Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 54-17 (2021) 93–98

A data-driven neural network model predictive steering controller for a bio-inspired quadruped robot

Paolo Arena * **Luca Patanè** ** **Pierfrancesco Sueri** * **Salvatore Taffara** *

* *Università degli Studi di Catania, Viale Andrea Doria 6, 95125 Catania, Italy (e-mail: paolo.arena@unict.it, salvatore.taffara@phd.unict.it)*

** *Università degli Studi di Messina, Contrada di Dio, 98166 Messina, Italy (e-mail: lpatane@unime.it)*

Abstract: This paper proposes a particular type of nonlinear data-driven Model Predictive Control (NMPC) strategy, called Neural Network Model Predictive Control (NNMPC), applied to a simulated neuro-inspired quadruped robot. The locomotion control is realised using a central pattern generator (CPG) implemented through oscillators synchronized through environmental feedback. The NMPC provides a descending command to the robot for steering control. This is realized by regulating a parameter governing the dynamics of the CPG structure. In order to test the performance obtained applying the NMPC, the results are compared with those obtained using a linear MPC. Carrying out a comparative analysis, the differences between the two methods will be highlighted.

Keywords: nonlinear MPC, CPG, quadruped, neural network, bio-robotics, heading control.

1. INTRODUCTION

The field of mobile robot control has been the focus of active research in the past decades and the Model Predictive Controllers (MPCs), with the increasing computing power of microprocessors, have been increasingly used for the control of wheeled and legged robots. An MPC includes a large range of control methods that make explicit use of a process model, exploiting the prediction capabilities, generating suitable control signals by minimizing a cost function.

There are a lot of research works focused on the study of MPCs applied to both quadrupedal and wheeled robots. In Horvat et al. (2017) the main goal is to improve the balance of a static walking quadruped robot using the MPC to provide a center of mass projection reference kept within support polygons to ensure stability. A different solution is proposed in Shi et al. (2019) where the MPC uses a 3D model of a reverse pendulum to represent the dynamics of the center of mass for path planning. The footholds are then automatically generated. This MPC optimization problem can be reformulated as quadratic programming, there are four weighted combinations of the cost function. Lages and Vasconcelos Alves (2006) presents an optimal control scheme for a wheeled mobile robot with nonholonomic constraints related to trajectory tracking. To solve this problem, a linear MPC Quadratic programming is used, working on successive linearizations of an error model of the robot.

⋆ This research was funded by MIUR project CLARA - Cloud platform for LAndslide Risk Assessment grant number SNC_00451.

Linear MPC (LMPC) refers to the simplest MPC family in which linear models are used to predict the system dynamics and represents a fairly mature theory (Lee, 2011). When it is necessary to control intrinsically nonlinear systems, the performance obtained by LMPC is not sufficient. In these cases, a nonlinear MPC (NMPC) can be adopted (Gros et al., 2016). Neunert et al. (2018) tested the NMPC performance to locomotion control of two different quadruped robots, identifying and optimizing the leg ground contacts efficiently.

The use of a nonlinear model implies higher complexity in terms of calculation of the control law and, in the stability analysis of the obtained closed-loop system. The formulation of the optimization problem is the principal part of the control design involving numerous decisions that are fundamental for the control performance and the stability issues (Grancharova and Johansen, 2012).

Farshidian et al. (2017) uses an NMPC for the motion planning of legged robots and the SLQ algorithm to solve the related optimal control problems with nonlinear dynamics, cost, and equality constraints (Farshidian et al., 2017). In Lim et al. (2008), the authors focused on the schematization of an NMPC with obstacle avoidance for the control of a robot on two wheels. In literature, the works related to the control of quadruped robots, in most of the cases, use the LMPC solutions. Very recently the interest in legged robots grew a lot, due to the large potential of these machines both in reaching very unstructured environments and also to cover a large range of speeds. On the other hand, the control problems of these sophisticated structures are the real challenge. In the last few years,

