Incorporating Forgetting Mechanism in Q-learning Algorithm for Locomotion of Bipedal Walking Robot

Rashmi Sharma , Inder Singh , Deepak Bharadwaj , Manish Prateek

Abstract:A walking bipedal is a kind of humanoid which resembles human. Bipedal are programmed for some specific tasks. The work studied biped walking with ZMP to control balance mechanism using reinforcement learning(RL). The proposed forgetting Q-learning algorithm helps the bipedal to learn to walk without any prior knowledge of dynamics model of the system. In this work, the study is carried out to examine improvement to reinforcement learning(RL) algorithm in order to successfully relate with the continuously changing environment. The bipedal navigation is studied by implementing forgetting mechanism in the traditional Q-learning algorithm. Simulations were performed on each of the six joints of both legs of bipedal to evaluate the feasibility study of the proposed algorithm. The optimal policy for navigation was evaluated. Incorporating forgetting mechanism improves the learning time of the RL agent to a certain extent in a dynamic environment. The learning architecture was developed to solve complex control problems. It uses different modules that consists of simple controllers with RL forgetting Q-learning algorithm.

Index Term/Keywords: ***Bipedal, Reinforcement Learning, Q-Learning Algorithm, Walking Robot, Optimal Policy, Forgetting Mechanism.***

I. INTRODUCTION

The bipedal walking has been an active action area of research after the developed of humanoid robots. Many robotic structures: wheeled robots[7-9], hexapod[1-3], quadruped[4-6] lacks flexibility and adaptability to the environment. The bipedal walking mimics human characteristics and so was selected to simulate movements and actions of human. Bipedal robots are assumed to operate with much greater efficiency than any other type of robot in a human environment [10]. Bipedal can perform tasks which are risky for human being. These can be any dangerous environmental conditions such as fire rescue operation, explosives and can also assist in other more complex and complicated tasks.

At present, the most basic issue for humanoid biped robot is how to walk steadily in uncertain and continuously changing environment. Many researchers has controlled the ZMP position for walking stability[11-13].

A. Reinforcement Learning(RL)

Reinforcement Learning(RL) seeks to maximize the numeric reward signal. In supervised learning method, we exploit the correct actions which are known prior to the execution. RL methods uses trial and error methods i.e. achieve learning by trying almost all the feasible actions and learning which of those actions produce the maximum value of reward. RL algorithm are used to solve those problems in which RL agent learns through interaction with its dynamic and uncertain environment. Reinforcement Learning(RL) or Self Learning technology is usually considered as a goal-oriented method for solving problems in uncertain and dynamic environment. A RL model consists of : an RL agent, dynamic environment in which RL agent operates, a method of action selection, a method to determine the immediate reward of each action taken and a method of estimate delayed reward of action taken.[14] (Figure 1) RL methods provide solution of many diverse problems.

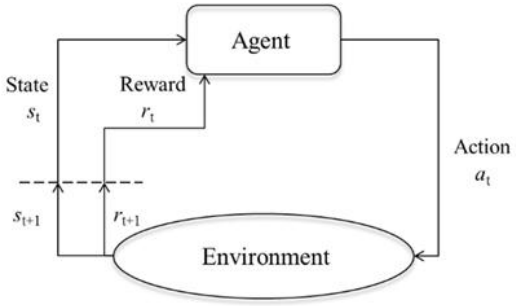

Figure 1. Architecture of RL

Application designers of RL methods faces drawback of very slowly learning which results in poor performance in uncertain and dynamic environment. Another drawback of RL method is tradeoff between exploration and exploitation. The RL agent has to exploit i.e. has to use previous knowledge in order to get reward and to reach goal as soon as possible(Exploitation).

Revised Manuscript Received on May 07, 2019.

Rashmi Sharma, Research Computer Science Engineering, University Of Petroleum & Energy Studies, Dehradun, India

Dr. Inder Singh Computer Science Engineering, University Of Petroleum & Energy Studies, Dehradun, India .

