A Reinforcement Learning Approach to Score Goals in RoboCup 3D Soccer Simulation for Nao Humanoid Robot

Mohammad Amin Fahami, Mohamad Roshanzamir and Navid Hoseini Izadi
Department of Electrical and Computer Engineering
Isfahan University of Technology
Isfahan, Iran
mafahami@gmail.com, mohamad.roshanzamir@ec.iut.ac.ir, navid.hoseini1@ec.iut.ac.ir

Abstract— **Reinforcement learning is one of the best methods to train autonomous robots. Using this method, a robot can learn to make optimal decisions without detailed programming and hard coded instructions. So, this method is useful for learning complex robotic behaviors. For example, in RoboCup competitions this method will be very useful in learning different behaviors. We propose a method for training a robot to score a goal from anywhere on the field by one or more kicks. Using reinforcement learning, Nao robot will learn the optimal policy to kick towards desired points correctly. Learning process is done in two phases. In the first phase, Nao learns to kick such that the ball goes more distance with minimum divergence from the desired path. In the second phase, the robot learns an optimal policy to score a goal by one or more kicks. Using this method, our robot performance increased significantly compared with kicking towards predetermined points in the goal.**

Keywords— Reinforcement Learning; Nao Humanoid Robot; RoboCup 3D Soccer simulation

I. INTRODUCTION

Our motivation for using reinforcement learning is to get rid of hand-coded algorithms so that our agents can learn and make complex decisions autonomously in complex environments. As the robots and what they do are getting more sophisticated, the role of reinforcement learning to make them more intelligent becomes more apparent. Several researches has been carried out in this field. For example in [1, 2] policy gradient reinforcement learning is used to optimize and speed up moving skill of quadrupedal Sony Aibo and Kondo KHR2 humanoid robot. In [3, 4] reinforcement learning is used for autonomous helicopter flying. This technique is also used a lot in RoboCup soccer competitions to teach different skills to various types of robots [5-7]. In [8], the performance of three different types of reinforcement learning algorithms have been compared on Nao penalty kick skill in standard platform league.

In this research, a two-phase learning method is proposed. In the first phase, the robot learns how to adjust itself with respect to the ball for a successful kick. The objective of this phase is learning to kick the ball as far as possible with minimum divergence from a desired direction. In the second phase, the robot learns how to move the ball towards opponent's goal to score a goal in one or more steps based on the kick learned during the first phase.

In section 2, the proposed method is explained. Then, it is described how it is used for training the robot. In section 3, experimental results are discussed and finally in section 4 conclusions and future works are presented.

II. PROPOSED METHOD

In this section, we explain the two phases of our method. In the first phase we use simulated annealing algorithm to optimize the kick skill. In the second phase, the robot learns how to score a goal with minimum number of kicks using Q-learning algorithm.

A. First phase: kick optimization

In this phase, we want to optimize kick skill so that the robot can use it efficiently. We put ball in the middle of the field and put the robot behind it. As illustrated in Fig. 1, to kick efficiently, the robot must learn the following parameters:

Radius: It is the distance between robot's feet and the ball along x axis. It is necessary to know what distance has the best kick performance. We assume that the ball is in the middle of the field (i.e. at point (x=0, y=0)). The player stands on x = Radius. We are looking for the optimal value of Radius.

Offset: Suppose that the result of learning parameter Radius is R. It means that the robot must stand behind the ball on line x = R. However, we have not determined the y component of player's position, yet. Offset is what we must look for. In other words, the player must stand on (x = Radius, y = Offset). The robot orientation with respect to the ball will be discussed in section D.

These two parameters are continuous variables. There are many different algorithms for continuous variables optimization. We used simulated annealing algorithm [9] as we have only two variables in this phase and simulated annealing requires much less conditions than other search techniques. It is one of the simplest algorithms for heuristic search implementation. It is a simple loop that continuously moves towards the direction of increasing fitness and stops in the peak of fitness function where there is no neighbor candidate with higher fitness. This algorithm is the advanced version of hill climbing [9]. It avoids local optima by taking some downhill steps during the search. It moves randomly instead of always picking the best move. If the movement improves the fitness

