ACTIVE PROTECTION SYSTEM SOFT-KILL USING Q-LEARNING

Arvind Rajagopalan (*DSTG*)
Adelaide, Australia

Abstract— **In this article, an active protection system design using a Q-learning based reinforcement learning (RL) is described and evaluated. It is applicable in the context of defending against fully observable threats that possess high mobility. The Q-learner is implemented with an Artificial Neural Network, which is a model-free RL method to solve what is effectively an optimal control problem that can be difficult to solve analytically. Numerical experiments are provided to illustrate the performance of Q-learning under various approaching directions of the high mobility threat. Finally, discussions for further improvements with applying Q-learning as well using substituting with policy-optimization RL techniques are provided.**

Keywords—Reinforcement-Learning, ***Q-learning, Active Protection System***

I. INTRODUCTION

Active protection systems (APS) are required to prevent guided anti-tank missiles/projectiles from hitting and damaging stationary/mobile land-based military assets [1]. Examples include entities such as an Armored Personnel carrier (APC). One strategy, applied as part of an APS is through applying soft-kill techniques. Under the umbrella of techniques applied for soft-kill strategies, one approach is to seduce the incoming threat using a decoy [2]. Such a decoy needs to be able to make agile control decisions under tight time constraints since the type of threat (e.g. guided anti-tank missiles) can be highly mobile [3]. The threats can approach the asset from any direction. The decoy needs to be able to successfully cope and defend the asset by drawing the missile threat away from the asset appropriately under motion constraints. In this paper, a machine-learning (ML) based control approach is applied for designing the decoy controller. The motivation for applying a ML-controller is to illustrate the application of this type of decoy-controller. Machine learning controllers are being developed for scenarios where there is a need to rapidly infer solutions for complex problems (e.g. complex dynamics) in a memory efficient manner [4]. This will become more important when extending the context to support the complex problem of coordinated multi-decoy strategy to defend several vehicles against multi-axis threats.

In this paper, reinforcement learning (RL), which is a powerful set of techniques within ML based control, has been selected. In RL, an agent interacts with an unknown environment over a certain period of time. At each time-step, the agent observes the state, takes an action, and receives a reward or penalty as the result of its action. The goal of the agent is to learn an optimal policy (i.e., a mapping from states to actions) that maximises its long-term return. In the long run, those actions that yield the higher rewards according to the current knowledge of the environment will be chosen to enhance the cumulative rewards [6]. RL algorithms are divided into the dynamic programming family and policy optimization family [7]. Here, we are looking at the dynamic programming family where we apply the well-known Q-learning for this decoying problem. Q-learning is a relatively well understand algorithm within RL. Therefore, it has been chosen as the method to devise the controller. For more advanced situations, there is a wide variety of emerging algorithms to choose from such as given in [8].

II. RESEARCH PROBLEM/CONTEXT

The threat such as an anti-tank missile is assumed to have an on-board seeker to find, track and guide the missile towards the target [3]. Using an IR/RF emitter, the decoy can present an IR source with suitable heat signature or mimic the RF signature of the target and can then seduce the threat towards the target [9]. RF seeker equipped threats appear to be less common than those using IR techniques. In the context of IR threats, for the passive IR variety with no man-in-the-loop, it is the author's view that seduction methods such as one described in this paper may have some usefulness as part of the countermeasure strategy. The decoy has to manoeuvre in such a manner to separate as far from the target as possible in angle with respect to the threat while adhering to some geometric constraints. The constraint introduced here is that the separation distance between the decoy and the range information of the decoy and the target to the missile cannot differ by more than 50 meters. This constraint is added to observe if the Q-algorithm is capably of solving the tightly constrained control problem with good predictive behaviour. Here, the notional application of the constraint is associated with preventing the threat from being alerted it is being seduced. This constraint may need to be suitably modified when applied to the land-domain. The seduction capability of the decoy payload in this problem is considered to be perfect. The representation of decoy goals and constraints is shown visually in the diagram given in Figure 1. In Figure 1, p_s denotes the true target, p_m denotes the threat and p_i denotes the decoy. The arrows in the diagram depict the transition of each entity in a single time-step. The symbols suffixed with $'$ (e.g. p_i') next states after the transition states at the end of the single time step. The angle θ shows the direction of motion selected by the decoy controller,

