Attitude control in the Mini Cheetah robot via MPC and reward-based feed-forward controller

Paolo Arena, Fabio Di Pietro, Alessia Li Noce [*]
Luca Patanè [**]

[*] *Università degli studi di Catania*
Catania, Italy (e-mail: paolo.arena@unict.it).
[**] *Università degli studi di Messina,*
Messina, Italy
(e-mail: lpatane@unime.it)

Abstract: In this paper, a model-free approach, based on a generalization of unsupervised Self Organizing Feature Maps, is introduced to compensate for attitude control errors in a simulated mini cheetah quadruped platform. Traditional techniques mainly exploit the potentialities of convex model predictive control (MPC) to efficiently regulate the robot attitude while moving in unstructured environments. However, they are based on the knowledge of the analytical model of the robot. If the robot structure undergoes significant modifications due, for example to an unbalancing of the robot weight caused by a load charged off the robot center of mass, the added value of a model-free, adaptive unsupervised nonlinear controller, acting as a feed-forward error compensator, shows its real effect when integrated with the model-based approach. Moreover, the proposed solution, acting as a self-learning feedforward controller, contributes to the control action only when needed, preserving the basic performance of the ongoing MPC action. The design of the control scheme and simulation results will be reported, showing the impact of the introduced model-free compensation on the quadruped robot locomotion.

Keywords: Legged robots; unsupervised nonlinear control; reward-based learning; feed-forward control; MPC.

1. INTRODUCTION

Legged robot locomotion control is a really challenging task that deserves to be fully investigated, especially when unstructured environments need to be explored (Arena et al., 2021). In literature, legged locomotion control is designed following a low-level and a high-level approach. The former imposes specific periodic trajectories to the robot legs, as well as specific phase shifts among them, to produce specific pre-assigned locomotion gaits; the latter is in charge of maintaining other references, such as attitude signals. These, if in the case of hexapods are feasible (Arena et al., 2002b), in the case of quadrupeds and bipeds attitude compensation is strictly needed. Under this perspective, several approaches are presented in the literature, but the most promising, and already working finely, are based on the fast computation of simplified linear models of the robot dynamics, exploited to predict, in a finite time horizon, the system outputs, subject to control actions to be optimised in order to minimize a prescribed performance index. This strategy, which goes under the name of model predictive control (MPC), known since the late 80s, is nowadays acquiring an ever increasing interest due to the availability of low-cost high-performance computing hardware allowing real-time onboard processing (Mayne, 2014). Moreover, it was demonstrated that the approximation of the complex nonlinear dynamics of quadruped robots through linear time-varying models is able to guarantee suitable results in locomotion and balance control in experiments involving running quadrupeds on flat terrains. However, as with all model-based control techniques, the performance of the control action heavily relies on the feasibility of the prediction model. For this reason, in the literature, alternative strategies based uniquely on model-free, data-driven nonlinear structures, such as neural networks, were introduced into the control loop. The capabilities of universal approximation, typical of neural networks, can be exploited to have a reliable prediction of the system outputs only on the basis of experimental data. Some examples of this approach refer to reinforcement learning techniques (Yue, 2020), for improving the capability of attitude maintenance while moving in unstructured environments (Sun et al., 2021). The main weakness of this approach is that, in principle, there is no guarantee of preserving stability in the control loop. Moreover, the knowledge of the system dynamics is completely discarded in the controller design. So the primary motivation of our approach is to maintain the suitability of a linear MPC and to add compensation in case of unexpected disturbances. In fact, a mixed approach is introduced, which takes into account the suitable results of linear MPC for maintaining locomotion and balance in nominal conditions, maintaining a prescribed attitude signal by correcting the feedback error, as recorded by the onboard inertial sensors (García et al., 2021). In such cases when MPC reveals unavoidable weaknesses, for example, in case of unexpected load, an additional nonlinear controller is added, in feedforward with the MPC block. The controller, which belongs to the family of nonlinear self-organising feature maps (Barreto et al., 2003), called Motor Map Controller (MMC) provides outputs (i.e., force signals) in addition to the control signals produced by the MPC. In this way, the MMC action is provided when needed. In the examples reported in this work, disturbances are introduced through an additional load in one of the legs; this unbalances the overall dynamical structure, affecting the reliability of the robot model. To the best of our knowledge, such mixed control scheme, aimed at optimising the MPC performance in front of typically real life disturbances (i.e. additional loads) in a quadrupedal structure has not yet been faced with. It has to be underlined that the MMC approach, belonging to the family of model-free strategies, can be easily generalized to different robotic structures, constituting an additional block designed to overcome disturbances and compensate for the nonlinearities discarded in the design phase. The remaining of this paper is organized as follows: Section 2 introduces the Motor Maps, Section 3 describes the designed control architecture and the simulation framework, Section 4 presents the simulation results and finally the conclusions are drawn in Section 5.

