Learning Push Recovery Strategies for Bipedal Walking

Dicksiano C. Melo[1] **, Marcos R. O. A. Maximo**[1] **, Adilson Marques da Cunha**[2]

[1]Autonomous Computational Systems Lab (LAB-SCA)
Computer Science Division, Aeronautics Institute of Technology (ITA)

[2]Computer Science Division – Aeronautics Institute of Technology (ITA)

dicksianomelo@gmail.com, mmaximo@ita.br, cunha@ita.br

Abstract. *The present work provides an implementation of a Push Recovery controller that aids the walking engine used by a humanoid simulated robot. The simulation environment is the Robocup Soccer 3D Simulation League. The learned movement policies exceeded our original walking engine. In addition, we evaluated the policies and detected undesired biases. New methodologies were introduced in order to eliminate it.*

Resumo. *O artigo contribui com a implementação de um controlador Push Recovery que melhora o desempenho do motor de caminhada usado por um agente simulado humanóide do ambiente RoboCup Soccer 3D Simulation. A política de movimentos aprendida foi capaz de superar as baselines com significância estatística. Finalmente, propomos duas abordagens para remover vieses indesejados em nossas políticas finais.*

Student level: MSc. Date of conclusion (defense): 01/11/2021. To be considered CTDR.

1. Introduction

Bipedal locomotion is one of the hardest motions for a human body due to its complex physics and natural instability that requires constant adjustment of balance. Therefore, although human beings have perhaps the most sophisticated brain in the animal kingdom, a kid needs about nine months for learning how to walk while some quadruped or hexapod animals (whose body structure is inherently balanced) take just a few hours after birth for walking [Nandi et al. 2009].

In that context, the implementation of a biped walking engine is one of the biggest challenges in Mobile Robotics. Robots equipped with wheels perform satisfactorily in various environments. Nevertheless, wheeled robots usually need human support when navigating in uneven terrains. In this way, legged robots are more interesting to navigate in real environments.

Therefore, in order to make more tasks achievable by these agents, it is primordial that they be able to navigate on uneven terrain and to recover from external disturbances such as pushes, since modifying the environment occupied by humans to make it passable by these agents is impracticable. Also, legged systems present high-dimensional and non-linear dynamics. As a consequence, humanoid walking poses a great scientific problem [Tedrake 2004].

Humanoid walking is a field where nature is still far beyond technology: even nonathletes usually presents walking and running much superior than the state-of-the-art biped robots. Although engineers are capable of manufacturing legs kinematically mimicking their biological counterparts, animals and humans are usually far superior in terms of agility, robustness, flexibility, and energy efficiency. Therefore, it represents a knowledge field with a lot to be explored.

Model-based approaches for humanoid walking heavily relies on analytical models of the robot dynamics. Our final goal is to improve our model-based walking engine [Maximo and Ribeiro 2016]. Instead of learning the walking policy from scratch, we create a Push Recovery layer through model-free methods. Therefore, our controller was implemented without a mathematical model of the agent dynamics.

Simulation environments allow running experiments in a nearly automatic way, which is mandatory for applying machine learning techniques. In a real robot setting, this is hard to achieve [Maximo et al. 2017]. Therefore, we choose RoboCup 3D Soccer Simulation which is one of the leagues within RoboCup Soccer. RoboCup is an international initiative which focuses on two main domains: Robotics and Artificial Intelligence. This scientific reunion is held annually and is composed by a Symposium and a robotics competition which is made of several leagues. This event is conducted since 1997 and its first edition happened in the city of Nagoya. RoboCup's mission is that a team of humaonoid robots will be able to beat the human team champion of World Cup by 2050 [Kitano et al. 1998].

In total, five learning tasks are reported in this work. Each task, along with the machine learning algorithm, allowed our simulated robot to learn a desired behavior. Our final policy was able to outshine the baselines: using the original walking engine without any learned policy. Our hypothesis and experiments will be reported in Section 3.

Concerning to scientific research, along this work we searched for the answers of the research questions: **RQ1**: *Is it possible to learn a Push Recovery Controller using only a "naive" reward (pure Early Termination)?* **RQ2**: *Is it possible to learn a Push Recovery Controller in synergy with a walking engine?* **RQ3**: *Given a Push Recovery task, how does the policy trained with a "naive" reward, pure Early Termination, compares with a policy trained using a reward signal endorsed by physical principles?* **RQ4**: *Are learned Push Recovery Controllers zero-shot task learners?*

Finally, we evaluated the learned policies and detected undesired biases. In order to eliminate it, two methodologies were introduced in Section 4. As result, the final policies were able to improve the robot's balance when exposed to a wide range of perturbations. We emphasize that this work is an extended version of a conference paper [Melo et al. 2020] – awarded as one of the Best Papers from LARS 2020.