Deepak Bharadwaj, Mechanical Engineering, University Of Petroleum & Energy Studies, Dehradun, India

Manish Prateek, Computer Science Engineering, University Of Petroleum & Energy Studies, Dehradun, India

The RL agent has to explore i.e. for making better action selections depending upon the value of reward and to learn more about their environment in order to enhance future performance (Exploration)[15].

B. Bipedal Navigation

Several ways exists to design a stable walking gait pattern. The most common and simplest approach is the applied inverted pendulum model(IPM) for balance control mechanism. The humanoid is modeled as a double inverted pendulum i.e. one for thigh and one for calf and the joints considered are hip , knee and ankle. While designing online controllers for bipedal, the bipedal does not fall or tip over is assured. The motions are so designed that the controllers need to be sufficient to ensure the stability of bipedal. The Zero Movement point(ZMP) considers both the static and dynamic forces. Therefore, many researchers have proposed ZMP methods for stable walking of Bipedal [11,16]. When the control designed keeps the ZMP position within the limit of the soles, then the robot have steady walk. (Figure 2)

Biped robots learning methods for walking is studied to increase the robustness of the existing proposed algorithm[17-19]. Training experiences is collected by RL agent and the learning policy is updated through interaction in dynamic environment. To determine stable state of a robot and to prevent them from drop down, researchers have analyzed the feedback position of ZMP. ZMP of biped should be maintained within a support polygon defined as the convex hull formed by all floor contact points. After carrying out a long learning process, the system gets an efficient walking policy that is suitable for present uncertain and dynamic environment.

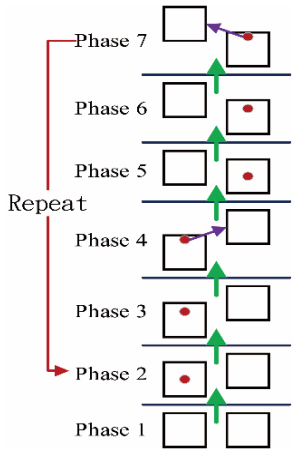

Figure 2. ZMP position of a biped walking sequence.

II. PROPOSED ALGORITHM

This section deals with proposed solution which overcomes the difficulties of applied RL based learning system for autonomous navigation problem in dynamic environments.

A. Incorporating Forgetting Mechanism in Traditional Q- learning

The RL agent when interacting with the dynamic environment may make use of the previously learned knowledge which is now outdated as the dynamics of the environment has changed. This difficulty usually is due to the tradeoff dilemma between exploration and exploitation. It is proposed in this paper that a forgetting mechanism be incorporated into traditional Q-learning algorithm to mitigate the effect of using outdated knowledge. In deterministic environment, the next state following an action is already known, which simplifies the Q-learning process. The Q-learning process in deterministic environment stores only values associated with each state instead of each state-action pair. In dynamic environment, state-action pair has to be stored along with the dynamic reward calculated on the fly.

1: Initialize Q(s,a) arbitrarily;
2: **repeat**(for each episode):
3: Initialize s;
4: **repeat**(for each step of episode):
5: Choose a from s using policy derived from Q;
6: Take action a, observe r, s';
7: $Q(s,a) \leftarrow Q(s,a) + \alpha\,[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$;
8: $s \leftarrow s'$;
9: **until** s is terminal
10: **until** all episodes end.

Figure 3. Q-Learning Algorithm by Watkins (1989): Single step algorithm[20]

The state value function maintains a reward function for a given state which keeps a record of total penalty/reward which the RL agent has to pay for being in that specific state. Initialize the state-value function with zeros. As the RL agent explores the dynamic environment, it learns and records the reward gained in each state. The Q-value function for the visited state is updated after each time step. (Figure 3)

B. Action Selection Policy

Action policy depends on random number generation between(0,1). For the first run, if value is in between (0,0.5) we exploit i.e. choose an action which is chosen maximum times and if it's between (0.5,1) we explore i.e. randomly select any action from the action set. For the subsequent runs, the range of the exploitation changes from (0,ε*ε-decay) and for exploration changes from (ε*ε-decay,1). This shows that the random actions are chosen more frequently so it favors exploration rather than exploitation which results in randomness in environment. It trains the agent, to avoid to stuck in the same action again and again.