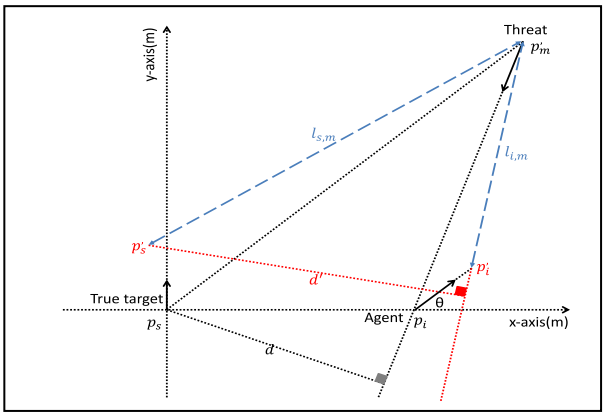

Fig. 1. Block diagarm depicting the form of function approximator

The decoy is initially co-located with the true-target. The goal for the decoy controller is find the policy (rule for actions) that will steer the decoy in the direction of maximum miss distance. Without loss of generality, we assume that the true target is moving north with a constant speed starting from the origin of the Cartesian coordinate system. The threat missile in this instance, is assumed to home in towards its respective target by applying the widely applied Proportional Navigation (PN) guidance law [10].

III. Q-Learning Controller

The central idea of Q-learning is to recognise and learn the sequence of actions through trial and error. By receiving a reward when performing an action, the agent learns about the quality of its action choices for a given state. The agent stores this knowledge it observes for each state-action pair in a form of a Q-value, denoted by $Q(s,a)$. Here, s represents the state of the system which includes information from the agent and the environment in which it operates. The action a is the chosen action from the set of feasible actions. Over time, the agent learns the optimal action-value function $Q^*(s,a)$ which is the unique solution of the following Bellman's equation

$$Q(s_k,a_k) \leftarrow Q(s_k,a_k) + \alpha(r_{k+1} + \gamma \max_a Q(s_{k+1},a) - Q(s_k,a_k)) \quad (1)$$

In Equation 1, α = (0; 1] is the learning rate, γ = [0; 1] is the discount factor, a_k is the action taken at time k, and r_{k+1} is the reward received in the next state. Q-learning has been proven to converge to the optimal policy when a finite state-action space is present [11]. In our case, we applied a learning rate to be equal to 1 as there are no stochastic transitions. It is generally the case that finding the tabular version of the Q-value function becomes challenging for the large state space problem due to the onerous task of having to having to store all the Q-values in memory for the large state space and infeasibility of visiting all the states. One way to overcome this problem is to use artificial neural networks (ANN) to estimate the Q-values and apply the interpolation capability to determine Q-values over the spread state-space (Q-surface). A basic form of the experience replay technique [12] which retrains multiple times on data collected during exploration is also applied. Introducing experience replay improves the behaviour of the agent when applying the trained Q-learner because the Q-values produced during training are tuned faster towards convergence. Also, due to the random sampling applied, when retraining over the same batch of tuples, the data presented for supervised learning is made uncorrelated. The form of the Q-learner used in our implementation is given in the block diagram in Figure 2. The reward/penalty logic applied to train the Q-learner can be summarised as follows. For each transition of the states shown in Figure 2, if (i) The miss prediction associated with the destination state decreased or (ii) The collision prediction associated with the destination state decreased or (iii) If the range between the true target and decoy with respect to the threat exceeded beyond the 50 m threshold, a suitable set of penalties scaled to the size of the change in the states was applied. This penalty was applied to calculate the appropriate Q-value estimate for that particular state transition.