10.1016/j.ifacol.2021.11.031

MPC was applied to such machines, as previously introduced. In particular, when dealing with the robot model, often a simplified linear dynamic structure was adopted, which neglected several aspects, including, for example, leg dynamics (Carlo et al., 2018). This approximation, although relevant, allowed to formulate a linear analytical representation of the robot and to apply convex, real-time optimization algorithms for the control law design. In other cases first, a more complex nonlinear state representation of the robot was derived, but afterwards, a linearised version was used for carrying out the MPC calculations, reaching real-time performance (Ding et al., 2019). On the other hand, it is interesting to analyse the role of nonlinearities and maintaining them through the whole optimization task. To this aim, it is convenient to adopt the same ground for the model design. Since it is fairly complicated to derive a nonlinear accurate model, in this work a data-driven approach was adopted and referred to the same quadrupedal structure.

A particular type of NMPC, i.e., the Neural Network Model Predictive Controller (NNMPC) (Piche et al., 2000), will be applied for the steering control of a quadruped robot simulated in a dynamic environment, in particular the yaw control is chosen to be the test case to validate the proposed method. The use of a dynamic simulator is essential in the designing phase. Even if the model dealt with in this manuscript has no real counterpart nevertheless, for several existing quadrupedal prototypes, efficient dynamic models were developed, exactly to try novel control strategies before real implementation. In particular, we developed an NNMPC-based heading control for a bio-inspired quadruped robot actuated using the CPG paradigm. Experiments in a simulated environment using the LMPC and NNMPC controllers as control techniques will be carried out. Finally, a comparison of the results of these two approaches will be presented.

Our quadruped model takes into account a bio-inspired hierarchical structure, in which a low-level neural locomotion controller is responsible for the motion aspects, at the level of the single legs, and the coordination of them, using the Central Pattern Generator (CPG) paradigm. The low-level controller is bio-inspired, whereas the high-level one is based on NMPC, exploiting a data-driven approach.

This paper is structured as follows: the robotic neural locomotion architecture used is shown in Section 2. In Section 3 the implementation details and the NNMPC outcomes are reported. Section 4 shows the test outcomes of the dynamic simulator and the results of the comparisons between NNMPC and LMPC. Finally, conclusions are drawn in Section 5.

2. ROBOTIC STRUCTURE AND LOCOMOTION CONTROLLER

A bio-inspired locomotion control architecture for a 12-degrees-of-freedom quadruped robot, introduced in Arena et al. (2018); Arena et al. (2019), is here recalled and then exploited for an NNMPC-based navigation control system. The robot model was created in a dynamic simulation environment named CoppeliaSim (Rohmer et al., 2013). The model consists of four legs actuated by three revolute joints, as seen in Fig.1. Each foot has a cylindrical shape connected to the knee through a prismatic passive joint acting as a shock absorber. As reported in Fig. 1, the robot is composed of two main bodies representing the front and the hind trunk, linked by a bar with a negligible mass. In this way, the robot mass is evenly located just above front and rear legs.

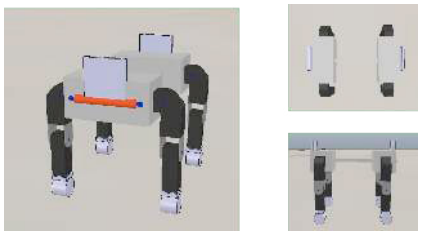


Fig. 1. Quadruped model developed in the dynamic simulation environment.

2.1 Neural locomotion architecture

The architecture for the neural controller is stated schematically in Fig.2, for more details please refers to Arena et al. (2018). Each leg is controlled by a half-center oscillator implemented by two complex integrated systems that imitate extensor and flexor muscle dynamics and is inspired by the Matsuoka neuron model (Matsuoka, 1987). In this implementation, a hybrid centralized locomotion controller was implemented. In fact, whereas in traditional CPGs the neural oscillators are connected among themselves with a rigid network structure, in our implementation, coordination among the neurons controlling each leg, arises only as a consequence of environmental feedback through load sensors placed on the leg tips.

