Intelligent Environments 2021
E. Bashir and M. Luštrek (Eds.)

doi:10.3233/AISE210092

Learning to Move an Object by the Humanoid Robots by Using Deep Reinforcement Learning

Simge Nur ASLAN [a,1], Burak TAŞÇI [a,2], Ayşegül UÇAR [a,3], and Cüneyt GÜZELİŞ [b]
[a1,3] *Firat University, Department of Mechatronics Engineering, Elazig, Turkey*
[a1,2] *Firat University, Vocational School of Technical Sciences, Elazig, Turkey*
[b] *Yaşar University Department of Electrical and Electronics Engineering, Izmir, Turkey*

Abstract. This paper proposes an algorithm for learning to move the desired object by humanoid robots. In this algorithm, the semantic segmentation algorithm and Deep Reinforcement Learning (DRL) algorithms are combined. The semantic segmentation algorithm is used to detect and recognize the object be moved. DRL algorithms are used at the walking and grasping steps. Deep Q Network (DQN) is used to walk towards the target object by means of the previously defined actions at the gate manager and the different head positions of the robot. Deep Deterministic Policy Gradient (DDPG) network is used for grasping by means of the continuous actions. The previously defined commands are finally assigned for the robot to stand up, turn left side and move forward together with the object. In the experimental setup, the Robotis-Op3 humanoid robot is used. The obtained results show that the proposed algorithm has successfully worked.

Keywords. Humanoid robots, DQN, DDPG, deep semantic segmentation, object manipulation, locomotion

1. Introduction

The humanoid robots are expected to navigate by themselves in environments such as the house, office, and hospital in a normal daily life and move the desired objects [1-4]. The combination of the locomotion and manipulation skills of the robots is required for the application. Especially, the robots should be capable of object detection and recognition capabilities. Hence, the development of artificial intelligence algorithms for robots to autonomously transport objects is still a challenging research topic.

Deep learning algorithms were used for a lot of applications in smart environments [5-9]. In this paper, we investigate different deep learning algorithms for the walking

[1] Simge Nur Aslan, Corresponding author, Department of Mechatronics Engineering, Firat University, 23119, Elazig, Turkey; E-mail: simgeaslan124@gmail.com.
[2] Burak Taşcı, Author, Vocational School of Technical Sciences, Firat University, 23119, Elazig, Turkey; E-mail: btasci@firat.edu.tr.
[3] Ayşegül Uçar, Author, Department of Mechatronics Engineering, Firat University, 23119, Elazig, Turkey; E-mail: agulucar@firat.edu.tr.
4 Cüneyt Güzeliş, Author, Department of Electrical and Electronics Engineering, Yaşar University, Izmir, Turkey; E-mail: cuneyt.guzelis@yasar.edu.tr

task towards the object and the manipulating task it. In this context, the recent attention has focused on Deep Reinforcement Learning (DRL) for solving complex and high dimensional problems [9-18]. In the literature, the locomotion and manipulation tasks were generally taken into consideration in separate scenarios. In [10-15], the DRL algorithms were worked the basic tasks such as pushing, grasping, and pulling were worked on robotic manipulators. In [16], Reinforcement Learning (RL) method was used for reaching task for Baxter humanoid robot arm to a target pose that is controlled by a human. [17] proposed two RL based hierarchies plans to perform the task of manipulating and grasping the objects with uncertain disturbance for a humanoid like mobile robot having the human like upper body includes two robotic arms. In [18-20], some deep learning methods were applied for object detection and manipulation in cluttered environment on the Nao Robot, the REEM-C robot, and Romeo. They did not include DRL. In [21-26], few powerful DRL algorithms were developed for locomotion tasks of both biped and humanoid robots. In our earlier works, we carried out separately the locomotion and grasping tasks for the humanoid robots [27-30]. We applied the demonstration learning by using different deep learning models such as Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks in [27]. In [28], we applied semantic segmentation for grasping tasks. We deployed DRL for different scenarios including walking tasks in [29-30].

In this paper, the locomotion and manipulation tasks are combined by using DRL as different from the works in the literature. The semantic segmentation algorithm is proposed to detect the target object. Pyramid Scene Parsing Network (PSPNet) is used for semantic segmentation [31]. The outputs of the network are imported to the input of the next stage. Two modules for walking and manipulation are planned. Deep Q Network (DQN) is prepared to walk towards the target object by using all combinations of the defined actions using the gait manager and the different head positions of the robot. Deep Deterministic Policy Gradient (DDPG) network is employed for manipulating the object [32-35]. The previously defined commands are finally assigned for the robot to stand up, turn left side and to move forward together with the grasped object. In the experimental setup, the Robotis-Op3 humanoid robot [36] is used. The obtained results show that the proposed algorithm successfully worked.

The rest of this paper is organized as follows. In section 2, the fundamentals of CNNs, PSPNet, DQN, and DDPG are introduced. Section 3 gives the details of the proposed vision based algorithm. The experimental setup and results are presented, respectively in Section 4. The conclusions and future direction are presented in Section 5.

2. Theoretical background

This section briefly reviews the fundamental concepts of CNNs, PSPNet, DQN, and DDPG.