This epsilon(ε) controls the policy update extent which is based on the value of the next state. Large value of ε denotes that the domain is very much dependent on the values of the next state. Smaller value of ε indicates that function/policy is more dependent on the state to state transition reward. The value of ε is usually a positive value between (0,1). If value of ε is taken very close to 1 little forgetting takes place in the algorithm and it behaves similar to traditional Q learning algorithm. If value of ε is taken close to 0 almost all the penalties/rewards are forgotten between the episodes.

This results in exploration of the dynamic environment by RL agent in each episode. *Assumed ε=0.5 for current work*. After each episode it is updated as ε=ε* ε-decay.

III. METHODOLOGY USED

A. Proposed Algorithm

The steps of proposed algorithm are :

1. The environment parameters are initialized: learning rate(α),
   epsilon(ε), epsilon decay, discount factor(λ).
2. Q matrix is initialized to zero.
3. Do for each episode -
   A. Initial state is selected (initially stated and then after evaluated by the proposed algorithm).
   B. Do while the desired goal state of RL agent is not reached.
      a) Select any one actions from the available action set for present state using random value generator (Exploit/Explore).
      b) Using this possible action calculate the random reward.
      c) Using this possible action, find the following(next) state.
      d) Maximum Q value is found for this next state based on all possible available actions.
      e) Compute value of Q(s,a).
      f) Data stored in excel file for the intermediate results, present state, next state, action taken, time taken and reward.
      g) For each episode, a new excel sheet should have the intermediate results.

   end do

   C. Data stored in new excel file for the final Q-matrix, the optimal policy, mean random value, total reward, total time taken.

   D. For each episode, a new excel sheet should have the final result.
4. Store the value of each episode and the number of iterations took to reach goal state along with total duration for each process in the third excel file.
5. Plot the graphs for: Total time in each episode, Total reward in each episode, Mean random value in each episode.

B. Characteristics of the Designed System

On considering a dynamic environment with following constraints:

1. States have been uniformly distributed between Start state and Goal state(known set).
2. The action considered are (0,1) degree movement(known set). For some cases also taken as (-1,1) .
3. The environmental parameters are initialized to following values.
   Learning rate (α)= 0.9 Discount Factor(λ) =0.9
   Exploration probability(ε) = 0.5 (ε to explore and 1- ε to exploit)
4. Epsilon decay(ε-decay) =0.98 (the forgetting factor).
5. Reward/Penalty given to the RL agent depends on the gap between present state and goal state
   *reward/penalty = exp(-α *(goal state of RL agent - present state of RL agent)*
6. Tradeoff between Exploitation and Exploration-
   - Exploitation respond slowly to the environmental changes.
   - Exploration achieves flexibility.
   - Our system is encouraging exploration which is more effective for dynamic environment.

Key Points of the developed system :-

- Storing and maintaining the number of states, action taken and next state individually, reduces the memory space.
- When agent explores it learns the reward earned in each state, the value function approximates for that state.
- After each step
  A. Actions are selected depending on random value generated.
  B. Reward/penalty is calculated.
  C. Next following state is evaluated from present state and reward calculated.
  D. Q-value (Policy) for the visited state is updated.
  E. Epsilon is recalculated by epsilon*epsilon decay.
  F. Total rewards/penalty calculated.
  G. Total Time for execution calculated.
  H. Gap between present state and goal state is evaluated i.e. distance between them.
  I. How far is the agent from goal state is tracked.

  On exploration, the states far from goal state approach maximum values where as nearby states approach minimum values. The values are a function of distance between present state and goal state.