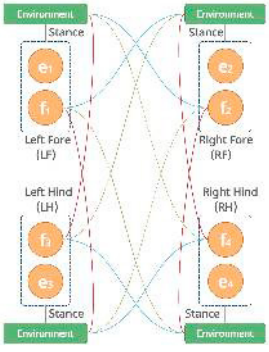


Fig. 2. CPG scheme controlling the robot locomotion. The load feedback connections are reported: ipsilateral connections (red line), diagonal connections (green line), contralateral connections (blue line).

The low level locomotion controller for each leg is represented by the nonlinear dynamics reported below, describing the behaviour of a flexor-extensor neural coupling:

$$\dot{x}_{1_{ei}} = \epsilon_r(-x_{1_{ei}} - bx_{2_{ei}} + \gamma y_{fi} + s + feed1_{ei}), \quad (1a)$$
$$\dot{x}_{2_{ei}} = \epsilon_a(-x_{2_{ei}} + y_{ei}), \quad (1b)$$
$$y_{ei} = x_{1_{ei}} H(x_{1_{ei}}) \quad (1c)$$

$$\dot{x}_{1_{fi}} = \epsilon_r(-x_{1_{fi}} - bx_{2_{fi}} + \gamma y_{ei} + s + feed1_{fi} + feed2_{fi}), \quad (2a)$$

$$\dot{x}_{2_{fi}} = \epsilon_a(-x_{2_{fi}} + y_{fi}), \quad (2b)$$

$$y_{fi} = x_{1_{fi}} H(x_{1_{fi}}) \quad (2c)$$

where $H(x)$ is the Heaviside function.

The dynamic behaviour of the neuron is shaped by the sensory input signals which act through the $feed1$ and $feed2$ signals described below:

$$feed1_{\{e,f\}} = \pm\, k_1 \cdot (\theta_i - \theta_0), \quad (3a)$$

$$feed2_f = [feed2_{f1}, feed2_{f2}, \cdots feed2_{f4}]^T = K_2 \cdot L, \quad (3b)$$

$$K_2 = k_{ip} Ip + k_{co} Co + k_{di} Di \in \mathbb{R}^{4\times 4}, \quad (4a)$$

$$L = [l_1, l_2, l_3, l_4]^T \in \mathbb{R}^4 \quad (4b)$$

$$Ip = \begin{bmatrix} 0&0&1&0 \\ 0&0&0&1 \\ 1&0&0&0 \\ 0&1&0&0 \end{bmatrix} \; Co = \begin{bmatrix} 0&1&0&0 \\ 1&0&0&0 \\ 0&0&0&1 \\ 0&0&1&0 \end{bmatrix} \; Di = \begin{bmatrix} 0&0&0&1 \\ 0&0&1&0 \\ 0&1&0&0 \\ 1&0&0&0 \end{bmatrix} \quad (5a)$$

The indexes e, f describe the extensor and flexor neurons, whereas i specifies the considered leg (i.e., 1: left front, 2: right front, 3: left hind, 4: right hind).

Every single cell of the CPG controller consists of a 4-th order nonlinear system, including two sub-units representing the flexor-extensor couple (Fukuoka et al., 2015). The membrane potential ($x_{1_{\{e,f\}i}}$) and the recovery variable ($x_{2_{\{e,f\}i}}$) are considered as two state variables. The $x_{2_{\{e,f\}i}}$ recovery vector inhibits the $x_{1_{\{e,f\}i}}$ membrane potential through the b parameter, determining the time constant used for deciding the specific frequency of the CPG. The other parameters of the model are reported in Table 1.

The terms $feed1\{e,f\}i$ are particularly relevant, they are evaluated using sensory feedback based on the hip joint angle, fundamental to provide the CPG rhythm during the steps. The terms $feed2\{e,f\}i$ stand for the sensory input from the load sensors. In particular, $feed2\{e,f\}i$ contains the afferent loads from the adjacent, legs described by the vector $L \in \mathbb{R}^4$, as also shown in Fig. 2. Depending on the selected locomotion gait, the following gains on the load sensors are tuned: k_{ip} (ipsilateral), k_{co} (contralateral) and k_{di} (diagonal). $y_{\{e,f\}i}$ are the outputs of the i-th leg extensor and flexor neurons; these are discontinuous and non-negative terms due to the $H(\cdot)$ Heaviside function. All the parameters are the same for each leg.