Given the success of *Deep Reinforcement Learning* (DRL) techniques to solve several problems of Continuous Control, this approach was adopted to address this problem. The DRL algorithm chosen is the called Proximal Policy Optimization (PPO) [Schulman et al. 2017]. Instead of implementing the PPO algorithm from scratch, we used its implementation from OpenAI Baselines [Dhariwal et al. 2017], which is a set of high-quality implementations of Reinforcement Learning algorithms.

2. Push Recovery Strategies

A biped standing erect is in a state of metastable equilibrium with respect to gravity [Nashner and McCollum 1985]. The center of mass of the body is located above its support. As consequence, no active muscular control is required.

Human balance is the target of many works [Nashner 1981, Nashner and McCollum 1985, Horak and Macpherson 1996]. Those works study how the human body responds to postural perturbations through multiple muscle synergies, or strategies. Namely, the "ankle" and "hip" strategies [Horak and Macpherson 1996, Horak et al. 1997] served as inspiration for this work.

2.1. Ankle Strategy

From the mechanical perspective, this strategy consists of rotating the body about the ankle joint [Nashner and McCollum 1985]. The body acts like an inverted pendulum with superior joints fixed. The ankle strategy moves the center of pressure (CoP). As a consequence, the tangential ground reaction force horizontal component is changed, restoring the equilibrium [Stephens 2007].

2.2. Hip Strategy

As the location of the CoP is limited under the feet, a second strategy is to create a moment around the center of mass (CoM). This results in a new point called the centroidal moment pivot (CMP). When there is zero moment about the CoM, the CMP is equal to the CoP. The CMP location moves outside of the area under the support foot when centroidal moment is not zero [Stephens 2007].

By applying torque to the hip joints, the upper body rotates forward and downward, which generates a backward rotation on the lower body. There is a decreasing of the moment of inertia about the ankle, which amplifies the ankle torque effect.

Biological experiments show that humans prefer using the ankle strategy first, if postural disturbances are small, and resort to the hip strategy only if disturbances are large. These biological studies have observed these strategies in humans, but have not explained the underlying biomechanical causes for this endeavor. Some of these studies have speculated that this behavior is a matter of preference, or is due to energy efficiency considerations [Hofmann 2006].

3. Methodology and Results

Our five experiments are **continuing Push Recovery tasks**: the robot is pushed periodically until its fall. There are two experiments (JPL and WSC) where the agent does not walk. On the other hand, the agent walks in the last three experiments (RANP, RAS, RAU). As a consequence, those experiments (RANP, RAS, RAU) are also **continuing Push Recovery walking tasks**. We now explain our evaluation hypothesis:

Evaluation Hypothesis I: *The best evaluation for a **continuing Push Recovery task** is the average duration of each episode. If an episode ends when the robot falls, the growth of the average duration of each episode indicates if the policy is performing the desired behavior, except for Reward Gaming [Leike et al. 2017] cases.*

Evaluation Hypothesis II: *A proper baseline for a* ***continuing Push Recovery walking task*** *is the average duration of each episode without any push recovery strategy – only the original walking engine.*

We ran each experiment multiple times. For each specific experiment we applied 95% bootstrap confidence interval [Efron and Tibshirani 1986]. The confidence interval is represented as shadowed areas in Figures 2 and 3. This approach is an attempt to achieve more consistent and reproducible results given the stochastic nature of the simulation.

3.1. State Space description

In order to design a practical push recovery controller, we use commonly available sensory data. We emphasize that for **all results** reported in this paper, the state is composed by: **(a)** The agent's torso angular velocity data; **(b)** The agent's torso acceleration data; **(c)** The agent's foot pressure sensor data; **(d)** The position of the agent's joints; **(e)** The height of the CoM.

Notice that this state space can be reproduced in many other environments. Except for the height of the CoM, most of the information comes from sensors that are available in several models of robots. The complete dimension of the State Space $\mathcal{S} \in R^{39}$.

3.2. Action Space description

The actions are position commands for the ankle joints in Ankle Strategy. Hip Strategy also include the hip joints. How each joint command is calculated will be described in the next Subsections. The Action Space dimensions are $\mathcal{A}_{Ankle} \in R^4$ and $\mathcal{A}_{Hip} \in R^{10}$. For JPL and WSC experiments, we used a Multilayer Perceptron (MLP) with 8 units per hidden layer. For the other experiments (RANP, RAS and RAU), we used a MLP with 64 units per hidden layer. All MLP has 2 fully-connected hidden layers and hyperbolic tangent as activation function.