IV. EXPERIMENTAL FINDINGS

Experimental results are stored and shown in two ways. The first form is the graphical representation of the results in the form of graphs and the second form is to store the data evaluated in the excel files for the future use.

A. **Simulation Results**

Case I- Random value generated between (0,1)
The algorithm has characteristics as-

1. Updating rule for Q(s,a).
2. Evaluating function for Action.
3. Mechanism of Forgetting being incorporated.
4. Randomness in reward/penalty depending on present state of the agent and goal state.

Goal is to utilize exploratory behavior to enhance the performance in dynamic and uncertain environment. Due to the exploratory behavior a larger set of possible solution has to be maintained in comparison to kept by traditional Q-learning algorithm.

The intermediate and final data of the algorithm is saved in two different excel files along with the header for the future use. The intermediate file contains the state, action transitions along with penalty/reward, random value generated in each episodes along with the time required to run each episode and the distance between present state of the agent and goal state which RL agent has to reach. The final data excel file has the mean value of the random values generated to reach goal state from the starting state, the final Q(s,a), the optimal policy obtained along with the actions selected in optimal policy, the total reward/penalty to reach the initial state to goal state along with the total time.(Figure 4 and Figure 5) The graph are plotted for-

1. Target state - present starting state of agent in each episode.
2. Target state - next state in each episode.
3. Mean random value and random value of each episode.
4. Distance between present state and goal state.
5. Reward earned in each episode.
6. ε value (ε* ε-decay) in each episode incorporation forgetting mechanism.

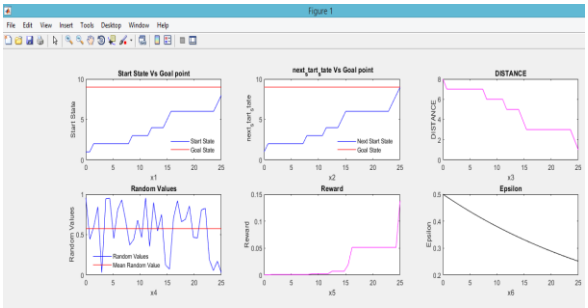

Figure 4. Case - I Reached Goal State in 36 Iterations

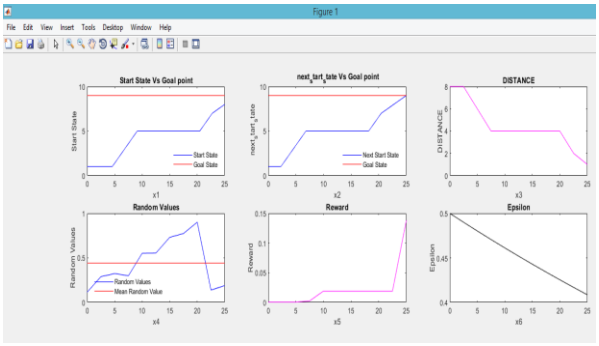

Figure 5. Case I- Reached Goal State in 12 Iterations

CASE II When the range of random number generation is fixed (0.4 to 0.6), with the same previously defined state set and action set (known set),all parameters are same as the previous simulation. We observed that it runs the maximum number of episodes but does not reach goal state but get stuck in between start state and goal state. This is due to the reason that we are considering an action of 0 degree due to which it remains in the state for infinite duration. (Figure 6)

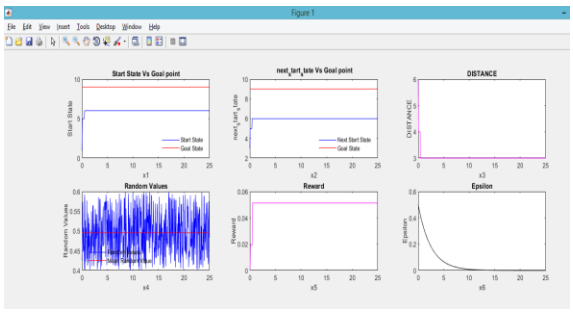

Figure 6. Case II - Does not reach goal state even after excecuting 100 episodes

CASE III - When considering action deg(0) does not reach goal as it is stuck in the same state loop. Without considering action deg(0) the algorithm converges fast and it explores more as compared to exploit and does not fully exploit decay factor (forgetting mechanism). (Figure 7)

