Abstract—Markov games can be effectively used to design controllers for nonlinear systems. The paper presents two novel controller design algorithms by incorporating ideas from game-theory literature that address safety and consistency issues of the ‘learned’ control strategy. A more widely used approach for controller design is the \( H_\infty \) optimal control, which suffers from high computational demand and at times, may be infeasible. We generate an optimal control policy for the agent (controller) via a simple Linear Program enabling the controller to learn about the unknown environment. The controller is facing an unknown environment and in our formulation this environment corresponds to the behavior rules of the noise modeled as the opponent. Proposed approaches aim to achieve ‘safe-consistent’ and ‘safe-universally consistent’ controller behavior by hybridizing ‘min-max’, ‘fictitious play’ and ‘cautious fictitious play’ approaches drawn from game theory. We empirically evaluate the approaches on a simulated Inverted Pendulum swing-up task and compare its performance against standard Q learning.

Keywords—Fictitious Play, Cautious Fictitious Play, Inverted Pendulum, Controller, Markov Games, Mobile Robot.

I. INTRODUCTION

In this paper we concentrate on the quality of the policy learned by the controller in a Reinforcement Learning (RL) [8] framework. In particular, we propose algorithms that are ‘safe’ meaning that they guarantee the controller at least minmax payoff and ‘consistent’ meaning that the learned policy should do at least as well as playing the best response to the empirical average of the play if the opponent’s play is given by independent draws from a fixed distribution. The second algorithm, proposed in this paper, strives for not just ‘consistency’ but ‘universal consistency’ meaning that the control strategy should be heavily weighted towards a ‘safe’ or the minmax strategy and in later stages, when the experiential information is good enough, the strategy should incorporate a solution element obtained either via the FP as done in the first proposed algorithm or the CFP as in the second proposed algorithm.

II. MARKOV GAMES AND SOLUTION APPROACHES

A Markov Game is represented by the tuple \( \langle N, \Omega, A_{1,...,N}, C_{1,...,N}, T \rangle \) where \( \Omega \) is the set of states, \( N \) is the number of agents, \( A_{1,...,N} \) is the collection of action sets for the agents \( 1,...,N \), \( C_i \) is the cost function for the agent \( i \), i.e., 
\[
C_i: \Omega \times A_1 \times A_2 \times \cdots \times A_N \rightarrow \mathbb{R},
\]
\( T: \Omega \times A_1 \times A_2 \times \cdots \times A_N \rightarrow \mathbb{P}(\Omega) \) and 
\[
T(s, a_1, a_2, \ldots, a_N, s') = \text{probability of moving from state } s \text{ to } s' \text{ when each agent takes an action } (a_i \in A_i) \text{ at the state } s.
\]

A. Minimax-Q

We can define \( Q(s, a, o) \) value for tuple \( \langle s, a, o \rangle \) as the expected cost for taking action \( a \) when the opponent takes action \( o \) at state \( s \) and continuing optimally thereafter: i.e.,
\[
Q(s, a, o) = c(s, a, o) + \alpha \sum_{s'} T(s, a, o, s') V(s')
\]  
(1)
where \( T(s, a, o, s') \) = Probability of transition from state \( s \) to \( s' \) and \( c(s, a, o) \) = one step cost incurred by the agent, when the first player or the agent takes action \( a \in A \) and the second player or the opponent takes \( o \in O \) at state \( s \).

Minimax-\( Q \) algorithm is similar to \( Q \) learning, except that the term \( \min \limits_{\pi_{a \in P(A)}} Q(s', a, o) \) is replaced by the value of the game played between the two players at state \( s' \), i.e.,

\[
V(s') = \min \limits_{\pi_{a \in P(A)}} \max \limits_{\pi_{o \in P(O)} \times \pi_{a \in P(A)}} \sum_{s' \in A} Q(s', a, o) \pi_a, \quad \pi_a = \text{Probability distribution over agent’s action set.}
\]

\( Q \) values are updated as:

\[
Q(s, a, o) = Q(s, a, o) + \eta [c(s, a, o) + \alpha V(s') - Q(s, a, o)]
\]

where \( \eta \) = learning-rate parameter and \( V(s') = \text{Value of the game played between the agent and the opponent at state } s' \).

A completely specified version of minimax-\( Q \) can be found in [1].

Minimax control strategy is safe; unfortunately minimax play does not have the minimal learning property of ‘consistency’ [11].