2.1. Convolutional neural networks

CNNs are a kind of artificial neural networks with multiple layers [37-38]. The inputs of the network are the images and/or the signals. They are processed end-to-end. Classification, detection, and recognition tasks are carried out without using a feature extraction method. A CNN structure consists of the combinations of the input layer, the convolutional layers, the Batch Normalization layers (BN), the pooling layers, and the

fully-connected layers [39]. The input layer accepts two-dimensional signals or three-dimensional images. If we consider using the images for our aim, the images are then filtered in the convolutional layer by sliding the filter matrices. In the pooling layer, the dimensions of image matrices are reduced by using the functions average, max poling, and min. In other ways, the layer is a feature selection layer. In BN layer, the layer inputs are normalized to provide zero mean and unit variance. Rectified Linear Unit Activation Function (ReLU) activation function is usually used after BN layer [40]. ReLU provides zero output for negative input, and itself input for the others. In a fully-connected layer, each unit is exactly connected to the next layer similar to feed forward neural networks. The data in a vector form are obtained. The final layer of CNN includes a softmax or a linear/nonlinear activation function. The softmax activation function calculates the class probabilities relating to each output [38-39].

2.2. Pyramid scene parsing network

PSPNet is a network used for semantic segmentation. The network consists of the encoder and decoder parts. The structure is shown in Fig.1 [31]. Fig. 1 consists of the encoder part and the decoder part. The encoder part can be generated by a vanilla CNN constructed with (BN), ReLU activation, max pooling, and zero padding or famous CNN models such as ResNet, VGG16, MobileNet, and FCNs [41-44]. The decoder part is a single standard structure. The decoder part is depicted by the blocks called Pool, Conv, Upsample, Concat, and Conv. Pool, Conv, and Upsample blocks are generated from four layers of average pooling, convolution, BN, ReLU activation, and resize. The feature map dimensions are [64, 128, 256, 256, 256] and the polling outputs are [1x1, 2x2, 3x3, 6x6]. The Concat block means the concentration as a single tensor of the inputs at the final. The final Conv block includes multiple layers such as the convolution, BN, ReLU, convolution, resize, reshape, and Softmax activation layers, respectively.

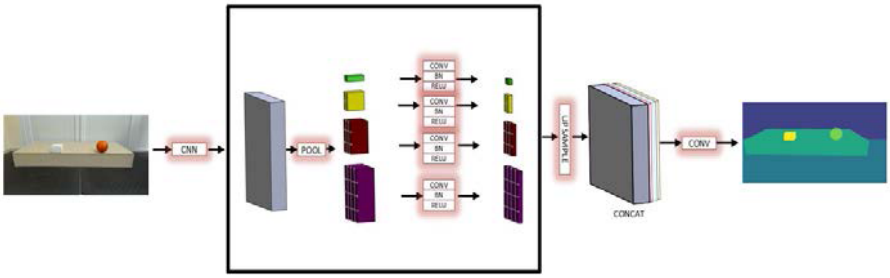

Figure 1. PSPNet architecture [31].

2.3. Deep Q network

There are three learning methods classified as supervised, unsupervised, and RL. RL methods use a reward and penalty values. The correct action provides a reward and the wrong action provides a penalty. The agent interacts with the environment, receives the values, and then applies the appropriate action [32-33]. Thus, the best policy is provided. AlphaGo game is the best known example of RL. The computer algorithm being powered with the RL has won the game against the best Go player [34].

Fig. 2 shows a kind of DRL algorithm called DQN. In Fig. 2, the agent is a CNN. It decides to an action $a_t \in A$ from action space with respect to the state S_t by interacting

with the environment at t∈[0, T] and obtains a reward or penalty R_t with respect to the selected action and pass to a new state s_{t+1}. It generates the policy $a_t = \pi(s_t)$.

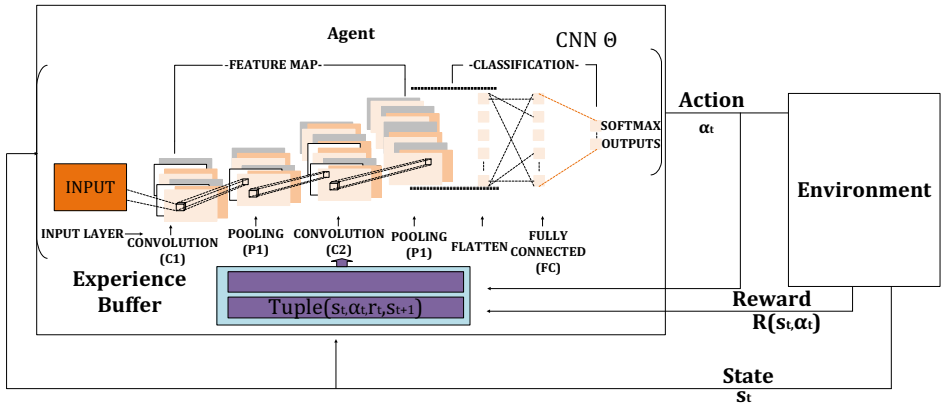

Figure 2. Structure of deep Q network.