2. MOTOR MAPS: MATHEMATICAL FORMULATION

Motor Maps (MMs) are neural structures proposed as an upgrade of the more traditional Kohonen self-organising feature maps (Kohonen et al., 2001). These are inspired by the topology-preserving maps identified in the brain, which encode the representation of sensory input signals into space-organised responding units; these, in turn, elicit an action in response to a given input stimulus. Kohonen networks formalize a self-organizing process that, at the end of the learning phase, generates a topographic map where neighbouring neurons are excited by inputs possessing similar features. MMs expand the capabilities of Kohonen networks by adding the possibility to associate to a specific active neuron a required action to be executed (Ritter et al., 1992). To realise such a task, Kohonen networks are augmented with an additional output layer that hosts a weight vector specific to each neuron site. Therefore, the overall network consists of two different layers: the first one is a traditional Kohonen layer, which hosts input weights, whereas the other generates the output signal. Typically, during the learning phase, both the input and the output weights are updated. MMs are perfect candidates to act as adaptive self-organizing controllers. These networks were successfully adopted in a variety of applications, including legged robot locomotion control (Arena et al., 2004), chaotic systems control (Arena et al., 2002a) and as a fundamental block in modelling the perception-action loop (Arena et al., 2009).

Formally, an MM can be defined as an array of neurons mapping the space V of the input patterns onto the space U of the output actions:

$$\Phi : V \longrightarrow U \tag{1}$$

The basic structure of the Motor Map is presented in Fig.1.

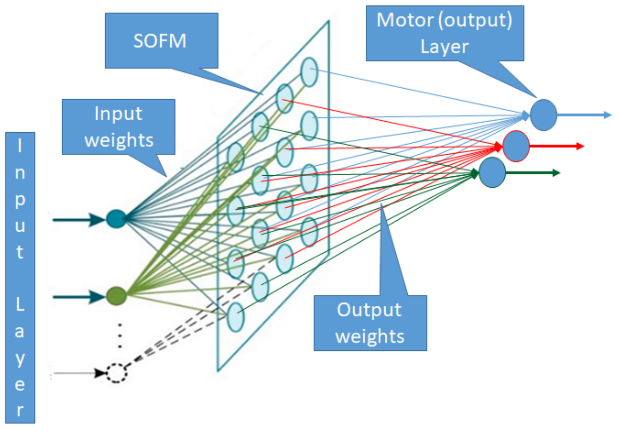


Fig. 1. Motor Maps basic structure: the SOM layer is connected with the input layer and the output layer through weights that are learned with an unsupervised clustering mechanism (i.e., input weights) and a reward-based strategy (i.e., output weights).

It is basically a very simple architecture, made up of three layers: the input and the output layers and a hidden layer with spatially organized neurons. The hidden layer, the so-called self-organizing feature map (SOFM), contains neurons encoding common features of the input signals, spatially organized to create clusters. The output layer provides the control action as a function of the SOFM neuron activation. The learning algorithm is the key to obtaining a suitable spatial arrangement of both the input and output weights of the map, which have the role of embedding the dynamic aspects of the neural controller. This is achieved by considering a learning rule which extends the capabilities of the classical winner-take-all algorithm. Upon the presentation of an input pattern, the neuron that best matches the input pattern is elected as the winner. Then, a weight update is realised not only for the winning neuron but also for those neurons belonging to a neighbourhood of a given radius. In the case of unsupervised learning, no a priori information on the appropriate control action is available. The algorithm has to autonomously find the correct control action exploiting the input pattern information and the performance index to be optimised, under the guidance of the so-called reward function. Output weight update is realised only if the applied control improves the reward function, in relation to the system being controlled. It is clear that the most critical aspect relies on the selection of the reward function. As mentioned above, the MM is introduced to control the robot balance. This is achieved following the scheme presented below in Fig.2, by acting incrementally on the leg joint torque control signals. Restricting, for simplicity only to the pitch (sagittal) plane, the control objective is to maintain the pitch angle of the robot as much as possible next to a reference angle. So the reward function takes into account the error between the reference and the actual pitch of the robot.

The most important aspect is the self-learning, model-free characteristic of the MMC. It is able to learn the right control law on the basis of the reward function to be maximized. Its definition for the proposed application is here reported:

$$Reward = -(\Theta - \Theta_{ref})^2 - \frac{(\dot{\Theta} - \dot{\Theta}_{ref})^2}{(\Theta - \Theta_{ref})^4 + 1}\beta \quad (2)$$

where Θ_{ref}, Θ, $\dot{\Theta}_{ref}$, $\dot{\Theta}$ are the reference and actual pitch, and the reference and actual pitch speed of the robot, respectively. The gain $\beta \in [0,\ 1]$ and the denominator in the pitch derivative part of the Reward are introduced to establish a priority of the pitch speed on the pitch position error: as the pitch error decreases, more relevance is given to the pitch speed error; moreover, the smaller the β values the smaller the range of positions in which the speed correction has an impact on the reward. $\beta = 0.001$ was selected as a suitable compromise in the simulation performed below.

2.1 The learning algorithm

In this paper, with the aim of autonomously learning a nonlinear balance controller for the mini cheetah robot, the unsupervised version of the MMC will be considered. The steps adopted to design the controller and apply the learning algorithm are reported in the following. A schematic flow chart is reported in Fig.2.

