Deep Reinforcement Multi-Directional Kick-Learning of a Simulated Robot with Toes

Martin Spitznagel
David Weiler
Klaus Dorer
Institute for Machine Learning and Analytics
Offenburg University, Germany
{mspitzna,dweiler}@stud.hs-offenburg.de, dorer@imla.ai

***Abstract*—This paper describes a thorough analysis of using PPO to learn kick behaviors with simulated NAO robots in the simspark environment. The analysis includes an investigation of the influence of PPO hyperparameters, network size, training setups and performance in real games. We believe to improve the state of the art mainly in four points: first, the kicks are learned with a toed version of the NAO robot, second, we improve the reliability with respect to kickable area and avoidance of falls, third, the kick can be parameterized with desired distance and direction as input to the deep network and fourth, the approach allows to integrate the learned behavior seamlessly into soccer games. The result is a significant improvement of the general level of play.**

***Index Terms*—Deep Reinforcement Learning, Proximal Policy Optimization (PPO), toed robot, kick learning**

I. Introduction

Behavior learning plays a key role since the early days of robotics and in RoboCup specifically. Good progress has been made by applying genetic learning algorithms to the optimization of parameters of model based behaviors [5], [8] and even model free behaviors [3]. However, especially the model free approaches have the limitation that they are open loop: the behaviors learned do not take the current observations and state of the agent into account. They are replayed as learned and fail, if they are triggered in situations that differ from the situation during learning.

Reinforcement Learning does not have this limitation, but had been limited to small observation and action spaces. In recent years however, deep reinforcement learning (DRL) algorithms like off-policy algorithms DDPG or DQN or on-policy like A2C or PPO [6] have overcome these limitations and work in comparably huge continuous observation and continuous action spaces. By using those algorithms, professional e-sport teams, for example, could be beaten by artificial neural networks in the competitive multiplayer game Dota2[1].

While this has already been applied very successfully to learning to walk in the simspark domain [2], [4], progress on kick behaviors and the usage of toed robots remains an open issue. Also, in these works there is a gap between the extremely successful behavior performance in training setup and the benefit from it during real game play. This paper describes an approach to overcome these limitations.

The remaining of the paper is organized as follows: Section II presents related work on learning humanoid robot behaviors with PPO. Section III provides background of the simspark domain used for learning. Section IV explains the approach and setup used for this evaluation. Section V presents in depth results to validate our approach. Finally, section VI concludes and shares ideas for future work.

II. Related Work

Deep Reinforcment Learning and especially PPO have gained increasing interest in robotics and was first introduced into the simspark domain with the work of Abreu et al. [1]. They demonstrated a stable walk (better say run) learned through PPO that was 2.5 times faster than the best teams at that time in the 3D soccer simulation league. They were able to integrate the walk into team play, but the overall performance in game play so far did not profit in the same manner. The observation space used in their work was used as the starting point for this work and adjusted to the kick task.

The extreme walking speed of 2.5 m/s has meanwhile been increased in work of [4] to more than 3 m/s. They examined the advantage of end rewards, changes in neural network architecture, reward scaling, and the entropy coefficient. The authors conclude that hyperparameters could dramatically change the way the agent learns, depending on the environment and task. Since the model-based optimization of hyperparameters is extremely time consuming, they tackle the problem by randomly searching for multiple sets. Hyperparameters including timesteps, batch size, epochs, clip range, gamma and lambda were randomly distributed with a limited number of possible values. Thus, the authors limited the search space in a number of possible combinations to make random searches feasible. The results of 60 random hyperparameters showed that PPO is susceptible to changes in hyperparameters, as the results vary widely. In this paper, we systematically look at the effects of various hyperparameters in more detail.

Although seemingly easier to learn, work on kick learning is rare so far. [9] presented a setup for a static kick and a kick in motion using DRL. The learned model for the static kick produced a failure rate of 1%, in which the robot did not

[1]https://openai.com/blog/openai-five-defeats-dota-2-world-champions

978-1-6654-3198-9/21/$31.00 ©2021 IEEE

hit the ball at all. 93% of all kicks had an average kicking distance of 8.53 meters. However, the robot was falling after these kicks almost always. The reward function did not contain benefit in keeping upright. Also, the robot did not have toes. In this work, a robot with toes is used that learns a kick that is stable. The kick in this work is further multi-directional and tolerant to a big range of relative ball positions.

III. Learning Domain

Learning is performed in the RoboCup 3D soccer simulation environment which is based on SimSpark[2] and initially initiated by [7]. It uses the ODE physics engine[3] and runs at a speed of 50Hz. Simspark provides variations of NAO robots with 22 DoF for the robot types without toes and 24 DoF for the type with toes (NAOToe henceforth) used in this work. More specifically, the robot has 7 (6) DoF in each leg, 4 in each arm and 2 in its neck.

The feet of NAOToe are modeled as rectangular body parts of size 8x12x2cm for the foot and 8x4x1cm for the toes (see Figure 1). The two body parts are connected with a hinge joint that can move from -1 degrees (downward) to 70 degrees.

