Obstacle negotiation learning for a compliant wheel-on-leg robot

Arthur Bouton, Christophe Grand, Faïz Ben Amar

Obstacle Negotiation Learning for a Compliant Wheel-on-Leg Robot

Arthur Bouton[1,2], Christophe Grand[3] and Faïz Benamar[1,2]

***Abstract*— Building versatile control for wheel-on-leg robots on various uneven terrains still constitutes a challenge. In this paper, we propose to combine a continuous state space Q-learning algorithm with a compliant wheeled locomotion structure capable of measuring forces applied by the environment. These forces are used to choose a behavior from the modulation of actuation distribution to simple configuration adjustments. Then, the robot does not need any prior knowledge on the ground geometry and is able to react to non-anticipated obstacles. The learned policy proves to be generic and allows the robot to negotiate complex obstacles that were not seen during learning.**

I. INTRODUCTION

Wheeled locomotion systems that are able of high relative wheel displacements are designed to enable good performances on both flat and uneven terrains. However, the control complexity significantly raises when attempting to cover the large variety of possible situations.

Passive structures offer native ground adaptation but do not control posture and force distribution, so their versatility is very limited and primarily consists in equilibrating the load on wheels for the largest variations of configuration [1]. Mainly for stability purpose, actuation can be added alongside passive joints while preserving the static determinacy of the structure [2], [3], but crossing capabilities still are entrusted to the wheel traction with no adaptation of the load distribution.

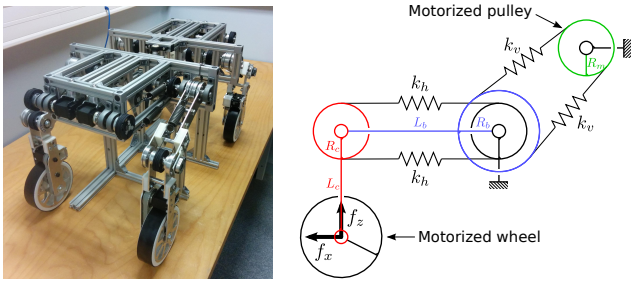
On the other hand, fully actively articulated robots can use their structural redundancy in order to adapt their configuration and literally lift wheels over obstacles [4]. However, active adaptation relies on prior knowledge of the ground geometry in order to plan the motion of each wheel [5]. A depth sensor can be used to map the environment [6], but in addition to data noise, accurately estimating the robot's pose with respect to the map, without external sensors, remains a challenging issue. Therefore, we can not guarantee that the optimal wheel placement will match the actual ground geometry under the robot.

Machine learning techniques have already been tested on wheel-on-leg robots for identifying the terrain type among very few classes in order to select whether the proper wheel speed control [7] or the predefined gait to use [8]. However, to our knowledge, the combination of compliant behavior and reinforcement learning in order to produce obstacle negotiation strategies has never been explored before.

[1]Sorbonne Universités, UPMC Univ Paris 06, UMR 7222, ISIR, F-75005, Paris, France {bouton,amar}@isir.upmc.fr

[2]CNRS, UMR 7222, ISIR, F-75005, Paris, France

[3]ONERA – The French Aerospace Lab, Toulouse, France christophe.grand@onera.fr



(a) The robot prototype (b) Kinematic principle of one leg

Fig. 1. The robot prototype related to this study weighs around 10 kg and the wheelbase is 0.58 m long for a 0.36 m wide track. The wheel diameter is 0.15 m and initial leg height is 0.21 m. All the characteristics of the robot's physical model used in the simulations, including the actuation modelization, are based on the prototype's data.

Here, we propose to take advantage of a compliant design where the measured external forces are used to learn how to deal with obstacles. Also, complex and computationally heavy plannings are replaced by few simple elementary behaviors independent from leg placements, while the forward movement of the robot is freely managed by the wheels. As a result, the robot clambers without lifting a wheel, so it is ensured to exploit every traction force, and adjusts its global control on the fly to the incoming geometries pressuring the structure.

Results prove that the robot is then able to cross complex and unknown obstacles with different shapes that were not seen at training time.