- The network topology and the hyperparameters are defined, typically adopting a grid search and a heuristic strategy. The input and output weights are randomly initialized;
- an input pattern is presented at time t ($I(t)$) and, using the Euclidean distance metrics, the winner neuron N_s is selected which minimises the distance between $I(t)$ and the input weight vector linking N_s to the input nodes;
- for that winner neuron N_s, the corresponding output weight $w_{N_s,out}$ is used to produce the control action. More in details, the disturbed version $O(t) = w_{N_s,out} + \alpha_s \lambda$ is adopted, adding to the winner neuron weight $w_{N_s,out}$ the Gaussian, zero-mean random variable λ, multiplied by the gain α_s associated to N_s. This α_s, starting from an initial value, and fixed a final value, is linearly decreased between these two values as the learning phase proceeds with a defined slope η_s;
- the actual Reward function $R(t)$ is evaluated together with the reward increment $\Delta R = R(t) - R(t-1)$. If $\Delta R > b_s$, where b_s is an average increase associated to the neuron N_s. The input and output weights connected to the winner neuron N_s and its neighbouring ones are updated according to the rule:

$$w_{i,in}(t+1) = w_{i,in}(t) + \eta_{in}\zeta(I(t) - w_{i,in}(t)) \quad (3)$$

$$w_{i,out}(t+1) = w_{i,out}(t) + \eta_{out}\zeta(O(t) - w_{i,out}(t)) \quad (4)$$

and b_s is updated as follows:

$$b_s^{new} = b_s^{old} + \sigma(\Delta R - b_s^{old}) \quad (5)$$

where $\sigma > 0$ and η_{in}, η_{out} are the input and output learning rates finally, $I(t), O(t), w_{i,in}, w_{i,out}$ are the input pattern, the output control signal and the input and output weights, respectively. For each neuron N, ζ is the neighborhood function: if $\mathcal{N}_{\mathcal{R}}$ represents the neighbour of radius $\mathcal{R}$, around the current winner neuron N_s, ζ is a binary value:

$$\zeta = 1 \iff N \in \mathcal{N}_{\mathcal{R}}; \quad (6)$$

So ζ establishes the radius, around N_s, within which the weight update takes place. Of course, larger ζ values, typically taken at the beginning of the learning phase, guarantee that most of the neurons undergo adaptation. As learning proceeds, ζ values decrease towards the unit value. When learning is always active, the reward evaluation and weight adaptation are always repeated. The learning rate η is a function of the learning epoch: during the starting phase (i.e., when a new reference pitch angle is considered) a high learning rate guarantees rapid learning, while subsequently, its decrements guarantee a good trade-off between memory and innovation capabilities.

The MM controller is included in the overall control loop, as specified in Section 3. In particular, within each single learning iteration, an input pattern is presented, the winning neuron N_s is selected and the output signal is provided. Moreover, in a window $dt = 0.01s$, the Reward is evaluated and the learning phase is performed, i.e. $w_{i,in}, w_{i,out}, b_s, \alpha_s$, are updated. After 10 iterations an update of the other learning hyperparameters (i.e., learning rates, Neighbour radius, etc.) is performed. Table 1 summarises the hyperparameters used in the simulations.

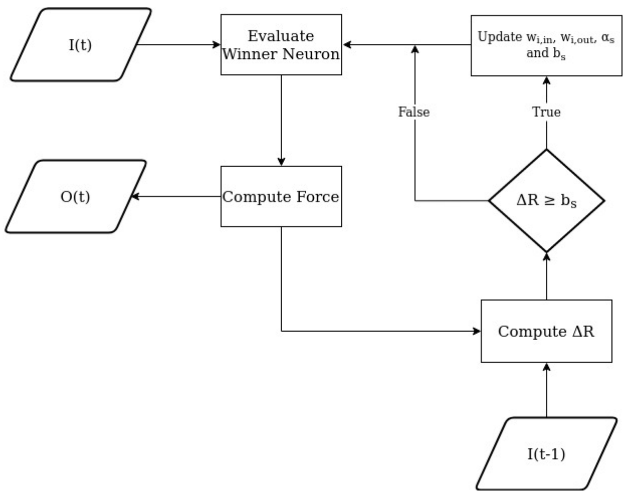


Fig. 2. Flowchart of the control loop for each i-th learning iteration. Each 10 iterations all the other parameters, besides $w_{i,in}, w_{i,out}, b_s, \alpha_s$, are updated.

Table 1. Motor Map learning parameters

Input learning rate η_{in}	1
Output Learning rate η_{out}	0.8
Learning explorative variable η_s	0.05
Maximum Neighborhood radius	10
Learning epochs	1000
dt	0.01
σ	0.8
α_s	(1, 1, 5)

3. THE MINI CHEETAH ROBOT CONTROL SYSTEM AND SIMULATION FRAMEWORK

The quadruped robot taken into account to evaluate the performance of the proposed controller is an MIT Mini Cheetah-like robot, shown in Fig.3. It is a torque-controlled four-legged robot developed at the MIT Biomimetic Robotics Lab (Katz et al., 2019).