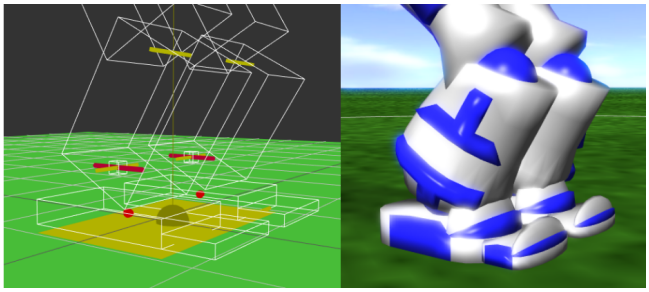

Fig. 1. Wire model of the NAO with toes (left) and how it is visualized (right).

All joints can move at an angular speed of at most 7.02 degrees per 20ms. The simulation server expects to get the desired speed at 50 Hz for each joint. If no speeds are sent to the server it will continue movement of the joint with the last speed received. Joint angles are noiselessly perceived at 50Hz, but with a delay of 40ms compared to sent actions. So only after two cycles the robot knows the result of a triggered action. A controller provided for each joint inside the server tries to achieve the requested speed, but is subject to maximum torque, maximum angular speed and maximum joint angles.

The simulator is able to run 22 simulated NAOs in real-time on reasonable CPUs. It is used as competition platform for the RoboCup 3D soccer simulation league[4]. For training, only a single robot is on the field.

Learning is performed using the OpenAI stable baselines implementation of PPO named PPO2[5]. Technically, an environment wrapper was created to send actions via sockets to the java client controlling the robot and receive the observations and rewards from the robot. The java client itself also uses sockets to send the motor commands to the simspark server (written in C++) and receive the sensor information.

[2]https://gitlab.com/robocup-sim/SimSpark
[3]http://www.ode.org/
[4]https://www.robocup.org/leagues/23
[5]https://github.com/hill-a/stable-baselines

TABLE I
Observation Space.

Index	Count	Observation
0	1	Counter
1-4	4	**head joints***
5-20	16	**arm joints***
21-48	28	**hip, knee, ankle, toe joints***
49-54	6	3D relative ball position*
55-102	48	**foot and toe force sensors***
103-108	6	**accelerometer***
109-114	6	**gyroscope***
115-116	2	torso up vector x,y
117	1	ball relative angle
118	1	desired kick direction (-90..90°, relative)
119	1	desired kick distance (0..20m)

IV. Approach

A. Observation Space

The observation space has been inspired by work of [1]. Table I shows the 120 entries of the observation space. Entries in bold font are raw sensor values and their derivatives (marked with * when applicable). The force sensors include the 3D point of force as well as the force vector itself resulting together with the derivatives in twelve values per sensor. In difference to [1], the usage of the NAO robot with toes not only adds two additional joint angle sensors, but also one force sensor per toe. Also, it turned out useful to add the torso's x and y components of the up vector, which are derived values, mainly from camera localization. The relative angle of the ball used is somewhat redundant to the relative ball position.

Inputs 118 and 119 are used to tell the network which direction and distance the kick is desired to achieve. It was fixed to 0 and 20 for the straight kicking experiments of section V-C. In section V-E these 'observations' were successfully used to learn kicks in a range of -30 to 30 degrees and for distances from 3 to 10 meters.

B. Action Space

For the kicking behavior, only the leg joints are part of the 28 entries action space (see Table II). For each joint, the action space contains two values: the destination angle to achieve, which is mapped to the possible values of each joint and the maximum angular speed to be used. Variants that only use the desired angle or only uses the angular speed produced worse results. Also, having angle and speed for each motor makes it possible to use the genetically learned kicks, that also use angle and speed, for pretraining the networks with expert behaviors.

C. Reward Function

Since kicking is a relatively short behavior (18 cycles), the reward function did not contain continuous reward, but only end reward. The final reward for the straight kicking

TABLE II
ACTION SPACE.

Index	Count	Joint	Orientation
0-5	6	left hip	YawPitch, Roll, Pitch
6-7	2	left knee	Pitch
8-11	4	left foot	Roll, Pitch
12-13	2	left toe	Pitch
14-27	14	right leg	YawPitch, Roll, Pitch

experiments included the sum of achieved kick distance in x-direction, the negative absolute deviation in y-direction and a negative penalty for falling: $reward = x - |y| - (1 - s/88)$, where s is the number of stable cycles after triggering the kick. To save time, an episode is stopped if the agent falls or if after 88 cycles (approx. 1.5s) after triggering the kick we can be sure that the agent is stable. In both cases, the achieved kick position is estimated as the 8s future ball position.

The final reward for the multi-directional kicking experiments is a mixture of the relative distance to the desired kick position and a penalty for falling: $reward = a_1 * (d_0 - d)/d_0 - a_2 * (1 - s/88)$, where d_0 is the desired kick distance and d the distance of the (estimated) ball end position to the desired kick position and a_1, a_2 are parameters balancing both penalties tuned to 100 and 25 respectively in earlier learning runs.

D. Training Setup

The setup of the learning plays an important role with respect to creating initial conditions that are similar to the conditions during a game. This usually increases learning time, but simplifies the transition from a learned to a successfully used behavior. The goal is that the robots use the learned kick during games when stepping slowly towards the ball or at the ball. Therefore, a training episode is designed as follows: the NAO robot is beamed randomly into a rectangle near the ball (see section V-D). After few initialization cycles, it steps in place for one second without trying to achieve a desirable position! The kick behavior with the right leg is then triggered when foot force sensors indicate just touching the ground (first force indication after at least five cycles without force). The robot then performs an 18 cycles deep kick behavior that is subject of learning. 18 cycles was found to be sufficient in preliminary experiments to kick the ball and stabilize after the kick. After the kick, the robot steps again in place for 1.4s to see if the robot was able to avoid a fall. This means that only 18 of the roughly 150 simspark simulation cycles of an episode are learning steps with respect to PPO.