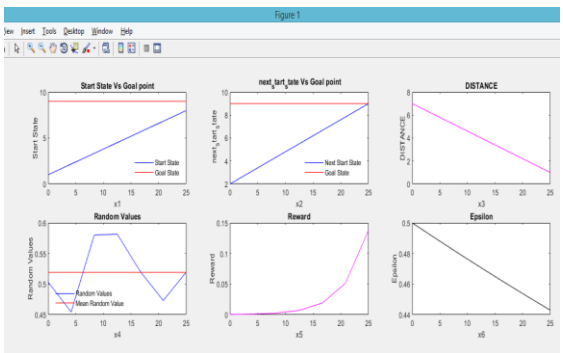

Figure 7. Case III - Reaches Goal State in 8 Iterations

Data Saved in excel Files for future use

The results found after the simulations are stored in excel files. Each episode stores its value in different sheet and the final result of each episodes are stored for future use. (Figures 8 - 13)

V. RESULTS AND FINDINGS

Results are analyzed according to the case considered above.
Case I : The number of episodes in which RL agent reaches goal state is not fixed as shown in two results in one it reaches in 36 episodes and in the other it reaches in 12 episodes, accordingly the total reward and total time varies
Case II: The action of deg(0) is exploited more as compared to exploration of other actions. The experiment uses exploitation and not exploration.
Case III :This approach converges very fast and encourages exploration.
This approach converges very fast and encourages exploration.

