. Q-learning was developed by Watkins, formally reported in 1989 [18]. The Q-learning method in its simplest form is a single step temporal difference learning method that is capable of maximizing the action value of an agent regardless of the policy being followed [3, 16, 18]. The concept of a single step algorithm is that it looks into the future one step in advance when estimating the best course of action to take from the current state. The best action is generally the choice that maximizes the future discounted reward available from all actions pertaining to the current state. It is important to note that for Q-learning it does not matter what policy is being followed, it will always maximize the action value [18]. Q-learning falls into the category of off-policy algorithms [16]. This implies that the decisions made for selecting a course of action do not necessarily follow the policy that is exploring the state space [16].

During the initialization of the Q-learning algorithm, a policy must be established to determine a preliminary (most likely sub-optimal) mapping from states to actions. As indicated in Alg. 1, this policy is not greedy, which means that there is a possibility that it will not always follow the action with the highest immediate reward. There must be at least a small chance that this policy will explore alternate actions providing potential visits to sub-optimal actions that may return greater long term rewards. Throughout the learning process of single step Q-learning, state-action pairs are examined one step ahead. Rather than following the original non-greedy policy that is selecting actions, a greedy policy is used to determine the best action to take from the current state.

$$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \cdot \max_{a'} Q(s', a') - Q(s, a)] \quad (2)$$

From Eq. 2, we see that although the non-greedy policy is selecting actions, the update is affected by the maximizing greedy policy inherent in the Q-learning update rule. The algorithm parameters that can be found in Q-learning are  $\gamma$ , and  $\alpha$  which correspond to the discount factor and the learning rate step size adjustment, respectively.

The term *episode* refers to a length or the amount of time steps present in an episode of analysis for the reinforcement learning process. The difference from classical Q-learning for the rough feedback version lies in the update rule presented in Eq. 2, it was modified to include the average rough coverage value to adjust the value of  $\alpha$ . As the average rough coverage value increases, the value of  $\alpha$  decreases, reducing the step size change to prevent overshoot when approaching a desirable policy. Conversely, for situations where the average rough coverage is low, the value of  $\alpha$  will increase, causing the adjustment to be greater in the hopes of finding more suitable behaviours.

---

**Algorithm 1:** The Q-Learning Method With Rough Feedback

---

**Input** : States,  $s \in S$ , Actions  $a \in A(s)$ , Initialize  $Q(s,a)$ ,  $\bar{v}$ ,  $\alpha$ ,  $\gamma$ ,  $\pi$  to an arbitrary policy (non-greedy);

**Output:** Optimal action value  $Q(s,a)$  for each state-action pair;

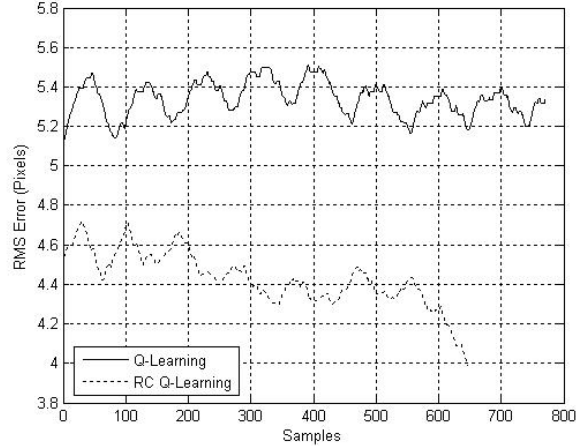
```
while True do
  for ( $i = 0; i \leq \#of\ episodes; i++$ ) do
    Initialize  $s$  and data table;
    Choose  $a$  from  $s$ , using policy derived from  $Q$ ;
    for Repeat(for each step of episode): do do
      Take action  $a$ ; observe reward,  $r$ , and next state,  $s'$ ;
      Record state, action, and associated reward in data table;
       $Q(s,a) \leftarrow Q(s,a) + (1 - \bar{v})\alpha[r + \gamma\max_a Q(s', a') - Q(s, a)]$ ;
       $s \leftarrow s'$ ;  $a \leftarrow a'$ ;
      until  $s$  is terminal;
    end
    Generate  $\bar{v}$  from results recorded in data table for current episode;
    Update new value of  $\bar{v}$ ;
    Clear data table;
  end
end
```

---

## 5 Experimental Results

The experimental work was done using a camera system identical to that found on the line crawling robot (seen in Fig. 1. Noise was introduced into the environment in the form of random movements of the platform, similar to what would be experienced in light wind conditions. This section includes preliminary results and a brief discussion of their implications.

The parameters of the Q-learning algorithms were selected as 0.1 for both the learning and discount rate ( $\alpha$  and  $\gamma$  respectively). The length of the experiments were 5 minutes each and a sample result is included (See Fig. 2). The number of samples differs between the two methods since the rough feedback method requires more calculations to generate the rough coverage values. The trade-off of spending more time processing data is improved results as seen in the results. The rough-feedback implementation is able to adjust to the movements of the camera and this can be seen with the reduced RMS error pertaining to the target location. As the target moves, the algorithms attempt to learn the best movements in any given situation for tracking. The adjustable learning rate allows the rough-feedback method to react more quickly or more slowly depending on how well it performs. The classical method maintained a reasonable rate of performance but since the environment was somewhat noisy, it was unable to adjust as quickly and provided a stable error rate around 1 pixel greater than that of the rough feedback method.