II. DESCRIPTION OF THE ROBOT

A. Four Compliant Legs

The robot stands on four wheels, each is linked to the chassis through two pivoting segments building up a serial kinematic chain. A set of antagonist springs, noted k_h on Fig. 1(b), aims to maintain the lower segments aligned with the vertical direction of the main body, so these segments tend to remain parallel. A second set of antagonist springs, k_v, connects the upper elements to an angular actuator, so as to form a series elastic actuation (SEA) at the top of each leg. Following the SEA principle, the actuator rotation is here controlled according to the elastic deformation of the k_v springs, which is deduced from a simple angle difference, in order to apply a desired torque on the joint.

Around the initial reference configuration where lower links are vertical and upper ones are horizontal, the mechanism offers a nearly orthogonal decomposition between passive and active stiffnesses, while relying robustness provided by pivot joints. Then it is possible to distinguish horizontal

forces passively absorbed by the legs from vertical ones that are controlled through the SEA at the end of the lever arm embodied by the upper segments. While these last forces are used to balance the chassis of the robot, horizontal actions applied by the environment on the legs can still be isolated. These actions reveal the presence of anything affecting the forward motion of the robot, such as obstacles, and give an image of the relative motion difficulty of each wheel. The analysis of inverse kinematics gives the expression of the horizontal force f_x applied on the leg as a function of the angles θ_c, θ_b and θ_m of respectively the lower segment, upper segment and motor pulley :

$$f_x = \frac{\pm L_c \sin\theta_c \tau_b + L_b \cos\theta_b \tau_c}{L_b L_c \cos(\theta_b - \theta_c)} \quad (1)$$

where $\tau_b = 2{R_b}^2 k_v \left(\frac{R_m}{R_b}\theta_m - \theta_b\right)$, $\tau_c = -2{R_c}^2 k_h \theta_c$ and the sign in front of L_c depends on whether we are considering a front or rear leg.

The balance of the chassis is regulated with a PD controller on the roll, pitch and elevation of the main body. The torque distribution on top joint of each leg is then given by the pseudo-inverse of the matrix expressing the influence of these torques on the main body dynamics. More details on this control can be found in [9]. Relying on a vertical projected force servoing allows the robot to balance regardless of differences in ground height under each wheel. The ground variations are naturally filtered by the SEA and the four wheels are ensured to keep the contact with the soil without the need of any prior knowledge on ground geometry.

B. Steering Pivot

As shown in Fig. 2(c), the robot is endowed with a pivot joint at the center of the main body that allows the symmetrical front and rear parts of the robot to rotate relative to each other along the vertical axis. The robot is then able to control its direction without theoretically any slipping if each wheel speed is properly adapted in accordance with the pivot control. With this design, the path curvature is limited, but it avoids extra weight, complexity and weakness of a vertical pivot over each wheel. Also, skid-steering is unpractical in our case, as the wheelbase is not stiff.

In addition, rotation of the chassis is also of great asset when dealing with obstacles, as shown in [10]. Two sequentially performed functions can thus be achieved by a unique strong joint.

C. Wheel Speed Adaptation

In order to assist the horizontal wheel positioning, each wheel speed ω is modulated according to the angular deviation θ_c of the vertical segment from its rest position, as expressed by :

$$\omega = (1 + k_\omega \theta_c)\frac{V_d}{R} \quad (2)$$

where V_d is the desired robot velocity, R the wheel radius and k_ω a proportional gain. This control prevents persistent internal stresses due to horizontal relative wheel displacements that can be maintained by ground friction. The measured horizontal forces are then ensured to come from ground elements pressuring the locomotion structure.

III. DEFINITION OF THE ACTIONS

The chassis balance control through SEA handles all ground height variations, so its combination with an even load distribution on wheels suffices to deal with smooth surfaces. However, discontinuities that are sharp enough to constitute an obstacle need a change of the applied forces on wheels in order to be surmounted. Therefore, we can define several adaptations relying on the same control scheme so as to either offer convenient load distributions, or modify the current approach configuration. In all cases, the wheel speed control keeps focusing on the desired robot's velocity tracking. Then, in every situation that can be encountered, for any obstacle layout, the robot will have to find the best sequence of actions that will allow it to continue advancing.

The seven actions, that can be seen as high-level control variants, are listed in the four following sections and an overview of actions dedicated to the crossing situations is given by Fig. 2.

