Policy Learning in Imperfect-information Infinite Dynamic Games

Paper ID: #247

ABSTRACT

Dynamic games (DGs) play an important role in distributed decision making and control in complex environments. Finding optimal/approximate solutions for these games in the imperfect-information setting is currently a challenge for mathematicians and computer scientists, especially when state and action spaces are infinite. This paper presents an approach to this problem by using multi-agent reinforcement learning techniques. We hence propose a method of policy-space search based on variable resolution state/action abstraction. Stated precisely, a non-uniform partitioning of the state-action space of a perfect-information game version is used to parameterize stochastic policies to learn. We study in detail the application of this method to two-player zero-sum pursuit-evasion games (PEGs), including an anti-missile game. Experimental evaluations on this realistic PEG demonstrate good performance of our learning method and show that it gives better solutions than those given by traditional analytical methods.

General Terms

Learning, game theory.

Keywords

Multi-agent learning, policy-space search, variable resolution, infinite dynamic games, pursuit-evasion games.

1. INTRODUCTION

In this paper, we consider the application of multi-agent learning to the field of dynamic game theory [1], seen as a child of game theory and optimal control theory. Its emphasis has been more on obtaining effective solutions, rather than on solution existence questions. It has a more versatile character than that of its parents, because it involves a dynamic decision process evolving in (discrete or continuous) time, with more than one decision maker (or agent) possibly having access to different information.

XXXXX 'XX XXX, XXX

Researchers have recently been interested in DGs with imperfect information [11, 12, 16, 17], *i.e.* in the situation where some agents cannot obtain complete knowledge of the state variables. Since there is currently no general optimal solution method for such games, approximate methods have been considered. However, in the case where the game state space and the agent action spaces are all infinite, the complexity of approximate calculations is still prohibitive.

We thus introduce an approach to find approximate solutions for imperfect-information infinite dynamic games by using multi-agent reinforcement learning techniques. We propose a method of *policy search* [2, 4, 15, 21] based on *variable resolution* [20, 26, 27], which particularly uses Monte Carlo estimations and evaluations. Our method and its application to a realistic PEG have been evaluated on a specialized simulation environment.

Research on (perfect-information) infinite dynamic games (IDGs) has mainly concentrated on optimal deterministic solutions. Stochastic playing has become a necessary condition for imperfect information: an agent randomizes its play if it cannot obtain enough information to play optimal deterministic policies. Mathematicians have been able to calculate sub-optimal stochastic policies in some cases [13, 24], but generally, they have not succeeded in obtaining effective global optimal stochastic policies for imperfect-information IDGs.

We are the first to apply learning tools to the approximation of such stochastic policies. To deal with infinite state and action spaces, we rely upon the direct search in a parameterized policy space, and adapt it to multi-agent problems (see 5.4). Many classes of deterministic IDGs have optimal solution methods [6, 18] which can "guide" randomized plays. The problem is how to link the parameterization of a stochastic policy to an optimal deterministic solution (possibly) provided. We intend to non-uniformly partition the state-action space of the corresponding perfect-information IDG, then focus the stochastic policy on state-action areas where it can reach the optimal payoff (a maximum or minimum). These areas are definitely among those in which the payoff function varies most. With some modifications, variable resolution methods can help with partitioning such a state-action space.

In comparison to previous works, Harmon et *al.* [7] and Sheppard [23] applied their learning algorithms to IDGs in order to approximate deterministic policies, even in some situations with imperfect information. They therefore entirely ignored the stochastic playing in question. Concerning other multi-agent learning methods, little work has tackled infi-

nite problems. In this context, our work contributes to the same research orientation as that of Bowling and Veloso [4], who scaled up their GraWoLF algorithm for a stochastic multi-robot problem with infinite state space. Nevertheless, we recognize that the application of GraWoLF to imperfect-information IDGs would be inefficient because the stochastic policies' parameterization is not linked to deterministic optimal solutions, or in other words, these solutions (possibly calculated) cannot be used to speed up learning.

This paper is organized as follows. Section 2 presents IDGs by describing two typical games, *i.e.* the theoretical *homicidal chauffeur* game and a PEG model of an anti-missile problem. Section 3 specifies the imperfect-information setting for IDGs, with illustrations from these two games. Section 4 serves as an introduction to our learning approach, including game modeling for learning. Section 5 proposes the method of *resolution-based policy search*. Section 6 summarizes experimental evaluations performed within the framework of a research project related to the anti-missile problem; we compare our method to traditional analytical solution methods proposed for this problem. Finally, Section 7 draws conclusions and discusses future work.

2. INFINITE DYNAMIC GAMES

An infinite dynamic game (IDG), *i.e.* with infinite game state and agent action spaces, normally involves difference (in discrete time) or differential (in continuous time) equations [1], which describe the evolution of the underlying decision process. The optimal agents' policies are determined by the solution of these functional equations.

IDGs are supposed to be deterministic because research on these games has mainly focused on optimal deterministic solutions.

Basic results have been obtained for two-player zero-sum IDGs. In such a game, at each moment two agents compete for the same prize: an agent (called minimizer agent) minimizes a payoff $J(s)$ while the other agent (maximizer agent) maximizes it. $J(s)$ depending on the current game state s is defined as the final outcome if the agents play optimally in the remainder of the game. The solution of the functional equations determines the optimal payoff J^*:

$$J^* = \min_{\pi_1} \max_{\pi_2} J(s_1) \tag{1}$$

where π_1, π_2 are agents' policies, and s_1 is an initial game state.

This paper focuses on pursuit-evasion games (PEGs) [5, 10, 22], a well-known class of IDGs. A two-player zero-sum PEG is played by a *pursuer* agent, that tries to catch the other agent, and by an *evader* agent, whose mission is to prevent the capture.