E. Asynchronous Training

PPO2 allows to use multiple threads that collect data from parallel environments to train one model. To do so, it has to receive an observation from each environment per cycle. Since the actual kick is only performed in 18 of 150 cycles, it often happens that several threads have to wait for a few other environments to setup or wait for the result.

In this work, we designed an asynchronous mode to provide observations. The environment wrapper waits for k% of the environments to return an observation. For all other environments still busy performing preparation cycles, a dummy observation with zeros is returned to PPO and the corresponding action ignored. The learning process and quality do not decrease by this. This reduces the syncing problem between all environments and makes sure that threads are less likely to freeze and wait for some other thread to finish. The results are shown in Table III with a used threshold of a minimum of 70% valid observations per cycle. The table contains the average reward received after 48 hours of learning, the network updates performed in that time and completed episodes in 20 minutes. It can easily be seen that asynchronous training increases performance and made the following research much more feasible in a given amount of time.

TABLE III
SYNCHRONOUS VS. ASYNCHRONOUS.

	Sync	Async
Average Reward, 48h	6.834	8.967
Updates, 48h	166	1627
Episodes in 20min	5644	20782

F. Mirroring Behaviors

Kicking is an unsymmetric behavior. To reduce learning times, the robots learn a right leg kick that is then mirrored to its left leg. Mirroring a DRL-kick is considerably more difficult than mirroring a genetically learned kick. The later only requires to mirror the action space since they are typically open loop. For a DRL-kick, also the observation space has to be mirrored.

For the action space, right side joints have to be replaced with left side joints. Also, all roll joints and the arm yaw joints need to be mirrored. Mirroring means to negate the desired joint angle. This can either be done after the mapping to the co-domain of the network of [-1..1]. If done before, also the joint min and max values have to be mirrored, at least for the roll joints that have a non-symmetric co-domain (e.g. of -25 to 45 degrees).

For the observation space, all joints have to be mirrored as described above. Additionally, the 3D relative ball position needs to be mirrored at the y-axis, the foot force and its force origin have to be mirrored at the y-axis, the accelerometer at the y-axis, the gyroscope at the y- and z-axis, the torso roll angle and the ball relative angle also need to be mirrored. Finally, also the relative desired kick direction is mirrored.

V. RESULTS

In order to evaluate the described approach, extensive simulation runs of more than 7000 server hours (Intel Xeon E5-2630 v4 @ 2.2 GHz, 10 Cores) were conducted and are documented in this section. The first part concerns the DRL approach PPO, while the second part deals with the results of the domain itself.

A. Hyperparameter Evaluation

PPO defines a couple of hyperparameters that are evaluated in this subsection. The following experiments were started from one initial hyperparameter set and only the specified parameter was adjusted. In this evaluation, the interdependence of hyperparameters had to be ignored due to the enormous time consumption of a single run.

1) Learning Rate: The learning rate controls how quickly the neuronal net is adapting to a possible solution for its kicking task. The results for straight kick learning using six different learning rates are compared in Figure 2. Each run took 48 hours (t=1) of training. The final reward roughly maps to the achieved distance.

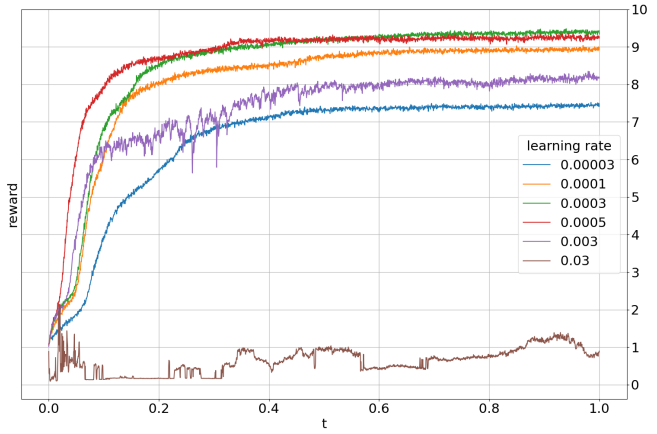

Fig. 2. Influence of the learning rate on straight kick learning.

The default value used is 0.0003 and it seems like learning rates close to this value show similar results. Even though, it is still visible that the slightly higher learning rate of 0.0005 is learning faster in the first couple of hours. The learning progress and final result only degrade using extremely high or low learning rates. It can also be seen that a very low learning rate like 0.00003 visibly slows down the actual learning progress and the neural network converges towards a significantly worse result.

2) Gamma: Gamma is a discount factor that makes sure that actions closer to the end of the episode get a higher share of the episode reward. Figure 3 shows the results of various values of gamma when learning a straight kick. As expected, small values of gamma show poor performance, while values above 0.99 show the best results. The difference between the three highest values 0.99, 0.995 and 1 is not significant.

