An Accelerated Approach to Decentralized Reinforcement Learning of the Ball-Dribbling Behavior

D. Leonardo Leottau and **Javier Ruiz-del-Solar**
Advanced Mining Technology Center - Department of Electrical Engineering, Universidad de Chile
{dleottau, jruizd}@ing.uchile.cl

Abstract

In the context of soccer robotics, ball dribbling is a complex behavior where a robot player attempts to maneuver the ball in a very controlled way, while moving towards a desired target. To learn when and how to modify the robot's velocity vector is a complex problem, hardly solvable in an effective way with methods based on identification of the system dynamics and/or kinematics and mathematical models. We propose a decentralized reinforcement learning strategy, where each component of the omnidirectional biped walk (v_x, v_y, v_θ) is learned in parallel with single-agents working in a multi-agent task. Moreover, we propose an approach to accelerate the decentralized learning based on knowledge transfer from simple linear controllers. Obtained results are successful; with less human effort, and less required designer knowledge, the decentralized reinforcement learning scheme shows better performances than the current dribbling engine used by *UChile Robotics Team* in the SPL robot soccer competitions. The proposed decentralized reinforcement learning scheme achieves asymptotic performance after 1500 episodes and can be accelerated up to 70% by using our approach to share actions.

1. Introduction

In the context of soccer robotics, ball dribbling is a complex behavior where a robot player attempts to maneuver the ball in a very controlled way, while moving towards a desired target. In case of humanoid biped robots, the complexity of this task is very high, because it must take into account the physical interaction between the ball, the robot's feet, and the ground, which is highly dynamic, nonlinear, and influenced by several sources of uncertainty.

Very few works have addressed the ball dribbling behavior with humanoid biped robots (Macalpine et al. 2012; Alcaraz et al. 2011; Meriçli et al. 2011; Latzke et al. 2007; Leottau et al. 2014). Besides, not many details about the specific dribbling modeling (Tilgner et al. 2013; Röfer et al. 2014), or performance evaluations for the ball-control or accuracy to the desired path are mentioned.

The main goal of this paper is to apply and evaluate a decentralized Reinforcement Learning (RL) scheme (Busoniu et al. 2006) as a first approach to Multi-Agent Systems (MAS) (Stone & Veloso 2000) applied to *inwalk-ball-pushing* based behaviors (Leottau et al. 2014) like the ball-dribbling in the context of the biped soccer robotics. In this way, this paper proposes to address the ball-dribbling behavior based on a Decentralized-RL (D-RL) strategy (Busoniu et al. 2006), where each component of the omnidirectional biped walk (v_x, v_y, v_θ) is learned in parallel as a single agent working in a multi-agent task (Tuyls et al. 2005). In addition, the *nearby action sharing* (NASh) approach is proposed to accelerate the decentralized learning, where knowledge from simple linear controllers is transferred to the D-RL agent.

Performance indices for dribbling speed and ball control are measured for the proposed D-RL solution and different configurations of the accelerated approach. These are compared with the *RL + Fuzzy-Logic-Control* (FLC) method reported in (Leottau et al. 2014), which is the current dribbling engine used by *UChile Robotics Team* (Yañez et al. 2013) in the SPL robot soccer competitions.

Experiments show that D-RL schemes are able to outperform the RL-FLC approach with less human effort and less required designer knowledge. Likewise, a decentralized agent, learning from scratch, achieves asymptotic performance after 1500 episodes, which can be accelerated up to a 70% by applying the proposed NASh approach.

2. Related Work

Although several strategies can be used to tackle the ball-dribbling problem in the context of the biped soccer robotics, we classify these in three main groups:

- *Based on human experience and/or hand-code:* (Latzke et al. 2007) presents an approach that uses imitative reinforcement learning for dribbling the ball from different positions into the empty goal, meanwhile (Meriçli et al. 2011) proposes an approach that uses corrective human demonstration for augmenting a hand-coded ball dribbling task performed against stationary defender robots.
- *Based on identification of the system dynamics and/or kinematics-mathematical models*: (Alcaraz et al. 2011) presents an approach to incorporate the ball dribbling as part of a closed loop gait, combining a footstep and foot trajectory planners for integrating kicks in the walking engine. Since this work is more focused to the theoreti-

cal models and controllers of the gait, there is not included a dribbling engine final performance evaluation.