The CPG output for each leg (if larger or equal to zero), is used to select the two phases of the leg (i.e., stance or swing). In each of these two phases, a PI speed controller enables the leg to reach the corresponding reference position for each of the two phases, realizing in this way a closed-loop cycling motion for each leg.

Adopting the set of parameters shown in Table 1, the proposed CPG with sensory feedback allows the generation of different locomotion gaits and the migration between them.

Table 1. Parameters adopted in the CPG structure.

Parameters	**Gait**			
	Lateral Sequence	Trot	Canter	Gallop
ϵ_a	1.67			
b	3			
γ	2			
θ_0	0			
k_1	3			
s	2.2	2.6	3	3
ϵ_r	6.25	8.33	16.67	16.67
k_{ip}	-0.04	0	0.08	0.08
k_{co}	0.04	0.04	-0.04	-0.04
k_{di}	0	0.08	0	0

Finally, a third feedback signal has been adopted to generate a heading control. The input is applied as an additive term to Eq. (2a) adopting the following relation:

$$feed3_{fi} = c_i \cdot S_c \qquad i = 1, \ldots, 4 \quad (6)$$

where S_c is the steering command and c_i is the i^{th} element of a vector specified as:

$$C = \begin{bmatrix} +1 \\ -1 \\ +1 \\ -1 \end{bmatrix} \quad (7)$$

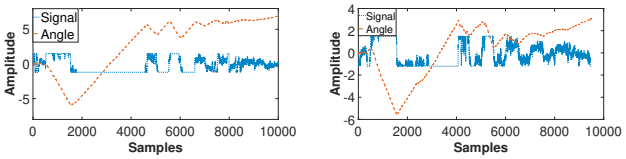
which governs the contribution between the legs of the steering command. The MPC-based strategy is, therefore, used to control the robot steering and thus to follow an imposed trajectory by acting on S_c.

3. NNMPC DESIGN OF A QUADRUPED ROBOT

In order to implement an NNMPC control a fundamental step is the system identification to create a model of the system to be controlled. We defined the quadruped robot model in our application using a data-driven approach. In particular, a neural network has been identified to model the relationship between the input corresponding to the steering command given to the robot, in the form of the S_c signals, and the output corresponding to the yaw angle of the robot.

Five different datasets have been used to identify the robot model, one for the training procedure and the others to test it. The sampling time is set to 50 ms. In Fig.3, the relationship between the steering and the yaw values is shown. The dataset for the training, composed of 10000 samples, is shown in Fig.3(a). The path followed by the robot is characterized by a yaw ramp with a negative slope, followed by a positive slope ramp, and gradually a more complex behaviour is generated with rapid heading changes. The steering control signal S_c in eq.6 covers the entire allowed range (i.e. from -1.2 to 1.5) representing a hard constraint on the control output signal to avoid robot instabilities. In Fig.3(b), one of the test datasets is shown. The simulated robotic structure is very complex: during simulation campaigns the robot needed a bias steering signal of about 0.2 to follow a straight trajectory.

In Fig.4 a high-level scheme of the system architecture, developed in Matlab-Simulink, is reported. The reference



(a) (b)

Fig. 3. Training and test datasets used for the identification of the robot model, considering the S_c signal as input and the yaw angle as output. (a) Training dataset; (b) Test datasets. The red line reflects the Yaw Output and the steering control action is reported in blu. The yaw angle is indicated in radians.

signal X_{ref}, representing the desired yaw, is an input for the NN Predictive Control block, together with the actual yaw of the robot, whereas the output is the control signal S_c which modulates the CPG locomotion network. The communication with the CoppeliaSim dynamic framework has been performed using the 2-level s-function available in Matlab. The MPC parameters used are the following: sample time: 0.05s, number of manipulated variables: 1 (steering), number of the measured outputs: 1 (yaw), prediction horizon: 150, control horizon: 40, closed-loop performance: fixed, speed of state estimation: fixed.