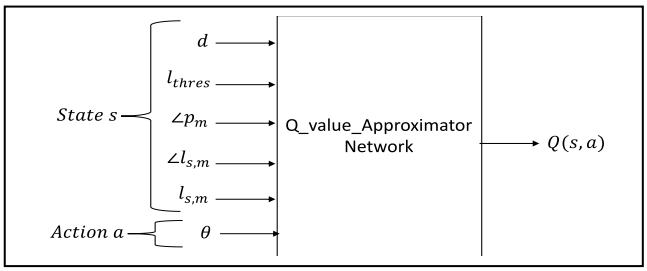

Fig. 2. 'Block diagarm depicting form of function approximator

Many tuples of $<s_t, a_t, r_t, s_{t+1}>$ were collected for transitions over multiple episodes in order to give as much coverage of the state-space as possible. The function approximator applied for Q-learning was simply a multi-layer perceptron (MLP) with a single hidden layer. Applying a single hidden layer was justified for this problem as the theory for function approximation using neural networks only requires a single hidden layer with sufficient number of neurons [13]. In Figure 2, d refers to the miss distance prediction between the missile and the target, given the current position and velocity of the threat. l_{thres} refers to the range threshold constraint between the missile and the agent with respect to the threat. $\angle p_m$ refers to direction of the velocity vector for the threat. $\angle l_{s,m}$ refers to the line of sight angle between the asset and the threat and $l_{s,m}$ refers to the range-to-go between the threat and the target. Finally, θ represents the action performed by the decoy, which, in this case, is simply to adjust its direction of motion.

Since the input features used by the Q-controller are continuous, an ANN based approximator is applied to estimate the Q-values in place of tabular Q-learning. Large miss distance is desirable but a sufficiently large enough miss (e.g. >100m) was acceptable. With the range-constraint, however, violating that early in the engagement would be unacceptable as it would have potentially triggered a 'being duped' signal in the threat. Since

adherence to the range constraint was highly critical during seduction, the penalty value for violating this threshold was configured as being five times greater compared to penalty applied for decrease in miss. In the simulation**,** this ratio worked out to give satisfactory performance.

IV. Simulation Configuration

For the problem targeted in this paper, 30 neurons in the single hidden layer [14] was applied for the function approximator representing the Q-estimator. This choice was determined through trial and error. Increasing the number of neurons in the hidden layer beyond this did not improve results in terms of maximizing miss/adhering to range constraint while reducing training time. Decreasing the number of neurons below 30 was not performed but could be conducted as future extension to this study. All reported results are averaged over 20 episodes of training for each choice of missile angle. The simulation was setup with the following initial conditions for initial conditions of the threat, false target and asset.

TABLE I. Simulation Configuration

Description	***Value***
Initial Distance between threat and true target	18000 m
Speed of true/false target	10.28 m/s
Speed of high-mobility threat	306 m/s
PN factor of the threat	4
Distance constraint: l_{thres}	50m

The author recognizes that, for the land context, the choice of initial separation range may need to be made smaller. However, this has not been attempted because the Q-learner technique is able to scale and the repetition of training is time-consuming. However, the experiment can be repeated in the future with smaller choice of initial separation range if necessary.

V. Results and Observations

The performance of the Q-learner controller under various approaching angles of the threats is provided below:

TABLE II. Simulation Results

Threat Approaching Angle (degrees)	***-90***	***-45***	***0***	***45***	***90***
Miss Distance achieved by decoying (m)	***78***	***825***	***1218***	***730***	***66***

As can be seen in Table 2 and Figure 3, the Q-learner controller was able to appropriately steer the decoy in order to sufficiently maximise the miss performance while adhering to the range-threshold constraint for most of the flight.