978-1-5386-0804-3/17/$31.00 ©2017 IEEE

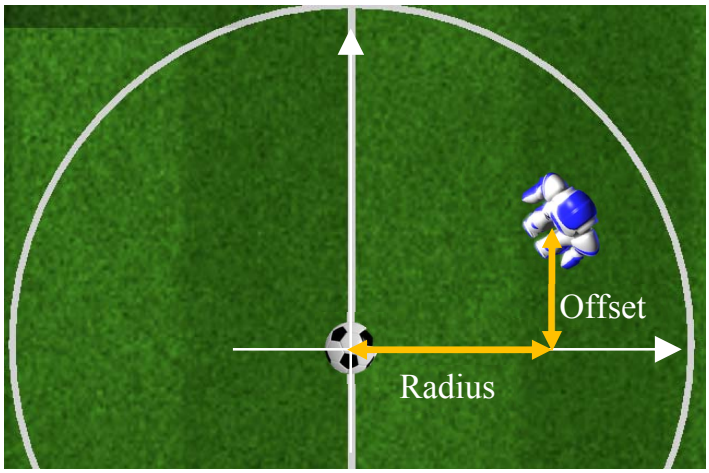


Fig. 1. The player position is in (Radius, Offset) coordinates.

value, it is always executed. However, it makes the movement with probability less than one.

In this problem, we define two-dimensional states (kickRange, kickOffset). kickRange and kickOffset are two parameters that the environment returns for each state. kickRange is the distance the ball travels after being kicked and kickOffset is the ball divergence from kick point. For example if initially the ball is at point (x = 0, y = 0), and after the kick, the ball stops at (x = a, y = b), then kickOffset = b. This parameter is important because we like the robot to kick as straight as possible (i.e. with kickOffset = 0).

The fitness function is defined according to these two parameters. kickRange has positive impact because we like to have long kicks and kickOffset has negative impact as we do not like kick trajectory divergence from straight line. So, we define fitness function as Eq. 1

$$\text{fitness(kickRange, kickOffset)} = \alpha \times \text{kickRange} - \beta \times \text{kickOffset} \quad (1)$$

where α and β are constant positive values and are chosen according to the importance of kickRange and kickOffset.

B. Second learning phase

In this section at first Markov Decision Process (MDP) [10] is explained. Then, we formulate our problem as an MDP. Finally, we describe the reinforcement learning algorithm for the second phase of learning.

C. Markov Decision Process (MDP)

An MDP consists of a set of states S, a set of actions A, a reward function $R(s,a)$, and a transition function $P(s'|s,a)$. $P(s'|s,a)$ is probabilistic distribution function that indicates the probability of transition from state s to s' when action a is done. For each state-action, $Q^*(s,a)$ is calculated by Eq. 2 known as Bellman equation

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s'} P(s'|s,a) \max_{a'} Q^*(s',a') \quad (2)$$

where $0 \leq \gamma \leq 1$ is the discounting factor. The Q^* value is calculated by repeating Bellman equation until it is converged [10]. The goal is to find policy π that maps each state to an action so that discounted total reward is maximized. The optimal policy π^* is defined by Eq. 3.

$$\pi^*(s) = \arg\max_a Q^*(s,a) \quad (3)$$

D. MDP in the second learning phase

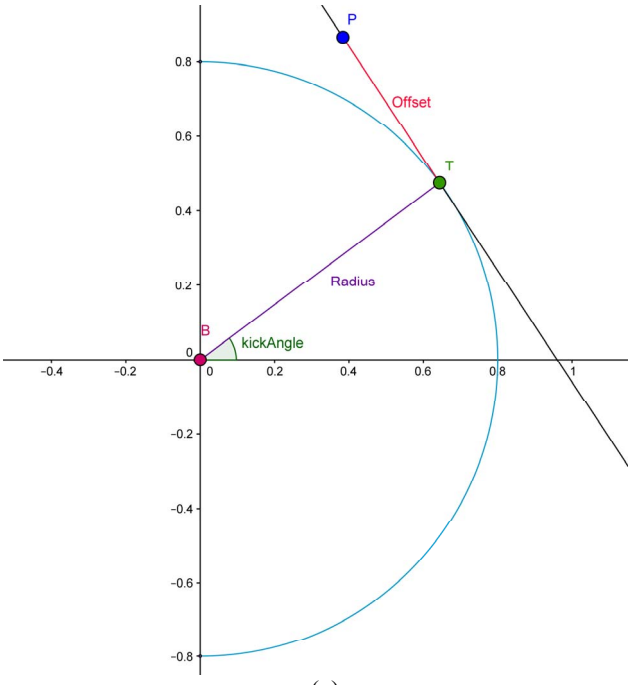
In the first phase, the robot learned the optimal kick. Now, the robot knows where it must stand behind the ball so that the kicked ball travels the maximum distance with minimum divergence. However, the robot does not know the appropriate kick direction to score a goal with minimum number of kicks. For example, if the kick range allows the robot to score a goal with one shot, then what is the most appropriate kick direction? If not, where must the robot kick the ball so that the next kicks have the best chance to enter the goal? So, in the second learning phase, we want to train the robot to stand in suitable direction with respect to the ball. Our objective is that the robot learns how to score a goal with minimum number of kicks.