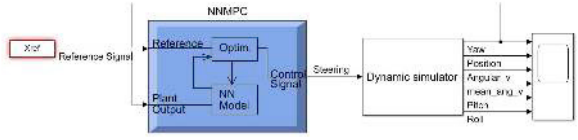


Fig. 4. Scheme of the control architecture developed in Simulink.

Besides the other information, the fundamental parameters of MPC are the prediction and control horizon. According to the MPC guidelines, the prediction horizon was chosen to have 20-30 samples covering the open-loop transient system response, whereas the control horizon was selected among the 10% and 30% of the Prediction Horizon. The NNMPC block employs a Neural Network-based model of the robot behaviour able to describe and predict the effect of the steering control signal on the robot heading. The best control sequence is derived via a numerical optimization algorithm minimizing the following performance criterion J over the specified horizon:

$$J = \sum_{j=1}^{N} (y_r(t+j) - y_m(t+j))^2 + \rho \sum_{j=1}^{N_U} (u'(t+j-1) - u'(t+j-2))^2 \quad (8)$$

where N defines the cost horizons over which the tracking error is evaluated and N_U defines the dimension of the control horizon, in which the control increments are analyzed. The u' variable is the tentative control signal, y_r is the reference (desired response), while y_m is the neural network model response. The ρ value weights the contribution of the sum of the squared control increments on the index J; its value was fixed to $0,05$. The optimization block determines the control input u' minimizing J, and then the optimal u is provided as input to the robot CPG controller. The minimization algorithm used is *csrchbac*; it is one-dimensional minimization routine based on a backtracking technique (Dennis and Schnabel, 1983).

The Search Parameter α works as a stop criterion for the minimization routine: if the minimization between two consecutive control input candidates is less than α, the routine stops (Press et al., 2007).

For comparisons, a linear MPC controller was also developed, as briefly explained below. The linear model was obtained through analyzing different transfer function structures, containing a varying number of poles and zeros (from 1 to 7). The outcome of this identification procedure was a transfer function characterized by 5 poles and 4 zeros; this was obtained using the same optimization criterion reported below for the NN structure.

Based on this analysis for the design of the linear model, the regressors for the input and output variables were used as input features for the neural network structure. Therefore the input layer of the networks consisted of a nine dimension vector. The network has been trained offline in batch mode, using the training dataset previously defined and adopting the Levenberg-Marquardt learning technique with a maximum number of epochs equal to 400.

To find the best number of hidden neurons a supplementary analysis was carried out, in which the number of neurons was varied in the range $[3, 12]$. The index used to choose the best model, both for the neural network and for the linear model, is based on the Normalized Root Mean Square Error (NRMSE):

$$Fit_{NRMSE} = \left(1 - \frac{||(x_{ref} - x)||}{||(x_{ref} - \bar{x}_{ref})||}\right) \cdot 100 \quad (9)$$

where the desired output is x_{ref}, the mean value is $\bar{x}_{ref}$, the actual output is x, and $||\cdot||$ indicates the vector 2-norm. The results of this hyperparameter analysis are shown in Fig.5.

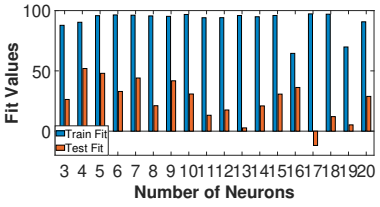


Fig. 5. Network performance as function of the number of hidden neurons. The blue bars represent the Fit_{NRMSE} on the training dataset, while the red bars represent the mean of the Fit_{NRMSE} calculated on the test datasets.

The most suitable number of hidden neurons selected is four. This network has a training fit value of 94 and a mean test fit value of 65, which represents an improvement if compared to the linear model identification, which was able to reach fit values in training and test of 78 and 52, respectively. In Fig.6 the comparison between the neural model and the linear transfer function is reported.

4. SIMULATION RESULTS

In this section, the results of the NNMPC controller are presented and discussed. Moreover, a comparison with the performance obtained using the LMPC controller is carried out.