2.1 Homicidal chauffeur game

The *homicidal chauffeur* game (HCG) [9, 18], a theoretical PEG, has become an important reference for IDGs. In this game, the driver of a car (pursuer P) seeks to capture a pedestrian (evader E). P moves faster, at a fixed speed, but manoeuvres slower, with a radius of curvature bounded by a given quantity R. E moves with simple motion, at a slower speed, and steers by choosing its direction of travel. A capture occurs when the distance between P and E is inferior to a capture radius. The payoff J is the time of capture.

Isaacs [9] and Merz [18] provided the solution to the HCG as partially illustrated in Figure 1. In the situation at (a), E is in front of P and a little to his right. P traverses the right circle of maximal curvature (the sharpest possible right turn) to P_1, then follows the tangent as shown. E also pursues along this tangent until capture occurs.

At (b) of Figure 1, E starts from a rear position with respect to P. This one acts less directly; first, it goes away from E and, when far enough away, it starts a more direct pursuit (as the path sketched). On the other hand, E strives to delay P's achieving an adequate distance by following P to some point E_1, at which it turns tail and flees. The game concludes with a direct chase as at (a).

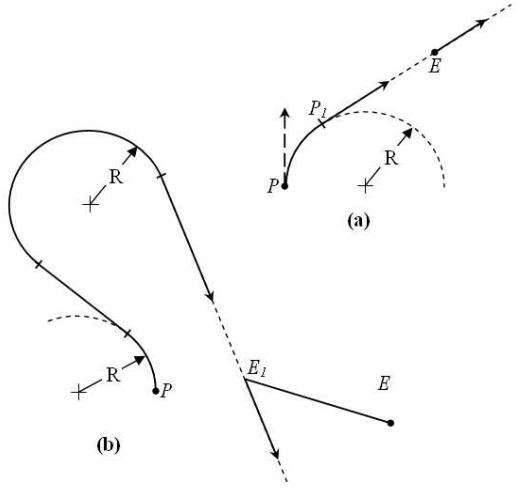

Figure 1: Partial solution of the HCG.

2.2 Anti-missile game

Another PEG that we present here has played an important role in the research on anti-missile problems [13]. Although its motion is only one-dimensional, it has more state variables than the HCG. These variables are, moreover, bounded by realistic constraints.

The situation considered is when a defensive missile launched from a ship has to intercept an incoming missile. The velocity elements subjected to the straight line along which the threatening missile comes to the ship are assumed to be constant. All the policies that the two agents can play are thus concerned with the variable velocity and acceleration elements perpendicular to this line. The pursuer P_m (defensive missile) can move faster but manoeuvre slower than the evader E_m (incoming one). In addition, the game is constrained by time, trajectory and control limits.

A particular state variable called *zero-effort miss distance* Z [13] is calculated at each moment. It is in fact the minimal relative separation that two agents can reach if they do not change their acceleration. The payoff, that P_m minimizes and E_m maximizes, is the final value of Z at the end of the game.

Gutman [6] introduced the following solution of the game:

$$\frac{a_p^*(\theta)}{(a_p)_{max}} = \frac{a_e^*(\theta)}{(a_e)_{max}} = \text{sgn}(Z(\theta)), \forall \theta < \theta_{endgame} \tag{2}$$

where θ is the normalized remaining time, $a_p(\theta)$ and $a_e(\theta)$

are respectively the accelerations of P_m and E_m, $a_p^*(\theta)$ and $a_e^*(\theta)$ are their optimal values.

This result suggests that before the interception moment (if it happens) E_m performs its maximal manoeuvre (lateral acceleration) in only one direction during at least $\theta_{endgame}$, and P_m follows it by performing its maximal manoeuvre.

3. IMPERFECT-INFORMATION SETTING

An IDG is said to be in the imperfect-information setting if some agent, called "blind" agent, cannot obtain complete knowledge about the positions and actions of the other agent.

3.1 Stochastic playing

Because of the lack of information about the other agent, a "blind" agent plays a stochastic policy while hoping for some probability of success. Consequently, any "no blind" agent, in its turn, also plays a stochastic policy, still takes advantage of full information. At each moment, the agents cannot directly minimize/maximize the targeted payoff J of the original perfect-information game (see Equation 1), and thus seeks to minimize/maximize the probability J_{pr} that this initial payoff is superior to some threshold ζ:

$$J_{pr} = Pr\{J > \zeta\} = 1 - Pr\{J \leq \zeta\} \quad (3)$$

The choice of ζ for a specific imperfect-information game can be based on the optimal payoff J^* of the original perfect-information game (if it exists) and on which agent is "blind". For example, if J^* is known and only the maximizer agent is "blind", one can choose ζ such that $J^* - \epsilon < \zeta < J^*$, with $0 < \epsilon < J^*$. In this case, the "blind" agent tries to increase the chance of obtaining a payoff close to the optimal maximized payoff.

Such stochastic IDGs still cannot be totally solved. Researchers mainly use solutions of perfect information games as guides for stochastic analyzes of imperfect-information games [9, 13].

It is supposed that, in the HCG (see 2.1), the "blind" evader E has imperfect information as to P's position and the degree of imperfection does not change with time. For instance, Isaacs [9] introduced a typical situation in which at each instant the only information granted E is that P is equiprobably within a sphere of radius r. He also recommended a randomized (stochastic) play for this imperfect-information game, which is decomposed into two phases. At first, E estimates the suspected locale of P, then evades with much the same tactics as if it had precise knowledge. Later, when P is close upon its heels, the E's play is randomized. P counteracts by a random pursuit.