3.3. Joint Position Learning – JPL

In order to disturb the agent, we periodically push the agent using the ball. Each collision has the duration of one simulation cycle. In RoboCup Soccer 3D Simulation League each simulation cycle lasts $20\ milliseconds$. We highlight that the ball points to the robot's CoM. The task is illustrated in Figure 1. For JPL, $\|\boldsymbol{v_{agent}}\| = 0.0$ in Figure 1.

The raw output $\hat{y}$ of the neural network for each target joint lies within the range $[-1,\ 1]$. Notice that the output of a neural network with hyperbolic tangent as activation funtion lies within the range $[-1,\ 1]$. The final joint command is computed using Equation (1). In other words, we map the value $\hat{y}$ from the range $[-1,\ 1]$ to the range $[\theta_{min}, \theta_{max}]$ for each joint. This method was inspired by [Abreu et al. 2019, Melo and Maximo 2019].

$$\theta_{action} = \theta_{min} + \ (\hat{y} + 1) \times \left(\frac{\theta_{max} - \theta_{min}}{2} \right). \tag{1}$$

The reward signal is represented in (2) where R_{factor} is a scaling constant. The longer the agent "survive"under external disturbance, more reward is summed.

$$\mathcal{R} = \begin{cases} 0, & \text{if agent has fallen} \\ R_{factor}, & \text{otherwise.} \end{cases} \tag{2}$$

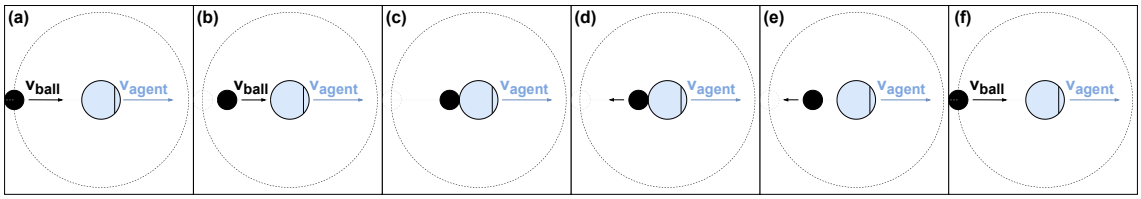

Figura 1. The black circle represents the ball, the blue circle represents the agent, v_{ball} represents the velocity of the ball, v_{agent} represents the velocity commanded to the agent's walking engine. The experiment sequence of events: (a) Ball is at maximum displacement from the agent ($R_{collision}$), its initial position; (b) Ball moves towards the agent; (c) Ball collides with the agent (one timestep); (d) After the collision, the ball's velocity direction is inverted; (e) Ball moves away from the agent; (f) Ball reaches the initial position. Then, the cycle is restarted.

Notice the growth of the average duration of the episodes in Figure 2(a). It indicates that the agent learns the desired behavior. The video was recorded to a training session of JPL[1]. It shows that the initial policy is driven to the desired behavior through the learning algorithm. We repeated the JPL experiment 12 times, generating the a small confidence interval in Figure 2(a), which reinforces the legitimacy of our results.

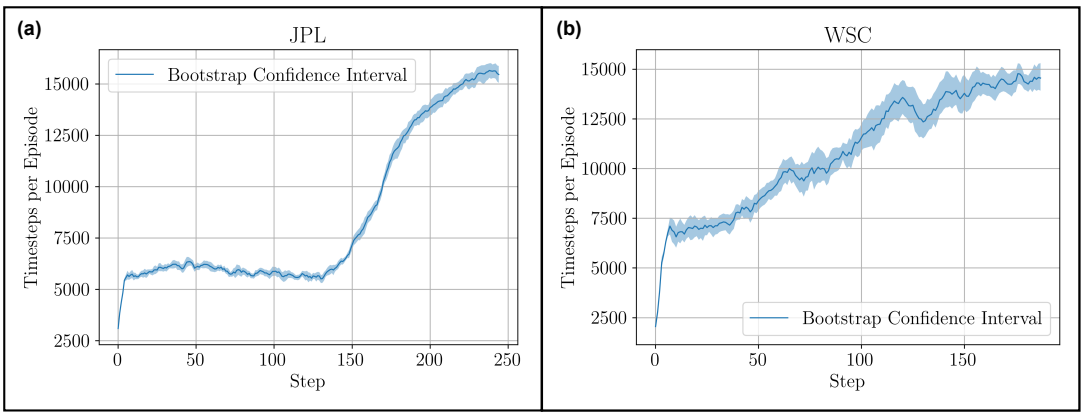

Figura 2. Episode duration during (a) JPL and (b) WSC. The shadowed region represents 95% bootstrapped confidence interval of the average duration of the episodes from 12 distinct sessions.