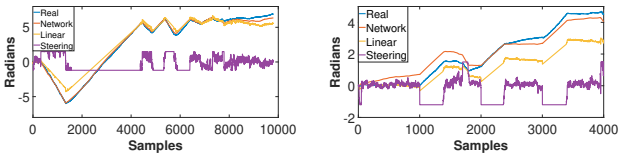


Fig. 6. Comparison of the identification performance for a linear and nonlinear model on the (a) training dataset and (b) one of the test dataset.

The NNMPC was tested on six different reference signals: two representing an ideal square route (one clockwise and the other counterclockwise), two representing an ideal equilateral triangle route (also in this case, in both directions), a one-period sine reference, and two semicircular routes.

The results are reported in Fig.7, as a function of the searching parameter α, used as a stopping criterion. In particular, in Fig.7(c), when the first negative step is applied in the reference yaw, the controller does not properly work with $\alpha = 0.001$. Here, the control signal starts to oscillate between the two saturation limits with the results that the output yaw is almost constant and loses the reference command tracking.

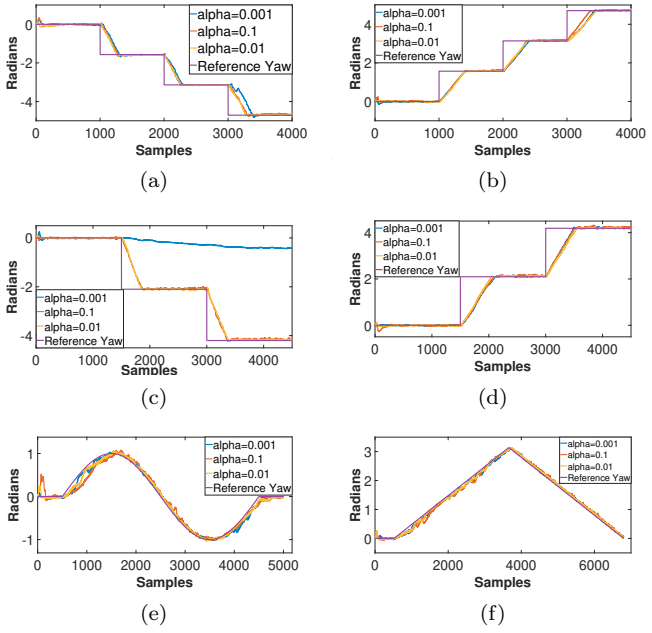


Fig. 7. Behaviour of the quadruped robot while following the reference steering command generated by the NNMPC with α varying between 0.1, 0.01, 0.001: (a) Clockwise square reference; (b) Counterclockwise square reference; (c) Clockwise triangle reference; (d) Counterclockwise triangle reference; (e) Sine reference; (f) Semicircular reference.

To better evaluate these results, the Fit_{NRMSE} between the robot yaw and the reference signal was evaluated. The fit value is affected by the maximum angular velocity that the robot can reach, due to the physical constraints and to the constraints added in the control action, i.e. the steering bounds. In fact, the reference signal is an ideal figure, and it assumes that the robot should have the capability to turn on the spot: this is not possible in our case, unless at the expenses of complicated maneuvers, not conceived within the trotting gait, which is the locomotion patterns shown by our robot.

Table 2 shows the outcome of this analysis: the best results are obtained with α= 0.01 (i.e., Mean $Fit_{NRMSE} = 77$).

Table 2. Fit values for the six datasets obtained through the NNMPC, for different values of α.

References	Fit_{NRMSE} α=0.1	Fit_{NRMSE} α=0.01	Fit_{NRMSE} α=0.001
CW sq	74.51	75.77	70.35
CCW sq	71.60	69.57	69.67
CW tr	71.44	70.49	-44.38
CCW tr	64.48	64.26	64.16
sine	80.87	88.97	88.69
semi	90.04	92.85	92.57
Mean	**75.49**	**76.99**	**56.84**

Finally, the NNMPC and the LMPC performance were compared. The results in terms of Fit values and maximum absolute error (MaxAE) are outlined in Table 3. The NNMPC performs better in almost all cases as also demonstrated in Fig.8. The accuracy is even more evident when the yaw signal undergoes continuous variations, as depicted in Fig.8 (e)-(f). Here the NMPC controller does not suffer from any tracking delay and is also smoother than the MPC controller result.