3.2 Randomized missile control

In the anti-missile game (AMG) presented in 2.2, the evader E_m can be "blind", *i.e.* it cannot obtain any information of the positions of the pursuer P_m. It does not know the launching moment of P_m, so it does not know the interception moment either. Lipman et *al.* [13] proposed a randomized play for this game: E_m's purpose is to maximize the avoidance probability Pr_a, *i.e.* the probability that $|Z(0)|$ is superior to P_m's interception radius; P_m, in its turn, aims to minimize Pr_a.

Lipman et *al.* [13] analyzed the game in some cases. By following the suggestion from the solution of the perfect information game, E_m can perform its maximal manoeuvre, and switch its manoeuvre direction at random moments. It is also suggested that it maintains a manoeuvre direction during at least $\theta_{endgame}$ before each switching. The random duration between any two consecutive direction switches can follow a distribution $F(\theta)$, *e.g.* the following random telegraph type:

$$F(\theta) = 1 - \exp(-\lambda\theta) \quad (4)$$

This is an example of E_m's stochastic policy. To ensure some interception probability $(1 - Pr_a)$, P_m follows E_m by performing randomly biased manoeuvres (see [13]).

We recognize that the two assumptions that E_m always performs its maximal manoeuvre and that its direction switching follows the random telegraph rule are not general, and of course, do not guarantee an optimal solution. In fact, the complexity of the mathematical calculation of optimal stochastic policies is so prohibitive that researchers have no choice but to impose such specific assumptions.

3.3 Approximate solutions

When a problem cannot be optimally solved yet, one tries to find approximate solutions. A natural way to do that is to rely on the discretization, which leads to numerical algorithms.

We now return to the imperfect-information AMG described in 3.2. Shinar and Silberman [24] proposed a discrete model of this game, then solved it by establishing a global matrix game from all the non-dominated deterministic policies (pure policies), calculated for both agents. The two stochastic policies to be found are in fact the Nash equilibrium of the matrix game. Such policies must be totally pre-calculated before one makes the agents play the game. The size of the matrix as well as the policy calculation complexity are exponential in state variables, *e.g.* in the normalized remaining time (discretized θ).

4. LEARNING APPROACH

4.1 How to learn stochastic policies?

With the aim of reducing the complexity of the stochastic policies' calculation while ensuring their high quality, we rely on the application of multi-agent reinforcement learning techniques.

As shown by Shinar [24], the state/action abstraction's resolution strongly affects numerical solutions of IDGs. This is not only the question of how high the resolution is, but also if the resolution is not uniform among different parts of the state or action space. Sheppard [23] proved that an uniform sampling does not guarantee sufficiently good solutions for an IDG. We thus propose a new method called *resolution-based policy search* in Section 5.

We intend to learn stochastic policies for IDGs in the imperfect-information setting. If we use a computer simulation as learning environment, we can provide perfect information to all the agents. This allows, firstly, learning policies for an *intermediate IDG version* (modeled in 4.2), that can be defined as an IDG in which the agents are provided perfect information, but still play stochastic policies and seek to minimize/maximize the same statistical quantity as in the imperfect-information game (see Equation 3). The overall learning process can thus be decomposed into two steps:

- the agents'policies for an intermediate IDG version are learned off-line by using the *resolution-based policy search*;
- starting from these policies, those for the corresponding imperfect-information IDG are estimated by using Monte Carlo experiments (illustrated in 6.1).

4.2 Game modeling

To make agents learn their own stochastic policies, we model an imperfect-information IDG as a discrete-time decision-making process. Consequently, the (discrete or continuous) time is uniformly (re-)discretized into time steps $\Delta\tau$. After each $\Delta\tau$, each agent falls into a state $s_{ag_i} \in \mathbf{S}_{ag_i}$ and chooses an action $a_{ag_i} \in \mathbf{A}_{ag_i}$ according to a stochastic policy, $\pi_{ag_i} : \mathbf{S}_{ag_i} \times \mathbf{A}_{ag_i} \rightarrow [0,1]$. The objective of the agent's playing is to maximize the discounted sum of a reward function, $R_{ag_i} : \mathbf{S}_{ag_i} \times \mathbf{A}_{joint} \rightarrow \Re$, over a finite horizon of T time steps. This game model is not a Markov game yet [14] because the reward on the joint state, $\widetilde{R}_{ag_i} : \mathbf{S}_{joint} \times \mathbf{A}_{joint} \rightarrow \Re$ cannot be mathematically represented. ($\mathbf{S}_{ag_i}$, $\mathbf{A}_{ag_i}$, $\mathbf{S}_{joint}$ and $\mathbf{A}_{joint}$ are all infinite.)

Notice that the time, as a state variable, always keeps its continuous nature or original small discrete steps, independent of the decision-making time step $\Delta\tau$. That means that if we change $\Delta\tau$, the time elements of any state remain the same, and the agents can always use the same policies $\pi_{ag_i}, i = 1, ..., N_{agent}$, for both playing and learning.

In a simulation environment, we can provide all the information to all the agents, firstly to learn policies for an intermediate IDG version (see 4.1). Based on the imperfect-information modeling previously presented, such a game version can be modeled in the form of a Markov game [14]. Indeed, because of full information provided, at any decision-making moment, the state of each agent $s_{ag_i} \in \mathbf{S}_{ag_i}$ is equivalent to the joint state $s_{joint} \in \mathbf{S}_{joint}$. Hence, there exists, for each agent, only one reward function, $\widetilde{R}_{ag_i} \equiv R_{ag_i}$. Moreover, this kind of Markov game has a deterministic transition function, $Tr : \mathbf{S}_{joint} \times \mathbf{A}_{joint} \times \mathbf{S}_{joint} \rightarrow \{0,1\}$.

4.3 Discussions

Our work begins with the application of reinforcement learning to the imperfect-information AMG (see 3.2) within the framework of a research project. The randomized play of this game has been better defined by researchers than that of the HCG (see 3.1), in such a way that we do not need to add assumptions. Our learning method has behaved well in experiments performed on the AMG (see Section 6).