- *Based on the on-line learning of the system dynamics*: (Macalpine et al. 2012) presents the evolutionary parameter learning of a biped omnidirectional walk for some common soccer subtasks such as the drive-ball to goal in the RoboCup 3D simulation environment. The distance the ball travels toward the goal during 30 seconds is the fitness function. This is the most related work regarding the proposed method in this paper. However, our approach also considers the ball control in the learning modeling and performance evaluation.

The dribbling problem has been addressed more extensively for the wheeled robots case (Carvalho & Oliveira 2011; Riedmiller et al. 2008; Riedmiller et al. 2009), however, these approaches are not directly applicable to the humanoid case, due to the much higher complexity of the biped case. Furthermore, oppositely to our decentralized proposal, these works address centralized RL schemes where the joint action state is composed by discretized combinations of each component of the velocity vector.

To the best of our knowledge, no publications related to distributed or D-RL agents working in a multi-agent task to learn individual behaviors such as the soccer ball-dribbling or other similar task have been reported. Some distributed control applications particularly applied to robot manipulators are (Busoniu et al. 2006; Martin & Lope 2007).

3. Proposed modelling for the ball-dribbling

The description of the dribbling behaviors will use the following variables: v_x, v_y, v_θ, the robot's linear and angular speeds; α, the robot-target angle; γ, the robot-ball angle; ρ, the robot-ball distance; ψ, the ball-target distance; β, the robot-target-ball angle; and, φ, the robot-ball-target complementary angle. These variables are shown in Figure 1, where the desired target ($\oplus$) is located in the middle of the opponent goal, and a robot's egocentric reference system is considered whit the *x* axis pointing forwards.

The ball-dribbling behavior can be splitted in two task which have to be executed in parallel: *the alignment* that keeps the robot aligned to the ball-target line ($\varphi = 0, \gamma = 0$) while approaching the ball; and *the inwalk-ball-pushing*, whose objective is that the robot walks as fast as possible and hits the ball in order to change its speed, but without losing its possession.

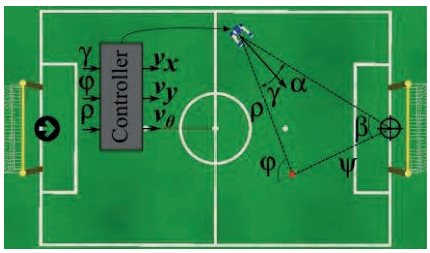

Figure 1: Variables definition for the dribbling modeling.

Reinforcement Learning of the ball-dribbling

Why a RL based controller? −In the *inwalk-ball-pushing* task of the dribbling behavior, the ball must be kept near ($\rho = 0 \wedge \gamma = 0$) while it is hit towards the desired target ($\varphi = 0 \wedge \psi_{k+1} < \psi_k$). The modeling of the robot's feet–ball–floor dynamics is complex and inaccurate because kicking the ball could generate several unexpected transitions, due to uncertainty on the foot's shape and speed when it kicks the ball (note that the foot's speed is different to the robot's speed v_x). Moreover, an omnidirectional biped walk intrinsically has a delayed response, which varies depending on the requested velocity (v_x, v_y, v_θ). To learn when and how much the robot has to slow down or accelerate is a complex problem, hardly solvable in an effective way with methods based on identification of the system dynamics and/or kinematics and mathematical models (Alcaraz et al. 2011; Li et al. 2007; Zell 2008). Thus, to solve that as a Markov Decision Process (MDP) with a RL scheme for learning simultaneously the ball-dribbling dynamics is a promising approach.

Decentralized Reinforcement Learning

The D-RL emerges as a Distributed Artificial Intelligence technique where systems with several branches working together towards a common goal (Stone & Veloso 2000). If resources and information of those branches are managed separately in a behavioral way, and, under a joint environment, it could be considered as a MAS.

Since this work is a first approach to MAS applied to the RL of *inwalk-ball-pushing* based behaviors, a simple scheme with single-agents without a coordination mechanism is considered. According to (Tuyls et al. 2005), single-agents working in a multi-agent task are able to converge to a coordinate equilibrium under certain parameter and for some particular behaviors.

Why a D-RL scheme? − The MAS perspective gives several potential advantages if the problem is approached with decentralized learners and the coordination problem is solved: learning speed might be higher with respect to a centralized agent because this searches into an exponentially larger action space; the state space could be reduced if not all the state information is relevant to all the learning agents; memory and processing time requirements will also be smaller because parallel computation (Busoniu et al. 2006).