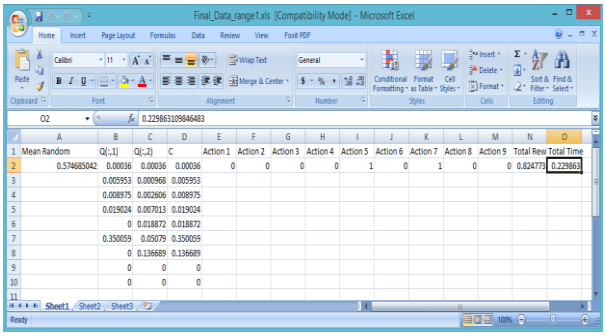

Figure 8. Case I Final Result Sheet

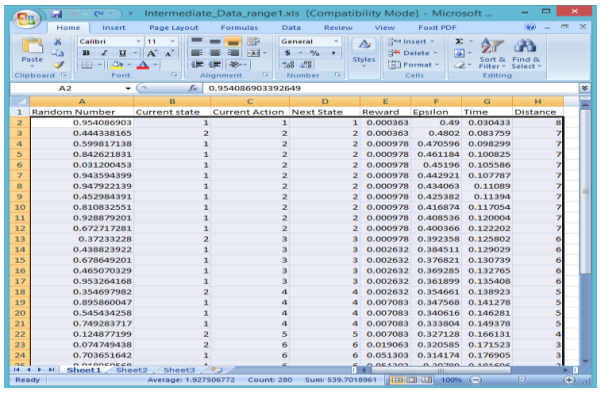

Figure 9. Case I Intermediate Result Sheet

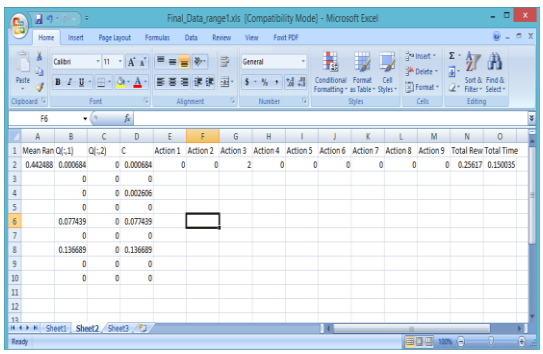

Figure 10. Case II Final Result Sheet

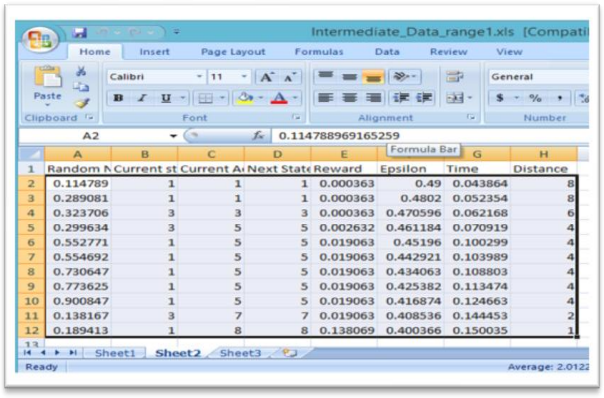

Figure 11. Case II Intermediate Result Sheet

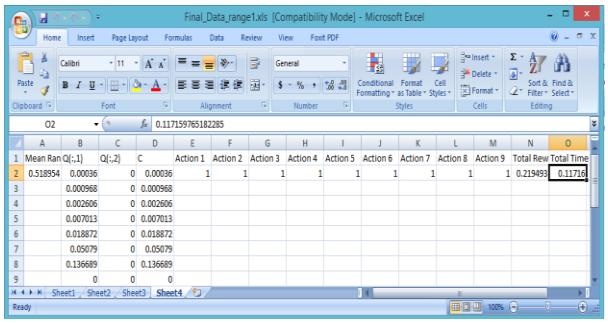

Figure 12. Case III Final Result Sheet

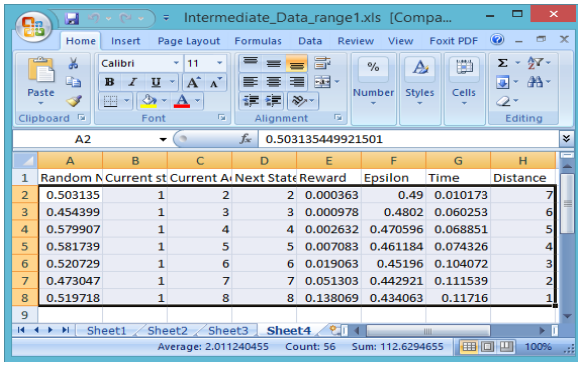

Figure 13. Case III Intermediate Result Sheet

VI. CONCLUSION

This paper proposes solution for performing biped navigation by using RL algorithm in dynamic environment. The proposed algorithm has been presented and analyzed. Forgetting mechanism in Q-Learning is proposed for improving performance in dynamic and uncertain environment by considering the possible options of navigation which may not be acceptable by traditional Q-learning. This mechanism was implemented as a decay factor which helps in reducing unvisited state values which is result of higher exploration tendency, resulting in increased performance in dynamic environment.

The proposed algorithm architecture is tested on real time control on robot walking. The RL agent learn to choose an appropriate action in this state to maintain the ZMP of the biped in the restricted region. When robot is in single support condition ZMP is intentionally shifted to a designated position. For learning an efficient optimal policy, a dynamic and continuous reward function is designed in order to reduce the learning cost on balance control.