We afterwards examine the application of the method to an important theoretical IDG, *i.e.* the HCG, whose randomized play in the imperfect-information setting has been not adequately formalized by previous works [8, 9]. We hence determine typical situations in which we can learn stochastic policies. For example, P and E can learn their policies for the final phase of the game (see 3.1), when P gets close upon E's heels and the suspected locale of P is not relevant to E's play anymore. Our preliminary results on the HCG strengthen our belief in the extension of the work to other IDGs.

One could find that our method does not depend on a specific game. However, its performance must be carefully examined. For instance, to ensure the learning efficiency we currently suppose that the initial agents' states with respect to the learning process are fixed (see 5.3). This assumption has been commonly chosen by researchers while learning stochastic policies [2, 25]. But it is not always obvious for a game like the HCG, *e.g.* at the beginning of the final phase, P's position relative to E is not necessary fixed. We deal with this problem by allowing the initial states to be limitedly variable: P's start position is within a sphere of radius r_{ini}. The dependance of the learning efficiency on the initial states' variation is to be studied.

It is also straightforward to extend the method to general-sum IDGs, in which the agents use independent payoff functions. A needed change is that each agent's policy must be parameterized according to its own payoff function.

5. RESOLUTION-BASED POLICY SEARCH

Similar to any policy search method, we aim to make the agents learn their parameterized policies, *i.e.* progressively update their policies' parameters. The policy parameterization is particularly based on the *variable resolution*. The agents' capacity to adapt to the opponent exposes the multi-agent property of our learning method.

5.1 Variable resolution

As discussed in 4.1, the state/action abstraction's resolution strongly affects numerical solutions of IDGs. We thus employ an existing variable resolution method [20, 26, 27] to non-uniformly partition the state-action space of a perfect information IDG version, then use this partitioning to build parameterized stochastic policies for the corresponding intermediate IDG version (see 5.2). Additionally, we seek to limit the number of state-action areas.

To use variable resolution methods to abstract a perfect information IDG, we must make sure that its abstraction can be represented by a Markov decision process (MDP) without loss of generality. This condition is satisfied by most of the IDGs we have studied (including the HCG and the AMG) because researchers often impose a Markovian restriction so that agents' policies depend only on time and the current game state. In this case, the "*argmax* action" at each MDP's state and the corresponding immediate reward need to be replaced respectively by the agents' optimal joint action at each game state and by the corresponding change of payoff ΔJ (see Section 2).

The existing variable resolution algorithms are mainly based on the definition of state/action splitting criteria. We are interested in the use of valued criteria (not yes/no criteria) [20, 26, 27]. We begin with a uniform partitioning and then improve it. At each "splitting iteration", we select N_{sa} state-action areas with the highest criterion values and split them. At each "merging iteration", we select N_{sa} adjacent area pairs with the lowest criterion values and merge each of them. This process stops after N_{it} iterations or when the criterion values of all the state-action areas are inferior to some threshold.

In this way, we could obtain m state-action samples $s_{joint,i} \in \mathbf{S}_{joint} \times \mathbf{A}_{joint}, i = 1, ..., m$, from which we can derive for each agent m state-action samples $s_i \in \mathbf{S} \times \mathbf{A}, i = 1, ..., m$, and the corresponding set of n state-action areas $\{(\Delta s)_j | j = 1, ..., n\}$.

5.2 Parameterized policy

Our objective is to build for each agent a policy function. We have partitioned the state-action space into areas. We

now assume that all the state-action areas initially have the same probability. Afterwards, we define q parameters that modify the probability of the q state-action areas considered as the most significant. The policy parameterization algorithm can be represented as follows:

Inputs A sampling $s_i \in \mathbf{S} \times \mathbf{A}, i = 1, ..., m$ (or n state-action areas); the required number of parameters q.

Output A policy function $\pi(s)$ having q parameters w_j, with $j = 1, ..., q$.

1. Assuming that the initial probability of being in any state-action area $(\Delta s)_i, i = 1, ..., n$, is the same and equal to $1/n$, calculate for each area
$$\widehat{\pi}((\Delta s)_i) = \frac{1}{n(\Delta s)_i} \tag{5}$$

2. Determine the q most significant state-action areas $(\Delta s)_j, j = 1, ..., q$, whose $\widehat{\pi}((\Delta s)_j)$ is to be modified (see Appendix A).

3. Define q parameters $w_j, j = 1, ..., q$, satisfying the following q linear equations:
$$(\Delta s)_j \widehat{\pi}((\Delta s)_j) = \tag{6}$$
$$(\Delta s)_j w_j - \frac{1}{q-1}\Big(\sum_{k=1}^{j-1}(\Delta s)_k w_k + \sum_{k=j+1}^{q}(\Delta s)_k w_k\Big)$$
We can represent $\widehat{\pi}((\Delta s)_j) = f_{(\Delta s)_j}(w_1, ..., w_q)$, with $j = 1, ..., q$.

4. Build the policy function $\pi(s)$ by smoothly interpolating all the $\widehat{\pi}((\Delta s)_i), i = 1, ..., n$, including the q linear functions of the parameters $f_{(\Delta s)_j}(w_1, ..., w_q), j = 1, ..., q$; then normalize $\pi(s)$.

Concerning the notation, from now on we use $\pi(s, a)$ instead of $\pi(s)$.

