2022 International Conference on Robotics and Automation (ICRA), 2022, pp. 214-220,
doi: 10.1109/ICRA46639.2022.9812317.

Task-Oriented Generation of Stable Motions for Wheeled Inverted Pendulum Robots

Marco Kanneworff, Tommaso Belvedere, Nicola Scianca, Filippo M. Smaldone,
Leonardo Lanari, Giuseppe Oriolo

***Abstract*— We present a whole-body control architecture for the generation of stable task-oriented motions in Wheeled Inverted Pendulum (WIP) robots. Controlling WIP systems is challenging because the successful execution of tasks is subordinate to the ability to maintain balance. Our feedback control approach relies both on partial feedback linearization and Model Predictive Control (MPC). The partial feedback linearization reshapes the system into a convenient form, while the MPC computes inputs to execute the desired task by solving a constrained optimization problem. Input constraints account for actuation limits and a stability constraint is in charge of stabilizing the unstable body pitch angle dynamics. The proposed approach is validated by simulations on an ALTER-EGO robot performing navigation and loco-manipulation tasks.**

I. Introduction

Balancing mobile robots are constituted by a statically unstable body, typically mounted on wheels, which can perform manipulation tasks by leveraging one or more actuated arms with multiple degrees of freedom. These platforms offer several advantages with respect to statically stable mobile robots. They are more suitable to traverse uneven terrains and they can perform fast and dynamic motions while carrying weights, moving at speeds comparable to those of humans. However, these capabilities implicitly require that the robot is able to maintain dynamic balance at all times in order to prevent falls. The underactuation of the dynamics renders this a rather complex control problem.

Early research studies on balancing mobile robots, both including transportation vehicles [1] and manipulators [2], [3], [4], highlighted the potentialities of these platforms which encouraged research groups to develop new robots such as Golem Krang [5], the Ballbot [6], and the more recent ALTER-EGO [7] shown in Fig. 1.

Fig. 1. ALTER-EGO, a prototype WIP robot developed at the University of Pisa [7].

In the literature, several approaches have been proposed to control robots of this kind. It is possible for instance to model the platform as an inverted pendulum with the only goal of maintaining balance, and account for the manipulator motion as a disturbance to be compensated by the mobile base, e.g., via sliding mode control [8]. In [9] a PD control law has been used for balancing coupled with a PID controller for navigation while performance has been improved with a disturbance observer. Others, such as [10], propose a whole-body control law for balancing, which also allows the robot to gently interact with the environment. Task-space control was demonstrated to be applicable to WIP-based platforms in [11], allowing to maintain balance while controlling the end effector. This approach was further investigated by [12], where task-space control has been extended to the class of WIP robots in which the wheel motors are connected to the robot base link that is then subject to the wheel reaction torque.

Model Predictive Control (MPC) offers a powerful approach to the control of underactuated systems, especially when constraints are to be enforced. Feedback linearization in conjunction with MPC was used in [13] for a wheeled inverted pendulum without arms. In [14] a hierarchical structure is proposed, composed by an MPC-based reference trajectory generator and an inverse dynamics controller to track the reference trajectories. Whole-body MPC algorithms instead allow to jointly perform re-planning and control, but their implementation can be cost-demanding due to the complex dynamics of the robots. Remarkable results were achieved in [15] for a Ballbot-like omnidirectional mobile manipulator, where real-time performance was achieved by exploiting custom nonlinear solvers. These methods, although proven to be effective, do not directly address the stability problem related to the unstable balancing dynamics. In fact, it is often required for the MPC to have a long prediction horizon in order to generate stable behaviors, either by regulating to zero the body pitch angle — possibly limiting performance — or by tracking a pre-computed pitch trajectory.

In this paper, we propose a whole-body MPC controller

The authors are with the Dipartimento di Ingegneria Informatica, Automatica e Gestionale, Sapienza Università di Roma, via Ariosto 25, 00185 Roma, Italy. E-mail: {lastname}@diag.uniroma1.it.