A. Regular Advance Mode

The first action, which can also be seen as the default one, consists in applying an even torque distribution on legs while regulating the chassis balance to the upright position, which corresponds to a center of mass (CoM) lying midway between the supporting points. Energy consumption and internal stresses are therefore minimal, while the robot's stability is ensured.

Let $\mathbf{w}$ a reduced dimension wrench of the three actions, two torques and a force, having directly effect on each independent variable to be regulated by the postural control, i.e. the roll, pitch and elevation, and $\boldsymbol{\tau}$ the vector containing the four leg actuation torques. We can express the 3×4 matrix $\mathbf{G}$, which depends on the geometry of the current robot configuration, such that $\mathbf{w} = \mathbf{G}\boldsymbol{\tau}$. Then, the torque distribution $\boldsymbol{\tau}$ can be computed by applying the pseudo-inverse of $\mathbf{G}$ to the output of the PD controller in charge of the roll, pitch and elevation regulation. This gives the actuation distribution with the minimal quadratic sum of torques that leads to the desired action $\mathbf{w}$ on the chassis.

As this mode is the most energy-efficient and well balanced, the robot has to keep using it as much as possible, as the default mode. So any other action should only be chosen when the robot is likely to be blocked if it persists in this mode.

When this mode is active, it means that the geometry of the ground is smooth enough not to engage particular crossing capabilities, so the central pivot of the main body is free to be used for steering along the desired trajectory. As soon as any other action is chosen, the priority is given to the obstacle crossing, so the steering is interrupted. Once the difficulties are passed and the robot comes back in the regular advance mode, the steering pivot is free to correct the course again.

B. Modulation of the Torque Distribution

As the system relies on four supports, each one subject to a single force control, there is one extra degree of freedom in choosing the actuation distribution while producing the same effect on the acceleration of the three posture variables. Using the notation $\mathbf{w} = \mathbf{G}\boldsymbol{\tau}$ defined in section III-A, actions 2 and 3 can thus consist in modifying the torque distribution $\boldsymbol{\tau}$ along the kernel of $\mathbf{G}$, so that $\mathbf{w}$ remains unchanged. In the general case, a vector $\mathbf{u_0}$ of $\ker(\mathbf{G})$ can be found with a singular value decomposition. Then, we look for the greatest value of λ such that each torque of $\boldsymbol{\tau}' = \boldsymbol{\tau} + \lambda\mathbf{u_0}$ keeps exerting a positive vertical action on legs, so wheels with the minimal load remain in contact with the ground. $\boldsymbol{\tau}'$ is then the new distribution of torques to be applied. The difference between both actions is thus defined by the direction of $\mathbf{u_0}$. In practice, this direction determines which couple of legs along the diagonal of the robot is going to be relieved, while torques applied on the two other legs will increase as a balance.

C. Displacement of the Center of Mass

Because of the unilateral torques generated by every wheel while crossing an obstacle, the previous defined action cannot lead to a full cancellation of the vertical force applied on one of the rear wheels, according to the static equilibrium. This is why we add the 4th and 5th actions consisting in the displacement of the robot CoM over three supporting legs in order to liberate the fourth one, i.e. either the rear left or rear right leg. To do this, we modify the desired roll and pitch of the chassis so the robot bents forward and to the opposite side of the leg to be freed. According to the linear inverted pendulum model described in [11], the CoM is thus displaced above the support area of both front wheels and one of the rear wheels. The column of $\mathbf{G}$ accounting for the influence of the leg outside this area is then set to zero in order to transfer the robot's mass to the other legs when computing the torque distribution. It is to note that the postural control still regulates the roll, pitch and elevation even when the robot is bent and relies on three legs.