REFERENCES

1. P. G. de Santos, E. Garoia, R. Ponticelli, and M. Armada, "Minimizing Energy Consumption in Hexapod Robots," Advanced Robotics, vol. 23, pp.681-704, 2009.
2. E. Burkus and P. Odry, "Autonomous Hexapod Walker Robot "Szabad(ka)"," Acta Polytechnica Hungarica, vol. 5, pp. 69-85, 2008.
3. J. Iovine, "Hexapod Walker Robot," Poptronics, vol. 2, pp. 42-46, 2001.
4. L. Skrba, L. Reveret, F. Hetroy, M. –P. Cani and C. O'Sullivan, "Animating Quadrupeds: Methods and Applications," Computer Graphics Forum, vol. 28, pp. 1541-1560, 2009.
5. T. Yaginuma, T. Takesima, E. Shimizu, M. Ito and J. Tahara, "Quadruped Walking with Parallel Link Legs," Artificial Life and Robotics, vol. 15, pp. 555-559, 2010.
6. T. Zielinska and J. Heng, "Multifunctional Walking Quadruped," Robotica, vol. 20, pp. 585-593, 2002.
7. V. M. Budanov and Ye. A. Devyanin, "The Motion of Wheeled Robots," Journal of Applied Mathematics and Mechanics, vol. 67, pp. 215-225, 2003.
8. Yu. G. Martynenko, "Motion Control of Mobile Wheeled Robots," Journal of Mathematical Sciences, vol. 147, pp. 6569-6606, 2007.
9. T. Thueer and R. Siegwart, "Mobility Evaluation of Wheeled All-Terrain Robots," Robotics and Autonomous Systems, vol. 58, pp. 508-519, 2010.
10. N. Elliot, "Bipedal Dynamic Walking in Robotics," Thesis (BEng). The University of Western Australia, 1998.
11. Napoleon, "Balance control analysis of humanoid robot based on ZMP feedback control", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, pp. 2437-2442, 2002.
12. T. S. Li, Y. T. Su, S. H. Liu, J. J. Hu, C. C. Chen, "Dynamic Balance Control for Biped Robot Walking Using Sensor Fusion Kalman Filter and Fuzzy Logic", *Industrial Electronics IEEE Transactions on*, pp. 4394-4408, 2012.
13. A. W. Salatian, K. Y. Viand, Y. F. Zheng*, "Reinforcement learning for a biped robot to climb sloping surfaces", *Journal of Robotic Systems*, vol. 14, no. 4, pp. 283-296, 1997.
14. J.A. Coelho, E.G. Araujo, M. Huber, R.A. Grupen, *Dynamical categories and control policy selection. Intelligent Control*, pp. 459-464, 1998.
15. Song Kai-Tai, C.C. Chang, "Reactive navigation in dynamic environment using a multisensor predictor", *IEEE Trans. Syst. Man Cyber.*, vol. 29, no. 6, pp. 870-880, 1999.
16. D. Kim, S. J. Seo and G. T. Park, "Zero-Moment Point Trajectory Modelling of a Biped Walking Robot Using an Adaptive Neuro-Fuzzy System," in *IEE Proceedings on Control Theory and Applications*, 8 July 2005.
17. J. Lee and J. H. Oh, "Biped Walking Pattern Generation Using Reinforcement Learning," in *7th IEEE-RAS International Conference on Humanoid Robots*, Pittsburgh, PA, 2007.
18. A. W. Salatian, K. Y. Yiand and Y. F. Zheng, "Reinforcement Learning for a Biped Robot to Climb Sloping Surfaces," *Journal of Robotic Systems*, vol. 14, no. 4, pp.283-296, 1997.
19. J. Morimoto, G. Cheng, C. G. Atkeson and G. Zeglin, "A Simple Reinforcement Learning Algorithm for Biped Robot," in *2004 IEEE International Conference on Robotics and Automation*, New Orleans, LA, 2004.
20. Rashmi Sharma, Manish Prateek, Ashok K. Sinha " Use of Reinforcement Learning as a Challenge: A Review" in International Journal of Computer Application, New york, 28-34 May 2013 DOI :10.5120/12105-8332, Vol 69-Number 22. On line available: https://www.ijcaonline.org/archives/volume69/number22/12105-8332
21. K. Berns, R. Dillmann and U. Zachmann (1992) : Reinforcement-learning for the control of an autonomous mobile robot, in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Raleigh, NC, 1808–1815 .
22. R. S. Sutton *et al. (1998): Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA.
23. C. Watkins (1989): Learning from delayed rewards, PhD Dissertation, Cambridge University.
24. S. D. Whitehead (1992): Reinforcement learning for the adaptive control of perception and Action, *Technical Report 406*. University of Rochester.
25. Gerhard Weib (1995): Distributed Reinforcement Learning, *Robotics and Autonomous Systems*, vol. 15, Elsevier, 135-142.
26. Changjiu Zhou (2002): Robot learning with GA-based fuzzy reinforcement learning agents, *Information Sciences*, Vol 145, Elsevier, 45–68.
27. Gary G. Yen, Travis W. Hickey (2004): Reinforcement learning algorithms for robotic navigation in dynamic environments *ISA Transactions*, Vol.43, 217–230.
28. Yi-Chi Wang, John M. Usher (2005): Application of reinforcement learning for agent-based production scheduling, *Engineering Applications of Artificial Intelligence* Vol. 18, Elsevier, 73–82.
29. Yong Duan, Qiang Liu, XinHe Xu (2007): Application of reinforcement learning in robot soccer, *Engineering Applications of Artificial Intelligence* Vol. 20, Elsevier, 936–950.
30. Maryam Shokri (2011): Knowledge of opposite actions for reinforcement learning , *Applied Soft Computing* ,Vol. 11 , Elsevier, 4097–4109.
31. Darío Maravall, Javier de Lope, Raúl Domínguez (2012): Coordination of communication in robot teams by reinforcement learning, *Robotics and Auto nomous Systems* doi:10.1016/j.robot 2012.07.007 , Elsevier.
32. Wang Qiang , Zhan Zhongli (2011) : Reinforcement Learning Model , Algorithms and its application ,*International conference on Mechatronic Science, Electric Engineering and Computer*.
33. Singh S(1997): *Agents and reinforcement learning [M]*. San Matco, CA, USA: Miller freeman publish Inc.
34. R.S. Sutton, A.G. Barto(1998): *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA.
35. L.P. Kaelbling, M.L. Littman, A.W. Moore (1996): Reinforcement learning: a survey, *Journal of Artificial Intelligence Research*, Vol 4, Elsevier, 237–285.
36. Richard S. Sutton.(1998): Learning to predict by the method of temporal differences. *Machine Learning*, Vol 3(1), 9-44.
37. Christopher J. C. H. Watkins (1989) Learning from Delayed Rewards. PhD thesis, King's College, Cambridge, UK.
38. Prasad Tadepalli, DoKyeong Ok(1998) : Model-based average reward reinforcement learning, *Artificial Intelligence*, Vol. 100 ,Elsevier , 177-224