**Fig. 2.** Plotted results, Samples (x-axis) vs. RMS pixel error (y-axis)

## 6 Conclusions

The preliminary results are promising for the rough feedback implementation of the Q-learning algorithm. The error rate recorded was a significant improvement over the classical implementation. The difference in the number of samples over a five minute period was 125 extra samples for the classical method over the rough feedback method. This was expected since creating the data tables and the required processing to generate the average rough coverage values is more work.

The performance boost is significant enough that it is worth considering as an alternative approach to target tracking in our environment. The only drawback is the amount of processing and extra time required which necessitates more battery power for an autonomous robot and the extra time weakens the tracking capabilities, since it pauses during heavy computations to preserve power. An attempted possible solution was to reduce the episode sizes when generating data tables. This resulted in less computational power and time but at the cost of reducing the sample size of behaviours drawn upon when revising the learning rate. The extra computational cost reduced the number of samples by 16% and improved the accuracy by an average of 20% over the classical method. Reducing the sample size did not adversely affect the learning process as the experimental environment did not have many variables. However, in more complex environments, reducing the sample size could introduce poorer results if the selected samples did not reflect the general behaviour.

In conclusion, the preliminary results the new form of Q-learning demonstrate that adjusting the approximation space-based learning rate based on average rough coverage provides improved accuracy for the target tracking process. Further experimental work is required to verify consistency and an optimal episode

size as well as the time required to generate the rough coverage values. Future work on Q-learning will include ways to improve the proposed method. This is definitely possible by hearkening back to the run-and-twiddle adaptive learning method introduced by Oliver Selfridge in 1984.

## References

1. Distante, C., Anglani, A. and Taurisano, F.: Target Reaching by Using Visual Information and Q-learning Controllers, *Autonomous Robots*, vol.9, no. 1, pp. 41-50, Springer Netherlands, August 2000.
2. Gaskett, C.: *Q-Learning for Robot Control*, Ph.D. Thesis, Department of Systems Engineering, Supervisor: A. Zelinsky, The Australian National University, 2002.
3. Kaelbling, L.P, Littman, M.L. and Moore, A.W.: Reinforcement Learning: A Survey, *Journal of Artificial Intelligence Research*, vol. 4, pp. 237-248, May 1996.
4. Komorowski J., Polkowski, L. and Skowron, A.: Rough Sets: A Tutorial, In S.K. Pal and A. Skowron, editors, *Rough-Fuzzy Hybridization: A New Method for Decision Making*, Springer-Verlag, Singapore(in Print), 1998.
5. Munakata, T. and Pawlak, Z.: Rough Control Application of Rough Set Theory to Control, *Fourth European Congress on Intelligent Techniques and Soft Computing*, vol. 1, pp. 209-218, Aachen, Germany, September 1996.
6. Nakamura, T. and Asada, M.: Motion Sketch: Acquisition of Visual Motion Guided Behaviors, *Fourteenth International Joint Conference on Artificial Intelligence*, vol. 1, pp. 126-132, Montreal, Canada, 1995.
7. Pawlak, Z.: Rough Sets, *International Journal of Computer and Information Sciences* vol. 11, no. 5, pp. 341-356, 1982.
8. Pawlak, Z.: *Rough Sets. Theoretical Reasoning about Data*, Theory and Decision Library, Series D: System Theory, Knowledge Engineering and Problem Solving, vol. 9, Kluwer Academic Pub., Dordrecht, 1991.
9. Pawlak, Z.: Rough Set Theory and its applications, *Journal of Telecommunications and Information Technology*, 3, 2002.
10. Peters, J.F.: Classification of objects by means of features. In: Fogel, D., Mendel, J., Yao, X., Omori, T. (Eds.), Proc. First IEEE Symposium on Foundations of Computational Intelligence (FOCI'07), 1-5 April 2007, *in press*.
11. Peters, J.F., Henry, C., Lockery, D., Borkowski, M., D. Gunderson, Ramanna, S.: Line-Crawling Bots That Inspect Electric Power Transmission Line Equipment, *The 3rd International Conference on Autonomous Robots and Agents (ICARA)*, Palmerston North, New Zealand, December 2006.
12. Peters, J.F., Lockery, D.A., and Ramanna, S.: Monte Carlo Off-Policy Reinforcement Learning: A Rough Set Approach, *Proceedings of The Fifth International Conference on Hybrid Intelligent Systems*, November 2005.
13. Peters, J.F., Borkowski, M., Henry, C., Lockery, D.: Monocular vision system that learns with approximation spaces. In: Ella, A., Lingras, P., Slezak, D., Suraj, Z.: *Rough Set Computing: Toward Perception Based Computing*. Idea Group Publishing, Hershey, PA (2006), 1-22.
14. Skowron, A. and Stepaniuk, J.: Tolerance Approximation Spaces, *Fundamenta Informaticae*, vol. 27, no. 2/3, pp. 245-253, 1996.
15. Stepaniuk, J.: Approximation Spaces, Reducts and Representatives, in L. Polkowski and A. Skowron (Eds.), *Rough Sets in Knowledge Discovery 2, Studies in Fuzziness and Soft Computing*, 19, pp. 109-126, Heidelberg: Springer, 1998.



16. Sutton, R.S., Barto, A.G.: *Reinforcement Learning: An Introduction*, Cambridge, MA: The MIT Press, 1998.
17. Swiniarski, R.W. and Skowron, A.: Rough set methods in feature selection and recognition, *Pattern Recognition Letters* 24, pp. 833-849, 2002.
18. Watkins, C.J.C.H.: Learning from Delayed Rewards, *Ph.D. Thesis, supervisor: Richard Young, King's College*, University of London, UK, pp. 25-36, May 1989.