3.4. Walking Stabilization Controller – WSC

The second experiment has the same learning task and the same reward signal from JPL. An illustration of WSC task is represented in Figure 1 with $\|\boldsymbol{v_{agent}}\| = 0.0$. The main difference to JPL is that we no longer use the output from the neural network as joint positions themselves. In WSC, the output from the neural net are summed into the current joint positions. Thereby, the position for each joint is calculated by the equation:

$$\theta_{action} = \theta_{current} + \theta_{min} + (\hat{y} + 1) \times \left(\frac{\theta_{max} - \theta_{min}}{2} \right). \quad (3)$$

[1] https://www.youtube.com/watch?v=YqzH3Mr2dTY&t=0s

The second difference is that the initial position of the agent was engineered to be as close as possible to the walking pose. The goal was to drive the learned behavior to be closer to the walking movement. Notice the growth of the average duration of the episodes in Figure 2(b). It indicates that the agent learns the desired behavior. We also observe a small confidence interval generated repeating the same experiment in 12 sessions. We present a video[2]. It shows that the initial policy is driven to the desired behavior through the learning algorithm. The last video [3] evidences that WSC policy being deployed in our original walking engine.

3.5. Running Agent with No Priors – RANP

The learning task JPL and WSC were inspired by similar works [Yang et al. 2017, Yang et al. 2018] where the simulated robot was periodically pushed while trying to keep upright. In this work we faced a new challenge: trying to learn a Push Recovery policy while the robot runs. Similarly to JPL and WSC, the agent is pushed periodically and the reward signal is the same. Also, an episode ends when the robot falls. A detailed illustration of RANP task is represented in Figure 1 with $\|\boldsymbol{v_{agent}}\| > 0$. The final action follows the same format used in WSC, described in Equation (3). The initial position was the same engineered for WSC.

Notice the growth of the average duration of the episodes in Figure 3(a). It indicates that the agent learns the desired behavior. In order to answer the research question **RQ2**, we compare the average episode duration with the baseline measured using the original walking engine without any policy.

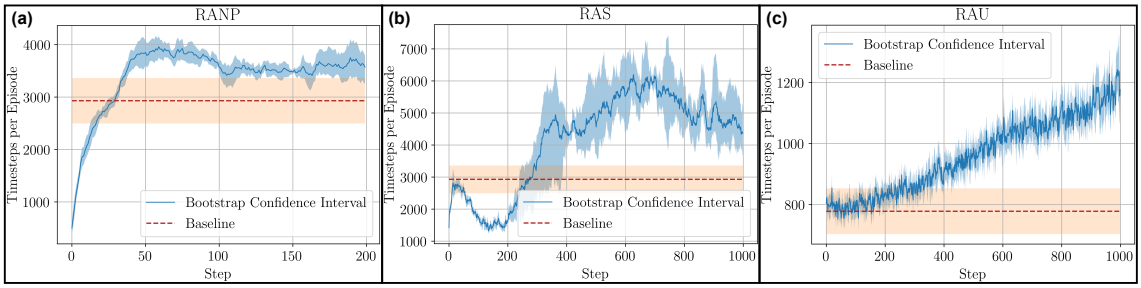

Figura 3. Episode duration during (a) RANP training, (b) RAS training and (c) RAU training. The shadowed region represents 95% bootstrapped confidence interval from 5 training sessions (RANP), 4 training sessions (RAS) and 12 training sessions (RAU) repeated in the same conditions. The evaluation was ran for 5 million timesteps: no policy, just our orginal walking engine [Maximo and Ribeiro 2016].

3.6. Running Agent Sagittal – RAS

Through the results obtained so far, we could answer research question **RQ1**. We conclude that a Push Recovery Controller can be learned through DRL using a "naive" reward signal. Our "naive" reward was based in Early Termination [Peng et al. 2018] as mentioned.