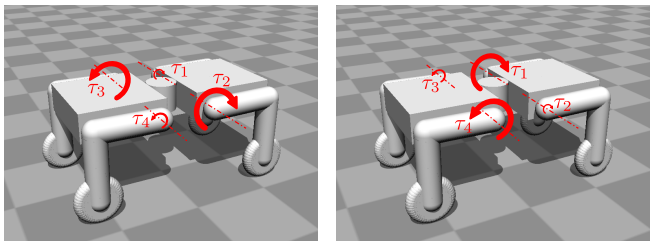
D. Rotation of the Steering Pivot

The pivot joint at the center of the robot allows it to steer, but also to change the relative spaces between the equilibrium positions of front and rear wheels. Indeed, it can bring left wheels closer while taking away the right ones, or vice versa. This can be particularly useful when several wheels are encountering a difficulty at the same time. The reconfiguration of wheel positions can then be used in order to help sequencing the obstacle crossing, one wheel after an other. Thus, the central pivot provides two more actions, 6 and 7 : a constant speed rotation in each direction.

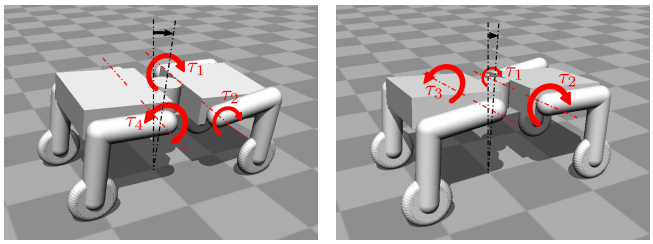
IV. LEARNING PROTOCOL

A. Reinforcement Learning Algorithm

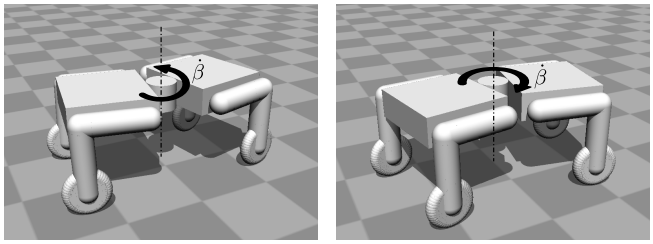
To make the robot learn the proper action to execute according to the situation and to allow it to develop strategies for complex obstacles, we use a Q-learning algorithm.



(a) Modulation of the Torque Distribution ($a = 2$ or 3)



(b) Displacement of the Center of Mass ($a = 4$ or 5)



(c) Rotation of the Steering Pivot ($a = 6$ or 7)

Fig. 2. Overview of the actions dedicated to the crossing situations. Black arrows represent displacements while leg actuation torques are depicted with red arrows whose sizes correspond to the expected relative intensities.

Indeed, the obstacle crossing problem can be expressed as a stochastic Markov decision process. Let s be a vector of $\mathbb{R}^4$ representing the state of the robot, i.e. the situation it is facing according to its means of perception, as it will be described in more detail in section IV-B. Then, let $a \in [\![1, 7]\!]$ be the variable representing the action that can be chosen among the seven possibilities defined in section III. The robot has to select the action a to execute according to its state s : this behavioral mapping is called the policy π. In Q-learning, the decision is conducted by the so-called function $Q : \mathbb{R}^4 \times [\![1, 7]\!] \to \mathbb{R}$, such that the action with the maximal Q-value is picked :

$$\pi(s) = \arg\max_{\mathrm{a}} Q(s, a) \tag{3}$$

The learning of the Q-function is directed by the reward function R, telling at each learning step how good was the previous choice $s \to a$. In our case, we define the reward function by :

$$R = \mathrm{sign}(v_x) \times {v_x}^2 + \max(0, v_z)^2 \tag{4}$$

where v_x and v_z are respectively the horizontal and vertical absolute velocity of the wheel subject to the largest longitudinal force f_x that is opposite to the desired robot velocity. So the reward function focuses on the wheel that slows down

the more the robot in order to let it deal with the current main difficulty with a direct feedback on it. Squares in the formula let v_x and v_z be considered as components of a 2D velocity vector. The sign of v_x is preserved so forward progresses are rewarded while backward moves are penalized. The term $\max(0, v_z)$ allows the climbing movements to be rewarded, but without penalizing downward ones, as they can occur because of a gap in front of the wheel, e.g. after a bump obstacle. As the velocities vary between two learning steps, as well as the elected wheel they come from, the value of R is averaged over this period.