In this phase, we put the ball in an arbitrary position B=(b_X, b_Y). This position determines the first state of the player in the training process. As illustrated in Fig. 2, the optimal player position for kicking (i.e. P) depends on point T which resides on the half circle centered at the ball. Moreover, opponent's goal is located on the negative side of x axis.

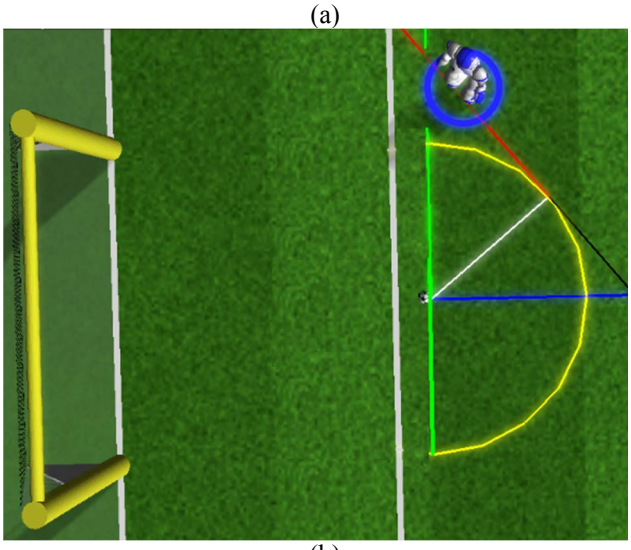
To determine point P, point T and Offset must be known. Since Radius and Offset have been learned in the first phase, in the second phase only kickAngle needs to be learned (see Fig. 2).

The MDP of the second phase is defined as follows:

1. State set (S): A set of states is define as:
   - $state\ (R,C)$: it is the discretized ball location. As the ball position is a 2D continuous vector, we partition the soccer field into $M \times N$ blocks to be able to solve the problem as an MDP. In the other words, we will have $M \times N$ states of type (R,C). Each state is represented by its row and column indices (R,C). M and N must be selected carefully so that the learning process will be computationally tractable. As the robot uses same kickAngle for all points within a block, our method does not work well for large blocks. However, the block size cannot be too small either, otherwise number of discrete states will be high which makes the problem hard to solve.
   - GoalState: after kicking the ball, if the robot scores a goal, its state is changed into this state which is a final one.
   - FailState: after kicking the ball, if the ball is caught by opponent goalie or goes out of the field, the state is changed into FailState. It is also a final state.
2. Action set (A): each action is defined as a unary (kickAngle) in which the player selects a value for kickAngle. According to kickAngle and learned parameters from previous phase i.e. (Radius, Offset), the robot kicks the ball. Actually, action selection is done using ε-greedy exploration. In this method, with



(a)



(b)

Fig. 2. Learning parameters in the first and second phases of learning in (a) mathematical schema and (b) real field.

probability ε, a random explorative action is selected and with probability $1-\varepsilon$ the optimal action is chosen. We run Q-learning with the $\varepsilon = 0.1$.

3. Reward function $R(s,a)$: The reward function gives a positive reward for scored kicks and zero for failed kicks.

   We define the reward function in Eq. 4

$$Reward = \begin{cases} \lambda + \phi \times |\text{ballY}| & \text{If scored a goal} \\ 0 & Otherwise \end{cases} \quad (4)$$

   where ballY is the y coordinate of the ball after it stops moving. We define the reward function according to the final position of the ball in the goal. The closer the ball is to goal posts, the more reward is assigned to the scored goal as it is harder for the goalie to catch the ball. λ is the base reward value for a scored goal and ϕ is the coefficient that shows the importance of ballY.

   In general, it is possible to use heuristics driven from real matches to assign different rewards for different points of the opponent's goal.

4. Transition function $P(s'|s,a)$: in this problem as we do not know the model, we use Q-learning as a model-free algorithm.

E. *Calculating player position in the second phase of learning*

The robot must determine its position according to Radius and Offset learned in the first phase of learning and kickAngle which is learned in the second phase. In this section, we describe how the player position is calculated.

Suppose the ball is located at B=(b_X, b_Y). The player position P=(p_X, p_Y) can be calculated using Eq. 5 and 6.