AUTHORS PROFILE

Ms. Rashmi Sharma, received her Master's in Computer Application (1999) DAVV Indore (M.P.) and Master's in Technology (2010) UPTU, Lucknow. At present, doing research in field of Reinforcement Learning and Bipedal. Interest area includes - Soft Computing, Artificial Intelligence, Machine Learning and Machine Vision.

.

Dr. Inder Singh, M.Sc.(IT), M. Tech. (IT), Ph.D., Microsoft Certified Professional, IBM DB2 certified, Dell EMC's Certified Data Science Associate. He is an Assistant Professor (S.G.) at School of Computer Science and Engineering, UPES, Dehradun. He has over 17 years of working experience. He has started his career as Systems Administrator and switched to teaching profession after 6 years. His area of research includes Computer Networks, Cloud Computing and Virtualization, and Data Science. So far, he has published and presented more than 30 research papers in different International Journals and conferences

Mr Deepak Bharadwaj, received his M.Tech Degree in 2008-10 from the MDU university Rohtak.. At present he is working as Assistant Professor in mechanical engineering department of UPES Dehradun. His area of research is robotics & Automation. At present he is developing the reinforcement control algorithm for an autonomous humanoid robot.

. Professor Manish Prateek, currently working as professor of computer science and Engineering. His area of research is soft computing and wireless network. He had published may research paper on robotics. Currently he is the Dean of School of computer science department Of UPES Dehradun.