5.3 Policy update

The purpose of the learning process for a single agent is to maximize the global performance ρ of the policy $\pi(s, a)$, according to [2, 25]:
$$\rho(\pi) = \sum_{se} V_\pi(se), \tag{7}$$
$$\text{with } V_\pi(se) = \sum_{t=1}^{T}\Big[\prod_{j=1}^{t}\pi(s_j, a_j)\Big]\gamma^t R(s_t, a_t)$$

where the sequence $se = (s_1, a_1, ..., s_T, a_T) \in \mathbf{SE}_T$ represents a trial, and $R(s, a)$ is the reward function (to simplify the formalism, we do not take into account the other agent yet).

After performing a trial (or several trials) by following the current greedy policy with some exploration (not detailed here), we update the parameters of the policy function by climbing the gradient of performance:
$$\Delta w_i = \alpha \frac{\partial V_\pi(se)}{\partial w_i}$$
$$= \alpha \sum_{t=1}^{T}\Big(\sum_{j=1}^{t}\frac{\partial \ln \pi(s_j, a_j)}{\partial w_i}\Big) P_{\pi,t}(s_t)\gamma^t R(s_t, a_t), \tag{8}$$
$$\text{with } P_{\pi,t}(s_t) = \prod_{j=1}^{t}\pi(s_j, a_j)$$

where α is the learning rate, γ is the discount factor, and $P_{\pi,t}(s_t)$ is the probability to be in s_t after t steps while following π from a single start state s_1, assumed to be fixed.

5.4 Adaptation to the opponent

The main challenge of the application of a mono-agent learning technique to a multi-agent problem is the adaptation to the other learning agents which make the world not stationary. In this paper, we deal with the simplest case of adaptation to a single opponent. Hence, the policy update is reformulated as follows:
$$w_i \leftarrow w_i + \alpha \sum_{t=1}^{T}\Big[\Big(\sum_{j=1}^{t}\frac{\partial \ln \pi_1(s_{1,j}, a_{1,j})}{\partial w_i}\Big)$$
$$P_{\pi_1,\pi_2,t}(s_{1,t}, s_{2,t})\gamma^t R(s_{1,t}, s_{2,t}, a_{1,t}, a_{2,t})\Big] \tag{9}$$

where the subscript 1 indicates the agent which makes the update, and the subscript 2 indicates the opponent.

Although one of the agents is "blind", during an off-line learning process using simulations, we can provide perfect information to all the agents (see 4.2). The agents' state is therefore common, and the policy update is still reformulated as follows:
$$w_i \leftarrow w_i + \alpha \sum_{t=1}^{T}\Big[\Big(\sum_{j=1}^{t}\frac{\partial \ln \pi_1(s_j, a_{1,j})}{\partial w_i}\Big)$$
$$P_{\pi_1,\pi_2,t}(s_t)\gamma^t R(s_t, a_{1,t}, a_{2,t})\Big] \tag{10}$$

A well-known method to adapt to a learning opponent is "*Win or Learn Fast*" proposed by [3], which suggests that the learning rate should be small when the policy π is better than an average policy $\overline{\pi}$, and should be big otherwise, as follows:
$$\overline{\pi}_1(s, a) \leftarrow \overline{\pi}_1(s, a) + \frac{1}{N_{trial}}(\pi_1(s, a) - \overline{\pi}_1(s, a)) \tag{11}$$

$$\alpha = \begin{cases} \alpha_{min} & \text{if } V_{\pi_1,\pi_2}(se) > V_{\overline{\pi}_1,\pi_2}(se) \\ \alpha_{max} & \text{otherwise} \end{cases}, \tag{12}$$
$$\text{with } V_{\pi_1,\pi_2}(se) =$$
$$\sum_{t=1}^{T}\Big[\prod_{j=1}^{t}\pi_1(s_j, a_{1,j})\pi_2(s_j, a_{2,j})\Big]\gamma^t R(s_t, a_{1,t}, a_{2,t})$$

where N_{trial} is the number of performed trials.

6. EXPERIMENTAL EVALUATIONS

Within the framework of a collaborative research projet, the application of the *resolution-based policy search* to the AMG has been evaluated by experiments performed on a specialized simulation environment.

6.1 Learning

In the perfect information setting, the optimal deterministic play of the AMG introduced in 2.2 can be informally specified as follows:

- At first, the evader E_m begins to play and does not change its position;
- The pursuer P_m chooses its launching moment, then follows E_m by maximally accelerating;
- Based on the launching moment of P_m, E_m calculates the interception moment t_f and the end game's starting moment $t_f - \theta_{endgame}$ (see 2.2);
- At the right time, P_m starts the end game by maximally accelerating in an unique direction.

This solution can be presented in the agent's reduced state space (θ, Z) (see 2.2), as illustrated in Figure 2, which moreover shows the partitioning of this space (see 5.1) and the q state-action areas obtained (see 5.2) in the case where $\langle n, q\rangle = \langle 32 \times 9, 12 + 4\rangle$[1]. E_m's action a_e (lateral acceleration) is also divided into 3 areas: $[-(a_e)_{max}, -0.87(a_e)_{max})$, $[-0.87(a_e)_{max}, 0.81(a_e)_{max})$,$[0.81(a_e)_{max}, (a_e)_{max}]$. And a similar decomposition is obtained for P_m: $[-(a_p)_{max}, -0.89(a_p)_{max})$, $[-0.89(a_p)_{max}, 0.93(a_p)_{max})$, $[0.93(a_p)_{max}, (a_p)_{max}]$. We obtain similar results for $n = 32 \times 9, 50 \times 9, 72 \times 9$ and $q = 12 + 4, 16 + 4, 24 + 4$.

Starting from these results, we build 9 pairs of q-parameter policy functions $\langle \pi'_e(\theta, Z, a_e), \pi'_p(\theta, Z, a_p)\rangle$ for the corresponding intermediate game (defined in 4.1).