In order to naturally give priority to the regular advance mode, the suitable technique is to slow down the desired robot velocity V_d, e.g. by 70%, when the robot is doing any other action, reserved for crossing situations. This way, the reward defined by (4) is only greater for these actions when the robot is actually blocked by an obstacle and the regular advance mode cannot enable anymore to continue. Also, reducing the wheel speed during obstacle crossing lessen wheel slip, as the robot is retained and cannot go to its full speed anyway.

Then, the Q-function is iteratively built so as to tend to the expected sum of the discounted rewards over time while applying the policy π. This is done by updating at each learning step the Q-value of the last state s and chosen action a that have led to the new state s', while providing the reward R over the transition, as follows :

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[R + \gamma \max_{a'} Q(s',a') - Q(s,a) \right] \tag{5}$$

where $\alpha \in]0,1]$ is the learning rate and $\gamma \in [0,1[$ the discount factor that weighs the importance of distant future rewards compared to the immediate one. As we are in a stochastic case because of the uncertainties on action effects and state transitions, which depend on the ground geometry and characteristics, the learning rate α has to decrease gradually to zero in order to let the Q-function to converge to its probabilistic value [12], i.e. the value given by the Bellman optimality equation. In the same time, π will converge to the optimal policy in accordance with the defined reward function and discount factor.

There is several ways to choose the actions to execute during the learning. Here, we opt for an exploration strategy based on a Boltzmann distribution, i.e. at each step the next action is chosen randomly, but the probability $P_s(a)$ of choosing each action a at state s is determined by :

$$P_s(a) = \frac{e^{\frac{Q(s,a)}{t}}}{\sum_{i=1}^{7} e^{\frac{Q(s,i)}{t}}} \tag{6}$$

where t is a parameter referred as the *temperature* : the larger it is, the more the robot tends to try actions independently of the current best policy.

The period between two updates of the Q-function and choices of a new action has to be chosen wisely in order to give enough time for the actions to have a noticeable effect on the state and/or the reward, without entailing a penalizing reaction time. This period is here set to 300 ms.

B. Continuous State Space Representation

The action selection is guided by the horizontal force distribution on each leg. Indeed, these forces give an internal representation of the robot situation regarding the surrounding obstacles the wheels are in contact with. Let f_x^1, f_x^2, f_x^3 and f_x^4 the horizontal measured forces respectively on the front left, front right, rear left and rear right wheel. These are evaluated according to (1). As the force average accounts for dynamics lengthwise oscillations, whereas we are only interested by the horizontal static balance of the robot, we can use a three-dimensional space for depicting only the relative force distribution while discarding the absolute measures. There are many ways of arranging the four forces in three dimensions with only relative relations. Among the many tested possibilities, the representation that facilitates the most the learning and the decision making is the one expressing the difference of rear leg forces and both side front and rear leg force differences, i.e. $f_x^3 - f_x^4$, $f_x^1 - f_x^3$ and $f_x^2 - f_x^4$.

As the position of the steering pivot changes the leg placements relative to the CoM and thus influences the static equilibrium, the joint angle β is added to the state. This allow the robot to hunt for the best configuration to adopt according to the obstacles.

In order to make the learning more efficient, we adopt a symmetrical representation of the state space and actions with respect to the sagittal plane. Thus, the middle position of the steering pivot defines the symmetry point beyond which the role of left and right legs are reversed to match the symmetrical equivalent with the same deviation but in the opposite direction. With $\beta = 0$ at the middle position, i.e. when the front and rear bodies are aligned and the robot goes straight, the state s can then be defined by :

$$s = \begin{cases} \left[f_x^3 - f_x^4, f_x^1 - f_x^3, f_x^2 - f_x^4, \nu\beta \right] & \textit{if} \quad \beta \geq 0 \\ \left[f_x^4 - f_x^3, f_x^2 - f_x^4, f_x^1 - f_x^3, -\nu\beta \right] & \textit{otherwise} \end{cases} \tag{7}$$

where ν is a dilatation coefficient used to homogenize the scales.

So the state space is actually $\mathbb{R}^3 \times \mathbb{R}^+$ rather than $\mathbb{R}^4$.

The couples of symmetrical actions are also defined according to the direction of the central pivot's deviation. This way, every obstacle geometry learned also prepares for the symmetrical layout where right and left are inverted.