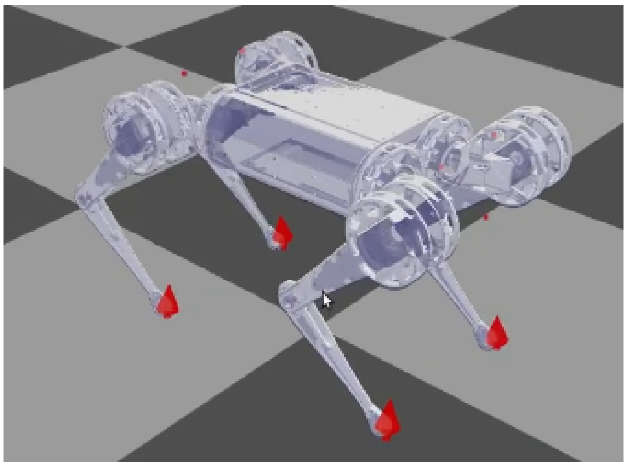
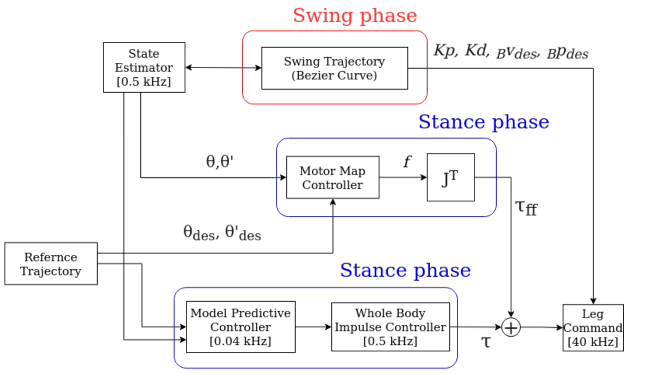


Fig. 3. The Mini Cheetah simulated robot

The real Mini Cheetah robot has a structure similar to the MIT Cheetah 3 robot (Bledt et al., 2018). The two models differ only in characteristics such as mass, length and motor gear ratio. The Mini Cheetah is roughly 60% the length of Cheetah 3; it weighs is 9 kg, with 0.21m and 0.18m lengths for the upper and lower links, respectively. The body length is about 0.38m. Both robots use identical, self-contained actuator modules at each degree of freedom (DoF) that incorporates together an electric motor, a single-stage planetary transmission and power electronics. The four identical legs are designed to maximise motion ranges, and allow the robot to work identically forward, backward or even upside down. The three degrees of freedom of the legs are controlled by three actuators that rotate the hip joint by a permitted angle of $\pm 120°$, the knee joint about $\pm 155°$ and ab/ad joint over $\pm 120°$. The actuator driving the knee joint is placed coaxially to the hip actuator, thus minimising the inertia of the leg with respect to the body. Torque is transmitted to the knee joint through a gates poly chain belt transmission which passes through the hollow upper link of the leg and provides an additional $1.55 : 1$ gear-up (Katz et al., 2019). For the simulations performed in this paper we used an accurate dynamic model of the Mini Cheetah consisting of 25 rigid bodies with 12 actuated DoF, 6 unactuated DoF and 12 constraints. The 25 rigid bodies consist of the robot main body (a rigid box floating base), 4 abduction/adduction (ab/ad) links, 4 hip links, 4 knee links, and 12 actuator rotors, that represent the actuated DoF. The 6 unactuated DoF are the base position and orientation. The dynamic model is described by a total of 37 state variables.

3.1 *The control scheme*

For controlling the pitch angle of the simulated Mini Cheetah robot, the pitch error and its derivative are considered input signals for the MM controller. Therefore, the MMC has two input units and twelve outputs, corresponding to the three force components along the three orthogonal axes $f_{i,x}, f_{i,y}, f_{i,z}$ for each actuated leg ($i = [1, \cdots, 4]$). Fig.4 shows a block scheme of the control loop. As depicted in Fig.4, the MM controller is added, as an optimization block, on the underlying traditional controller. In particular, the leg control commands are generated in two different ways, according to the (stance or swing) phase of


Swing phase
State Estimator [0.5 kHz]
Swing Trajectory (Bezier Curve)
Kp, Kd, $_Bv_{des}$, $_Bp_{des}$
Stance phase
θ,θ'
Motor Map Controller
f
J^T
τ_{ff}
θ_{des}, θ'_{des}
Refernce Trajectory
Stance phase
Model Predictive Controller [0.04 kHz]
Whole Body Impulse Controller [0.5 kHz]
τ
Leg Command [40 kHz]
Fig. 4. The Mini Cheetah control scheme.

the single leg. During the swing phase, the leg trajectory is simply planned using an interpolation of a series of waypoints using a Bezier curve approach (Saputra et al., 2016). The stance phase is particularly relevant for stability and attitude control. This is realised traditionally by a model predictive linear controller (MPC) with the addition of a whole-body impulse controller (WBIC) (Kim et al., 2019; García et al., 2021). Here, the MPC computes the forces needed to maintain the body in a standing position, optimised on a $40Hz$ window; then these forces are used as input signals for a faster ($500Hz$) WBIC controller, which provides the stabilizing joint torques. In this loop and during the stance phase, the MMC provides an additional nonlinear feedforward control action, in terms of forces, to be added to the underlying leg joint controller, after being transformed in joint torques $\tau_{i,ff}$ through the Jacobian matrix.