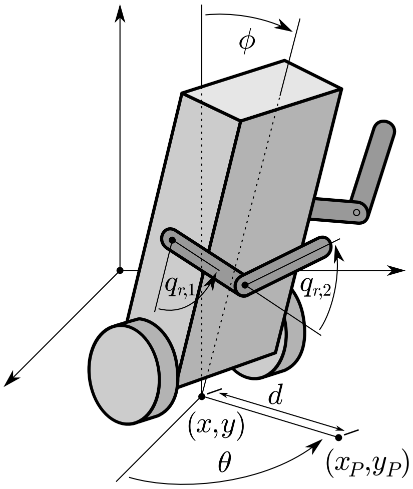

Fig. 2. Generalized coordinates for a WIP robot with $n_r = n_l = 2$ (left arm coordinates are omitted for clarity). Note the displaced point on the ground (x_P, y_P) used for navigation tasks.

for WIP robots which explicitly addresses the instability problem. The proposed scheme has a general formulation that can accommodate different objectives. In particular, we report examples of navigation and loco-manipulation tasks. In order to reshape the system in a convenient way, we perform MPC on a partially feedback-linearized system. This structure allows us to address the instability problem by introducing an explicit stability constraint. This idea is inspired by our previous work on humanoid robots, where we developed an Intrinsically Stable MPC (IS-MPC) [16] with guarantees of recursive feasibility and internal stability, and [17] where a similar approach was applied to anti-jackknifing control of a tractor-trailer vehicle.

The paper is organized as follows. Section II gives some background concepts by introducing the robot model, the partial feedback linearization and the generic task definition. Section III contains an overview of the proposed approach, while Sect. IV describes in detail the IS-MPC algorithm. Simulation results and concluding remarks are reported in Sect. V and Sect. VI.

II. Background

In this section we introduce the dynamic model of the considered WIP robot, describe the partial feedback linearization procedure and define the task to be executed.

A. Modeling

The configuration of a WIP robot is defined as $\boldsymbol{q} = (x, y, \theta, \phi, \boldsymbol{q}_r, \boldsymbol{q}_l)$, where (x, y) is the position of the differential-drive robot base in the world frame, θ and ϕ are respectively the yaw angle and the pitch angle of the body, and $\boldsymbol{q}_r \in \mathbb{R}^{n_r}$ and $\boldsymbol{q}_l \in \mathbb{R}^{n_l}$ are the right and left arm joint angles, respectively. The total number of dofs in the arms is $n_a = n_r + n_l$. Figure 2 shows the schematic of an example WIP robot having $n_r = n_l = 2$.

Denote the velocity vector by $\boldsymbol{\nu} = (v, \omega, v_\phi, \boldsymbol{v}_r, \boldsymbol{v}_l)$, where v is the pseudovelocity of the robot base, $\omega = \dot{\theta}$, $v_\phi = \dot{\phi}$, $\boldsymbol{v}_r = \dot{\boldsymbol{q}}_r$ and $\boldsymbol{v}_l = \dot{\boldsymbol{q}}_l$. The robot state is then $\boldsymbol{x} = (\boldsymbol{q}, \boldsymbol{\nu})$.

A reduced-order dynamic model of the WIP robot can be expressed in compact form as [18]

$$\begin{aligned} \dot{\boldsymbol{q}} &= \boldsymbol{G}(\boldsymbol{q})\,\boldsymbol{\nu} && (1)\\ \dot{\boldsymbol{\nu}} &= -\boldsymbol{M}^{-1}(\boldsymbol{q})(\boldsymbol{c}(\boldsymbol{q}, \boldsymbol{\nu}) - \boldsymbol{E}\,\boldsymbol{\tau}), && (2) \end{aligned}$$

where $\boldsymbol{G}$ is a matrix whose columns span the null space of the Pfaffian nonholonomic constraint on the robot base, $\boldsymbol{M}$ is the inertia matrix, $\boldsymbol{c}$ is a force vector containing the gravity, centrifugal and Coriolis contributions, $\boldsymbol{E}$ is the actuator selection matrix and $\boldsymbol{\tau}$ are the torques acting on the base wheels and the arm joints.

Due to the presence of the articulated arms and the associated inertial couplings, the above model is considerably more complicated than in the case of a WIP with no arms (see [19] for a model of the latter). Therefore, we omit the explicit expression of the various matrices in eqs. (1–2), with the exception of the actuator selection matrix which takes the form

$$\boldsymbol{E} = \begin{pmatrix} 1/R & 1/R & \boldsymbol{0}_{1\times n_a} \\ a/R & -a/R & \boldsymbol{0}_{1\times n_a} \\ -1 & -1 & \boldsymbol{0}_{1\times n_a} \\ \boldsymbol{0}_{n_a\times 1} & \boldsymbol{0}_{n_a\times 1} & \boldsymbol{I}_{n_a\times n_a} \end{pmatrix}, \qquad (3)$$

where R is the wheel radius and a is the semi-distance between the wheels. The linear dependence of the first and third rows is a consequence of the fact that the pitch angle ϕ is not independently actuated — which makes the WIP robot an underactuated system.