Decentralized RL modelling

The proposed control actions for the dribbling behavior are the requested speed to each axis of the biped walking engine (v_x, v_y, v_θ). A D-RL scheme (Busoniu et al. 2006; Martin & Lope 2007) can be considered in order to learn in parallel each component of that velocity vector as a MAS.

Our expected policy is walking fast towards the desired target while keeping the ball possession. That means: to minimize γ, φ; to maintain $\rho < \rho_{max}$; to maximize v_x; and, to minimize v_y and v_θ. So, the proposed modelling for learning the velocity vector (v_x, v_y, v_θ), depending on the

observed state (ρ, γ, φ) is detailed in Table 1, where $\rho_{th}, \gamma_{th}, \varphi_{th}$ are desired thresholds where the ball is considered controlled while it is reinforced to walk forward at maximum speed.

Table 1: States, actions, and reward function description for the three RL agents

Common states space: $s = [\rho, \gamma, \varphi]^T$				
		Min	*Max*	*# Bins*
$Feature_1$	ρ	0mm	600mm	13
$Feature_2$	γ	-50°	50°	11
$Feature_3$	φ	-50°	50°	11
Actions space: $a = [v_x, v_y, v_\theta]$				
		Min (0%)	*Max (100%)*	*# Actions*
$Agent_x$	v_x	0 mm/s	150 mm/s	21
$Agent_y$	v_y	-50 mm/s	50 mm/s	21
$Agent_\theta$	v_θ	-45 °/s	45 °/s	21
Reward function: $r(s,a) = [r_x, r_y, r_\theta]$				
		$r = [1,1,1]$		$r = -[1,1,1]$
Constraint	r_x	$if\ \rho < \rho_{th} \wedge \lvert\gamma\rvert < \gamma_{th} \wedge \lvert\varphi\rvert < \varphi_{th} \wedge v_x \geq v_{x.max}$,		otherwise
Constraint	r_y	$if\ \lvert\gamma\rvert < 5°$		otherwise
Constraint	r_θ	$if\ \lvert\gamma\rvert < 5° \wedge \lvert\varphi\rvert < 5°$		otherwise

4. Knowledge Transfer

Since the proposed D-RL scheme does not include direct coordination mechanisms, joint actions, or sharing information between agents, knowledge transfer is considered to guide the learning towards a partially coordinated policy in the early episodes in order to accelerate the learning process. According to (Taylor & Stone 2009), transfer is one possible approach to making such problems more tractable, as an alternative, decentralized agents could start the learning process over a subset of actions, and then increase or modify progressively the action space over time.

Two constraints are considered to select the knowledge transfer strategy: methods for transferring on any RL algorithm that uses an action value function (Taylor & Stone 2007; Mataric 1994; Knox & Stone 2012), and methods for transferring from different source types, as controllers, hand coded behaviors, RL policies, among others (Bianchi et al. 2012; Fernández et al. 2010). Thus, the *control sharing* method (Knox & Stone 2012), which accomplishes both constraints, is used here. In addition, a similar, but original method, called *nearby action sharing* (NASh), is introduced.

Control Sharing Approach

The *control sharing* (CoSh) method, originally proposed in (Knox & Stone 2010), acts only during action-selection, without affecting the Action-Value functions:
$P(a = a_{src}) = min(\beta, 1)$. *Otherwise use base RL agent's action selection mechanism.* The action a is chosen by *source-policy* ($\pi_{src}(s) = a_{src}$) with probability β, where β is annealed periodically by a predefined factor.

Nearby Action Sharing Approach

This method is introduced for transferring knowledge from continuous action spaces, when no information different to the suggested action in an observed state is available from the source of knowledge. The method has applicability in cases where the source of knowledge are controllers, hand coded behaviors, rule inference system, among others. The NASh method takes advantage of continuous actions spaces to compensate the lack of information about the quality of the actions, like an action-value function. It assumes that a measure of the quality of an state-action pair from source is related with its distance to the action suggested by the source (a_{src}). In this way a normal distribution along the universe of discourse centered in a_{src}could be considered, and the resulting nearby action to transfer is $a'_{src} = \xi(a_{src} = \pi_{src}(s), \varrho \cdot (1-\beta))$, where $\xi(\mu, \sigma)$ is a normally distributed random generator with mean $\mu = a_{src}$, and standard deviation $\sigma = \varrho \cdot (1-\beta)$, where ϱ is a scale factor for the continuous action space. Then, the transfer mechanism is similar to the CoSh method:
$P(a = a'_{src}) = min(\beta, 1)$. *Otherwise use base RL agent's action selection mechanism.*

As in the CoSh case, the action is chosen by *source-policy* with probability β; β is annealed periodically as well as the standard deviation of ξ. That means at the beginning of the learning process, NASh works similarly to CoSh, however, along the learning process, the probability of choosing an action a'_{src}, increasingly away from the action a_{src} increases with $\varrho \cdot (1-\beta)$.