If offset is zero player position will be given by Eq. 5.

$$\begin{aligned} p_X &= b_X + \text{Radius} \times \cos(\text{kickAngle}) \\ p_Y &= b_Y + \text{Radius} \times \sin(\text{kickAngle}) \end{aligned} \quad (5)$$

Otherwise, the player position is calculated using Eq. 6.

$$\begin{aligned} R' &= \sqrt{\text{Radius}^2 + \text{Offset}^2} \\ \theta &= Arcsin\left(\frac{\text{Offset}}{R'}\right) \\ p_X &= b_X + R' \times \cos(\text{kickAngle} + \theta) \\ p_Y &= b_Y + R' \times \sin(\text{kickAngle} + \theta) \end{aligned} \quad (6)$$

F. *Reinforcement learning for goal scoring*

In second phase of learning we use Q-learning algorithm as shows in Table I.

In this algorithm, when Q^* converged, a set of optimum values for kickAngle will be obtained. We can run this algorithm for each block of the field.

III. Experimental Results

This algorithm is applied on Nao type 4 in Simspark simulator. This robot has 58 cm height and 25 degrees of freedom. This robot is used in RoboCup 3D soccer simulation league. At this time, the simulator which is used in this league is Simspark [11]. RoboCup is an annual competition for developing artificial intelligence and robotics. Its main goal is that a team of soccer robots defeat human champion team in 2050.

We run these algorithm using UTAustinVilla3D base code [12, 13]. In the first phase of learning, optimal parameters and the achieved kick characteristics are shown in Table II.

In the first phase, to calculate the fitness function in Eq. 1, we selected $\alpha = 100$ and $\beta = 20$. Radius lower bound and upper bound were 0.1 and 0.3, respectively. Offset lower bound and upper bound were 0.05 and 0.11, respectively.

TABLE I. Q-LEARNING ALGORITHM FOR SCORING A GOAL IN THE SECOND PHASE OF LEARNING

A ← Set of Actions // Each action is a unary: (kickAngle)

S ← { (R, C) States, GoalState, FailState }

For each state ***(R, C)*** and each action ***a*** initialize the table entry $Q^*(s,a)$ to zero.

Observe the primary state s

Do until convergence:

- Select an action a = (kickAngle) using ε-greedy
- Set the position of player according to (Radius, Offset, kickAngle)

  $R' = \sqrt{\text{Radius}^2 + \text{Offset}^2}$

  $\theta = Arcsin(\frac{\text{Offset}}{R'})$

  $p_X = b_X + R' * \cos(\text{kickAngle} + \theta)$

  $p_Y = b_Y + R' * \sin(\text{kickAngle} + \theta)$
- Perform an action // Kick the ball
- Receive immediate reward :
  If goal occurs:
  - Reward = $\lambda + \phi * |\text{ballY}|$

  Else:
  - Reward = 0
- Observe the new state s'
- Update the table entry for $Q^*(s,a)$ as follows:
  $Q^*(s,a)$ = Reward + $\gamma \max_{a'} Q^*(s',a')$
- s ← s'

TABLE II. THE RESULTS OF THE FIRST PHASE OF LEARNING

Optimized parameters		**kick characteristics**		
Radius	***Offset***	***kickRange***	***kickOffset***	***Fitness***
0.202	0.053	5.561	0.126	553.58

In the second phase of learning we used UTAustinVilla3D goalie, the champion of 2016 RoboCup 3D soccer simulation as the opponent goalie [14]. Then, we tried to learn kickAngle. The continuous kickAngle interval $[-\frac{\pi}{2}, \frac{\pi}{2}]$ were discretized with resolution $\frac{\pi}{720}$. In the reward function, we set $\lambda = 100$ and $\phi = 10$. In Q-learning algorithm, the discount factor (γ) was set to 0.9. The selected values for M and N were 60 and 30, respectively.

This algorithm is applied for some points of the field and the results of learning is summarized in Table III. The first and second columns of the table show the center coordinates of the block in which the last kick is performed.

We tested each learned parameter set ten times. If all the kicks using a parameter set leaded to a goal, we considered it as a successful parameter set. The third column of Table III shows the number of successful parameter sets. We compared our proposed method with a hand-coded solution. In our hand-coded solution, the robot kicks towards the two regions between the opponent's goalie and the goalposts to increase goal scoring chance.

As it is clear from Table III, the success rate of our method is much higher than success rate of hand-coded algorithm in all positions, especially when the player is not in front of opponent's goal at the last kick (i.e. Y=2). For example, when the ball is within block with center (12, 2), the success rate of our method is 89.11% while this rate for hand-coded algorithm is only 30%.