B. Partial feedback linearization

To simplify the design of our control scheme, we perform a partial feedback linearization of model (1–2).

Let us focus on the following subset of equations from (2):

$$\begin{pmatrix} \dot{v} \\ \dot{\omega} \\ \dot{\boldsymbol{v}}_r \\ \dot{\boldsymbol{v}}_l \end{pmatrix} = \boldsymbol{\alpha}(\boldsymbol{x}) + \boldsymbol{\Psi}(\boldsymbol{x})\boldsymbol{\tau}. \qquad (4)$$

Define the following input transformation

$$\boldsymbol{\tau} = \boldsymbol{\Psi}^{-1}(\boldsymbol{x})(\boldsymbol{u} - \boldsymbol{\alpha}(\boldsymbol{x})), \qquad (5)$$

where $\boldsymbol{u}$ are the new transformed inputs. Using (5) in (1–2) results in a partially linearized dynamics which is conveniently reordered as

$$\begin{aligned} \dot{x} &= v\cos\theta \\ \dot{y} &= v\sin\theta \\ \dot{\theta} &= \omega \\ \dot{v} &= u_1 \\ \dot{\omega} &= u_2 \\ \dot{\boldsymbol{q}}_r &= \boldsymbol{v}_r \\ \dot{\boldsymbol{v}}_r &= \boldsymbol{u}_3 \\ \dot{\boldsymbol{q}}_l &= \boldsymbol{v}_l \\ \dot{\boldsymbol{v}}_l &= \boldsymbol{u}_4 \\ \dot{\boldsymbol{\eta}} &= \boldsymbol{f}_\eta(\boldsymbol{x}, \boldsymbol{u}), \end{aligned} \qquad (6)$$

having set $\boldsymbol{\eta} = (\phi, v_\phi)$. The vector field describing the pitch dynamics is now

$$\boldsymbol{f}_\eta(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{r}(\boldsymbol{x}) + \boldsymbol{S}(\boldsymbol{x})\boldsymbol{\Psi}^{-1}(\boldsymbol{x})\left(\boldsymbol{u} - \boldsymbol{\alpha}(\boldsymbol{x})\right),$$

where $\boldsymbol{r}$ and $\boldsymbol{S}$ are respectively the drift and the input matrix characterizing the original pitch dynamics in (1–2).

As intuition suggests, the pitch dynamics is inherently unstable in this kind of system [20]. For instance, one may easily verify that in a WIP robot the linearized pitch dynamics around the static equilibrium (CoM over the wheel axis) has a positive eigenvalue. A similar situation holds when the linearization of the pitch dynamics is computed around generic trajectories.

C. *Task definition*

We will consider general task functions defined as

$$\boldsymbol{r} = \boldsymbol{h}(\boldsymbol{q}), \tag{7}$$

with $\boldsymbol{r} \in \mathbb{R}^m$. By differentiating (7) we obtain

$$\dot{\boldsymbol{r}} = \frac{\partial \boldsymbol{h}(\boldsymbol{q})}{\partial \boldsymbol{q}}\,\dot{\boldsymbol{q}} = \frac{\partial \boldsymbol{h}(\boldsymbol{q})}{\partial \boldsymbol{q}}\,\boldsymbol{G}(\boldsymbol{q})\,\boldsymbol{\nu} = \boldsymbol{J}(\boldsymbol{q})\,\boldsymbol{\nu},$$

where $\boldsymbol{J}(\boldsymbol{q})$ is the $m \times (3 + n_a)$ task Jacobian.

For the simulations, we will consider two specific task functions. The first (*navigation* task) describes the coordinates (x_P, y_P) of a point P on the ground which is displaced[1] from the base by a distance d along the sagittal axis (see Fig. 2). The task function and the associated task Jacobian are easily computed as

$$\boldsymbol{h}(\boldsymbol{q}) = \begin{pmatrix} x_P \\ y_P \end{pmatrix} = \begin{pmatrix} x + d\cos\theta \\ y + d\sin\theta \end{pmatrix},$$

$$\boldsymbol{J}(\boldsymbol{q}) = \begin{pmatrix} \cos\theta & -d\sin\theta & \boldsymbol{0}_{1\times(n_a+1)} \\ \sin\theta & d\cos\theta & \boldsymbol{0}_{1\times(n_a+1)} \end{pmatrix}.$$

While the values of the arm generalized coordinates have no instantaneous effect on the navigation task, arm motions can dynamically contribute to the stabilization of the pitch dynamics.