Table 3. Comparison between the NNMPC and LMPC Fit values and MaxAE for the six test datasets adopted. In the last row the mean values calculated on the datasets are indicated. (CW sq: clockwise square; CCW sq: counterclockwise square; CW tr: clockwise triangle; CCW tr: counterclockwise triangle; sine: sine yaw; semi: semicircular).

Ref	**NNMPC**		**LMPC**	
	Fit	MaxAE	Fit	MaxAE
CW sq	75.77	0.20	72.31	1.63
CCW sq	69.57	0.24	71.02	1.58
CW tr	70.49	0.19	68.48	2.18
CCW tr	64.26	0.28	65.45	2.06
sine	88.97	0.08	74.88	0.30
semi	92.85	0.07	84.47	0.40
Mean	**76.99**	**0.17**	**72.77**	**1.35**

5. CONCLUSION

In this work, the effect of a nonlinear MPC-based control strategy applied on a quadruped robot, endowed with a bio-inspired locomotion controller is analyzed. In particular, the controller acts on the robot steering. The quadruped robot behaviour has been modelled using a neural network designed and optimized following a data-driven approach. From a comparative analysis carried out considering the fit values performance and the MaxAE index, it was possible to evaluate the control results in terms of matching between the reference signals of the yaw angle of the controlled robot simulated in a dynamic framework. A comparative analysis of the results obtained using a linear and a nonlinear MPC demonstrates the

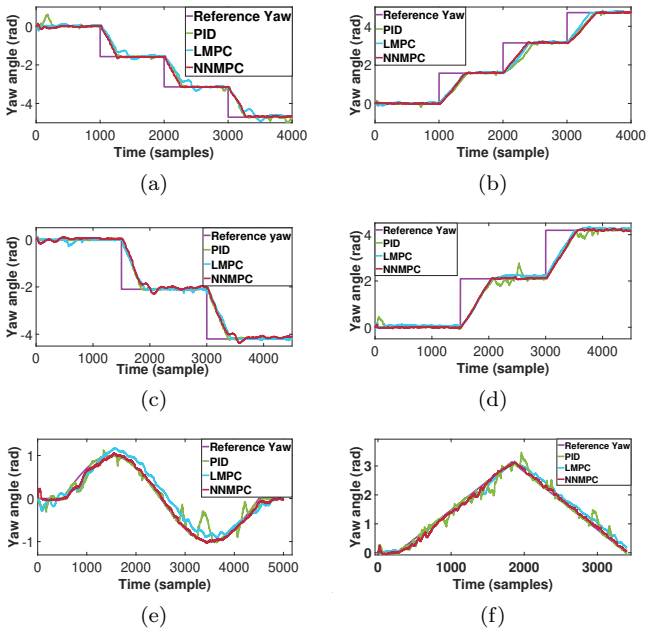


Fig. 8. Comparison between the robot heading and the reference signal obtained using the LMPC and NNMPC control strategies in the case of: (a) Clockwise square reference; (b) Counterclockwise square reference; (c) Clockwise triangle reference; (d) Counterclockwise triangle reference; (e) Sine reference; (f) Semicircular reference.

effectiveness of the NNMPC in particular in presence of continuously time-varying trajectories.

REFERENCES

Arena, P., Bonanzinga, A., and Patané, L. (2018). Role of feedback and local coupling in cnns for locomotion control of a quadruped robot. In *CNNA 2018; The 16th International Workshop on Cellular Nanoscale Networks and their Applications*, 1–4.

Arena, P., Bonanzinga, A., and Patané, L. (2019). *Emergence of Locomotion Gaits through Sensory Feedback in a Quadruped Robot*, 547–574. doi:10.1201/9781315167084-25.

Carlo, J., Wensing, P., Katz, B., Bledt, G., and Kim, S. (2018). Dynamic locomotion in the mit cheetah 3 through convex model-predictive control. 1–9. doi:10.1109/IROS.2018.8594448.