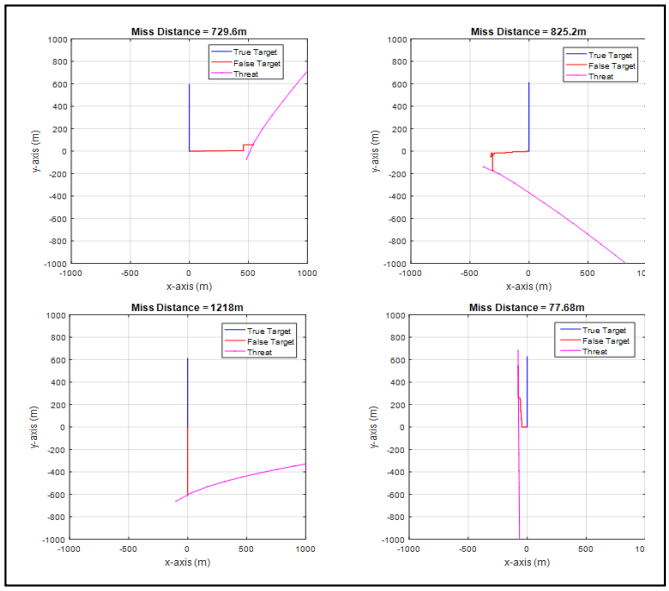

Fig. 3. Performance of Q-learning agent in directing false-target for different approach angles of threat.

The range-threshold adherence can be seen in Figure 3 to have applied all cases for most of the flight except for the case of the missile approaching from minus 90 degrees. In the 90 degree case, it can be seen that the learning agent has attempted to restrict the growth in the range-constraint, but cannot satisfy it due to the geometry and constraints on the false target speed. Otherwise, towards the end of the flight, for all cases, it becomes impossible to adhere to the range-constraint. However, it does not matter at this point for the range-constraint to be applied as the asset is now outside the sensing beam of the threat. Therefore, violating the threshold won't lead to the threat being alerted that it is being duped. The results presented in Figure 3, have only been provided for a subset of approach angles for the threat. It has been performed for the sake of communicating the trained Q-learner's capability. However, the Q-learner did behave appropriately for all other angles of the threat. Statistical analysis of the performance becomes relevant when allowing for stochastic state transitions. In our case following training, the Q-learner will lead to the same miss distance for repetitions of the simulation for different angles of threat approach.

VI. Future Research/Development Pathway

While the results indicate that the Q-learner can steer the decoy to seduce the threat appropriately, proving that it is converged is a different matter. This means being able to keep performing training until the Q-value evolution stops. In theory this happens when the state/action pairs are visited infinitely often for all state-action pairs [6]. However, due to the curse of dimensionality the state-space to be explored fully grows exponentially with number of features. In our case we have 5 features, each of which is continuous. Even if an ANN were to be applied as the approximator to

perform interpolation under the assumption of smoothness with change in q-values, the exploration requirement is still very large to get to near converged Q-surface. There have been several proposals to speed up the rate of convergence of Q-learning such as [15-17]. However, the visual results obtained here shown in Figures 3 and 4 suggest that there was sufficient training to produce a Q-learner capable of the successful completing the seduction task. These results were arrived at after 20 episodes of training with the choice of reward/penalty framework. So, it could be inferred that the trend for the Q-value surface can progress quickly towards converging to final q-values. However, to get to converged results (as in discovering the most optimal path for decoy through converged Q-values), further training maybe required. Finally the possibility of scaling this to the scenario of a coordinated multi-agent scenario needs to be investigated. The multi-agent scenario becomes relevant for protecting a group of land vehicles which can employ a distributed soft-kill strategy.

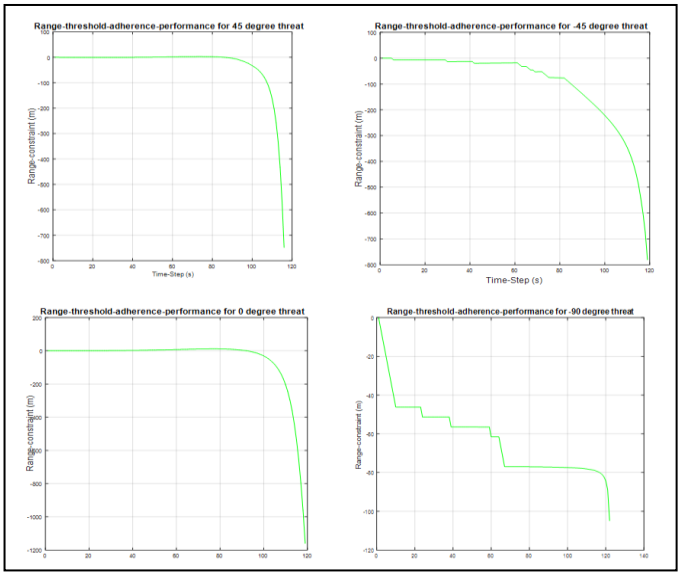

Fig. 4. Performance of Q-learning agent in directing false-target for different approach angles of threat.