In this way, the MMC is able to autonomously learn the nonlinear rules that link the reference pitch trajectory to the residual joint torques to refine the overall robot balance control. It has to be underlined that, in absence of any disturbance, the definition of the reward function makes the MMC action almost zero. Only when the error increases, the MMC contribution is evident with a nonlinear feedforward compensation.

3.2 *Experiment issue and setup*

The role of the MMC is to filter out potential disturbances that can affect the robot performance and cannot be taken into account in the model used in the MPC. In particular, we simulated the case in which the robot undergoes a weight variation in the anterior legs, whose mass is increased twice with respect to the posterior ones. This implies a variation of the center of mass position and an unbalancing of the whole robot. In this case, the positive role of the added MM control will be compared and evaluated on the linear model based MPC-WBIC control. The selected gait used within the simulations was the trot which, due to its symmetry and stability characteristics, is a perfect candidate to appreciate pitch variations. The results are shown in the Section 4.

4. SIMULATION RESULTS

The learning procedure of the MMC was applied following the reward-based learning procedure described in Section

2, using the data acquired from a dynamic simulation of the Mini Cheetah robot while it is trotting on the spot.

The optimization of the MMC in terms of the number of neural units was performed using a grid search. The best MMC structure able to provide a good trade-off between size and accuracy was selected considering 900 neurons arranged in a $30x30$ 2D lattice. Fig.5 reports the internal distribution of the weights within the Kohonen layer at the end of 1000 learning epochs. The pitch of the robot body and its derivative, considered as inputs to the MMC, are reported in the axes. The blue circle represents, as an example, the winning neuron, whenever the inputs fall within the neighbours of $\theta = 2.5°$ and $\dot{\Theta} = -5°/s$. It can be seen that the network organises in a range covering the poses typically spanned by this robot walking configuration. In fact, since the weight increase is concentrated on the front side, the pose of the robot tends to lean down frontally (i.e. towards positive pitch angle values). As a result, the Kohonen layer has to compensate for positive pitch errors. In addition, the organization of the Kohonen layer clearly emphasises the compensation action around small pitch angles: inputs outside the range $\Theta_{minmax} = [-1; 3.5], \dot{\Theta}_{minmax}[-25; 5]$ generate forces so as to make the following input to fall within the interval $[\Theta_{minmax}, \dot{\Theta}_{minmax}]$, making the MMC able to control the robot towards the reference signals.

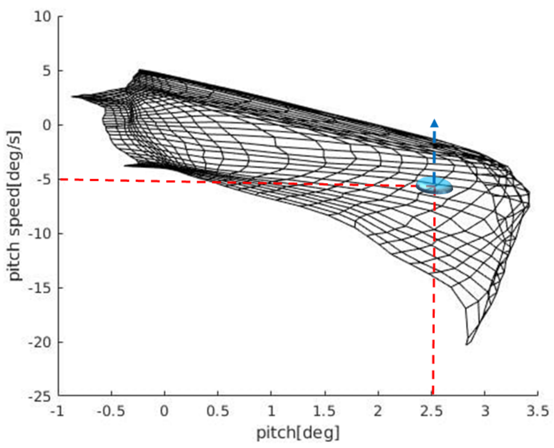


Fig. 5. The Kononen layer distribution at the end of a typical learning phase. A winning neuron is represented as an example, in front of inputs falling within a neighbour of the dotted lines.

Fig.6 depicts the MMC outputs, i.e. the force modules for each of the 4 legs. What has to be underlined is the highly symmetrical shape of the force modules: these are almost equal for both the anterior and rear legs, enhancing a highly balanced control action.

In order to assess the reliability of the results, being this approach based on unsupervised learning, a statistical analysis was performed, by running 10 different MMCs with a random weight initialization. The results obtained are depicted through the analysis of the Reward function, reported in Fig.7. In detail, the figure reports the mean value (i.e., yellow trace) and the variance values (i.e., black trace) of the reward function over 10 learning trials. It clearly appears that the reward values, after an initial fluctuation, tend to a mean value around $R = -0.5$ and an average variance $\sigma = 1.3$. This confirms the suitability of network convergence.