3) Nsteps: Nsteps is the number of collected cycles which are considered for one network update. Figure 4 shows the results of several nsteps sizes and its result for learning the straight kick behavior. With a high nsteps size, more data needs to be collected in order to perform an update of the neural network. This can clearly be seen in the results. The bigger the nsteps size, the slower the learning progress since fewer updates are performed and more data is collected on each step. More data per update has a positive impact in terms of quality

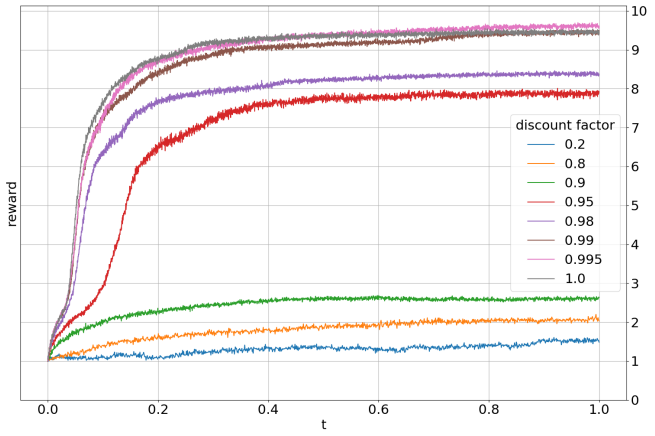

Fig. 3. Influence of the discount factor gamma on kick learning.

of the learned kick. The best results were achieved with an nsteps size of 2048 cycles per network update.

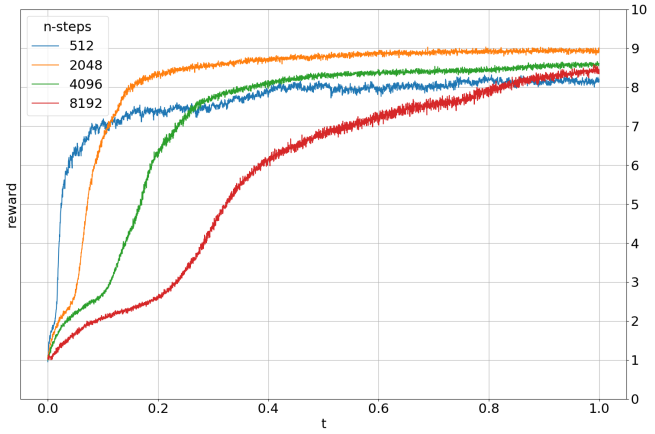

Fig. 4. Influence of the nsteps size on kick learning.

4) Minibatch Size: The minibatch size defines the amount of minibatches taken for each update for the minibatch gradient descent. In Stable Baselines, the minibatch size reflects the amount of batches into which the collected dataset is divided. A minibatch size of 32 with an nsteps size of 2048 corresponds to 32 minibatches with a size of 64 for the minibatch gradient descent. The results in Figure 5 show the smaller the minibatchsize, the better the quality of the resulting neural network. In addition to that, the amount of performed episodes are significantly higher in comparison with higher minibatch sizes after a 48 hours of learning.

B. Network Size

PPO2, by default, uses two fully connected hidden layers with 64 neurons each in the value and in the policy network. Figure 6 shows learning results for various network sizes. Surprisingly, a network with only four neurons in both hidden layers and both networks is still able to learn a reasonably well kick though with a significantly lower reward of 7. The movement looks rougher like gross motor skills, but still

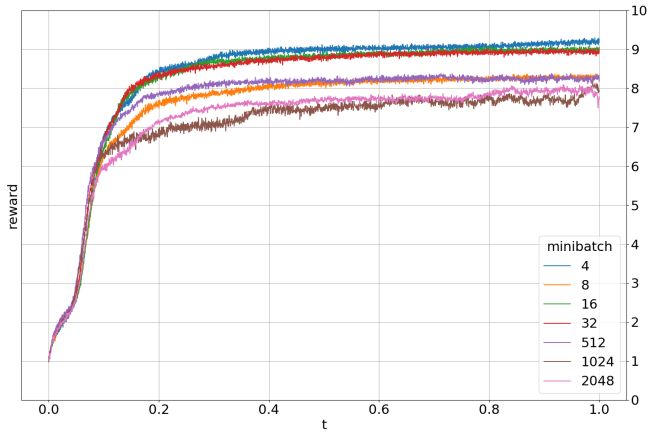

Fig. 5. Influence of the minibatch size on kick learning.

clearly recognizable as a kick. For eight neurons the result is slightly better, but still worse compared to bigger networks. With already 16 neurons in each layer, the kick behavior achieved comparable results to the default value of 64.

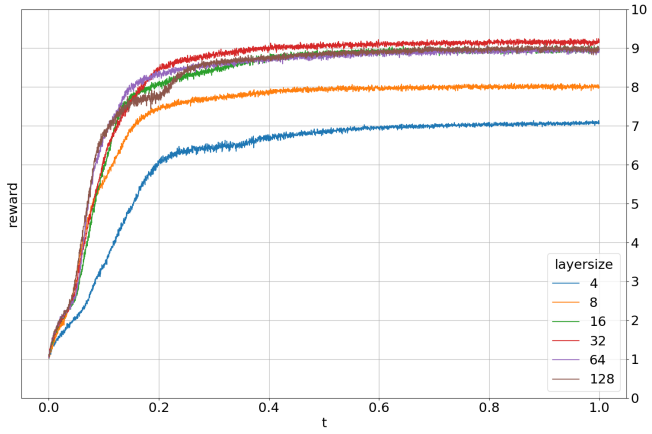

Fig. 6. Reward of kicks with different hidden layer sizes.