So we need to estimate the Q-value of each action over a continuous state space, what can be done with a local function approximator based on an artificial neural network. We choose to use a normalized radial basis function (NRBF) network because such networks, which are proven to be capable of universal approximation [13], have the particularity to spread the nearest NRBF weight in the unfilled areas or to the boundaries [14]. Then, we build as many NRBF networks, i.e. as many sets of weights w_i, as actions, such that the Q-function is computed by :

$$Q(s,a) = \sum_{i=1}^{N} w_{i,a} \frac{e^{-\frac{\|s-c_i\|^2}{{\sigma_i}^2}}}{\sum_{j=1}^{N} e^{-\frac{\|s-c_j\|^2}{{\sigma_j}^2}}} \quad (8)$$

where N is the number of NRBF covering the state space, σ_i their widths and c_i the center of each one.

All the weights of the last tested action a are then updated at each learning step according to the formula :

$$\forall i \in [\![1, N]\!],\ w_i, a \leftarrow \alpha\epsilon \frac{e^{-\frac{\|s-c_i\|^2}{{\sigma_i}^2}}}{\sum_{j=1}^{N} e^{-\frac{\|s-c_j\|^2}{{\sigma_j}^2}}} \quad (9)$$

where $\epsilon = R + \gamma \max_{a'} Q(s', a') - Q(s, a)$ is the temporal-difference error as expressed in (5).

A NRBF distribution that proves to be efficient in our case is to evenly dispose five gaussian units along each dimension of the state space, symmetrically arranged around zero, while σ is set to half the distance between two units. Both units in positive area and the one at zero are enough for the last dimension that encodes the central pivot angle, as it is always positive. Therefore, the network is comprised of $5 \times 5 \times 5 \times 3 = 375$ gaussian units.

C. Layout of the Training Terrain

The robot is trained on a track which consists of two elemental obstacles, representative of most situations that can be encountered, with the help of the robot reconfiguration during exploration. Obstacle samples are both upright and as tall as the wheel diameter in order to prepare the robot for the hardest cases, so it will be ensured to pass easier ones. As shown on Fig. 3, the first obstacle is a vertical rectangular parallelepiped placed in front of both right wheels of the robot. It prepares the robot for purely unilateral obstacles. It is chosen to be thin enough to let the front wheel reach the ground again before the rear wheel comes in contact with it, because it represents the hardest case for the rear wheel approach according to the static equilibrium. The second one is a vertical step subject to a random rotation around the vertical axis, such that the deviation angle from a frontal approach of the robot is uniformly picked between 0 and $15°$ at each new trial. This way, right and left wheels are subject to a large variety of relative force differences. It is assessed that no more than a $15°$ deviation is necessary for the training, as it would become equivalent to successive unilateral obstacles. Some space is left between both obstacles in order to let the robot recenter its trajectory before reaching the step. As the learning is symmetrical, the robot is trained in the meantime for the reverse situations where left and right are switched.

The training trails are carried by a physical simulations based on *Open Dynamics Engine* (ODE), with a $1\,\mathrm{ms}$ simulation step. We use a pyramidal approximation for the friction model with a unfavorable coefficient of 0.5, which corresponds to a tire on a wet road.

At each trial the robot starts at the same pose and the learning updates begin as soon as it has reached the initial speed of $0.3\,\mathrm{m\,s^{-1}}$. If the robot's CoM deviates more than $0.5\,\mathrm{m}$ sideways, reaches the end of the track, i.e. when the four wheels are on the step, or a maximum time is exceeded, the trial ends.