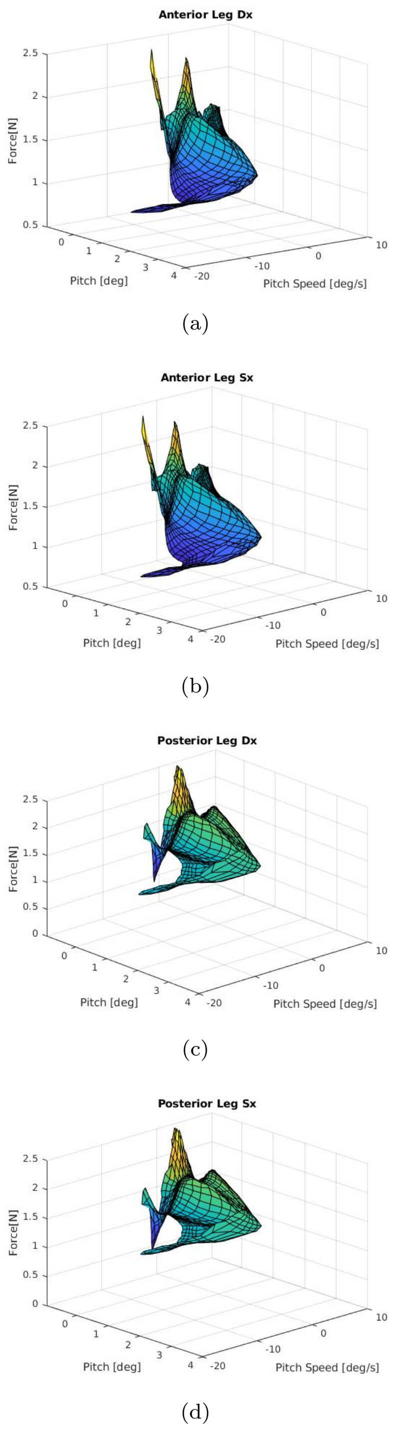


Fig. 6. Motor Map output, at the end of the learning phase as a function of the inputs: force module acting on the (a) front right leg, (b) front left leg, (c) rear right leg, (d) rear left leg.

Fig. 8 reports the time convergence of the body pitch angle for one of the 10 MMCs trained before. The pitch angle tends to zero, although showing a slightly positive bias towards the front down position around 0.5°.

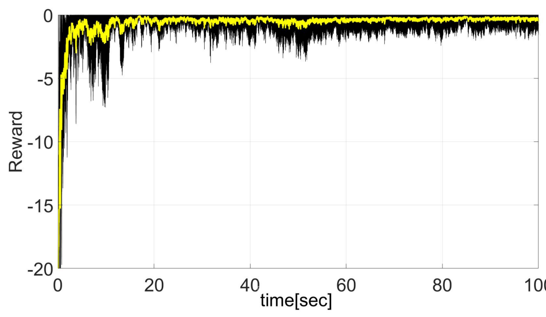


Fig. 7. The mean and variance of the Reward function were evaluated over 10 different MMC trials.

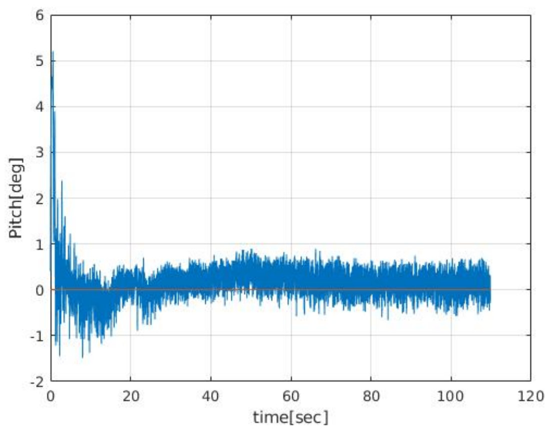


Fig. 8. Time convergence of the pitch angle obtained with the MMC.

The role of the MMC over the traditional one (MPC-WBIC) is well represented in Fig.9. Differently from the learning phase, where the network runs at $100Hz$, the testing phase is performed by applying the MMC at a frequency of $10KHz$. In fact, the MMC, at the end of the learning phase, involves only static processing, and so can be applied even in several steps within each WBIC iteration, which, in this application, runs at $0.5KHz$. Referring to Fig.9, in the starting phase only the traditional MPC-WBC is active and the robot shows a steady-state pitch error of about $4.5°$ conspicuously leaning front down. As soon as the MMC action is switched on, (after about $5s$) the pitch error suddenly decreases, settling at an average value of about $-0.1°$. Several successive similar on-off trials are reported in the figure, showing consistent behaviors. The suitability of the proposed structure is therefore evident.

An additional testing phase was performed, leaving the robot to move forward for 12s in a flat terrain with a trot gait. In this case, the weight of the front legs was increased by two times with respect to the default value: from $1,238Kg$ to $2,476Kg$. Being the additional weight increase equally balanced on the forward side, the reference strait trajectory is maintained and the robot is able to navigate at a speed of around $1m/s$. The relevant aspect is that the pitch balance improvements are preserved also during walking, although the MMC is being trained in standing conditions (i.e., trot on the spot). This can be appreciated in Fig.10 (a), where the pitch of the moving robot is reported. The forward locomotion causes an added

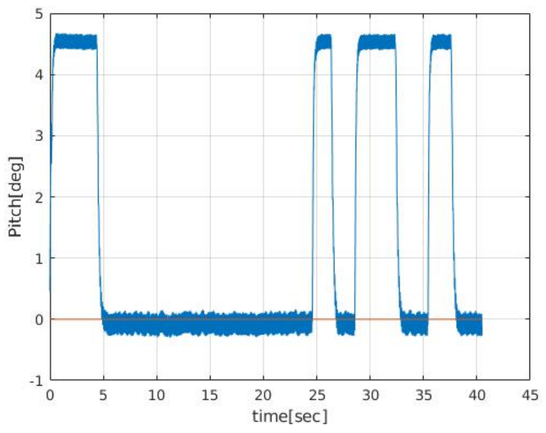
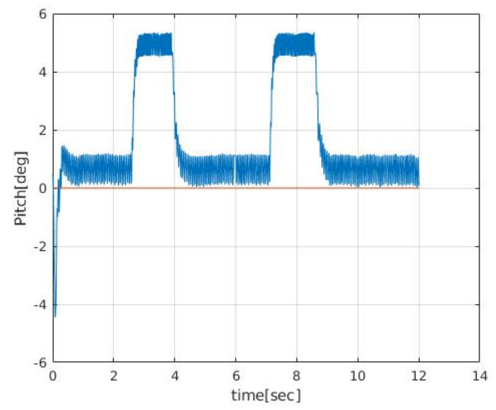


Fig. 9. The testing phase of the MM controller runs within the control loop at $10kHz$. The MMC is turned on for the first time at about 5s and then is turned on/off other three times.