C. Straight Kick Distance and Precision

Figure 7 shows a comparison of the final position of 100 kicks from starting coordinate (0,0). The PPO kick is slightly longer on average and has considerably less spread in x and y direction. Also, the number of failed kicks is significantly lower. In a series of 1000 kicks, the achieved distance of the PPO kick was only in three cases less than 5m.

D. Kickable Area

The usage of a DRL policy network for controlling the agent's behavior has a fundamental advantage over behaviors learned with genetic learning: they are closed-loop. The action is calculated based on the current observation while a genetically learned behavior in joint space is simply replayed no matter what the current situation is.

The DRL-Kick makes use of this, for example, with adjusted movements depending on the relative position of the

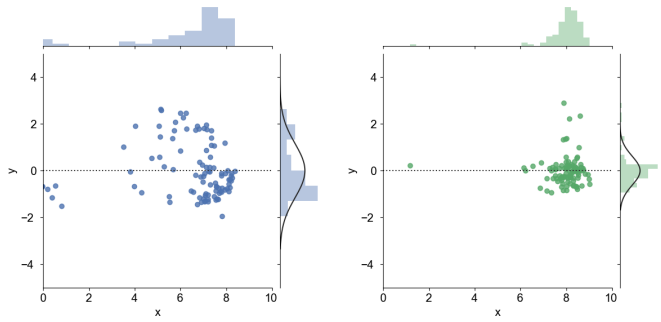

Fig. 7. Comparison of ball end positions of a genetically learned kick (left) and a PPO learned kick (right).

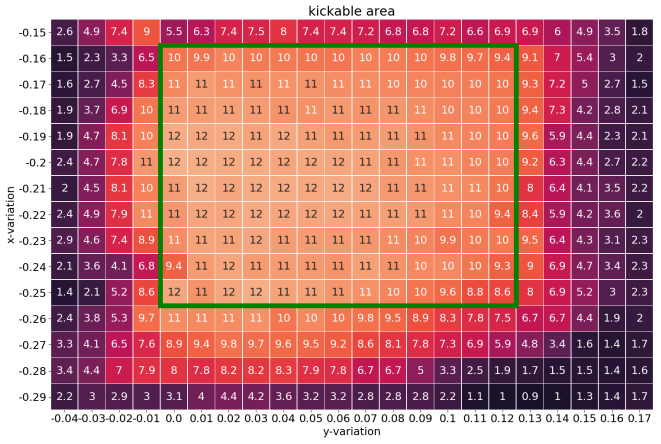

Fig. 8. The kickable area of the PPO kick. The green rectangle shows where the robot was beamed to during learning.

ball, which is part of the observation space. In order to show this, the initial position of the player relative to the ball during learning was randomly set within a rectangle of 10x13cm with respect to the ball (at position (0,0)) (see Figure 8). The vertical axis represents the position of the robot with respect to the ball in the frontal plane. A value of less than 0.15m causes the robot to touch the ball already while stepping in place. The horizontal axis represents the player position with respect to the sagittal plane.

Each position in Figure 8 is the average reward of 50 kicks from the corresponding position relative to the ball. As can be seen, the robot learned a successful kick in the whole area and slightly outside it. The huge difference in size of the kickable areas of the PPO and the genetic kick is shown in Figure 9. The genetic kick used as comparison uses a similar action space with joint angles and maximum speeds for each joint, but limited to four keyframes with learnable duration (for details see [3].

E. Multi-Directional Kick

The extreme flexibility of DRL is shown in this experiment. In addition to beaming the player into various positions relative to the ball, now the two steering inputs of desired distance and direction are used to learn a multi-directional kick.

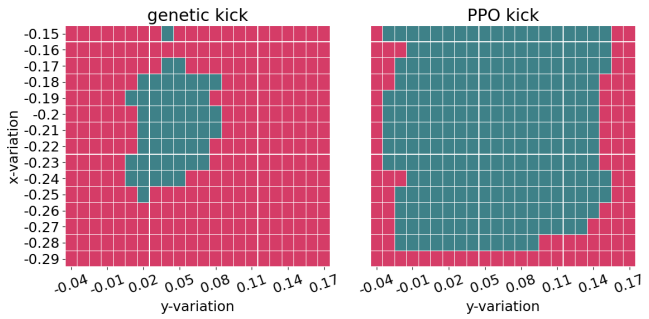

Fig. 9. Comparison of the kickable area.

In a first experiment, the network learned during straight kicking was used as a starting point for the NAOToe. During learning, the desired direction was randomly selected for each kick in a range from -45 to 45 degrees. The desired kick distance was randomly selected between 3 and 10 meters.

Figure 10 shows the results of roughly one week of kick learning. Each rectangle is an average of 100 kicks with a corresponding fixed desired direction and distance. As can be seen, the robot is able to learn to kick into a considerable range of directions and distances despite the big variance of the relative ball position. Two videos that demonstrate the kick are available here[6] and here[7].