**B. Stochastic Fictitious Play (FP)**

Model of FP suggests that the players choose their actions in each period to maximize that period’s expected payoff given their prediction or assessment of the distribution of the opponent’s strategy in that period. In a zero-sum setting the empirical distribution generated by FP must converge to Nash equilibrium [10]. In stochastic FP, the solution is in the form of a mixed policy, i.e., a probability distribution over crisp action set and has the advantage that small changes in the experiential data does not lead to abrupt changes in the agent’s policy and such a procedure is ‘consistent’. Suppose at time \( t \) the state is \( s \) and the opponent takes action \( o \in O \). Let \( k(s, o) \) be the times tuple \( < s, o > \) has been visited, then update \( k(s, o) \) with:

\[
k_{t+1}(s, o) = k_t(s, o) + \left\{ \begin{array}{ll}
1 & \text{if } o_t = o \\
0 & \text{if } o_t \neq o
\end{array} \right.
\]

(3)

Probability over opponent’s action set:

\[
P_{t+1}(s, o) = k_{t+1}(s, o) \sum_{o' \in O} k_{t+1}(s, o')
\]

(4)

Optimal policy of agent:

\[
\pi^*_a = \arg \min \limits_{\pi_a \in P(A)} \sum_{o \in O} U_a(s, a, o') p_t(s, o') \pi_a \quad \text{where } U_a(s, a, o) \text{ is the reward or utility accrued to the agent on taking } a \in A \text{ when opponent takes } o \in O \text{ at time } t.
\]

FP is well known not to approach the reader is referred to [11].

\( \eta \) is a constant, \( k > 1 \). For a detailed description of the CFP approach the reader is referred to [11].

**D. Proposed Hybrid Markov Game Algorithms**

1. **First Hybrid Markov Game Algorithm (HMG-1)**

For the FP part of the algorithm, we use the same opponent modeling approach as in FP but best response strategy is calculated based on the value of the game \( Q(s, a, o) \) and not on reward as done in standard FP, i.e., agent’s optimal policy is calculated as:

\[
\pi^*_a = \arg \min \limits_{\pi_a \in P(A)} \sum_{o \in O} U_a(s, a, o') p_t(s, o') \pi_a
\]

A matrix game is defined at a current state \( s \) by the game matrix \( C_a(s, o) \) consisting of \( Q_a(s, a, o) \) values, e.g., for \( |A| = 3, |O| = 3 \), the resulting game matrix \( C_{a, o}(s) \) at state \( s \) is:

\[
\pi = \begin{bmatrix}
\pi_1 & \pi_2 & \pi_3 \\
\pi_{a_1} & \pi_{a_2} & \pi_{a_3} \\
\pi_{a_1} & \pi_{a_2} & \pi_{a_3} \\
\pi_{a_1} & \pi_{a_2} & \pi_{a_3}
\end{bmatrix}
\]

where \( Q_{ij} = Q_a(s, a, o) \) and \( |A|, |O| \) stand for cardinality of sets \( A \) and \( O \) respectively.

The agent’s optimal policy \( \pi_a : \pi_a(s) \rightarrow P(A) \) is found as an annealed mix of the solutions of the matrix game defined by \( C_{a, o}(s) \), obtained using FP and Minimax-\( Q \). The algorithm incorporates a state-action pair visits dependent parameter \( \beta \in (0,1] \) that controls the amount of hybridization of the minimax-\( Q \) and FP solutions. Initially \( \beta \) is high for all unvisited state-action pairs which makes the policy ‘safe’ and in later stages with more visits at a particular state-action pair a high \( \beta \) value achieves a ‘safe’ and ‘consistent’ policy.
We generate action \( a' \in A \) at the next state \( S' \) according to a \( \epsilon \)-soft policy corresponding to \( \pi_a^{\epsilon} \) and update \( Q \) value using the standard \( Q \)-learning [8] update:

\[
Q_{t+1}(s,a,o) \leftarrow Q_t(s,a,o) + \eta [c(s,a,o) + \alpha Q^\epsilon - Q_t(s,a,o)]
\]

where \( \eta \) is the learning rate parameter, \( \alpha \) is the discount factor and \( c(s,a,o) \) is the cost of transition on taking action \( a \) at \( s \).

The agent’s optimal policy \( \pi^\epsilon \) is found as \( \pi^\epsilon = \arg \min \max (\pi^\epsilon, Q^\epsilon) \).

Then we use probability distribution specified by \( \pi_a^{\epsilon} \) to get \( a^\epsilon \).

Target \( Q \) value is found as

\[
Q^\epsilon \leftarrow \sum_{o' \in O} Q(s,a',o')p_i(s,o')
\]

**Min-max Part**

The game specified by the matrix \( C_{a,o}(s) \) is solved using the standard Linear Programming technique [5] to generate

\[
\pi_{a}^{\min}\cdot Q_{a}^{\min}\cdot Q_{a}^{\max}
\]

\[
Q_{a}^{\min} = \min_{a} \max_{a'} \sum_{o' \in O} Q(s,a',o')\pi_{a'}(s,a')
\]

The agent’s optimal policy \( \pi^\epsilon \) is found as an annealed mix of the solutions obtained using CFP and min-max-\( Q \). This algorithm also incorporates a state-action pair visits dependent parameter \( \beta \in (0,1] \) that controls the amount of hybridization of the minmax-\( Q \) and CFP solutions.

\[
\pi_{a}^{\min} \cdot Q^{\min} \cdot Q^{\max} 
\]

\[
Q_{a}^{\max} = \min_{a} \max_{a'} \sum_{o' \in O} Q(s,a',o')\pi_{a'}(s,a')
\]

\[
\pi_{a}^{\min} = \arg \min_{a} \max_{a'} \sum_{o' \in O} Q(s,a',o')\pi_{a'}(s,a')
\]