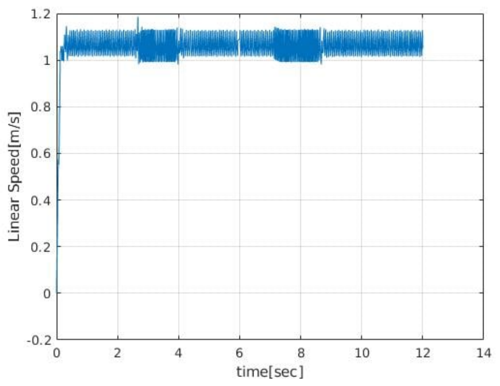
bias in the pitch error, so the robot once again tends to lean forward-down (positive pitch angles). Fig.10 (a) reports the alternative turning on and off of the MMC action. In particular, when the MM controller is active, for example in the interval $[0; 2.5s]$ the body pitch settles at about $+0.7 \pm 0.5°$, whereas, in the following $1s$ of simulation, the absence of the MM control leads the pitch error to attain values around $+5 \pm 0.4°$, much higher than with MMC. The overall robot behaviour is therefore significantly improved through the application of the MM controller, already learned only in standing conditions. Fig.10(b) reports the forward speed of the robot for the previous simulation. The trend of the robot forward velocity is a typical signal of a legged robot, showing an average value surrounded by a series of oscillations at a frequency related to the stepping pattern. It can be appreciated that the MMC does not cause potential destabilization. The picture testifies that even a sudden introduction of the MMC does not negatively affect the behavior of the robot. To show in action the role of the MM controller, in Fig.10(c) the trotting behavior of the robot with (left side) and without (right side) the MMC action is reported. The robot, in the latter condition, clearly leans forward down under the additional weight. It is to be underlined that we are controlling only along the sagittal plane, so, on average, the trajectory of the robot is not altered. In case of an asymmetrical load disturbance, which is possible if the robot has to hold additional unbalanced weight, the trajectory could be not retained, since a drift in the lateral direction is to be expected. To further evaluate the MMC capabilities, a different scenario including alternating slopes was considered. The MMC was able to learn the proper input and output weights to deal with this complex terrain soon after the first attempt (i.e., time interval [0s, 20s]) as reported in Fig. 11.

5. CONCLUSIONS

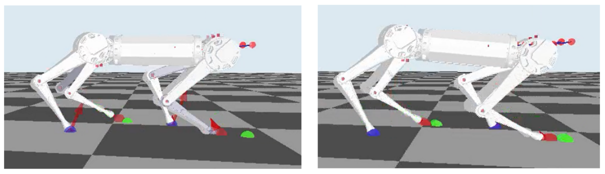
In this paper, a motor-map-based nonlinear controller is introduced with the aim to maintain pitch balance in a simulated quadruped robot undergoing body-weight disturbances which affect the efficiency of a more traditional body balance control based on a linear MPC-WBC con-



(a)



(b)



(c)

Fig. 10. Effect of the MMC on the pitch control while the robot is trotting: (a) pitch error; (b) forward velocity; (c) snapshots showing the robot attitude in the sagittal plane when the MMC is active (left panel) and disabled (right panel). The control signal is activated at 0s, around 4s and after 8s.

troller. The characteristics and advantages of the proposed controller are outlined below:

- The MMC is a nonlinear self-learning controller, uniquely based on measurements, without needing apriori knowledge of the robot physical characteristics, for the generation of the control law. In this paper, this advantage is exploited together with those deriving from a traditional controller, based on a simplified robot model.
- Once the MMC is trained, the controller does not need internal iterations, since a static map is created. This generates a real-time, fast control action, suitable to be applied at different frequencies, as shown in the paper.

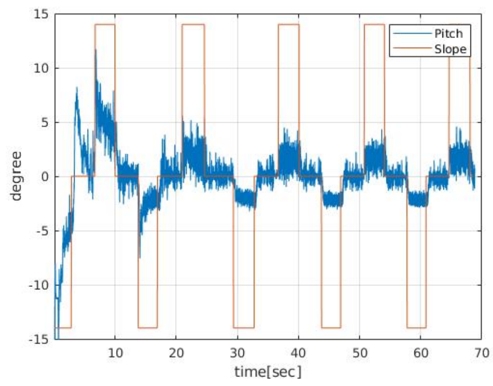


Fig. 11. Time evolution of the robot pitch while walking on a complex terrain characterized by alternating slope changes ($\pm 14°$).