Also, the dynamic programming approach is restrictive and instead a policy optimisation approach is recommended as the pathway to introduce RL learner for problems of this nature [18].

VII. Conclusion

Q-learning control has been used to guide a false-target to protect a true target against high mobility threat in a land-defence application. Results show that, in principle, the Q-learner based controller can train to seduce the threat for a range of approach angles while doing its best to adhere to the range constraint following limited training. Since several input variables (5) were required to uniquely describe the state, it is questionable whether the Q-learner explored the state space sufficiently to then have developed a converged Q-surface. Despite this, the results show that the controller has learned to steer the false target appropriately for the test-cases. To overcome this limitation, pathways have been identified to speed up Q-learning. Also, migration to policy optimization based RL approaches to serve as controller may be required in order to better provide for guarantees of convergence of the trained controller.

VIII. References

[1] R. Xiao-gang, "Foreign Tank Armored Vehicle Active Protection System Introduction," *Fire Control and Command Control*, vol. 35, pp. 4-6, 2010.

[2] M. Fegg and H. Bannasch, "Method for offering a phantom target, and decoy," ed: Google Patents, 2003.

[3] L. J. Z. Z. Z. Qingtai, "AN OPTIMUM LAW OF OVERHEAD ATTACK GUIDANCE FOR ANTI-TANK MISSILES [J]," *ACTA ARMAMENTARII*, vol. 2, p. 019, 1999.

[4] S. T. Duriez. Brunton, Noack B.R., "Machine Learning Control - Taming Nonlinear dynamics and Turbulence," in *Methods of Machine Learning*, ed Switzerland: Springer International Publishing Switzerland 2017, 2017.

[5] Z. Shiyu and Z. Rui, "Cooperative guidance for multimissile salvo attack," *Chinese Journal of Aeronautics*, vol. 21, pp. 533-539, 2008.

[6] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction* vol. 1: MIT press Cambridge, 1998.

[7] (2017, 2/05/2018). *Berkeley Deep RL Bootcamp*. Available: https://planspace.org/20170830-berkeley_deep_rl_bootcamp/

[8] K. Zhang, Z. Yang, H. Liu, T. Zhang, and T. Başar, "Fully Decentralized Multi-Agent Reinforcement Learning with Networked Agents," *arXiv preprint arXiv:1802.08757*, 2018.

[9] N. Bruce, "Expendable decoys," *Countermeasure Systems*, p. 288, 1996.

[10] P. Zarchan, "Tactical and strategic missile guidance(Book)," *Washington, DC: American Institute of Aeronautics and Astronautics, Inc, 1994.*, 1994.

[11] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, pp. 279-292, 1992.

[12] S. Adam, L. Busoniu, and R. Babuska, "Experience replay for real-time reinforcement learning control," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, pp. 201-212, 2012.

[13] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, pp. 303-314, 1989.

[14] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, pp. 359-366, 1989.

[15] M. G. Azar, R. Munos, M. Ghavamzadeh, and H. J. Kappen, "Speedy Q-learning," in *Proceedings of the 24th International Conference on Neural Information Processing Systems*, 2011, pp. 2411-2419.

[16] A. M. Devraj and S. P. Meyn, "Fastest Convergence for Q-learning," *arXiv preprint arXiv:1707.03770*, 2017.

[17] R. A. Bianchi, C. H. Ribeiro, and A. H. Costa, "Heuristically Accelerated Q–Learning: a new approach to speed up Reinforcement Learning," in *Brazilian Symposium on Artificial Intelligence*, 2004, pp. 245-254.

[18] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in neural information processing systems*, 2000, pp. 1057-1063.