The second task function used in the simulations will be the position of the end-effector, chosen as one of the two hands (*loco-manipulation* task). In this case, all generalized coordinates directly contribute to the task.

III. Proposed approach

The problem addressed in this paper is that of generating in real-time a whole-body motion of the WIP robot such that:

- the task function $\boldsymbol{r}(t)$ tracks a desired reference trajectory $\boldsymbol{r}_d(t)$;
- balance is maintained;
- constraints on both states (joint limits, velocity limits) and inputs (torque limits) are satisfied.

A block scheme of the proposed solution approach is shown in Fig. 3. The input is the task trajectory $\boldsymbol{r}_d(t)$ to be tracked. At each iteration, the IS-MPC algorithm solves an optimization problem over a receding control horizon to compute the transformed inputs $\boldsymbol{u}$, from which the original torque inputs $\boldsymbol{\tau}$ will be reconstructed. To achieve real-time performance, the optimization problem is formulated as a Quadratic Program (QP) by approximately linearizing system (6) around an auxiliary trajectory obtained from the solution predicted at the previous iteration, and using the same trajectory to compute the cost function as well as to map torque limits to linear constraints on the transformed inputs.

An essential component of the IS-MPC algorithm is the explicit stability constraint included in the QP. Such constraint will avoid the onset of instability in the pitch dynamics, ultimately guaranteeing that the robot maintains balance while executing the assigned task.

In the next section we describe in detail the IS-MPC algorithm and its various components.

IV. IS-MPC

The proposed method is based on the MPC paradigm and works over discrete sampling intervals of duration δ. At each sampling time t_k, an optimal control problem is solved over a control horizon $T_c = N\delta$.

Below we introduce the prediction model, the stability constraint, the input constraints and the resulting QP formulation.

A. *Prediction model*

At the k-th iteration, the prediction model is obtained by linearizing (6) around an auxiliary trajectory defined as follows. Denote by $(\boldsymbol{x}^{k-1}, \boldsymbol{x}^{k|k-1}, \boldsymbol{x}^{k+1|k-1} \ldots, \boldsymbol{x}^{k+N-1|k-1})$ the predicted state trajectory at the $(k-1)$-th iteration, where $\boldsymbol{x}^{k-1}$ is the current state at t_{k-1} and the following samples $\boldsymbol{x}^{k|k-1}, \boldsymbol{x}^{k+1|k-1} \ldots, \boldsymbol{x}^{k+N-1|k-1}$ are obtained by injecting the QP solution inputs at t_{k-1} into the corresponding prediction model[2]. From this trajectory, we can build the *auxiliary* trajectory $(\bar{\boldsymbol{x}}^k, \ldots, \bar{\boldsymbol{x}}^{k+N})$ by letting

$$\bar{\boldsymbol{x}}^{k+i} = \begin{cases} \boldsymbol{x}^k & i = 0, \\ \boldsymbol{x}^{k+i|k-1} & i = 1, ..., N-1 \\ \bar{\boldsymbol{x}}^{k+N-1} & i = N. \end{cases} \tag{8}$$

The current state $\boldsymbol{x}^k = \boldsymbol{x}(t_k)$ is used as first sample of the auxiliary trajectory to increase its precision and therefore the accuracy of the subsequent linearization procedure. Note also that the auxiliary trajectory is prolonged up to t_{k+N} by replicating the last sample $\boldsymbol{x}^{k+N-1|k-1}$ of the predicted state trajectory at the $(k-1)$-th iteration.

The prediction model at t_k can now be computed as the linear approximation of the partially feedback-linearized dynamics (6) around the auxiliary trajectory. In particular, for the pitch dynamics we will use the following model

$$\dot{\boldsymbol{\eta}} = \bar{\boldsymbol{A}}_\eta^{k+i}\boldsymbol{\eta} + \bar{\boldsymbol{B}}_\eta^{k+i}\boldsymbol{u}^{k+i} + \bar{\boldsymbol{f}}_\eta^{k+i}, \tag{9}$$

[1] A nonzero displacement guarantees that $\boldsymbol{J}(\boldsymbol{q})$ will be full rank, which is an implicit requirement of our MPC-based approach.

[2] At the first iteration, when a previous solution is not available, the inputs are simply set to zero.