	-45	-40	-35	-30	-25	-20	-15	-10	-5	0	5	10	15	20	25	30	35	40	45
3	54,9	62,9	66,8	72,6	72,5	72,3	75,0	74,6	74,2	75,7	68,4	70,5	73,1	69,4	63,2	60,9	56,5	50,2	48,1
3,5	62,7	66,2	69,7	77,9	77,2	77,3	78,7	80,7	80,2	81,0	74,2	76,8	77,8	73,2	75,2	72,1	66,8	63,0	59,1
4	62,0	70,5	74,4	77,4	82,8	86,2	86,9	84,3	85,6	82,8	81,4	82,2	77,4	77,7	75,9	75,0	69,7	71,6	64,8
4,5	64,2	70,0	74,6	80,8	84,0	85,7	82,8	87,5	86,9	87,2	86,9	83,5	82,4	79,2	78,1	76,1	75,0	68,1	64,5
5	60,9	71,3	76,2	83,5	85,7	83,4	87,4	87,3	86,4	86,1	85,3	84,7	83,3	79,6	75,3	75,2	72,9	70,7	66,6
5,5	63,6	70,5	79,0	83,1	86,7	83,5	84,5	87,7	87,6	86,6	85,0	84,4	82,7	79,8	78,8	74,2	73,0	68,9	65,9
6	64,8	73,2	75,1	82,3	82,6	85,2	87,1	86,8	88,9	87,7	86,2	83,4	81,4	75,8	77,7	81,3	70,3	69,3	67,0
6,5	63,4	70,5	78,5	83,1	84,8	82,4	88,1	89,1	88,0	86,5	81,9	82,7	80,1	78,3	78,2	73,7	76,2	70,1	64,1
7	67,9	73,5	80,5	85,0	86,2	85,8	85,5	86,6	86,4	84,7	82,8	82,7	80,5	80,9	78,1	78,1	72,8	66,6	63,9
7,5	65,6	71,8	81,6	86,1	86,7	88,2	88,6	89,4	86,2	86,3	83,0	82,8	81,5	80,9	76,6	68,6	74,0	69,2	64,5
8	68,4	72,2	81,0	86,3	86,9	88,6	90,5	90,6	88,7	87,9	85,8	83,5	80,9	77,9	75,4	69,8	67,8	65,9	58,1
8,5	64,0	74,8	77,2	85,8	87,1	88,0	88,2	87,4	86,4	88,6	85,2	83,5	80,5	77,0	79,5	74,0	69,9	62,3	57,8
9	63,9	71,1	78,1	84,5	88,2	87,7	85,7	86,0	86,8	85,7	84,8	83,8	78,4	76,8	73,5	68,4	64,8	61,1	56,1
9,5	64,0	69,5	77,1	84,0	83,6	85,2	85,3	84,7	83,9	84,4	83,6	81,8	79,5	72,9	72,9	65,1	59,0	56,7	53,1
10	62,2	67,2	74,8	78,0	81,7	82,0	80,6	81,8	80,2	80,0	78,1	80,2	77,2	72,4	68,6	68,3	62,4	55,6	54,4

Fig. 10. Reward with respect to desired direction and distance of NAOToe.

The same procedure was performed in a second experiment with a NAO robot without toes. Figure 11 shows the difference of the toed robot to the robot without toes for each direction and distance. As can be seen, the robot with toes has a considerable advantage on longer kick distances while the robot without toes is more precise on shorter kicks. For game playing, longer kicks are of higher value though.

Figure 12 shows typical toe movements over time. The first graph shows the toes of the kicking and support foot for a straight kick, the second for a kick 30 degrees to the left, the third for a kick 30 degrees to the right. The ball is hit in cycle 6 (dashed line). It should be noted that the movement for even each straight kick varies. This is true even if the randomness of different start positions is eliminated. The reason is, that there is still non-determinism from the simulator and the stepping in