Practical implementations of the NASh method can be carried out similarly to the *softmax* action selection (Sutton & Barto 1998), e.g., for N discrete actions, $Q(a_n) = \eta(a_n)$, $a_n = \{a_1, \dots a_N\}$, where $\eta(\mu, \sigma)$ is a normally distributed function with $\mu = a_{src}$, and $\sigma = \varrho \cdot (1-\beta)$. Other simpler alternative is directly gets a'_{src} bounding it into the action space with module or clip functions.

5. Experimental setup

The ball-dribbling RL procedure is carried out episodically. After a reset, the robot is set in the center of the own goal (black right arrow in Figure 1), the ball is placed ρ_{th} millimeters in front of the robot, and the desired target is in the center of the opponent goal ($\oplus$). The terminal state is reached if the robot loses the ball, or, the robot leaves de field, or, the robot crosses the goal line and reaches the target, which is the expected terminal state. The trained field has 6x4 meters. Fault-State constraints are set as:
$[\rho_{th}, \gamma_{th}, \varphi_{th}, v_{x.max}] = [250mm, 15°, 15°, 0.9 \cdot v_{x.max}]$.

The current scope of this paper is to evaluate the effectiveness and potential usefulness of knowledge transfer for D-RL agents as first approach to the MAS applied to *in-walk-ball-pushing* based behaviors. Thus, all the presented experiments are carried out in simulation.

The SARSA(λ) algorithm

The implemented RL algorithm is the SARSA(λ) with replacing traces (Sutton & Barto 1998) and a Radial Basis Functions (RBF) scheme. Based on previous work and after several trials, the SARSA(λ) parameters have been chosen prioritizing fastest convergences. In this way, the following parameters are chosen: learning rate α=0.5, discount factor γ=0.995, eligibility traces decay λ=0.9, and epsilon greedy action selection with an exponential decay along the trained episodes:
$\varepsilon = \varepsilon_0 \cdot exp(-episode \cdot \kappa / maxEpisodes)$, with ε_0=1, κ=15 constants, and $episode$ the current episode index and $maxEpisodes$=*2000* the trained episodes per run.

The same learning parameters are used for the three decentralized agents under evaluation in all the experiments. For the CoSh transfer and NASh experiments, $\beta = \varepsilon$.

Experiments

Four different experiments are presented in this paper. Methods considered for testing and comparing during each of these experiments are listed below:

i) **Decentralized Learner (DL):** the D-RL proposed scheme detailed in Table 1. Three initializations of the Q function are considered, pessimistic (*Qi=-5*), zeros (*Qi=0*), and optimistic (*Qi=5*). These initialization values are the best after several simulation trials with different values. So, three DL are considered for this experiment: *DL pessimistic*, *DL zeros,* and *DL optimistic.*

ii) **DL accelerated with CoSh:** the three initialization values considered for DLs are also tested. So, three CoSh DLs are considered: *CoSh pessimistic*, *CoSh zeros,* and *CoSh optimistic.* A hand tuned linear controller like the proposed in (Leottau et al. 2014) is used as source of knowledge.

iii) **DL accelerated whit NASh:** as in CoSh experiments, three DLs are considered: *NASh pessimistic, NASh zeros,* and *NASh optimistic.* The same source of knowledge is used for CoSh and NASh.

iv) **CoSh and NASh with different sources of knowledge:** in order to evaluate the impact of the source policy's quality in the learning process, two extra sources of knowledge are tested, both linear controllers with the same scheme considered for *(ii)*: *FastSrc* that is tuned to prioritize the dribbling speed (i.e., shortest t_{DP}, despite highest $\%t_{FS}$); and *SlowSrc*, which prioritizes the ball-control (i.e., smallest $\%t_{FS}$, despite longest t_{DP}). Thus, four additional learners are considered for this experiment: the best CoSh from experiment *(ii)* whit *FastSrc* and *SlowSrc,* and the best NASh from experiment *(iii)* whit *FastSrc* and *SlowSrc.*