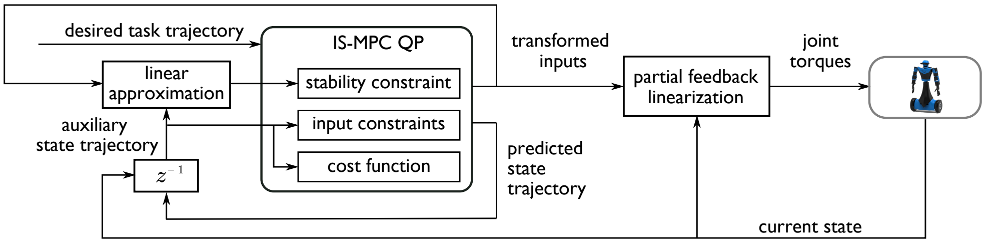

Fig. 3. A block scheme of the proposed approach. The z^{-1} block stores the solution of each iteration and makes it available at the next, after replacing its first sample with the current state, see (8).

for $i = 0, \ldots, N-1$, where

$$\bar{\boldsymbol{A}}_\eta^{k+i} = \left.\frac{\partial \boldsymbol{f}_\eta}{\partial \boldsymbol{\eta}}\right|_{\bar{\boldsymbol{x}}^{k+i}, \bar{\boldsymbol{u}}^{k+i}}, \quad \bar{\boldsymbol{B}}_\eta^{k+i} = \left.\frac{\partial \boldsymbol{f}_\eta}{\partial \boldsymbol{u}}\right|_{\bar{\boldsymbol{x}}^{k+i}, \bar{\boldsymbol{u}}^{k+i}}$$

$$\bar{\boldsymbol{f}}_\eta^{k+i} = \boldsymbol{f}_\eta\big|_{\bar{\boldsymbol{x}}^{k+i}, \bar{\boldsymbol{u}}^{k+i}} - \bar{\boldsymbol{A}}_\eta^{k+i} \bar{\boldsymbol{\eta}}^{k+i} - \bar{\boldsymbol{B}}_\eta^{k+i} \bar{\boldsymbol{u}}^{k+i},$$

and $(\bar{\boldsymbol{u}}^k, \ldots, \bar{\boldsymbol{u}}^{k+N-1})$ is the QP solution at t_{k-1}. Note that this piecewise time-invariant (but still time-varying) approximation neglects the effect of the variation of the arm coordinates $\boldsymbol{q}_r$, $\boldsymbol{q}_l$ with respect to the auxiliary trajectory.

B. Stability constraint

As already mentioned, the pitch dynamics in a WIP robot is unstable. This is obviously true also for its linear approximation (9). To cope with this instability, we include in the QP formulation a constraint *on the inputs* in order to guarantee that the evolution of ϕ does not diverge[3]. To formulate such a constraint, we will retain the time-varying prediction model (9) for $t < t_{k+N}$, and use a time-invariant approximation for $t \geq t_{k+N}$, as proposed in [21].

In particular, let $\boldsymbol{x}^* = (\boldsymbol{q}^*, \boldsymbol{0}_{(3+n_a)\times 1})$ and

$$\boldsymbol{q}^* = (\bar{x}^{k+N}, \bar{y}^{k+N}, \bar{\theta}^{k+N}, \phi^*, \bar{\boldsymbol{q}}_r^{k+N}, \bar{\boldsymbol{q}}_l^{k+N}). \tag{10}$$

where the configuration at the last sample of the auxiliary trajectory has been used, with the exception of $\bar{\phi}^{k+N}$ which has been replaced by ϕ^*, the value of the pitch angle that puts the WIP in static equilibrium when the arms are at $\bar{\boldsymbol{q}}_r^{k+N}$, $\bar{\boldsymbol{q}}_l^{k+N}$. The time-invariant approximation that we will use for $t \geq t_{k+N}$ is defined as

$$\dot{\boldsymbol{\eta}} = \boldsymbol{A}_\eta^* \boldsymbol{\eta} + \boldsymbol{f}_\eta^* \tag{11}$$

where the dynamic matrix and the constant drift, respectively

$$\boldsymbol{A}_\eta^* = \left.\frac{\partial \boldsymbol{f}_\eta}{\partial \boldsymbol{\eta}}\right|_{\boldsymbol{x}^*, \boldsymbol{0}}, \qquad \boldsymbol{f}_\eta^* = \boldsymbol{f}_\eta\big|_{\boldsymbol{x}^*, \boldsymbol{0}} - \boldsymbol{A}_\eta^* \boldsymbol{\eta}^*,$$

are computed at $\boldsymbol{x} = \boldsymbol{x}^*$ and $\boldsymbol{u} = \boldsymbol{0}$, and we have set $\boldsymbol{\eta}^* = (\phi^*, 0)$.