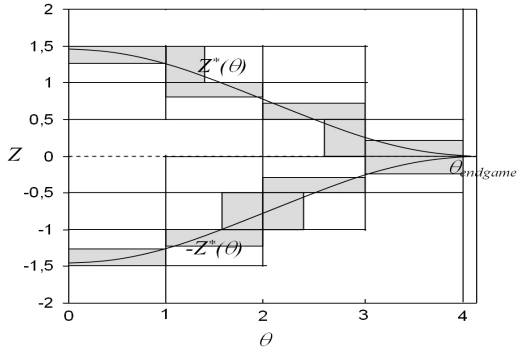

Figure 2: Partitioning of the (θ, Z) space ($n = 32 \times 9$, $q = 12 + 4$).

To learn π'_e and π'_p, we use a specialized simulation environment. We choose $\Delta\tau = 1s$, $\alpha_{max} = 0.25$ and $\alpha_{min} = 0.015$. $R_e(\theta, Z, a_e, a_p)$ and $R_p(\theta, Z, a_e, a_p)$ are respectively estimated by using the statistical avoidance probability of E_m and the statistical interception probability of P_m. These probabilities are initially set to 0.5, then accumulated and linearly interpolated after each trial. After performing 1000 trials for each pair $\langle n, q\rangle$, $n = 32 \times 9, 50 \times 9, 72 \times 9$, $q = 12 + 4, 16 + 4, 24 + 4$, we obtain 9 pairs of policies $\langle \pi'_e, \pi'_p\rangle$.

We now illustrate the Monte Carlo estimation of policies for the corresponding imperfect-information game. Because a "blind" evader does not know the positions and the launching moment of P_m, its policy can be formulated as

$$\pi_e(t, a_e) = \int Pr(\theta, Z \mid t)\pi'_e(\theta, Z, a_e)dt$$

where t is the (continuous or finely discretized) playing time calculated from the game's beginning. P_m does not choose a fixed launching moment, so its policy can be reformulated as

$$\pi_p(t, Z, a_p) = \int Pr(\theta \mid t)\pi'_p(\theta, Z, a_p)dt$$

If we rewrite $Pr(\theta, Z \mid t) = Pr(\theta \mid t)Pr(Z \mid t)$, we will directly estimate only two probabilities $Pr(\theta \mid t)$ and $Pr(Z \mid t)$ by using Monte Carlo experiments and the linear interpolation. Indeed, for each pair of policies $\langle \pi'_e, \pi'_p\rangle$, we perform 1000 simulation runs, with P_m's choice of launching moment following a uniform probability distribution. In this way, we obtain 9 pairs of policies $\langle \pi_e, \pi_p\rangle$ for the imperfect-information game.

6.2 Results

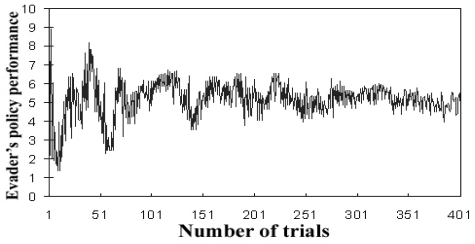

Figure 3: Evader's current policy performance while learning ($n = 72 \times 9$, $q = 24 + 4$, $\Delta\tau = 1s$).

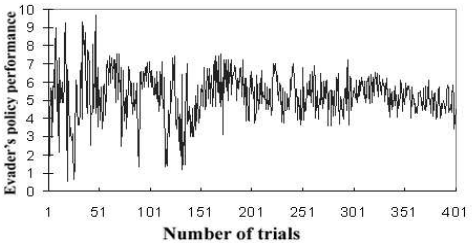

Figure 4: Evader's current policy performance while learning ($n = 72 \times 9$, $q = 24 + 4$, $\Delta\tau = 0.5s$).

Figure 3 and Figure 4 present the variation of E_m's current policy performance *while learning* (see 5.3) where $n = 72 \times 9, q = 24 + 4, \Delta\tau = 1s, 0.5s$. Both the graphs show clear convergence tendencies. By comparing these two, we recognize that the smaller decision-making time step $\Delta\tau$ is, the more the current policy performance varies, and as a result, the longer the learning time required is. From now on, we use $\Delta\tau = 1s$.

We evaluate any pair of policies $\langle \pi_e, \pi_p\rangle$, by performing 1000 Monte Carlo runs. The avoidance probability of E_m as well as the interception probability of P_m are all estimated.

Table 1 shows the avoidance probability of E_m for the 9 pairs of learned policies $\langle \pi_e, \pi_p\rangle$. We can see that n still

[1]Among 9 common action areas (3 individual action areas for each agent), we choose 4 areas (2 individual areas for each agent) that correspond to 4 parameters.

n	$q = 12 + 4$	$q = 16 + 4$	$q = 24 + 4$
32×9	0.412	0.432	0.459
50×9	0.477	0.508	0.514
72×9	0.525	0.574	0.632

Table 1: Avoidance probability against the learned pursuer.

significantly influences the obtained solution even if the partitioning of the state-action space is already optimized (see 5.1). On the other hand, because of E_m's stronger lateral acceleration capacity, the bigger n and q are, the freer E_m is, and therefore, the more it can avoid P_m's interception.

We afterward employ a technique, which was first used by Littman [14] and then by Bowling and Veloso [4], to evaluate the resistance of the learned policies to their worst-case opponents. The better this resistance is, the less exploitable the learned agents are. After learning the 9 pairs of policies $\langle \pi'_e, \pi'_p \rangle$, we fixe all E_m's policies and continue to train P_m's policies by performing 1000 trials for each pair. In this way, we transform P_m into E_m's worst-case opponent, called challenger. Consequently, we obtain 9 pairs of policies $\langle \pi'_e, \pi'_c \rangle$, where π'_c is the challenger's policy in the corresponding intermediate game. We then estimate 9 pair of policies $\langle \pi_e, \pi_c \rangle$ for the imperfect-information game in the same way as we have done in 6.1. We evaluate the resistance of the 9 policies π_e to their worst-case opponents by performing 1000 Monte Carlo runs for each pair $\langle \pi_e, \pi_c \rangle$.

n	$q = 12 + 4$		$q = 16 + 4$		$q = 24 + 4$	
32×9	0.394	-4.3%	0.416	-3.7%	0.445	-3.1%
50×9	0.460	-3.6%	0.501	-1.4%	0.507	-1.4%
72×9	0.512	-2.5%	0.568	-1.0%	0.624	-1.3%

Table 2: Avoidance probability against the challenger.