In DQN, the cumulative reward is maximized as in

$$R_t = \sum_{i=t}^{T} \gamma^{i-t} r_i \tag{1}$$

where γ is a discount factor in the range of [0,1]. Being given a policy, the optimal Q-value function is obtained by using Belmann equality

$$Q^{\pi}(s_t, a_t) = \mathrm{E}[R_t | s_t, a_t, \pi], \tag{2}$$

$$Q^{\pi}(s_t, a_t) = \mathrm{E}\big[r_t + \gamma \mathrm{E}[Q^{\pi}(s_{t+1}, a_{t+1}) | s_t, a_t, \pi]\big], \tag{3}$$

$$Q^{*}(s_t, a_t) = \mathrm{E}_{s_{t+1}}\big[r + \gamma max_{a_{t+1}} Q^{*}(s_{t+1}, a_{t+1}) | s_t, a_t\big]. \tag{4}$$

Q-learning is realized by:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t[r_{t+1} + \gamma max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]. \tag{5}$$

DDPG in Fig.3 integrates both the actor-critic and DQN methods to learn the policies in the continuous domain. It provides the continuous actions. In DDPG, $Q(s, a)$, Q-value network represents a critic function estimating the value of state-action pairs. The actor function called the policy network is updated by

$$\nabla_{\theta^{\mu}} \approx \mathrm{E}_{s_{t\sim\rho\pi}}\left[\nabla_a Q(s, a|\theta^{Q})|_{s=s_t, a=\mu(s_t)} \nabla_{\theta_{\mu}} \mu(s|\theta^{\mu})|_{s_t}\right] \tag{6}$$

where ρ_{π} means the transitions obtained from the π stochastic behavior policy. The policy is generally a Gaussian distribution with the center of $\mu(s|\theta^{\mu})$ depending on the parameter θ.

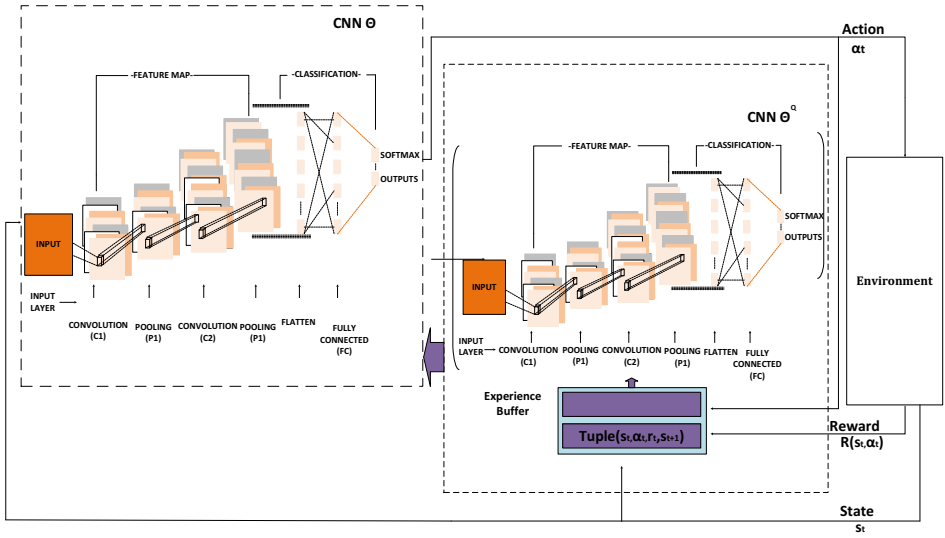

Figure 3. The structure of deep deterministic policy gradient network.

3. Vision-based walking and object manipulation

In this paper, a vision based experimental procedure is designed to walk to the target and to manipulate the object. The flowchart in Fig. 4 depicts the experimental plan for the pilot scenario. In this plan, Robotis-Op3 is controlled using a keyboard and the images are collected by using its camera as moving the robot. Three deep learning algorithms are then executed. One of them is the deep semantic segmentation algorithm and the others are DRL algorithms. PSPNet semantic model is applied to segment the objects in the environment and find the target object. To use at the training of the PSPNet model, all images are manually labeled in Matlab. DQN is applied for the robot to walk to the target. DDPG is applied for the robot to manipulate the target object. Finally, the networks are transferred to the real robot by means of Robot Operating System (ROS) and the pre-defined actions are applied to the robot in order.

4. Experimental results

In the study, we used the Robotis-Op3 humanoid robot and experimental environment consisting of the room of (2m x 4m) in Fig. 5 [36]. We aimed that the robot walks to the target object and to manipulate it. Robotis-Op3 consists of 20 axes and has a Logitech C920 HD Pro camera, 9 degrees of freedom inertial measurement unit, Intel NUC i3, Linux operating system, Dynamixel SDK, and ROS. In the experiments, we worked on a workstation having Nvidia Titan XP.