We generate action \( a_{t+1} \in A \) at the next state according to a \( \epsilon \)-soft policy corresponding to \( \pi_a^{\epsilon} \) and update \( Q \) value using the standard \( Q \)-learning [8] update of equation 9. Fig. 2 gives a pseudo-code for the proposed algorithm:
III. APPLICATION

Inverted Pendulum Swing-up

The details of the simulation model used for pendulum swing-up task can be found in [7]. We adopt a lookup table (LUT) approach by dividing state-space into discrete non-overlapping regions, as in the scheme of BOXES by Michie and Chambers [4]. Each trial is started from a position close to the origin of the system. During the trial plant parameters, i.e., mass and length of the pendulum were varied by [-20 20] % from nominal values while additive exogenous disturbances in [-10 10] Newton or 20% of the force magnitude continued to affect the controlled system. The performance of the controller in handling both these simultaneous disturbances was evaluated and compared against Q learning. Results are averaged over 100 experiments.

We take one step costs as:

\[
c = \begin{cases} 
4 & \text{if } |\theta| > 12^\circ \\
1 - \cos(\theta) & \text{otherwise}
\end{cases}
\]

Each experiment consists of a series of trials until either a trial resulting in pendulum remaining balanced for about 42,000 simulated steps corresponding to 14 minutes of real time or a maximum of 180 trials. Results are averaged over 100 experiments.

Controller Comparison: Performance

Table I summarizes trials needed to balance the pendulum for 42,000 simulated steps:

<table>
<thead>
<tr>
<th>Controller</th>
<th>Average</th>
<th>Maximum</th>
<th>Minimum</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMG-1</td>
<td>51.04</td>
<td>114</td>
<td>32</td>
<td>21.21</td>
</tr>
<tr>
<td>HMG-2</td>
<td>77.23</td>
<td>180</td>
<td>35</td>
<td>34.27</td>
</tr>
</tbody>
</table>

As can be seen from Table I, the average, maximum and minimum number of trials required to balance the pendulum is lower in case of both Hybrid Markov Game-1 and Hybrid Markov Game-2 controllers than the corresponding Q controller. Out of all the controllers HMG-1’s performance is the best. Fig. 3 displays a typical trajectory of the pole angle from a successful HMG-1 trial, for the first 300 balancing steps. It is to be noted that the pole angle’s maximum deviation from the vertical is less than 0.09 radians or \( \theta = 5^\circ \) even though the failure condition is \( |\theta| > 12^\circ \).

Controller Comparison: Consistency

Fig. 4 shows a comparative evaluation of the HMG-1 controller against Q controller, in terms of number of trials needed to balance the pendulum in each experiment, for 25 experiments. For Q controller, a number of experiments had to be stopped at 180 trials (without balance), clearly indicating the inability of the Q controller in handling the noise and parameter variations while none of the experiment in HMG-1 exceeded 114 trials. From Fig. 4 and a comparison of standard deviation values from Table I we see that the HMG-1 controller is far more consistent in performance than the Q controller.
Further, as can be seen from Fig. 5 and a comparison of standard deviation values from Table I, HMG-2 controller achieved a significantly better consistency than the $Q$ controller.

Here again HMG-1 controller outperformed the HMG-2 controller. The inferior performance of the HMG-2 controller is probably due to the fact that HMG-2 has been designed explicitly to optimize in non-stationary environments or for situations wherein we have an adaptive opponent. In presence of time-varying noise or with adaptive opponent, we expect HMG-2 to outperform the HMG-1 algorithm.

In terms of computational demand $Q$ controller has the least computation per iteration while in HMG-1 we need to solve two linear programs per iteration and one for HMG-2. The higher computational effort required for HMG-1 and HMG-2 in comparison to $Q$ controller is a very small price to pay when we consider the significant increase in the performance and consistency of the designed controllers. Further, the computational effort requirement in HMG-1 and HMG-2 can be reduced by approximations to the solution of the Linear Programs or iterative methods as suggested in [1].

IV. CONCLUSIONS AND FUTURE WORK

The paper presents two novel hybrid Markov game-theoretic algorithms for optimizing controllers that are ‘safe-consistent’ and ‘safe-universally consistent’. The algorithms advocate safe play when the environment or opponent is relatively unknown and a mixed strategy incorporating elements from the fictitious play or cautious fictitious play, when the transition information leads to a fair idea of opponent’s strategy. It exploits the capability of the fictitious play and cautious fictitious play to produce a payoff higher than the min-max and a more consistent behavior. Simulation results of applying the approach on a pendulum swing-up task and its comparison to $Q$ learning shows that the approaches produces a safe yet consistent controller. The results show that a Markov game formulation of the control problem gives better results than the $Q$ learning solution. Further, Markov game setup allows us to use efficient approaches like FP and CFP, from the game theory literature, for controller optimization. An important area for future research could be a hybrid game theoretic formulation for the control problem with a time varying model for the disturbances. We hope that such a formulation would address the problem to the fullest extent and may give better results as it fits the game-theoretic framework to a greater extent. These algorithms can be extended to optimize the behavior of an agent in multiplayer environments where several adaptive agents compete against each other.

REFERENCES