Performance indices

The evolution of the learning process of each scheme mentioned in the previous section is evaluated by measuring and averaging ten runs. In this way, the following performance indices are considered:

- The *Dribbling-Path Time* (t_{DP}): how long takes the agent to push the ball up to the target, finishing the dribbling episode.
- *% time in fault-state (*$\%\, t_{FS}$*)*: the cumulated time in fault-state t_{FS} during the whole episode. The fault-state is considered when the robot loses the ball possession, i.e., $\rho > \rho_{th} \vee |\gamma| > \gamma_{th} \vee |\varphi| > \varphi_{th}$, then: $\%\, t_{FS} = t_{FS}/t_{DP}$.

A final comparison between the best DL scheme from methods evaluated in experiment *(i)*, the best scheme with CoSh from methods evaluated in *(ii)*, and the best scheme with NASh from methods evaluated in *(iii)* is carried out by using their averaged mean rewards and a global fitness function introduced in order to evaluate both performance indices together. The global fitness is computed as follows, where t_{DP} is normalized by the maximum value of t_{DP}.

$$F = 1/2 \cdot [(1 - t_{DP}/t_{DP-max}) + \%t_{FS}/100] \quad (1)$$

6. Results and Discussion

Experiment (i): Figure 2 shows the learning evolution of the three decentralized learners: *DL pessimistic*, *DL zeros,* and *DL optimistic.* With respect to the learner initialized with zeros, the pessimistic scheme achieves a similar $\%t_{FS}$, and the optimistic one achieves a similar t_{DP}. However, the learner initialized with zeros shows the best performance for both indices, furthermore, it shows the fastest asymptotic convergence. Notes that pessimistic initializations favor the ball control (improving $\%t_{FS}$) meanwhile optimistic initializations favor the dribbling speed ing t_{DP}). This is because the agent gets negative reinforces were it losses the control of the ball and gets positive reinforces were it goes fast. Thus, the learner initialized neutrally whit zeros keeps a trade-off between $\%t_{FS}$and t_{DP}.

Experiment (ii)-(iii): the two proposed strategies for transferring knowledge are compared in Figure 3, the *Control Sharing* (CoSh) and the Nearby *Action Sharing* (NASh). As the experiments for DLs, pessimistic, with zeros, and optimistic initializations are tested.

Regarding CoSh, the pessimistic initialized learner shows the fastest asymptotic convergence and the best performances. The better performance of optimistic initializations for the CoSh method is well discussed in (Knox & Stone 2010), which is validated in this work with a different and more complex problem.

Comparison of the best schemes: According to experiments (i)-(iv), the fastest asymptotic convergence and best performances of each of the tested methods (DL, CoSh, NASh) are: the DL with zeros initialization (*DL zeros*) where knowledge transfer is not used; the accelerated DL with c*ontrol-sharing* and pessimistic initialization (CoSh pessimistic), and the accelerated DL with *nearby-action-sharing* and optimistic initialization (NASh optimistic). These learning evolution plots are shown in Figure 5,

where the mean reward of each decentralized agent and a global fitness evolution are included.

The policy of the run with better performance of the *DL zeros,* CoSh pessimistic, and NASh optimistic are stored. After that, these are tested and measured separately by a hundred of runs in order to obtain statistically significant results. Those performance are presented in Table 2, in addition to the RL-FLC reported in (Leottau et al. 2014).

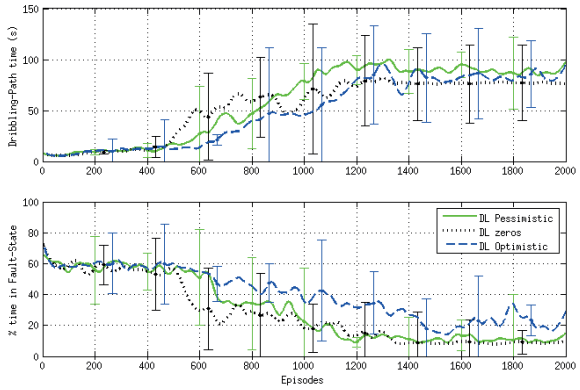

Figure 2: Learning evolution of the Decentralized Learners (DL).