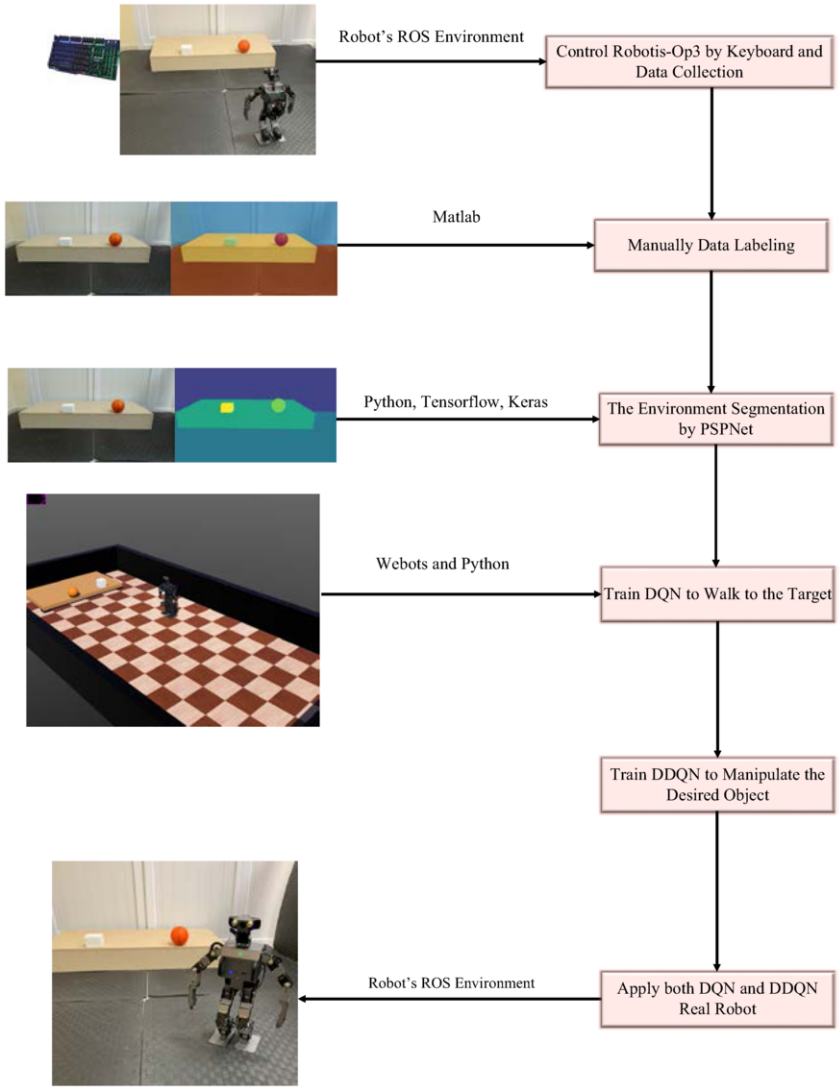

Figure 4. The flowchart of the proposed system.

Figure 5. Robotis-Op3 humanoid robot and the experimental environment.

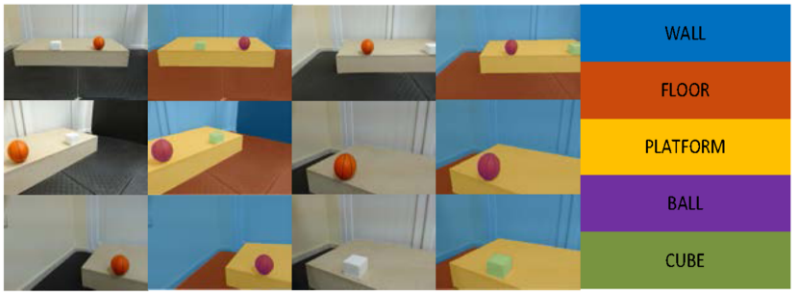

Figure 6. Some images used in the training data set.

To walk towards the desired object, we used the gait pattern generator that is available in Robotis-Op3 source code [36]. The pattern generator deploys three coupled oscillators, for the right foot, the left foot and the center of mass with sinusoidal trajectory in a synchronized way with respect to the movement. Moreover, it has closed loop control to balance using the gyro data to get rid of falling [45]. In our experiment, firstly, we controlled the robot by a keyboard. We captured a total of 3211 images consisting of 2269 training images and 942 validation images from the real Robotis-Op3 camera by using ROS to walk to target.

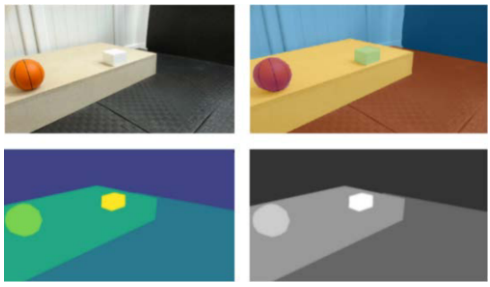

Figure 7. The RGB image used in the training data set and its labelled version (top), the segmented image and its depth version (bottom).

Fig. 6 shows the sample images in the training data set. We labeled five objects consisting of the ball, cube, platform, floor, and wall. Each object painted with a different color. Fig. 7 shows the captured real RGB image, the labeled images, the segmented images, and the depth images. We used completely real images for the training. We tried different models such as ResNet50-PSPNet, VGG16-UNet, VGG16-FCN32, VGG16-PSPNet, and VGG16-Segnet [41-44, 46-47]. We trained all models by the Adedelta optimization algorithm for 40 epochs, 2 batch size, and 100 steps [48]. We evaluated the network by the accuracy, the Dice coefficient, and the Intersection Over Union (IOU) coefficient to measure the segmentation performance [49-51]. The values near one of the accuracy, Dice, and IOU exhibit perfect object boundaries [49-50]. We used ResNet50-PSPNet in our scenario since it achieved better performance than the other networks. The results of ResNet50-PSPNet relating to the accuracy, Dice, and IOU are shown in Fig. 8a-c for each epoch, respectively. As can be seen from Fig. 8 and Table 1, the value of {0.9983, 0.9988, 0.9945} were obtained for accuracy, Dice, and IOU in the training stage, respectively by ResNet50-PSPNet. In the validation stage, we obtained the values of

{0.9971, 0.9978, 0.9928}. The results reveal that the ResNet50-PSPNet can successfully segment the experimental environment.