[6]https://youtu.be/sHlkRaljtjY
[7]https://youtu.be/F82hqicRYZQ

	-45	-40	-35	-30	-25	-20	-15	-10	-5	0	5	10	15	20	25	30	35	40	45
3	-20,0	-10,4	-5,6	-0,2	0,7	-4,4	-0,7	-3,2	5,8	-0,6	-10,9	-7,7	-1,1	2,2	-6,4	-5,0	-14,4	-10,0	-10,4
3,5	-5,9	-6,1	-7,4	-0,4	1,0	-3,4	-1,3	1,3	8,2	4,2	-4,6	-4,2	-2,9	-2,5	0,9	-5,7	-7,0	-6,1	-5,5
4	-6,8	-2,8	-1,3	-3,8	2,3	6,6	7,9	0,5	3,7	-1,2	-1,3	1,5	0,9	-0,5	-2,1	-1,4	-3,0	-4,3	-1,1
4,5	-0,3	3,2	2,6	3,9	3,5	4,0	0,6	6,6	5,2	3,1	5,1	6,5	3,9	5,9	4,1	8,0	2,5	1,8	1,6
5	-2,5	4,6	1,4	5,9	3,7	1,0	4,2	3,1	6,1	0,5	3,6	4,2	4,9	4,7	0,9	6,2	0,1	0,7	1,9
5,5	3,0	4,5	4,7	5,2	2,9	1,4	-2,9	4,1	2,0	-0,8	0,1	3,5	6,4	1,2	5,9	-0,9	-0,7	-2,3	3,4
6	5,8	7,7	3,4	1,5	-1,5	-0,3	2,1	-0,6	2,6	-0,6	1,1	-0,9	2,3	-1,4	1,8	2,5	-0,3	1,0	7,7
6,5	6,8	7,2	3,2	3,4	-1,9	-0,5	3,0	6,0	1,4	-1,9	-1,1	-0,2	-2,1	-2,3	11,9	2,1	14,3	8,6	3,8
7	10,3	5,2	8,7	6,5	-2,4	0,0	-1,9	-1,1	3,2	-3,0	-3,7	2,1	0,7	3,0	3,5	6,2	5,2	12,8	9,4
7,5	10,0	4,4	10,5	6,5	4,3	2,7	7,3	1,6	-1,1	-1,9	0,5	3,1	4,3	4,7	7,4	-1,7	7,4	13,7	11,9
8	13,1	11,0	9,6	8,8	6,5	4,3	10,6	7,8	4,4	5,3	3,7	5,5	8,0	3,0	5,4	7,6	10,7	9,1	7,9
8,5	9,8	12,8	10,2	11,4	9,1	10,6	12,3	10,0	6,9	7,9	7,1	7,5	8,0	5,5	15,1	11,4	11,2	9,4	8,6
9	10,0	10,9	11,3	14,2	14,4	20,0	12,5	9,6	9,8	7,4	7,8	9,0	7,6	10,7	9,5	9,0	8,0	8,6	6,7
9,5	8,9	16,5	15,5	14,6	11,8	18,0	10,9	12,4	9,4	11,0	15,2	12,8	11,7	9,7	11,3	6,5	3,6	6,0	7,2
10	10,3	12,2	12,3	15,3	16,8	9,6	8,4	12,2	9,9	14,4	11,7	14,3	12,7	8,3	10,3	10,6	9,4	6,8	14,1

Fig. 11. Relative kick success of NAOToe compared to a NAO without toes.

place before kick and the policy network adjusts the movement to each specific situation.

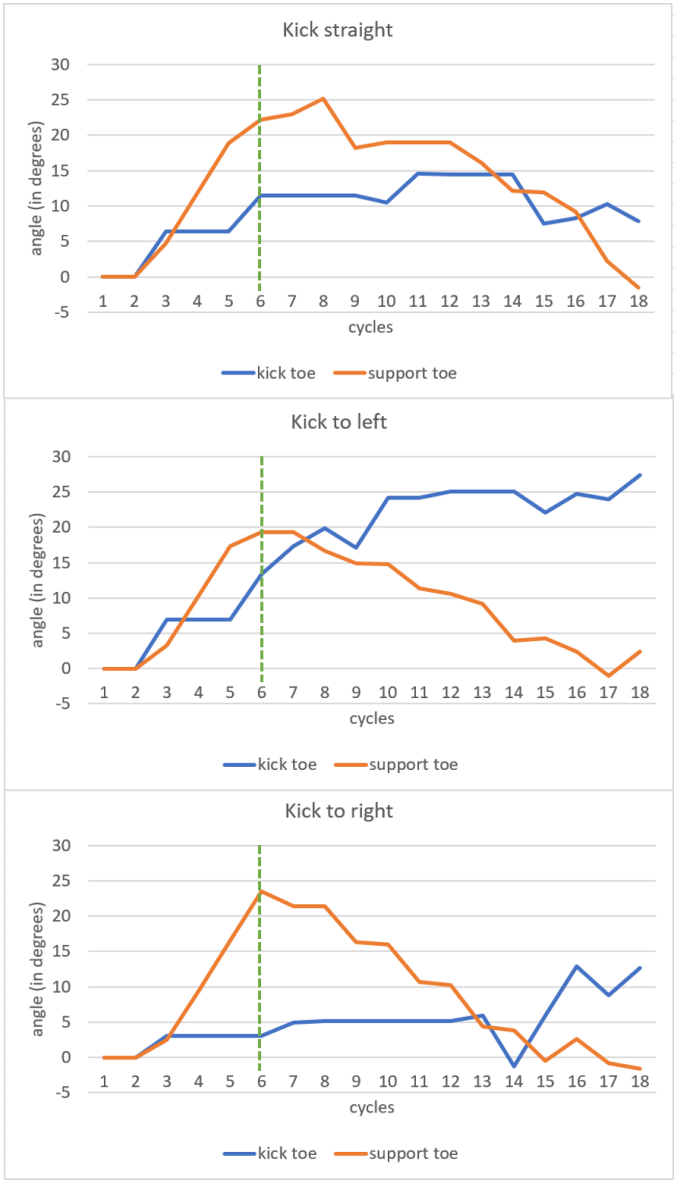

Fig. 12. Movements of the toe joints for a straight kick (1), a kick 30 degrees to the left (2) and 30 degrees to the right (3).