The particular choice of the state and inputs at which the linear approximation is frozen can be justified by noting that the state $\boldsymbol{x}^*$ is a static equilibrium for system (6) which requires no input ($\boldsymbol{u} = \boldsymbol{0}$). In the terminology of [16], this corresponds to using a *truncated tail* to derive the stability constraint. In practice, this is the most viable option due to the fact that the value of the input at t_{k+N} is neither a decision variable in the current QP nor available from the solution of the previous QP.

At this point, let us perform a coordinate transformation

$$\begin{pmatrix} \eta_s \\ \eta_u \end{pmatrix} = \boldsymbol{T}^* \boldsymbol{\eta},$$

with $\boldsymbol{T}^*$ chosen in such a way that the unstable component[4] η_u evolves according to

$$\dot{\eta}_u = \lambda_u^* \eta_u + f_u^*, \tag{12}$$

with $\lambda_u^* > 0$ is the positive eigenvalue of $\boldsymbol{A}_\eta^*$ in the time-invariant linearized pitch dynamics (11) and f_u^* is the corresponding constant drift on η_u.

As shown in [16], in spite of the instability of (12) the trajectory of $\boldsymbol{\eta}$ in (11) is guaranteed to be bounded provided that the following stability condition is satisfied

$$\eta_u^{k+N} = -\int_{t_{k+N}}^{\infty} e^{-\lambda_u^*(\tau - t_{k+N})} f_u^* \, d\tau = -\frac{f_u^*}{\lambda_u^*}. \tag{13}$$

This is a condition on the terminal state of the prediction model (9) which is easily rewritten as a constraint on the control inputs inside the control horizon.

C. Input constraints

The input constraints account for the physical limitations of the robot actuators, which take the form

$$\mathrm{abs}(\boldsymbol{\tau}^{k+i}) \leq \boldsymbol{\tau}_M \qquad i = 0, ..., N-1, \tag{14}$$

where $\mathrm{abs}(\cdot)$ denotes component-wise absolute value and $\boldsymbol{\tau}_M$ is the vector of symmetric torque limits. In view of the nonlinear mapping (5) between torques $\boldsymbol{\tau}$ and inputs $\boldsymbol{u}$, to transform (14) into a linear constraint on $\boldsymbol{u}$ we simply evaluate the mapping along the auxiliary trajectory, obtaining

$$(\bar{\boldsymbol{\Psi}}^{k+i})^{-1} \bar{\boldsymbol{\alpha}}^{k+i} - \boldsymbol{\tau}_M \leq (\bar{\boldsymbol{\Psi}}^{k+i})^{-1} \boldsymbol{u}^{k+i} \leq (\bar{\boldsymbol{\Psi}}^{k+i})^{-1} \bar{\boldsymbol{\alpha}}^{k+i} + \boldsymbol{\tau}_M \tag{15}$$

where $\bar{\boldsymbol{\Psi}}^{k+i} = \boldsymbol{\Psi}(\bar{\boldsymbol{x}}^{k+i})$ and $\bar{\boldsymbol{\alpha}}^{k+i} = \boldsymbol{\alpha}(\bar{\boldsymbol{x}}^{k+i})$.

Since in the auxiliary trajectory we have set $\bar{x}^k = x^k$, the linear constraint (15) on the first input $\boldsymbol{u}^k$ — which is the only one actually applied in a MPC algorithm — is exactly equivalent to the original constraint (14) on $\boldsymbol{\tau}^k$.

[3] Note that this is conceptually different from enforcing a box constraint on ϕ, which would unnecessarily restrict the range of admissible motions for the system.

[4] The unstable component η_u plays in our context a similar role to that of the capture point [22] (also known as the divergent component of motion [23]) in humanoid locomotion.

D. QP formulation

The cost function of our QP is built by adding three different terms aimed at solving the tracking problem.