Table 1. Segmentation performance of state-of-art algorithms.

Model	Train			Validation		
Encoder-Decoder	**Accuracy**	**Dice**	**IOU**	**Accuracy**	**Dice**	**IOU**
ResNet50-PSPNet	0.9983	0.9988	0.9945	0.9971	0.9978	0.9928
VGG16-PSPNet	0.9719	0.9730	0.9709	0.9663	0.9676	0.9705
VGG16-UNet	0.9844	0.9865	0.9740	0.9713	0.9726	0.9457
VGG16-FCN32	0.9834	0.9832	0.9810	0.9738	0.9663	0.9386
VGG16-Segnet	0.9811	0.9856	0.9534	0.9768	0.9767	0.9402

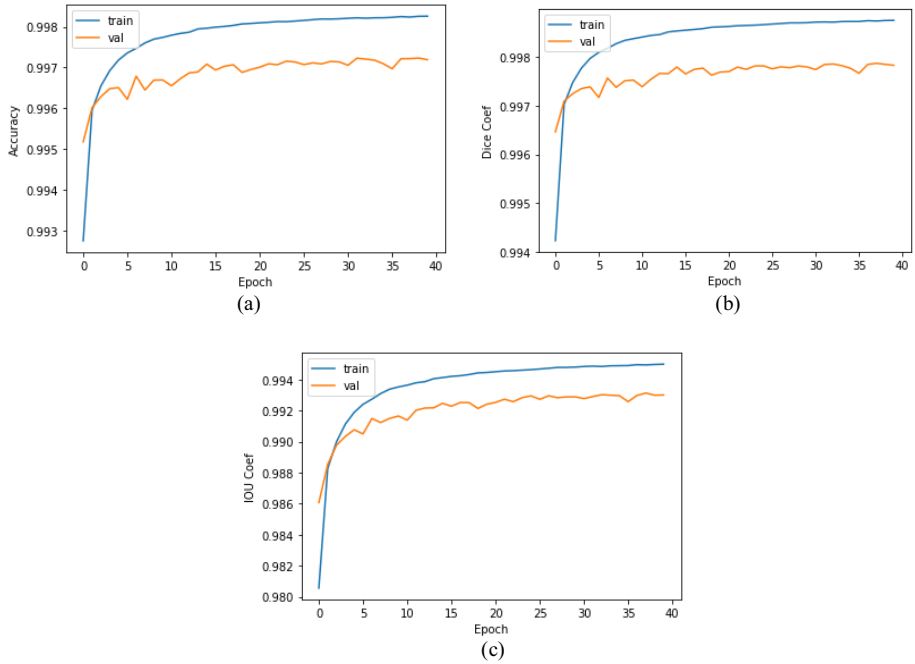

Figure 8. The training and validation results for the accuracy, Dice, and IOU.

Secondly, we constructed our experimental environment in the Webots simulation platform for the simulation works [52]. We trained our robot to learn to walk towards the target object by using DQN in Webots. We defined the combinations of discrete actions consisting of forward, back, left, and right of the movement and right, left, straight, down on the head and applied DQN structure in [29]. We used both the depth image and segmented image as the DQN inputs. Depth image corrupted with random noise and blur was used for achieving the robustness to the differences of illumination and appearance between the real and simulation environments. The segmented image was used to determine the target. We centered the object on the image taken from the robot camera in respect to the neck and head positions [36,52-53]. We calculated the distance and the orientation to the target using the head tilt angle and the neck pan angle, respectively [36,52-53]. We selected the eight actions such as forward, backward, turn left, turn right, and stop. Table 2 shows the selected discrete actions in respect to the robot velocities and orientation. We determined the reward with respect to the distance to the target. We

randomly selected the label of the target object in the environment by using the segmented image at the beginning of each episode. We fixed the distance between the objects since the robot is able easily to grasp one object, but we randomly changed the initial locations of both objects on the table at the beginning of each episode. We randomly re-initialized the location of the robot at each episode. When the robot fell down and the robot walked more than 8 minutes in the room of 2mx4m, we terminated and reset, respectively. The obtained mean reward from the used DQN at the walking task towards the desired object is given in Fig. 9. The results reveal that our agents can successfully learn to walk to the target object no matter what the size, shape, and location of it are.

Table 2. The actions to robot velocities and orientation.