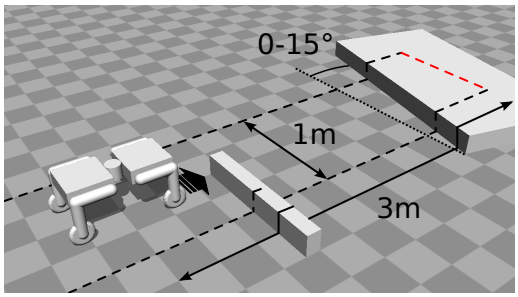


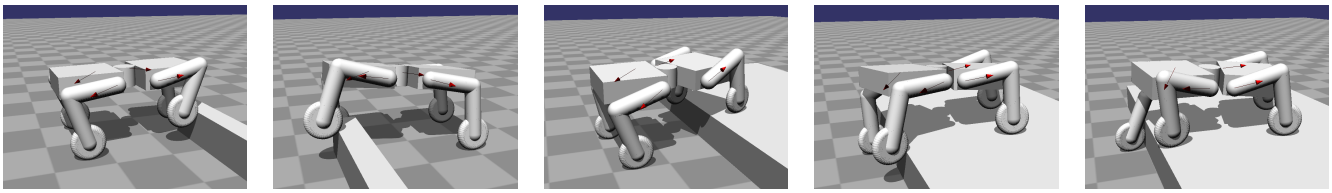
Fig. 3. Track used for the robot training

The discount factor γ is set to 0.7 and the learning rate α starts at 0.5 to decrease gradually while being multiplied by 0.997 between each trial. This last factor that has been chosen to let enough time for the robot to learn the best policy while converging quickly.

V. EVALUATION OF THE LEARNED CAPABILITIES

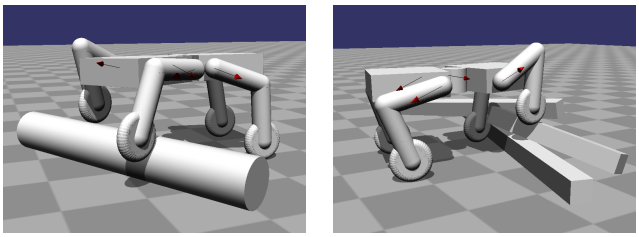
After few hundreds trials on the training track described previously and depicted on Fig. 3, the robot learns an effective policy that allows it to reach the top of the last obstacle for any approach angle. Fig. 4 gives an overview of the resulting policy after 500 trials. In this example, the rotation of the step is set to $0°$, so that the robot comes to it frontally. In Fig.4 and 5, as well as in the associated video, red arrows represent the angles of the motor shafts, so the resulting torques applied by the SEA can be assessed by the deviation between the arrows and the upper leg segments, as both are linked by an equivalent of angular springs. On Fig. 4, we first observe that the modulation of torque distribution suffices to make the front wheels cross a unilateral obstacle. However, the rear wheels need the displacement of the CoM to counteract the effect of wheel torques on the static equilibrium. Then, when dealing with obstacles spread over multiple wheels, the robot has to successively use the rotation of the steering pivot to the right and to the left, interspersed by the other actions, in order to sequence the wheel crossings.

The resulting policy is also tested in parallel on four other obstacles. The first three consist of a large cylinder laid on the ground, like a fallen tree trunk, which diameter is chosen as large as the wheel diameter. The cylinder angle relative to the robot's advance direction is set to several values, so that the robot has the different approach angles of 10, 20 and $30°$. The last obstacle is made up by a heap of three beams with a rectangular cross section as large as the wheel radius and heavy enough not to move on the robot passage. This gives an example of complex obstacle requiring more adaptation steps in order to be climbed. The resulting gaits for every obstacle can be appreciated in the related video, which can also be found at `http://www.isir.upmc.fr/vid/learning_obstacle_negotiation.mp4`.



(a) (b) (c) (d) (e)

Fig. 4. Crossing steps on the training track after 500 trials



(a) Crosswise cylinder (b) Beam heap

Fig. 5. Other tested obstacles

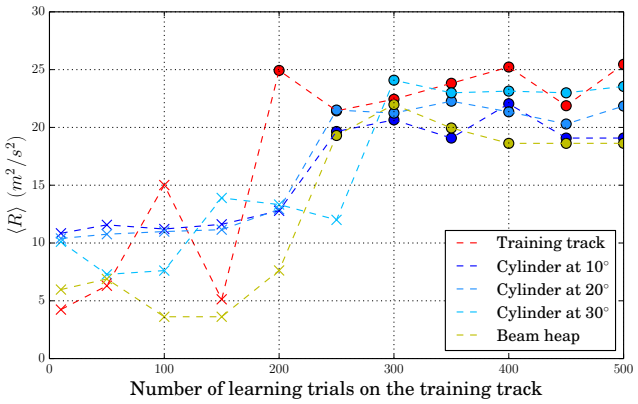


Fig. 6. Progress of the average reward got by the robot from various obstacle crossings. The learning trials are performed on the training track and the resulting *greedy* policy, i.e. without exploration, is then applied on every obstacles at different learning stages. Circles indicate that the robot succeed in reaching the other side of the obstacle, while crosses stand for failures, i.e. when the robot got stuck.