[2] https://www.youtube.com/watch?v=YqzH3Mr2dTY&t=207s
[3] https://www.youtube.com/watch?v=YqzH3Mr2dTY&t=548s

Now, our goal is to compare the impact of the reward signal in the learning process: investigate the impact of using the same reward signal used in similar works [Yi et al. 2011, Yi et al. 2013, Yang et al. 2017, Yang et al. 2018]. The new reward signal was inspired by related literature [Yang et al. 2017, Yang et al. 2018]. Generally speaking, this reward decomposes balancing into objectives and explicitly punishes undesired states. Each individual reward is calculated by $R_i = \exp\left[-c_i(x - x_0)^2\right]$, where x_0 is the target value, x is the actual value, and c_i is a normalization factor. Notice that $R_i \in (0,\ 1]$. Thereby, this formulation punishes large values of $\mid x - x_0 \mid$ and explicitly reinforces $\mid x - x_0 \mid \rightarrow 0$.

We modulate three values: center of mass's height $Z \in R$, the torso's angular velocity $[\omega_x\ \omega_y\ \omega_z]^T \in R^3$ and the torso's acceleration $[a_x\ a_y\ a_z]^T \in R^3$. A detailed illustration of RAS task is represented in Figure 1 with $\|\boldsymbol{v_{agent}}\| > 0$.

Again, notice the growth of the average duration of the episodes in Figure 3(b). It indicates that the agent learns the desired behavior. The new reward signal leads to a better result compared to RANP. This results answers **RQ3**. Therefore, is possible to achieve the desired behavior using a "naive" reward, pure Early Termination. However it presents an inferior performance when compared with a policy trained using a reward signal endorsed by physical principles. The video[4] was recorded in order to show the RAS policy being used.

3.7. Running Agent Uniform – RAU

The empirical observations of human postural responses to perturbations in the sagittal plane [Nashner and McCollum 1985] deeply inspired the design of our experiments so far. As the perturbations are not limited to the sagittal plane in real world environments, we propose a new experiment. Let us define $\alpha_{collision}$ as the angle between the ball's velocity $\boldsymbol{v_{ball}}$ and the agent's velocity $\boldsymbol{v_{agent}}$. The value of $\alpha_{collision}$ is altered after a collision. The values for $\alpha_{collision}$ will be uniformly sampled from the range $[-90°, 90°]$. See Figure 4 for a detailed explanation of the process.

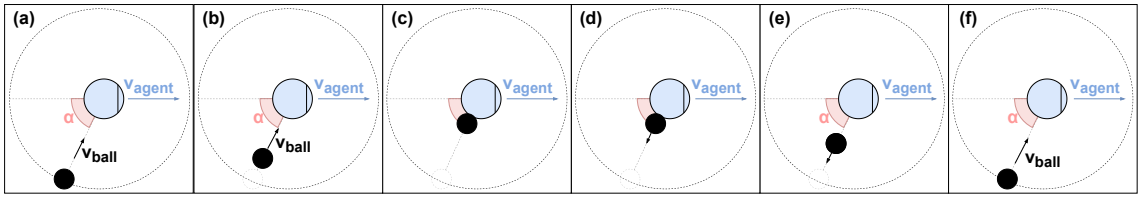

Figura 4. The black circle represents the ball, the blue circle represents the agent, v_{ball} represents the velocity of the ball, v_{agent} represents the velocity commanded to the agent's walking engine, α (in red) represents the angle $\alpha_{collision}$ between the agent's moving direction and the ball's moving direction when colliding with the agent. The experiment sequence of events: (a) Ball is at maximum displacement from the agent ($R_{collision}$), its initial position; (b) Ball moves towards the agent; (c) Ball collides with the agent (one timestep); (d) After the collision, the ball's velocity direction is inverted; (e) Ball moves away from the agent; (f) Ball reaches the initial position, new $\alpha_{collision}$ is sampled, the ball is dislocated to a new starting position and the cycle restarts.

[4] https://youtu.be/qb-SNPAvQgI

The new experiment presents a completely different dynamics. First, there are different kinds of collisions. Also, there is large difference between the range of the pitch and roll joints. Therefore, the action space does not have the same efficiency in different directions. The final action follows the same format used in WSC, described in Equation (3). The reward signal is the same from RAS.

Again, notice the growth of the average duration of the episodes in Figure 3(c). It indicates that the agent learns the desired behavior. Again, we compare the average duration of each episode with the baseline measured using the original walking engine without any policy. The value of the baseline is different from Figure 3(a) and Figure 3(b). It was expected because the new experiment presents a completely different dynamics as discussed before. Finally, the video [5] illustrates the learned policy being used.