Action	Linear (cm/s), Angle (degree)
Forward	[6, ±-15] and [1-3, ±[0-15]]
Backward	[1,7],[1,-15] [8,-7.5]
Noop	-

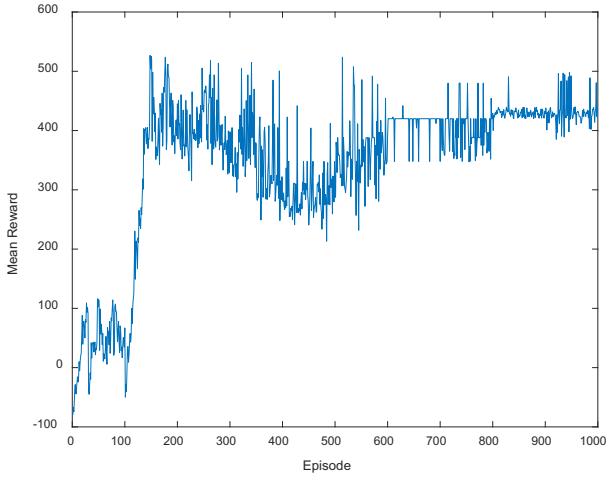

Figure 9. Mean reward with respect to training episodes for DQN.

To test the performance of DQN, we measured the success rate of the reaching to the target when moving the robot from four different initial positions and the rotation. We determined the middle point of the environment as (0,0) point. Each position was evaluated 20 times, resulting in the success percentages reported in Table 3. The results show that our agent is successful in both the real-world and simulation scenarios.

Table 3. Comparison of and simulation and real world experiments to walk to the target.

Initial Value (x,y, angle)	(0,0,35º)	(0,0,0º)	(0.6,0.6,35º)	(-0.5,-0.5,-90º)
Real-World	86.24 %	98.12 %	83.87 %	69.45 %
Simulation	92.03 %	100 %	93.45 %	88 .67%

Thirdly, in order to manipulate object, we then put the robot to the sitting position and run DDPG for grasping as an autonomous step in the algorithm. We controlled the position of right shoulder and left shoulder, right arm upper and left arm upper, right arm

lower and left arm lower by using the continuous actions thanks to DDPG. In DDPG having 4 layered actor and critic networks, we selected critic learning rate, actor learning rate, discount factor, critic l2 regularization as 1e-3, 1e-4, 0.9, 1e-2, respectively. We trained the robot in Webots. We received exactly the simulated RGB images as the inputs. We limited the robot arm's workspace and started a new episode and penalized the robot when it digresses off limits. We evaluated the relative location of the object and robot arm in the reward function. We determined the positive and negative reward values as the object moves up and down, respectively. The obtained mean reward is given in Fig. 10. As can be seen from Fig. 10, the models learn perfectly all steps to manipulate the object after 300 episodes.

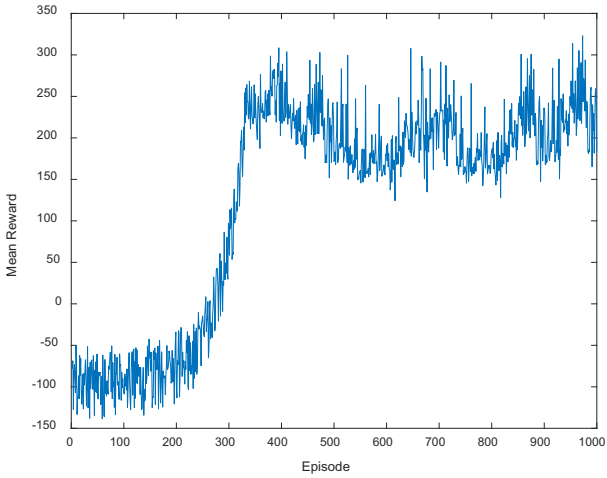

Figure 10. Mean reward with respect to training episodes for DDPG.

We applied all steps of the algorithm and embedded the DQN and DDPG networks into the robot in the ROS environment. To test the performance of DDPG, we measured the success rate of the object manipulation. We repeated 20 times the manipulation task for the object in both real world and simulation environments. As can be seen from Table 4, our agent provides the good results for both real world and simulation. Failure of manipulation trials are due to the robot hand with clamp shape. It is generally not seen as the failure of the agent. The robot catches the object, but it could not hold the object without dropping it due to the hand's clamp shape.

Table 4. Comparison of and simulation and real world experiments to manipulate the object

Model	Real-World	Simulation
DDPG	67 %	79 %

After carrying out the manipulation task, we finally applied the commands of standing up, turning left side, and straight walking, respectively to move the object.

5. Conclusions

In this paper, we presented new algorithm steps for the humanoid robots to learn to move an object by using DRL. The algorithm included three tasks, as object detection, locomotion, and manipulation. The first task was carried out by a deep semantic segmentation network, the second and third tasks deployed DQN and DDPG respectively.

All models were trained in the simulation environment and then the obtained networks were embedded into the robot. All algorithm steps were successfully performed on the real robot. The algorithm can be applied to walk towards one from previously labelled objects in a cluttered environment and to grasp it without re-training the networks. Moreover, the algorithm can be extended for the obstacle avoidance task with the appropriate reward and actions. Future studies will go ahead to work in the clutter environments with the same algorithm.

Acknowledgment

This work was supported by the Scientific and Technological Research Council of Turkey (TUBITAK) grant numbers 117E589. In addition, GTX Titan X Pascal GPU in this research was donated by the NVIDIA Corporation

References

[1] M. Grey, S. Joo, and M. Zucker, Planning heavy lifts for humanoid robots, in: 2014 IEEE-RAS International Conference on Humanoid Robots, IEEE, 2014: pp. 640–645.