- Being a model-free approach, the MMC opens the possibility to be easily transferred to other robotic platforms.
- The nature of the Self Organising Feature maps embeds a nonlinear topographic representation of the system, for the purpose of torque compensation. Suitable processing of the Kononen layer could be formulated to explain the control action and generate an alternative "linguistic-like" controller representation, along with the recent concept of "explainable AI".
- The feedforward MMC action, injecting an additional torque signal focalised to compensate for disturbances, preserves the reliability of the underlying MPC-WBIC controller, previously designed. The control action is therefore oriented to further optimise the balance performance. This is in contrast with other fully data-driven approaches (i.e. based on Reinforcement learning), that on the one hand discard the model (and related controller) available, and on the other hand, are particularly time-consuming and computationally heavier than our structure.
- The MMC unsupervised learning scheme is particularly suited in this case of study since if a supervised learning-based structure would be adopted, the target signal (i.e. the residual reaction force to be added to the controller in front of each specific error) is not easy to be calculated, to build the training set.
- The most critical aspect to be taken into account is the Reward function definition, which is crucial for the MMC success. This has to be defined after a series of trials, together with the typical neural network parameters (neuron number, learning rate, etc.)
- The structure, as demonstrated in the simulation results, thanks to the simplicity of the learning algorithms, has the capability to fast reach a maximization of the reward function after the suitable mapping of the input signals. This give the opportunity to implement a never ending learning, which means that a residual capability of weight plasticity is allowed to the structure, in order to react to further modifications of the robot model, such as a further weight variation. This leads to a a time-varying control law which adapts to environment disturbances.

In this paper, only pitch control is reported. The results obtained deserve further investigation and generalization of the whole balance problem.

REFERENCES

Arena, P., Di Pietro, F., Li Noce, A., Taffara, S., and Patane, L. (2021). Assessment of navigation capabilities of mini cheetah robot for monitoring of landslide terrains. 540–545. doi:10.1109/RTSI50628.2021.9597335.

Arena, P., Fortuna, L., and Frasca, M. (2002a). Chaos control by using motor maps. *Chaos*, 12(3), 559–573.

Arena, P., Fortuna, L., Frasca, M., and Sicurella, G. (2004). An adaptive, self-organizing dynamical system for hierarchical control of bio-inspired locomotion. *Trans. Sys. Man Cyber. Part B*, 34(4), 1823–1837. doi:10.1109/TSMCB.2004.828593.

Arena, P., De Fiore, S., and Patanè, L. (2009). Cellular nonlinear networks for the emergence of perceptual states: Application to robot navigation control. *Neural Networks*, 22(5), 801–811. doi:https://doi.org/10.1016/j.neunet.2009.06.024.

Arena, P., Fortuna, L., and Frasca, M. (2002b). Attitude control in walking hexapod robots: an analogic spatio-temporal approach. *International Journal of Circuit Theory and Applications*, 30(2-3), 349–362. doi:https://doi.org/10.1002/cta.203.

Barreto, G., Arajo, A., and Ritter, H. (2003). Self-organizing feature maps for modeling and control of robotic manipulators. *Journal of Intelligent and Robotic Systems*, 36, 407–450. doi:10.1023/A:1023641801514.

Bledt, G., Powell, M.J., Katz, B., Di Carlo, J., Wensing, P.M., and Kim, S. (2018). Mit cheetah 3: Design and control of a robust, dynamic quadruped robot. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2245–2252. doi:10.1109/IROS.2018.8593885.

García, G., Griffin, R., and Pratt, J. (2021). Time-varying model predictive control for highly dynamic motions of quadrupedal robots. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 7344–7349. doi:10.1109/ICRA48506.2021.9561913.

Katz, B., Carlo, J.D., and Kim, S. (2019). Mini cheetah: A platform for pushing the limits of dynamic quadruped control. *2019 International Conference on Robotics and Automation (ICRA)*, 6295–6301.

Kim, D., Carlo, J.D., Katz, B., Bledt, G., and Kim, S. (2019). Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control. *CoRR*, abs/1909.06586.

Kohonen, T., Schroeder, M.R., and Huang, T.S. (2001). *Self-Organizing Maps*. Springer-Verlag, Berlin, Heidelberg, 3rd edition.

Mayne, D.Q. (2014). Model predictive control: Recent developments and future promise. *Automatica*, 50(12), 2967–2986. doi:https://doi.org/10.1016/j.automatica.2014.10.128.

Ritter, H., Martinetz, T., and Schulten, K. (1992). *Neural Computation and Self-Organizing Maps; An Introduction*. Addison-Wesley Longman Publishing Co., Inc., USA, 1st edition.

Saputra, A.A., Tay, N.N.W., Toda, Y., Botzheim, J., and Kubota, N. (2016). Bézier curve model for efficient bio-inspired locomotion of low cost four legged robot. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 4443–4448.

Sun, H., Fu, T., Ling, Y., and He, C. (2021). Adaptive quadruped balance control for dynamic environments using maximum-entropy reinforcement learning. *Sensors*, 21, 5907. doi:10.3390/s21175907.

Yue, J. (2020). Learning locomotion for legged robots based on reinforcement learning: A survey. In *2020 International Conference on Electrical Engineering and Control Technologies (CEECT)*, 1–7. doi:10.1109/CEECT50755.2020.9298680.