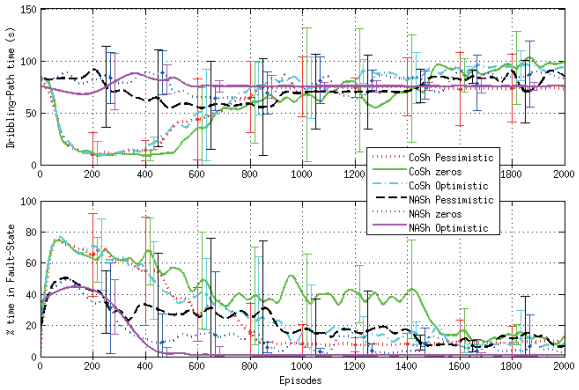

Figure 3: Learning evolution of the Control Sharing (CoSh) and Nearby Action Sharing (NASh) approaches.

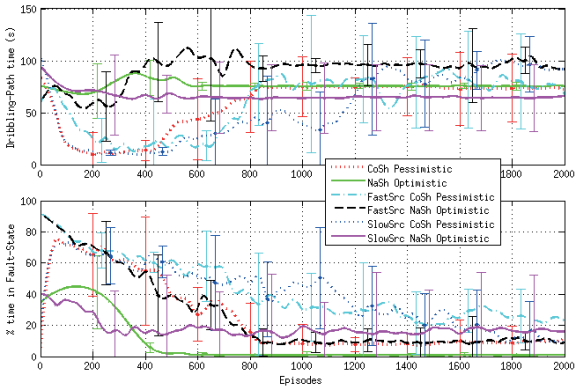

Figure 4: Learning evolution of the CoSh and NASh approaches transferring knowledge form sources with different performances.

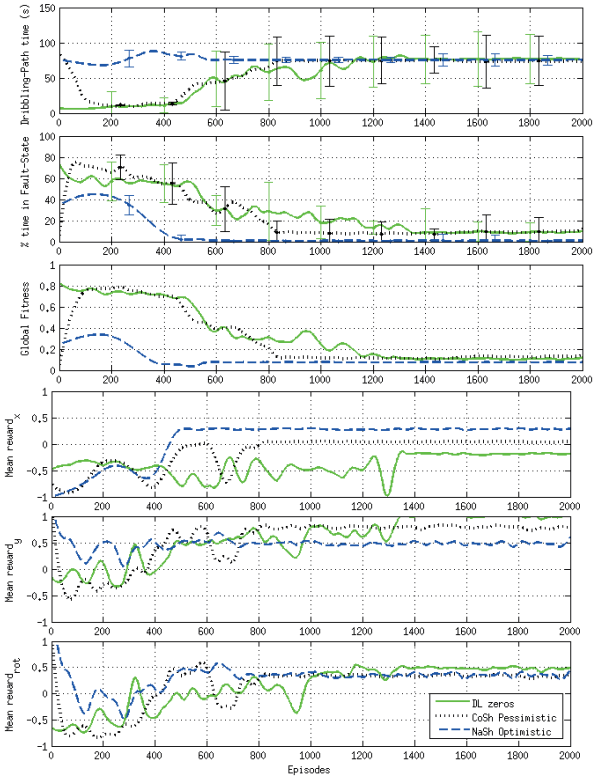

Figure 5: Learning evolution of the best performances parameter settings by each method: DL, CoSh and NASh schemes.

Table 2: Performance indices

Method	t_{DP} *(s)*	t_{DP} *St.Dev.*	$\%t_{FS}$	$\%t_{FS}$ *St.Dev.*
RL-FLC	62.19	3.30	14.57	5.13
DL zeros	58.08	6.64	4.07	7.75
NASh Optimist.	47.49	0.61	1.67	4.13
CoSh Pessimit.	56.50	5.07	1.83	5.44
Best Source	71.03	1.77	7.29	3.42
Slow Source	93.38	1.33	0.56	0.11
Fast Source	41.55	0.44	90.08	1.77

From Table 2, *DL* is on average 9% faster and commits 10.5% less faults than RL-FLC. NASh optimistic is the most effective learning scheme. It shows the fastest asymptotic convergence, and the best asymptotic performance. If a threshold of *F=0.12* is defined, according to Figure 5, the *time to threshold* (Taylor & Stone 2009) of NASh optimistic is 331 learning episodes, the 26.6% and 39.5% of the required learning time with respect to DL and CoSh. Moreover, from Table 2, NASh's dribbling time t_{DP} speedup the DL and CoSh in around 18% and 15.9% respectively, and it commits 2.4% and 0.16% less faults than DL and CoSh.