[2] S.A.A. Moosavian, A. Janati, and M.H. Ghazikhani, Object manipulation by two humanoid robots using MTJ control, in: 2011 IEEE International Conference on Mechatronics and Automation, IEEE, 2011: pp. 1286–1290.

[3] M.-H. Wu, A. Konno, S. Ogawa, and S. Komizunai, Symmetry cooperative object transportation by multiple humanoid robots, in: 2014 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2014: pp. 3446–3451.

[4] A. Rioux, and W. Suleiman, Autonomous SLAM based humanoid navigation in a cluttered environment while transporting a heavy load, *Robotics and Autonomous Systems*. **99** (2018) 50–62.

[5] S.I. Muzaffar, K. Shahzad, K. Malik, and K. Mahmood, Intention mining: A deep learning-based approach for smart devices, *Journal of Ambient Intelligence and Smart Environments*. (2020) 1–13.

[6] M.A. Guillén, A. Llanes, B. Imbernón, R. Martínez-España, A. Bueno-Crespo, J.-C. Cano, and J.M. Cecilia, Performance evaluation of edge-computing platforms for the prediction of low temperatures in agriculture using deep learning, *The Journal of Supercomputing*. **77** (2021) 818–840.

[7] J. Kar, M.V. Cohen, S.P. McQuiston, and C.M. Malozzi, A deep-learning semantic segmentation approach to fully automated MRI-based left-ventricular deformation analysis in cardiotoxicity, *Magnetic Resonance Imaging*. **78** (2021) 127–139.

[8] Z.-W. Hong, C. Yu-Ming, S.-Y. Su, T.-Y. Shann, Y.-H. Chang, H.-K. Yang, B.H.-L. Ho, C.-C. Tu, Y.-C. Chang, and T.-C. Hsiao, Virtual-to-real: Learning to control in visual semantic segmentation, *ArXiv Preprint ArXiv:1802.00285*. (2018).

[9] S. Kumra, S. Josh, and F. Sahin, Learning Robotic Manipulation Tasks through Visual Planning, *ArXiv Preprint ArXiv:2103.01434*. (2021).

[10] S. Gu, E. Holly, T. Lillicrap, and S. Levine, Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates, in: 2017 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2017: pp. 3389–3396.

[11] T. Haarnoja, V. Pong, A. Zhou, M. Dalal, P. Abbeel, and S. Levine, Composable deep reinforcement learning for robotic manipulation, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2018: pp. 6244–6251.

[12] A. Hundt, B. Killeen, N. Greene, H. Wu, H. Kwon, C. Paxton, and G.D. Hager, "Good Robot!": Efficient Reinforcement Learning for Multi-Step Visual Tasks with Sim to Real Transfer, *IEEE Robotics and Automation Letters*. **5** (2020) 6724–6731.

[13] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, Learning complex dexterous manipulation with deep reinforcement learning and demonstrations, *ArXiv Preprint ArXiv:1709.10087*. (2017).

[14] S. Joshi, S. Kumra, and F. Sahin, Robotic grasping using deep reinforcement learning, in: 2020 IEEE 16th International Conference on Automation Science and Engineering (CASE), IEEE, 2020: pp. 1461–1466.

[15] P. Florence, L. Manuelli, and R. Tedrake, Self-supervised correspondence in visuomotor policy learning, *IEEE Robotics and Automation Letters*. **5** (2019) 492–499.

[16] R. Silva, M. Faria, F.S. Melo, and M. Veloso, Adaptive indirect control through communication in collaborative human-robot interaction, in: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2017: pp. 3617–3622.

[17] Z. Li, T. Zhao, F. Chen, Y. Hu, C.-Y. Su, and T. Fukuda, Reinforcement learning of manipulation and grasping using dynamical movement primitives for a humanoidlike mobile manipulator, *IEEE/ASME Transactions on Mechatronics*. **23** (2017) 121–131.

[18] A. Hornung, S. Böttcher, J. Schlagenhauf, C. Dornhege, A. Hertle, and M. Bennewitz, Mobile manipulation in cluttered environments with humanoids: Integrated perception, task planning, and action execution, in: 2014 IEEE-RAS International Conference on Humanoid Robots, IEEE, 2014: pp. 773–778.

[19] P. Regier, A. Milioto, C. Stachniss, and M. Bennewitz, Classifying obstacles and exploiting class information for humanoid navigation through cluttered environments, *International Journal of Humanoid Robotics*. **17** (2020) 2050013.

[20] G. Claudio, F. Spindler, and F. Chaumette, Vision-based manipulation with the humanoid robot Romeo, in: 2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids), IEEE, 2016: pp. 286–293.

[21] D.R. Song, C. Yang, C. McGreavy, and Z. Li, Recurrent deterministic policy gradient method for bipedal locomotion on rough terrain challenge, in: 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV), IEEE, 2018: pp. 311–318.

[22] X.B. Peng, G. Berseth, and M. Van de Panne, Terrain-adaptive locomotion skills using deep reinforcement learning, *ACM Transactions on Graphics (TOG)*. **35** (2016) 1–12.

[23] X.B. Peng, G. Berseth, K. Yin, and M. Van De Panne, Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning, *ACM Transactions on Graphics (TOG)*. **36** (2017) 1–13.