Table 2 presents the avoidance probability of E_m for the 9 pairs of policies $\langle \pi_e, \pi_c \rangle$. The challenger's mission is to exploit E_m's policy, so it generally decreases the avoidance probability. Therefore, we associate to each probability a percentage that indicates the difference from the corresponding probability in Table 1 and represents the the policy's exploitation. It is obvious that these differences are quite small. The results also show that E_m's policy seems more difficult to exploit, or more robust, when n and q increase. However, this tendency is not totally justified when n and q become relatively big. We think that is because the quality of E_m's policy, including its robustness, moreover depends on the number of learning trials. The bigger n and q are, the bigger the number of learning trials required to ensure a given quality level is.

Finally, to compare solution methods for the AMG, we implement two kinds of agent, whose maximal manoeuvres have been mathematically proven to be optimal by traditional analytical methods [13, 24]:

- a deterministic pursuer that always follows E_m by maximally accelerating,
- an end-game evader that estimates the end game's starting moment according to a uniform probability distribution, then plays the end game by maximally accelerating in an unique direction.

$\langle n, q \rangle$	LE *vs.* LP	LE *vs.* DP	EE *vs.* LP
$\langle 50 \times 9, 16 + 4 \rangle$	0.508	0.918	0.277
$\langle 72 \times 9, 24 + 4 \rangle$	0.632	0.944	0.357

Table 3: Avoidance probability related to non-learned agents.

Table 3 shows the probability of avoidance for the three cases: a learned evader (LE) *vs.* a learned pursuer (LP), an LE *vs.* a deterministic pursuer (DP), an end-game evader (EE) *vs.* an LP. The LE and LP's policies are learned with $\langle n, q \rangle = \langle 50 \times 9, 16 + 4 \rangle, \langle 72 \times 9, 24 + 4 \rangle$. These results prove that, while playing against an LP, an LE always gets a higher avoidance probability than an EE. Similarly, while playing against an LE, an LP always gets a lower avoidance probability than a DP. Moreover, in comparison with the analytical results given by Lipman et *al.* [13] and Shinar and Silberman [24], our learning-based method gives better solutions in the sense that when an LE plays against an LP it always gets a better avoidance probability. In other words, from the defensive point of view we would better predict the strongest attacks, and methodologically we would better deal with imperfect information.

7. CONCLUSION

Finding stochastic policies for imperfect-information IDGs [1] has faced a big problem of computational complexity. In this paper, we have proposed a learning method, called *resolution-based policy search*, approximating such stochastic policies. The learning performance depends on several parameters, *e.g.* the number of abstract state-action areas n, the number of parameters of a policy function q, and the decision-making time step $\Delta\tau$. Experimental results have shown that we can choose these parameters' values so that we obtain good solutions after a reasonable number of trials. Concerning the AMG (a realistic PEG), with some appropriate learning parameter values, the learned pursuer and evader have proven their greater power, in comparison to two typical kinds of non-learned agent, whose manoeuvres have been mathematically proven to be optimal by traditional analytical methods [13, 24].

Future work will provide more investigation into the experimental behavior of the learning method presented, and into the non-uniform discretization of the decision-making time, as suggested by Munos [19].

8. REFERENCES

[1] T. Başar and G. J. Olsder. *Dynamic Noncooperative Game Theory, 2nd Ed.* Academic, London, 1995.

[2] L. Baird and A. W. Moore. Gradient descent for general reinforcement learning. In *NIPS'98*, pages 968–974, Cambridge, MA, USA, 1998.

[3] M. H. Bowling and M. M. Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136(2):215–250, 2002.

[4] M. H. Bowling and M. M. Veloso. Simultaneous adversarial multirobot learning. In *IJCAI'03*, Acapulco, Mexico, 2003.

[5] J. M. Eklund, J. Sprinkle, and S. Sastry. Implementing and testing a nonlinear model predictive tracking controller for aerial pursuit evasion games on a fixed wing aircraft. In *American Control Conference*, Portland, OR, USA, 2005.

[6] S. Gutman. On optimal guidance for homing missiles. *Guidance and Control*, 2:296–300, August 1979.

[7] M. E. Harmon, L. C. Baird, and A. H. Klopf. Advantage updating applied to a differential game. In *NIPS'95*, pages 353–360, Cambridge, MA, USA, 1995.

[8] R. Isaacs. *Games of pursuit*. Scientific report of the RAND corporation, Santa Monica, USA, 1951.

[9] R. Isaacs. *Differential Games: A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization*. J.Wiley and Sons, Toronto, Canada, 1965.

[10] V. Isler, S. Kannan, and S. Khanna. Randomized pursuit-evasion with local visibility. *SIAM Journal on Discrete Mathematics*, 1(20):26–41, 2006.

[11] M. Jimenez-Lizarraga and A. Poznyak. Near-nash equilibrium strategies for lq differential games with inaccurate state information. *Mathematical Problems in Engineering*, 2006.

[12] Y. Lipman and J. Shinar. Mixed strategy guidance in future ship defense. *Guidance, Control and Dynamics*, 19(2):334–339, June 1996.