Meanwhile, when the player distance from the opponent's goal was increased, the performance of hand-coded algorithm decreased significantly, while our proposed method could still score goals.

IV. CONCLUSION

This research is an example of using reinforcement learning to make humanoid robot more intelligent to learn difficult skills.

In this paper, we proposed a reinforcement learning algorithm for humanoid robot Nao to learn how to score a goal. This algorithm has two learning phases. In the first phase Nao tries to kick with maximum range and minimum divergence. In the second phase of learning Nao learns to score a goal. In this phase, the direction of robot with respect to ball is investigated and Nao learns to kick the ball towards opponent's goal according to what it has learned from the first phase of learning.

TABLE III. THE RESULT OF THE SECOND PHASE OF LEARNING.

The center coordinate of the block in which the last kick is performed		**Number of successful parameter sets**	**Number of successes in test phase**	**Success rate of our proposed method (%)**	**Success rate of hand-coded algorithm (%)**
X	**Y**				
10	0	44	33	75	40
10.5	0	45	40	88. 9	33.33
11	0	50	36	72	43.33
11.5	0	68	60	88.24	53.33
12	0	69	58	84.06	50
12.5	0	96	68	70.83	56. 67
13	0	157	122	77.71	73.33
10	2	40	30	75	16.66
10.5	2	41	31	75.61	20
11	2	60	54	90	13.33
11.5	2	80	75	93.75	23.33
12	2	101	90	89.11	30
12.5	2	124	123	99.19	36.66
13	2	135	131	97.04	43.33

The advantage of this method is that it can find the optimal parameters for each kick type. In other words, our algorithm can find the best way for using different kick skills. There is also the capability to give different rewards when goal is scored according to the location of ball in the goal. In this way, it is possible to find weak points where the goalie is less likely to catch the ball. Then, we shoot towards those points to increase the chance of scoring a goal.

As the future work, these parameters can be learned while there are more robots in the field. They can be opponent players or our teammates. If they are our teammates, ball passing skill is learned for scoring a goal.

V. References

[1] A. Givchi and M. Palhang, "Frame based humanoid walking using sequential policy gradient and gyro optimizing," presented at the The 21st Annual Conference of the Japanese Neural Network Society, 2011.

[2] N. Kohl and P. Stone, "Policy gradient reinforcement learning for fast quadrupedal locomotion," presented at the Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on, 2004.

[3] J. A. Bagnell and J. G. Schneider, "Autonomous helicopter control using reinforcement learning policy search methods," presented at the Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164), 2001.

[4] A. Y. Ng, H. J. Kim, M. I. Jordan, and S. Sastry, "Autonomous helicopter flight via reinforcement learning," presented at the Proceedings of the 16th International Conference on Neural Information Processing Systems, Whistler, British Columbia, Canada, 2003.

[5] A. Merke and M. Riedmiller, "Karlsruhe Brainstormers - A Reinforcement Learning Approach to Robotic Soccer," in *RoboCup 2001: Robot Soccer World Cup V*, A. Birk, S. Coradeschi, and S. Tadokoro, Eds., ed Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 435-440.

[6] M. Riedmiller, T. Gabel, R. Hafner, and S. Lange, "Reinforcement learning for robot soccer," *Autonomous Robots,* vol. 27, pp. 55-73, 2009.

[7] P. Stone, R. S. Sutton, and G. Kuhlmann, "Reinforcement Learning for RoboCup Soccer Keepaway," *Adaptive Behavior,* vol. 13, pp. 165-188, 2005.

[8] T. Hester, M. Quinlan, and P. Stone, "Generalized model learning for reinforcement learning on a humanoid robot," presented at the Robotics and Automation (ICRA), 2010 IEEE International Conference on, 2010.

[9] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*: Pearson Education, 2003.

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

[11] *Soccer Simulation League*. Available: Soccer Simulation League, http://wiki.robocup.org/Soccer_Simulation_League [Accessed: 26-Mar-2017]

[12] P. MacAlpine, M. Depinet, and P. Stone, "UT austin villa 2014: robocup 3d simulation league champion via overlapping layered learning," presented at the Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, Austin, Texas, 2015.

[13] P. MacAlpine and P. Stone, *UT Austin Villa RoboCup 3D Simulation Base Code Release*. Berlin: Springer Verlag, 2016.

[14] *AUSTIN VILLA ROBOT SOCCER TEAM*. Available: AUSTIN VILLA ROBOT SOCCER TEAM, http://www.cs.utexas.edu/~AustinVilla/sim/3dsimulation/. [Accessed: 26-Mar-2017]