4. Policy Evaluation and Improvement

Using *Evaluation Hypothesis I and II*, we will evaluate our learned policies. As there are 5 different experiments (JPL, WSC, RANP, RAS and RAU), there are 5 learned policies. For the sake of conciseness, we will only choose 2 policies. The **JPL** policies is not deployable with a walking engine due to the action format, therefore it will be excluded. The **WSC** policy was not learned together with our walking engine, thereby it will also be excluded. The policy learned from RANP will also be excluded. Comparing Figure 3(a) and Figure 3(b), we concluded that **RAS** was superior to **RANP**. Finally, the chosen policies are the ones learned from RAS and RAU. Notice that both experiments used reward signals endorsed by physical principles and the robot were walking while being pushed. Let us define $\pi_{\underline{S}}$ as the movement policy learned in RA**S**. Similarly, $\pi_{\underline{U}}$ is the movement policy learned in RA**U**.

Let us now define four different benchmarking tests. For task $\mathcal{T}_S$ all collisions are in the sagittal plane, similar to **RANP** and **RAS**. For task $\mathcal{T}_R$ all collisions are in the coronal plane on the right side (right lateral pushes). For $\mathcal{T}_L$ all collisions are in the coronal plane on the left side (left lateral pushes). Notice that the coronal plane divides the body into dorsal (back) and ventral (front) part. For task $\mathcal{T}_U$ the collisions follow the same structure from **RAU**: the angle $\alpha_{collision}$, already defined in Subsecion 3.7, is sampled from $[-90°, 90°]$.

We ran each learned policies π_S and π_U for 5 million timesteps in order to estimate its performance according to *Evaluation Hypothesis I and II*. Notice that π_S was learned using $\mathcal{T}_S$ and π_U was learned using $\mathcal{T}_U$. The values of the average number of timesteps per episode using π_S and π_U in the tasks ($\mathcal{T}_S$, $\mathcal{T}_R$, $\mathcal{T}_L$, $\mathcal{T}_U$) are reported in Table 1.

Notice that π_S was learned only with collisions in the sagittal plane. As it surpassed the baseline for $\mathcal{T}_R$ and $\mathcal{T}_U$, one may interpret it as some sense of generalization. On the other hand, π_S performed worse than the baseline in task $\mathcal{T}_L$. As result, π_S may not be considered a zero-shot learner [Singh et al. 2020, Oh et al. 2017] in terms of generalizing for all 4 tasks defined, because it was not able to generalize to the unseen task $\mathcal{T}_L$. This answer **RQ4**.

Also, π_U performed worse than the baseline in task $\mathcal{T}_S$. Neither π_S nor π_U surpassed the baseline in all proposed tasks ($\mathcal{T}_S$, $\mathcal{T}_R$, $\mathcal{T}_L$, $\mathcal{T}_U$). Now, we introduce two methods

[5] https://youtu.be/qb-SNPAvQgI

Tabela 1. Average number of timesteps per episode for π_S, π_U, π_{TL}, π_{IL}. Higher values indicates better performances.

Scenario	Baseline	π_S	π_U	π_{TL}	π_{IL}
$\mathcal{T}_S$	2930.02	4627.56	1453.55	3570.93	5689.32
$\mathcal{T}_R$	505.73	1569.98	858.1	1048.53	1877.35
$\mathcal{T}_L$	512.39	404.12	882.96	849.3	730.62
$\mathcal{T}_U$	778.91	899.92	1115.44	1498.85	1116.17

in order to combine those policies and derive our final policy. We emphasize that the goal is to surpass the baseline in all tasks.

4.1. Transfer Learning – π_{TL}

We choose the expert policy π_U as our source for transfer learning [Goodfellow et al. 2016]. As shown in Table 1, the only scenarios where π_U has a performance inferior to the baseline is $\mathcal{T}_S$. Thereby, we used π_U as seed for a new training session in RAS.

We highlight that the transfer learning approach was not quite straightforward. Generally speaking, long training sessions result in policies with good performances in $\mathcal{T}_S$, but forgot how to perform in $\mathcal{T}_U$ or $\mathcal{T}_L$, a known phenomena called *Catastrophic Forgetting* [Goodfellow et al. 2015]. The initial policy π_U was trained along $50\ M$ timesteps. The best result was achieved by using a second training session with $30\ M$ timesteps. Thereby, ratio between the duration of the pre-training and the duration second training session is 5/3 for the results presented here.

4.2. Imitation Learning – π_{IL}

Now, we aim to distill the knowledge from π_S and π_U in a new policy through Supervised Learning – Behavioral Cloning [Bain and Sammut 1995]. In that sense, we had to collect pairs of states and actions (s_i, a_i) using π_S and π_U.