[13] Y. Lipman, J. Shinar, and Y. Oshman. Stochastic analysis of the interception of maneuvering antisurface missiles. *Guidance, Control and Dynamics*, 20(4):707–714, July–August 1997.

[14] M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *ICML'94*, pages 157–163, New Brunswick, NJ, 1994.

[15] P. Marbach and J. N. Tsitsiklis. Approximate gradient methods in policy-space optimization of markov reward processes. *Discrete Event Dynamic Systems*, 13(1-2):111–148, 2003.

[16] W. M. McEneaney. Some classes of imperfect information finite state-space stochastic games with finite-dimensional solutions. *Applied Mathematics and Optimization*, 50(2):87–118, August 2004.

[17] W. M. McEneaney and B. G. Fitzpatrick. Control for uav operations under imperfect information. In *1st AIAA UAV Symposium*, Portsmouth, VA, USA, 2002.

[18] A. W. Merz. *The homicidal chauffeur - a differential game*. Technical report, Guidance and Control Laboratory 418, Stanford University, Stanford, CA, USA, 1971.

[19] R. Munos. Policy gradient in continuous time. *Machine Learning*, 7:771–791, 2006.

[20] R. Munos and A. W. Moore. Variable resolution discretization for high-accuracy solutions of optimal control problems. In *IJCAI'99*, pages 1348–1355, 1999.

[21] J. Peters, S. Vijayakumar, and S. Schaal. Natural actor-critic. In *ECML'05*, pages 280–291, 2005.

[22] L. Schenato, S. Oh, and S. Sastry. Swarm coordination for pursuit evasion games using sensor networks. In *International Conference on Robotics and Automation*, Barcelona, Spain, 2005.

[23] J. W. Sheppard. Colearning in differential games. *Machine Learning*, 33(2-3):201–233, 1998.

[24] J. Shinar and G. Silberman. A discrete dynamic game modelling anti-missile defense scenarios. *Dynamics and Control*, 5(1):55–67, 1995.

[25] R. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *NIPS'99*, pages 1057–1063, Cambridge, MA, USA, 1999.

[26] W. T. B. Uther and M. M. Veloso. Tree based discretization for continuous state space reinforcement learning. In *AAAI/IAAI'98*, pages 769–774, 1998.

[27] W. T. B. Uther and M. M. Veloso. Ttree: Tree-based state generalization with temporally abstract actions. In *Adaptive Agents and Multi-Agents Systems*, pages 260–290, 2002.

APPENDIX

A. Q STATE-ACTION AREAS

The aim of this algorithm is to robustly determine the q most significant areas among n state-action areas provided. These are where the deterministic payoff (see Section 2) varies most. Knowing that a modified variable resolution method (see 5.1) gives us n state-action areas which have uniform payoff variance (splitting criterion value), the q areas to be selected are the smallest or next to the smallest areas. A detailed description of such a robust algorithm would be complicated. The following presentation is for illustrative purposes only. Notice that adjacent areas have adjacent indexes.

Input A set of state-action areas $(\Delta s)_i, i = 1, ..., n$ generated by an abstraction sampling.

Output The q most significant state-action areas $(\Delta s)_j$, with $j = 1, ..., q$.

1. For each $(\Delta s)_i, i = 2, ..., n$, define $rep_{1,i} \triangleq (\Delta s)_i$ and calculate
$$avrg_{1,i} = \frac{1}{3}\Big[(\Delta s)_i + \frac{1}{2}(\frac{(\Delta s)_{i-1}+(\Delta s)_i}{2} + \frac{(\Delta s)_i+(\Delta s)_{i+1}}{2}) + \frac{(\Delta s)_{i-1}+(\Delta s)_i+(\Delta s)_{i+1}}{3}\Big]$$
2. For each pair $\langle(\Delta s)_i, (\Delta s)_{i+1}\rangle, i = 2, ..., n-1$, define $rep_{2,i} \triangleq min((\Delta s)_i, (\Delta s)_{i+1})$ and calculate
$$avrg_{2,i} = \frac{1}{3}\Big[\frac{1}{2}((\Delta s)_i + (\Delta s)_{i+1}) + \frac{(\Delta s)_i+(\Delta s)_{i+1}}{2} + \frac{1}{2}(\frac{(\Delta s)_{i-1}+(\Delta s)_i+(\Delta s)_{i+1}}{3} + \frac{(\Delta s)_i+(\Delta s)_{i+1}+(\Delta s)_{i+2}}{3})\Big]$$
3. For each triplet $\langle(\Delta s)_i, (\Delta s)_{i+1}, (\Delta s)_{i+2}\rangle$, $i = 1, ..., n-2$, define $rep_{3,i} \triangleq min((\Delta s)_i, (\Delta s)_{i+1}, (\Delta s)_{i+2})$ and calculate
$$avrg_{3,i} = \frac{1}{3}\Big[\frac{1}{3}((\Delta s)_i + (\Delta s)_{i+1} + (\Delta s)_{i+2}) + \frac{1}{2}(\frac{(\Delta s)_i+(\Delta s)_{i+1}}{2} + \frac{(\Delta s)_{i+1}+(\Delta s)_{i+2}}{2}) + \frac{(\Delta s)_i+(\Delta s)_{i+1}+(\Delta s)_{i+2}}{3}\Big]$$
4. For each pair $\langle rep_{r,t}, rep_{l,g}\rangle$ satisfying $rep_{r,t} \equiv rep_{l,g}$ and $avrg_{r,t} \leq avrg_{l,g}$, remove $rep_{l,g}$ from $REP \triangleq \{rep_{i,j} | \forall i, j\}$.
5. In the set REP, choose the q elements $rep_{i,k}$ whose $avrg_{i,k}$ are the smallest; these elements represent the q most significant areas $(\Delta s)_j, j = 1, ..., q$.