The first term is designed in such a way to implement a kinematic control law for the task variables $\boldsymbol{r}$. In particular, we let

$$\dot{\boldsymbol{r}}_r = \dot{\boldsymbol{r}}_d + \boldsymbol{K}\,(\boldsymbol{r}_d - \boldsymbol{r}), \tag{16}$$

where $\boldsymbol{K}$ is a positive definite matrix, and ask IS-MPC to generate inputs that force $\dot{\boldsymbol{r}}$ to track[5] $\dot{\boldsymbol{r}}_r$. Based on this, we let

$$L_r = \sum_{i=1}^{N} \|\dot{\boldsymbol{r}}^{k+i} - \dot{\boldsymbol{r}}_r^{k+i}\|^2_{\boldsymbol{W}_r}$$

$$= \sum_{i=1}^{N} \left\| \boldsymbol{J}(\bar{\boldsymbol{q}}^{k+i})\,\boldsymbol{\nu}^{k+i} - (\dot{\boldsymbol{r}}_d^{k+i} + \boldsymbol{K}\,(\boldsymbol{r}_d^{k+i} - \boldsymbol{h}(\bar{\boldsymbol{q}}^{k+i}))) \right\|^2_{\boldsymbol{W}_r}$$

where $||\cdot||_{\boldsymbol{W}}$ denotes the 2-norm weighted by matrix $\boldsymbol{W}$. Note that:

- the nonlinearity of $\boldsymbol{J}(\cdot)$ and $\boldsymbol{h}(\cdot)$ is dealt with by evaluating these terms over the auxiliary trajectory $\bar{\boldsymbol{q}}^{k+i}$;
- thanks to the partial feedback linearization, all components of the velocity vector $\boldsymbol{\nu}^{k+i}$ depend linearly on the transformed input samples (i.e., the decision variables at t_k), except for v_ϕ, which is then replaced by the corresponding term in $\bar{\boldsymbol{\nu}}^{k+i}$.

To regularize the optimization problem and to decrease the control effort, the cost function includes a second term

$$L_u = \sum_{i=0}^{N-1} \|\boldsymbol{u}^{k+i}\|^2_{\boldsymbol{W}_u}.$$

Finally, a third term introduces preferred positions and velocities for the arm joint coordinates

$$L_a = \sum_{i=0}^{N-1} \|\boldsymbol{q}_{r,l}^{k+i}\|^2_{\boldsymbol{W}_p} + \|\dot{\boldsymbol{q}}_{r,l}^{k+i}\|^2_{\boldsymbol{W}_v}.$$

The IS-MPC algorithm solves at each iteration the following QP problem:

$$\left\{ \begin{array}{c} \min\limits_{\boldsymbol{u}^k,\ldots,\boldsymbol{u}^{k+N-1}} L_r + L_u + L_a \\ \text{subject to:} \\ \bullet \text{ stability constraint (13)} \\ \bullet \text{ input constraints (15), } i = 0,\ldots,N-1. \end{array} \right.$$

To extend this formulation, one can easily add state constraints by rewriting them as linear constraints on the decision variables by means of the prediction model (9).

As usual with MPC, only the first input of the solution is applied to the robot. To this end, the first sample $\boldsymbol{u}^k$ of the optimal input is used to compute the joint torque command

$$\boldsymbol{\tau}^k = \boldsymbol{\Psi}^{-1}(\boldsymbol{x}^k)(\boldsymbol{u}^k - \boldsymbol{\alpha}(\boldsymbol{x}^k)). \tag{17}$$

The remaining input samples in the solution are used to predict the auxiliary trajectory for the next iteration.

V. Simulation results

The proposed method was validated by MATLAB simulations on the ALTER-EGO WIP robot shown in Fig. 1. For each arm, we have considered only two degrees of freedom, namely the shoulder pitch angle and the elbow angle. The robot has an overall mass $m = 21.32$ kg, with the arm links contributing respectively for 1.8 kg and 1.0 kg. The wheel radius is $R = 0.13$ m and the semi-distance between the wheels is $a = 0.248$ m. The torque limits are set to ±10 N·m for the wheels, ±6 N·m for the shoulders and ±1.1 N·m for the elbows.

The sampling time is chosen as $\delta = 0.02$ s while the MPC control horizon is $T_c = 0.5$ s. Each QP contains 150 decision variables and is solved using the `quadprog` function in MATLAB. The entire control loop runs in real-time (i.e., each iteration requires less than 0.02 s) on a standard laptop computer.

A. Navigation task

In the first simulation, the proposed method is applied to a navigation task. The desired ground trajectory is a sine wave in x, y, with an amplitude of 0.2 m and a a wavelenght of 1 m. The desired velocity along the x direction is 0.3 m/s.