[24] K. Lobos-Tsunekawa, F. Leiva, and J. Ruiz-del-Solar, Visual navigation for biped humanoid robots using deep reinforcement learning, *IEEE Robotics and Automation Letters*. **3** (2018) 3247–3254.

[25] S. Khatibi, M. Teimouri, and M. Rezaei, Real-time Active Vision for a Humanoid Soccer Robot Using Deep Reinforcement Learning, *ArXiv Preprint ArXiv:2011.13851*. (2020).

[26] Q. Shi, W. Ying, L. Lv, and J. Xie, Deep reinforcement learning-based attitude motion control for humanoid robots with stability constraints, *Industrial Robot: The International Journal of Robotics Research and Application*. (2020).

[27] S.N. Aslan, R. Ozalp, A. Uçar, and C. Güzeliş, End-To-End Learning from Demonstation for Object Manipulation of Robotis-Op3 Humanoid Robot, in: 2020 International Conference on INnovations in Intelligent SysTems and Applications (INISTA), IEEE, 2020: pp. 1–6.

[28] S.N. Aslan, A. Uçar, and C. Güzeliş, Semantic Segmentation for Object Detection and Grasping with Humanoid Robots, in: 2020 Innovations in Intelligent Systems and Applications Conference (ASYU), IEEE, 2020: pp. 1–6.

[29] R. Özalp, C. Kaymak, Ö. Yildirim, A. Ucar, Y. Demir, and C. Güzeliş, An implementation of vision based deep reinforcement learning for humanoid robot locomotion, in: 2019 IEEE International Symposium on INnovations in Intelligent SysTems and Applications (INISTA), IEEE, 2019: pp. 1–5.

[30] S.N. Aslan, A. Uçar, and C. Güzeliş, Development of Deep Learning Algorithm for Humanoid Robots to Walk to the Target Using Semantic Segmentation and Deep Q Network, in: 2020 Innovations in Intelligent Systems and Applications Conference (ASYU), IEEE, 2020: pp. 1–6.

[31] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, Pyramid scene parsing network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017: pp. 2881–2890.

[32] R.S. Sutton, and A.G. Barto, Reinforcement learning: An introduction, MIT press, 2018.

[33] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, and G. Ostrovski, Human-level control through deep reinforcement learning, *Nature*. **518** (2015) 529–533.

[34] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, and M. Lanctot, Mastering the game of Go with deep neural networks and tree search, *Nature*. **529** (2016) 484–489.

[35] T.P. Lillicrap, J.J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, Continuous control with deep reinforcement learning, *ArXiv Preprint ArXiv:1509.02971*. (2015).

[36] Robotis-Op3, http://emanual.robotis.com/docs/en/platform/op3/introduction/ Accessed 5 October 2020.

[37] A. Uçar, Y. Demir, and C. Güzeliş, Object recognition and detection with deep learning for autonomous driving applications, *Simulation*. **93** (2017) 759–769.

[38] Y. LeCun, Y. Bengio, and G. Hinton, Deep learning. nature 521 (7553), 436-444, *Google Scholar Google Scholar Cross Ref Cross Ref*. (2015).

[39] S. Ioffe, and C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: International Conference on Machine Learning, PMLR, 2015: pp. 448–456.

[40] V. Nair, and G.E. Hinton, Rectified linear units improve restricted boltzmann machines, in: Icml, 2010.

[41] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016: pp. 770–778.

[42] K. Simonyan, and A. Zisserman, Very deep convolutional networks for large-scale image recognition, *ArXiv Preprint ArXiv:1409.1556*. (2014).

[43] J. Long, E. Shelhamer, and T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015: pp. 3431–3440.

[44] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, Mobilenets: Efficient convolutional neural networks for mobile vision applications, *ArXiv Preprint ArXiv:1704.04861*. (2017).

[45] I. Ha, Y. Tamura, and H. Asama, Gait pattern generation and stabilization for humanoid robot based on coupled oscillators, in: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2011: pp. 3207–3212.

[46] O. Ronneberger, P. Fischer, and T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2015: pp. 234–241.

[47] V. Badrinarayanan, A. Kendall, and R. Cipolla, Segnet: A deep convolutional encoder-decoder architecture for image segmentation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **39** (2017) 2481–2495.

[48] M.D. Zeiler, Adadelta: an adaptive learning rate method, *ArXiv Preprint ArXiv:1212.5701*. (2012).

[49] A. Popovic, M. De la Fuente, M. Engelhardt, and K. Radermacher, Statistical validation metric for accuracy assessment in medical image segmentation, *International Journal of Computer Assisted Radiology and Surgery*. **2** (2007) 169–181.

[50] L.R. Dice, Measures of the amount of ecologic association between species, *Ecology*. **26** (1945) 297–302.

[51] S.V. des S. Naturelles, Bulletin de la Société vaudoise des sciences naturelles, F. Rouge, 1864.

[52] O. Michel, Cyberbotics Ltd. Webots™: professional mobile robot simulation, *International Journal of Advanced Robotic Systems*. **1** (2004) 5.

[53] W. Budiharto, B. Kanigoro, and V. Noviantri, Ball Distance Estimation and Tracking System of Humanoid Soccer Robot, in: Information and Communication Technology-EurAsia Conference, Springer, 2014: pp. 170–178.