We used π_S for the "correct" actions in $\mathcal{T}_S$. As π_U presented a weak performance in task $\mathcal{T}_S$, we inferred the "incorrect" actions with $\boldsymbol{\pi_U}$. Similarly, we used π_U for the "correct" actions in $\mathcal{T}_L$. As π_S presented a weak performance in task $\mathcal{T}_L$, we inferred the "incorrect" actions with $\boldsymbol{\pi_S}$. Notice that by "incorrect" we would like to enforce that the policy which presented a weak performance in a specific task, would probably infer an inefficient action for that task. The Loss Function used for the regression was inspired by the Triplet Loss [Parkhi et al. 2015]. We emphasize that our methodology was applied to distill the knowledge of two neural networks which were experts in different tasks in a final one. As result, the new neural network presents a good performance in all tasks. This idea can be extended for multiple scenarios.

4.3. Final Results

Comparing the values of the average number of timesteps per episode of the final policies π_{TL} and π_{IL} in the tasks ($\mathcal{T}_S$, $\mathcal{T}_R$, $\mathcal{T}_L$, $\mathcal{T}_U$) with the values measured from the baselines, we concluded that π_{TL} and π_{IL} outperforms the baseline in all evaluation scenarios. Please, refer to Table 1.

In order to check the feasibility of learned policies, we ran ten thousand inferences from the neural net π_s. All times were smaller than 0.00045 seconds. Each simulation cycle lasts 20 milliseconds. Therefore, in the worst case, the inference time represented 2.50% of the whole cycle. We conclude that the inference is much smaller than the total time to train the neural net. Also, as the biggest inference time took only 2.50% of a cycle, the solution is feasible to be deployed. We ran all inferences on an Intel i7-8750H CPU with 12 logical cores. We did not use GPU to accelerate the inferences, which would possibly improve the results.

5. Conclusions and Future Works

The main goal of this work was to improve the walking movement in a simulated humanoid robot by learning human-inspired behaviors called Push Recovery strategies. The technique to solve this problem was a Model-free Deep Reinforcement Learning algorithm called Proximal Policy Optimization. The choice was based on the complexity associated with the humanoid robot's dynamics. By using the Model-free algorithm, we avoid having to model the dynamics. Also, it is a interesting choice as human beings perform Push Recovery strategies without knowing its own dynamics a priori.

From the experiments conducted, we conclude that the present work was able to learn the desired behavior using a higher level reward function, pure Early Termination, compared to the related literature [Yi et al. 2011, Yi et al. 2013, Yang et al. 2017, Yang et al. 2018].

As the policy from **RAS** performed much better than the regular **RANP**, we do recommend the application of physical principles in the reward design for future works. We believe that this is the first work to explore the impact of a "naive"reward in the final policy.

Given our JPL[6] and WSC[7] videos, one may conclude that the desired behaviors were achieved. Indeed, both achieved robust movement policies that were able to resist against perturbations minutes straight. For all walking experiments, all were able to surpass the baseline from a large margin. This work was not only able to learn the desired behaviors but also to measure their improvements.

Although, transferring knowledge from simulated environments to real robots, *sim2real*, can be really challenging, we believe that this represent a trend in robotics research [Zhao et al. 2020]. Therefore, we also encourage applying our methodology to real robots.

An interesting research would be to study the final outcome in the Push Recovery policy when using different Action Spaces. In the present work, we have compared different tasks related do Push Recovery capabilities and also experimented different Reward Signals. However, we have not studied what would happen if only a small subset of joint was removed from the Action Space. The inspiration for this research is based on [Peng and van de Panne 2017].

The parameter $\|\boldsymbol{v_{ball}}\|$ is one of the most important values in this work. It was hard to tune, since a too small $\|\boldsymbol{v_{ball}}\|$ does not add relevant experience to the agent. On

[6] https://www.youtube.com/watch?v=YqzH3Mr2dTY&t=0s

[7] https://www.youtube.com/watch?v=YqzH3Mr2dTY&t=207s

the other hand, a large values does not allow the agent to learn, as it always fall. Therefore, would be interesting utilize the values presented in this work and apply Curriculum Learning in order to make the agent learn how to recover from stronger collisions.

Referências

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 *2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 1–8.

Bain, M. and Sammut, C. (1995). A framework for behavioural cloning. In *Machine Intelligence 15*.

Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., Wu, Y., and Zhokhov, P. (2017). Openai baselines. `https://github.com/openai/baselines`.

Efron, B. and Tibshirani, R. (1986). Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. *Statist. Sci.*, 1(1):54–75.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

Goodfellow, I. J., Mirza, M., Xiao, D., Courville, A., and Bengio, Y. (2015). An empirical investigation of catastrophic forgetting in gradient-based neural networks.