Dennis, J. and Schnabel, R. (1983). *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Englewood Cliffs, NJ, Prentice-Hall.

Ding, Y., Pandala, A., and Park, H. (2019). Real-time model predictive control for versatile dynamic motions in quadrupedal robots. In *2019 International Conference on Robotics and Automation (ICRA)*, 8484–8490. doi:10.1109/ICRA.2019.8793669.

Farshidian, F., Jelavic, E., Satapathy, A., Giftthaler, M., and Buchli, J. (2017). Real-time motion planning of legged robots: A model predictive control approach. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, 577–584. doi:10.1109/HUMANOIDS.2017.8246930.

Farshidian, F., Neunert, M., Winkler, A.W., Rey, G., and Buchli, J. (2017). An efficient optimal planning and control framework for quadrupedal locomotion. *2017 IEEE International Conference on Robotics and Automation (ICRA)*. doi:10.1109/icra.2017.7989016. URL `http://dx.doi.org/10.1109/ICRA.2017.7989016`.

Fukuoka, Y., Habu, Y., and Fukui, T. (2015). A simple rule for quadrupedal gait generation determined by leg loading feedback: A modeling study. *Scientific reports*, 5, 8169. doi:10.1038/srep08169.

Grancharova, A. and Johansen, T. (2012). *Explicit Nonlinear Model Predictive Control: Theory and Applications*, volume 429. doi:10.1007/978-3-642-28780-0.

Gros, S., Zanon, M., Quirynen, R., Bemporad, A., and Diehl, M. (2016). From linear to nonlinear mpc: bridging the gap via the real-time iteration. *International Journal of Control*, 93, 1–19. doi:10.1080/00207179.2016.1222553.

Horvat, T., Melo, K., and Ijspeert, A.J. (2017). Model predictive control based framework for com control of a quadruped robot. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3372–3378. doi:10.1109/IROS.2017.8206176.

Lages, W.F. and Vasconcelos Alves, J.A. (2006). Real-time control of a mobile robot using linearized model predictive control. *IFAC Proceedings Volumes*, 39(16), 968 – 973. 4th IFAC Symposium on Mechatronic Systems.

Lee, J. (2011). Model predictive control: Review of the three decades of development. *International Journal of Control, Automation and Systems*, 9, 415–424. doi:10.1007/s12555-011-0300-6.

Lim, H., Kang, Y., Kim, C., Kim, J., and You, B. (2008). Nonlinear model predictive controller design with obstacle avoidance for a mobile robot. In *2008 IEEE/ASME International Conference on Mechtronic and Embedded Systems and Applications*, 494–499. doi:10.1109/MESA.2008.4735699.

Matsuoka, K. (1987). Mechanisms of frequency and pattern control in the neural rhythm generators. *Biological cybernetics*, 56, 345–53. doi:10.1007/BF00319514.

Neunert, M., Stäuble, M., Giftthaler, M., Bellicoso, C.D., Carius, J., Gehring, C., Hutter, M., and Buchli, J. (2018). Whole-body nonlinear model predictive control through contacts for quadrupeds. *IEEE Robotics and Automation Letters*, 3(3), 1458–1465. doi:10.1109/LRA.2018.2800124.

Piche, S., Sayyar-Rodsari, B., Johnson, D., and Gerules, M. (2000). Nonlinear model predictive control using neural networks. *IEEE Control Systems Magazine*, 20(3), 53–62. doi:10.1109/37.845038.

Press, W., Teukolsky, S., Vetterling, W., and Flannery, B. (2007). *Numerical Recipes: The Art of Scientific Computing (3rd ed.)*. New York: Cambridge University Press.

Rohmer, E., Singh, S.P.N., and Freese, M. (2013). V-rep: A versatile and scalable robot simulation framework. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1321–1326.

Shi, Y., Wang, P., Li, M., Wang, X., Jiang, Z., and Li, Z. (2019). Model predictive control for motion planning of quadrupedal locomotion. In *2019 IEEE 4th International Conference on Advanced Robotics and Mechatronics (ICARM)*, 87–92. doi:10.1109/ICARM.2019.8834241.