From mean reward plots in Figure 5, it can be noticed that NASh scheme shows the highest values for *mean-reward*$_x$ but the lowest for *mean-reward*$_y$ and *mean-*

$reward_\theta$. The opposite of the *DL* which $mean\text{-}reward_x$ is the lowest, meanwhile its $mean\text{-}reward_y$ and $mean\text{-}reward_\theta$ are the highest. Thus, it can be said the dribbling problem does not require a very accurate controller for v_y and v_θ for achieving good performance, so, the key of the NASh's best performance is a good policy for the $Agent_x$.

Note that at the early episodes, $Agent_y$ and $Agent_\theta$ mean-rewards of the knowledge transfer schemes get highest values, those decrease during the transfer and learning process, and these converge to values lower than their initial values at the fists episodes. It can be explained taking into account the MAS benefits; and, although any coordination mechanism is implemented, each decentralized agent is able to adjust its own policy, observing a joint state space, interacting with a joint environment, and acting individually towards a common goal, achieving the so-called indirect coordination.

A video which shows some of the training process and the learned policies for the dribbling behavior can be watched in (Leottau 2014).

7. Conclusions

This paper has proposed and implemented the Decentralized-RL of the ball-dribbling behavior in the context of humanoid soccer robotics, where each component of the omnidirectional biped walk (v_x, v_y, v_θ) is learned in parallel with single-agents working in a multi-agent task.

Two methods for accelerating the D-RL, transferring knowledge from simple linear controllers have been tested and compared: the *control sharing* method, and the *nearby action sharing* which is introduced by authors in this paper.

Several schemes of D-RL are evaluated testing different Q initialization values, and different sources of knowledge for the accelerated learners.

The effectiveness and benefits of the RL-FLC approach have been pointed in (Leottau et al. 2014). However, a significant human effort and knowledge of the controller designer are required for implementing all the proposed stages. In that sense, a D-RL approach is able to learn almost autonomously the whole dribbling behavior, achieving best performances with respect to the RL-FLC with less human effort and less previous knowledge. An advantage that still remains the RL-FLC method is the considerably less RL training time, regarding the D-RL scheme (100 episodes vs. 1500 episodes approximately). In that scene, the knowledge transfer strategies for D-RL agents proposed in this work are able to reduce that learning time up to 330 episodes, which open the door to make achievable future implementations for learning *ball-pushing* based behaviors with physical robots.

Our ongoing research outline includes: (*i)* extending the current D-RL scheme to cooperative and joint actions learners with coordination mechanisms; (*ii)* extending the NASh method for addressing those cases where the agents re-learn all the policy more than enhance it weakness; (*iii*) validating the approach using physical robots.

Acknowledgments

This work was partially funded by FONDECYT under Project Number 1130153. The first author was funded by CONICYT, under grant: CONICYT-PCHA/Doctorado Nacional/2013-63130183.

References

Alcaraz, J., Herrero, D. & Mart, H., 2011. A Closed-Loop Dribbling Gait For The Standard Platform League. In *Workshop on Humanoid Soccer Robots of the IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*. Bled, Slovenia.

Bianchi, R.A.C., Ribeiro, C.H.C. & Costa, A.H.R., 2012. Heuristically Accelerated Reinforcement Learning: Theoretical and Experimental Results. In L. De Raedt, ed. *Frontiers in Artificial Intelligence and Applications*. pp. 169 – 174.

Busoniu, L., Schutter, B. De & Babuska, R., 2006. Decentralized Reinforcement Learning Control of a Robotic Manipulator. In IEEE, ed. *Ninth International Conference on Control, Automation, Robotics and Vision, ICARCV 2006, 5-8 December 2006*. Singapore, pp. 1–6.

Carvalho, A. & Oliveira, R., 2011. Reinforcement learning for the soccer dribbling task. In *Computational Intelligence and Games (CIG), 2011 IEEE Conference on*. Seoul, Korea, pp. 95–101.

Fernández, F., García, J. & Veloso, M., 2010. Probabilistic policy reuse for inter-task transfer learning. *Robotics and Autonomous Systems*, 58(July 2009), pp.866–871.

Knox, W.B. & Stone, P., 2010. Combining Manual Feedback with Subsequent MDP Reward Signals for Reinforcement Learning. In *Proc. of 9th Int. Conf. on Autonomous Agents and Multi-agent Systems (AAMAS 2010)*. Toronto, Canada: International Foundation for Autonomous Agents and Multiagent Systems.