Hofmann, A. (2006). Robust execution of bipedal walking tasks from biomechanical principles.

Horak, F., Henry, S., and Shumway-Cook, A. (1997). Postural perturbations: New insights for treatment of balance disorders. *Physical therapy*, 77:517–33.

Horak, F. and Macpherson, J. (1996). Postural orientation and equilibrium. in: Handbook of physiology. exercise: Regulation and integration of multiple systems. *MD: Am Physiol Soc*, pages 255–292.

Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., Osawai, E., and Matsubara, H. (1998). Robocup: A challenge problem for ai and robotics. In Kitano, H., editor, *RoboCup-97: Robot Soccer World Cup I*, pages 1–19, Berlin, Heidelberg. Springer Berlin Heidelberg.

Leike, J., Martic, M., Krakovna, V., Ortega, P. A., Everitt, T., Lefrancq, A., Orseau, L., and Legg, S. (2017). Ai safety gridworlds.

Maximo, M. R., Colombini, E. L., and Ribeiro, C. H. (2017). Stable and fast model-free walk with arms movement for humanoid robots. *International Journal of Advanced Robotic Systems*, 14(3):1729881416675135.

Maximo, M. R. O. A. and Ribeiro, C. H. C. (2016). ZMP-Based Humanoid Walking Engine with Arms Movement and Stabilization. In *Proceedings of the 2016 Congresso Brasileiro de Automática (CBA)*, Vitória, ES, Brazil. SBA.

Melo, D. C., Máximo, M. R. O. A., and da Cunha, A. M. (2020). Push recovery strategies through deep reinforcement learning. In *2020 Latin American Robotics Symposium (LARS), 2020 Brazilian Symposium on Robotics (SBR) and 2020 Workshop on Robotics in Education (WRE)*, pages 1–6.

Melo, L. C. and Maximo, M. R. O. A. (2019). Learning humanoid robot running skills through proximal policy optimization.

Nandi, G., Ijspeert, A., Chakraborty, P., and Nandi, A. (2009). Development of adaptive modular active leg (amal) using bipedal robotics technology. *Robotics and Autonomous Systems*, 57:603–616.

Nashner, L. (1981). Analysis of stance posture in humans.

Nashner, L. M. and McCollum, G. (1985). The organization of human postural movements: A formal basis and experimental synthesis. *Behavioral and Brain Sciences*, 8(1):135–150.

Oh, J., Singh, S. P., Lee, H., and Kohli, P. (2017). Zero-shot task generalization with multi-task deep reinforcement learning. *CoRR*, abs/1706.05064.

Parkhi, O. M., Vedaldi, A., and Zisserman, A. (2015). Deep face recognition.

Peng, X. B., Abbeel, P., Levine, S., and van de Panne, M. (2018). Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Trans. Graph.*, 37(4).

Peng, X. B. and van de Panne, M. (2017). Learning locomotion skills using deeprl: Does the choice of action space matter? In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA '17, pages 12:1–12:13, New York, NY, USA. ACM.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *CoRR*, abs/1707.06347.

Singh, A., Jang, E., Irpan, A., Kappler, D., Dalal, M., Levine, S., Khansari, M., and Finn, C. (2020). Scalable multi-task imitation learning with autonomous improvement.

Stephens, B. (2007). Humanoid push recovery. In *2007 7th IEEE-RAS International Conference on Humanoid Robots*, pages 589–595.

Tedrake, R. L. (2004). *Applied Optimal Control for Dynamically Stable Legged Locomotion*. PhD thesis, Massachusetts Institute of Technology.

Yang, C., Komura, T., and Li, Z. (2017). Emergence of human-comparable balancing behaviours by deep reinforcement learning. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 372–377.

Yang, C., Yuan, K., Merkt, W., Komura, T., Vijayakumar, S., and Li, Z. (2018). Learning whole-body motor skills for humanoids. In *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, pages 270–276.

Yi, S., Zhang, B., Hong, D., and Lee, D. D. (2013). Online learning of low dimensional strategies for high-level push recovery in bipedal humanoid robots. In *2013 IEEE International Conference on Robotics and Automation*, pages 1649–1655.

Yi, S.-J., Zhang, B.-T., Hong, D., and Lee, D. (2011). Online learning of a full body push recovery controller for omnidirectional walking. pages 1–6.

Zhao, W., Queralta, J. P., and Westerlund, T. (2020). Sim-to-real transfer in deep reinforcement learning for robotics: a survey. *CoRR*, abs/2009.13303.