Fig. 6 shows the average reward got by the robot while it tries to cross the different obstacles in a finite amount of time. Even if the robot has only learned from the training track, it becomes able to cross unseen obstacles, with wider approach angles or requiring more complex crossing procedures. 500 trials are enough to learn an efficient policy, what correspond to approximately 50 minutes of computation using a single core on a modern computer.

VI. CONCLUSIONS

In this work, we described the application of Q-learning for a compliant wheel-on-leg robot in order to produce a generic policy for all obstacles. Results prove that the robot can successfully negotiate complex obstacles without neither having any prior knowledge on their geometry, nor having encountered such shapes before. The robot is then free to roll faster without requiring detailed map of the environment. In future works, we will consider the application of the learned policy on the prototype.

ACKNOWLEDGMENT

The author thanks Anis Najar[1,2] and Ryan Lober[1,2] for their valuable advices.

REFERENCES

[1] T. Thueer, A. Krebs, and R. Siegwart, "Comprehensive locomotion performance evaluation of all-terrain robots," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2006, pp. 4260–4265.

[2] K. Iagnemma, A. Rzepniewski, S. Dubowsky, and P. Schenker, "Control of robotic vehicles with actively articulated suspensions in rough terrain," *Autonomous Robots*, vol. 14, no. 1, pp. 5–16, 2003.

[3] D. Wettergreen, S. Moreland, K. Skonieczny, D. Jonak, D. Kohanbash, and J. Teza, "Design and field experimentation of a prototype lunar prospector," *The International Journal of Robotics Research*, 2010.

[4] B. H. Wilcox, T. Litwin, J. Biesiadecki, J. Matthews, M. Heverly, J. Morrison, J. Townsend, N. Ahmad, A. Sirota, and B. Cooper, "Athlete: A cargo handling and manipulation robot for the moon," *Journal of Field Robotics*, vol. 24, no. 5, pp. 421–434, 2007.

[5] M. Tarokh, H. D. Ho, and A. Bouloubasis, "Systematic kinematics analysis and balance control of high mobility rovers over rough terrain," *Robotics and Autonomous Systems*, vol. 61, no. 1, pp. 13–24, 2013.

[6] W. Reid, S. Sukkarieh, *et al.*, "Actively articulated suspension for a wheel-on-leg rover operating on a martian analog surface," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 5596–5602.

[7] M. Sato, A. Kanda, and K. Ishii, "An environmental adaptive control system of a wheel type mobile robot for the rough terrain movement," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2007, pp. 3962–3967.

[8] S. Koos and J.-B. Mouret, "Online discovery of locomotion modes for wheel-legged hybrid robots: A transferability-based approach," in *Proceedings of CLAWAR*, 2011, pp. 70–77.

[9] A. Bouton, C. Grand, and B. Faïz, "Design of a compliant wheel-legged robot," in *CLAWAR 19th International Conference on Climbing and Walking Robots and Support Technologies for Mobile Machines*, 2016.

[10] J.-C. Fauroux, F. Chapelle, and B. Bouzgarrou, "A new principle for climbing wheeled robots: Serpentine climbing with the open wheel platform," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2006, pp. 3405–3410.

[11] A. Bouton, C. Grand, and B. Faïz, "Motion control of a compliant wheel-leg robot for rough terrains crossing," in *Robotics and Automation (ICRA), IEEE Int. Conf. on*, 2016.

[12] C. J. C. H. Watkins, "Learning from delayed rewards," Ph.D. dissertation, University of Cambridge England, 1989.

[13] M. Benaim, "On functional approximation with normalized gaussian units," *Neural Computation*, vol. 6, no. 2, pp. 319–333, 1994.

[14] R. M. Kretchmar and C. W. Anderson, "Comparison of cmacs and radial basis functions for local function approximators in reinforcement learning," in *Neural Networks, 1997., International Conference on*, vol. 2. IEEE, 1997, pp. 834–837.