Knox, W.B. & Stone, P., 2012. Reinforcement Learning from Simultaneous Human and MDP Reward. In *In Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), June 2012*.

Latzke, T., Behnke, S. & Bennewitz, M., 2007. Imitative Reinforcement Learning for Soccer Playing Robots. In G. Lakemeyer et al., eds. *RoboCup 2006: Robot Soccer World Cup X SE - 5*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 47–58.

Leottau, D.L., 2014. Video: Decentralized RL of the Ball-Dribbling Behavior. Available at: https://docs.google.com/document/d/1dYl-a-uRleQzLkicp6pF9lyG7QXteYm5PRQbQ63aBZM/edit?usp=sharing.

Leottau, D.L., Celemin, C. & Ruiz-del-solar, J., 2014. Ball Dribbling for Humanoid Biped Robots: A Reinforcement Learning and Fuzzy Control Approach. In RoboCup, ed. *RoboCup 2014: Robot Soccer World Cup XVIII Preproceedings*. Joao Pessoa, Brazil. Available at: http://fei.edu.br/rcs/2014/RegularPapers/robocupsymposium2014_submission_58.pdf.

Li, X., Wang, M. & Zell, A., 2007. Dribbling Control of Omnidirectional Soccer Robots. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, pp. 2623–2628.

Macalpine, P. et al., 2012. Design and Optimization of an Omnidirectional Humanoid Walk: A Winning Approach at the RoboCup 2011 3D Simulation Competition. In *Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI-12)*. Toronto, Ontario, Canada,.

Martin, J. & Lope, H. De, 2007. A distributed reinforcement learning architecture for multi-link robots. In *4th Int. Conf. on Informatics in Control, Automation and Robotics, ICINCO 2007*. Angers, Francia, pp. 192–197.

Mataric, M.J., 1994. Reward Functions for Accelerated Learning. In *In Proceedings of the Eleventh International Conference on Machine Learning*. Boca Raton, Florida, USA: Morgan Kaufmann, pp. 181–189.

Meriçli, Ç., Veloso, M. & Akin, H., 2011. Task refinement for autonomous robots using complementary corrective human feedback. *International Journal of Advanced Robotic Systems*, 8(2), pp.68–79.

Riedmiller, M. et al., 2008. Learning to dribble on a real robot by success and failure. In *Robotics and Automation (ICRA), 2008 IEEE International Conference on*. Pasadena, California: IEEE, pp. 2207–2208.

Riedmiller, M. et al., 2009. Reinforcement learning for robot soccer. *Autonomous Robots*, 27(1), pp.55–73.

Röfer, T. et al., 2014. B-Human Team Description for RoboCup 2014. In *RoboCup 2014: Robot Soccer World Cup XVIII Preproceedings*. Joao Pessoa, Brazil.

Stone, P. & Veloso, M., 2000. Multiagent Systems: A Survey from a Machine Learning Perspective. *Autonomous Robotics*, 8(3), pp.1–57.

Sutton, R. & Barto, A., 1998. *Reinforcement Learning: An Introduction*, MIT Press.

Taylor, M. & Stone, P., 2009. Transfer learning for reinforcement learning domains: A survey. *The Journal of Machine Learning Research*, 10, pp.1633–1685.

Taylor, M.E. & Stone, P., 2007. Cross-domain transfer for reinforcement learning. *Proceedings of the 24th international conference on Machine learning - ICML '07*, pp.879–886.

Tilgner, R. et al., 2013. Nao-Team HTWK Team Description Paper 2013. In *RoboCup 2013: Robot Soccer World Cup XVII Preproceedings*. Eindhoven, The Netherlands: RoboCup Federation.

Tuyls, K., Hoen, P.J. 'T & Vanschoenwinkel, B., 2005. An Evolutionary Dynamical Analysis of Multi-Agent Learning in Iterated Games. *Autonomous Agents and Multi-Agent Systems*, 12(1), pp.115–153.

Yañez, J.M. et al., 2013. UChileRT RoboCup 2013 Standard Platform League Team Description Paper. In *RoboCup 2013: Robot Soccer World Cup XVII Preproceedings, July 2013*. Eindhoven, The Netherlands: RoboCup Federation.

Zell, A., 2008. Nonlinear predictive control of an omnidirectional robot dribbling a rolling ball. In *2008 IEEE International Conference on Robotics and Automation*. Ieee, pp. 1678–1683.