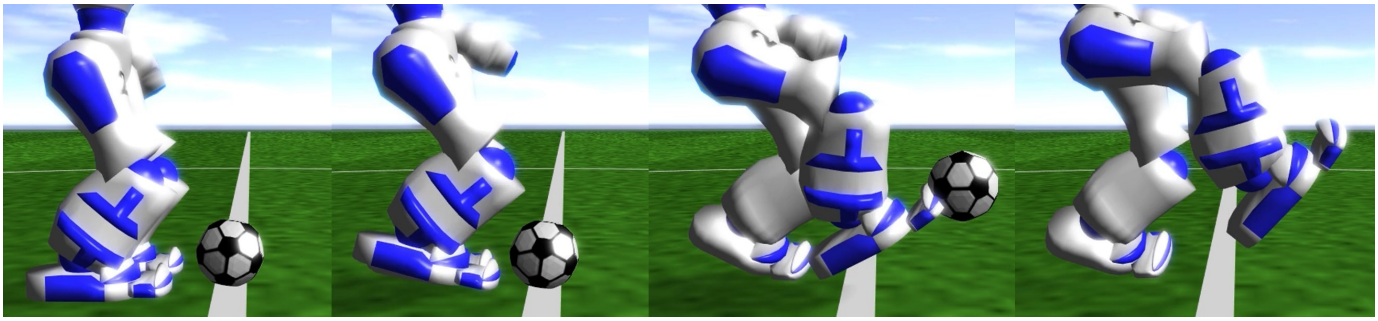

Fig. 13. Visualization of the kick phases of a straight kick.

As can be seen in Figure 13, the robot has learned to step on its support toe to improve the kicking geometry and keep the stability of its stand. In 93 of 100 kicks the NAOToe did not fall during or after the kick. This is only slightly less stable than the NAO (94 out of 100), but more effective for longer kicks (see Figure 11).

F. Performance in Games

The setup of the kick learning has been chosen in a way to simplify the introduction of that behavior into real games. In fact, the PPO kick behavior can replace the genetical kick by changing a single line of code, that adds the kick to the list of available kicks instead of its predecessor. The performance of the new straight kick was tested in a series of 200 games of two identical teams of eleven robots with the only difference that one team was using the PPO straight kick while the other used the old genetic kick instead. The PPO team scored 0.665 goals on average which is significantly more than the 0.385 goals for the comparison team. Of the 200 games, the PPO team won 83 games, tied in 76 and lost 41.

In another series of 200 games, seven of the eleven players of one team had toes and used the multi-directional kick (seven is the highest number of identical robot types allowed). The other team was identical except that the toed robots used the PPO straight kick learned. The multi-directional team scored 0.695 goals on average compared to 0.460 goals of the team without multi-directional kick (signif.). Of the 200 games, the multi-directional team won 79 games, tied in 72 and lost 49.

VI. Conclusion and Future Work

In this paper we have shown that PPO can be used to learn a multi-directional kick with a toed NAO robot that is highly flexible with respect to the relative ball position. The deep neural networks are able to learn situation dependent movements of the robots based on, among other, desired kick direction, kick distance and ball position. This demonstrates the main advantage of behaviors learned through DRL over pre-programmed or genetically learned behaviors.

The approach to setup a learning situation that is as close as possible to real game situations was made feasible with an asynchronous version of PPO2 that allows to define the number of instances to wait for. Finally, the multi-directional kick shows highly significant improvement of game play.

Future work includes learning a kick from full speed running for which initial results are very promising. Also, one problem to address is certainly the long learning times. Tests with pretraining of networks did not show promising improvement on learning time or quality. Other approaches could be to use off-policy learning algorithms. They promise to be more sample efficient and most of the learning time is due to collecting samples in the simulation domain.

Acknowledgment

We thank the magmaOffenburg team and particularly Jens Fischer for the implementation of the stable baselines environment wrapper used in this work and the FC Portugal team for very valuable discussions on this topic.

References

[1] Abreu M, Lau N, Sousa A and Reis L P (2019). Learning low level skills from scratch for humanoid robot soccer using deep reinforcement learning. In Proceedings of 2019 International Conference on Autonomous Robot Systems and Competitions (ICARSC). IEEE.

[2] Abreu M, Reis L P, Lau N (2019). Learning to Run Faster in a Humanoid Robot Soccer Environment Through Reinforcement Learning. In RoboCup 2019: Robot World Cup XXIII (pp. 3–15). Springer International Publishing.

[3] Dorer K (2017). Learning to Use Toes in a Humanoid Robot. In Akiyama H, Obst O, Sammut C, Tonidandel F: RoboCup 2017: Robot World Cup XXI, Nagoya, Japan, Springer Verlag.

[4] Melo L and Maximo M (2019). Learning Humanoid Robot Running Skills through Proximal Policy Optimization. arXiv:1910.10620.

[5] MacAlpine P (2017). Multilayered Skill Learning and Movement Coordination for Autonomous Robotic Agents. Ph.D. Thesis, The University of Texas at Austin, Austin, Texas, USA.

[6] Mnih V, Badia A, Mirza M, Graves A, Lillicrap T, Harley T, Silver D and Kavukcuoglu K (2016). Asynchronous Methods for Deep Reinforcement Learning. arXiv:1602.01783.

[7] Obst O and Rollmann M (2005). SPARK - A Generic Simulator for Physical Multiagent Simulations. Computer Systems Science and Engineering, 20(5).

[8] Peters J, Kober J, Mülling K, Nguyen-Tuong D and Kroemer O (2012). Robot skill learning. In 20th European Conference on Artificial Intelligence (ECAI 2012), pages 1–6.

[9] Teixeira H, Silva T, Abreu M and Reis L P (2020). Humanoid Robot Kick in Motion Ability for Playing Robotic Soccer. 2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), Ponta Delgada, Portugal, 2020, pp. 34-39, doi: 10.1109/ICARSC49921.2020.9096073