As explained in Sect. II-C, for navigation tasks it is necessary to use as task function the position of a point displaced from the base; in particular, we have set $d = 0.2$ m.

The weight matrices used in the QP cost function are $\boldsymbol{W}_r = 50\,\boldsymbol{I}$, $\boldsymbol{W}_u = \mathrm{diag}\{0.05, 0.05, 1, 1, 1, 1\}$ and $\boldsymbol{W}_p = \boldsymbol{W}_v = \boldsymbol{I}$, while the gain in eq. (16) is set to $\boldsymbol{K} = \boldsymbol{I}$.

Figure 4 is a stroboscopic view of the generated motion, while Fig. 5 shows the norm of the tracking error e_{xy} and the input torques. At the start of the simulation, there is an initial error due to the fact that the robot state is not matched with the desired trajectory. This error increases at first, a behavior which represents the typical undershoot characterizing the response of non-minimum phase systems. In fact, in order to achieve a stable forward acceleration of the base, the robot has to first move backwards so as to tilt the pitch angle in the forward direction. After this transient, the desired trajectory is followed with good accuracy.

The torque plots show that the arm actuators are actively involved in the constrained minimization of the cost function, confirming the whole-body nature of our approach.

An animation of the generated motion is included in the accompanying video, where it is also shown that removing the stability constraint leads to the robot immediately losing balance. The video also contains simulation results for two additional navigation scenarios: tracking a circular ground trajectory and reaching a navigation set-point (not shown here for lack of space). See also https://youtu.be/PyyoZNOeklE.

B. Loco-manipulation task

The second simulation deals with a loco-manipulation task. In particular, we assign as desired trajectory for the right hand a helix with a radius $r = 0.2$ m, placed at an average height of 0.9 m from the ground.

[5] In fact, one may easily verify that if $\dot{\boldsymbol{r}}$ converges to $\dot{\boldsymbol{r}}_r$ as given by (16), then $\boldsymbol{r}$ will converge to the desired trajectory $\boldsymbol{r}_d(t)$.

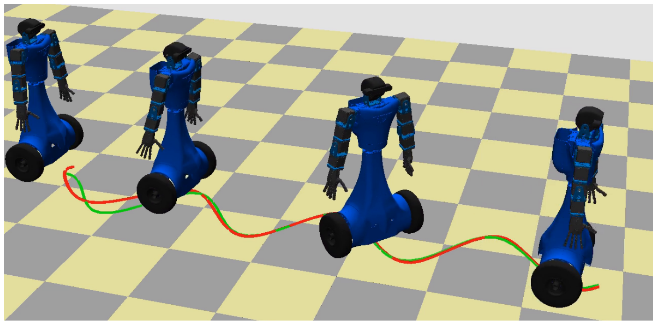

Fig. 4. Navigation task with ALTER-EGO: Stroboscopic view of the generated motion with actual (red) and desired (green) task trajectories.

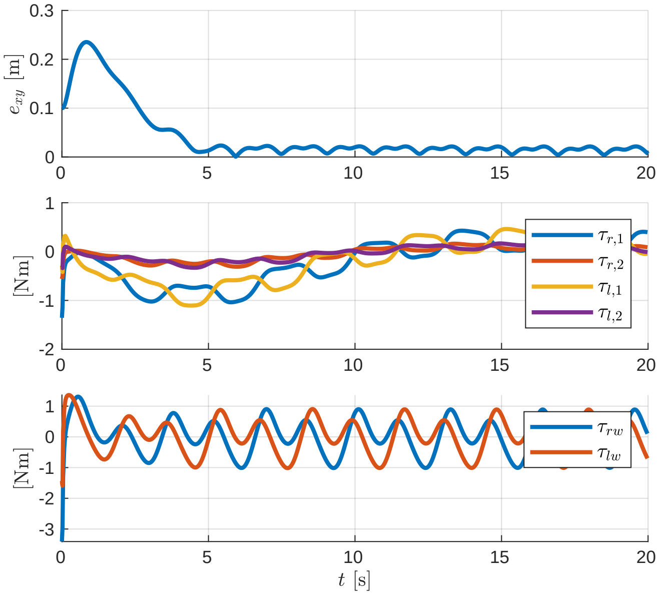

Fig. 5. Navigation task with ALTER-EGO: tracking error norm (top), arm torques (center), wheel torques (bottom). Torque limits are not shown since all actuators are far from saturation.

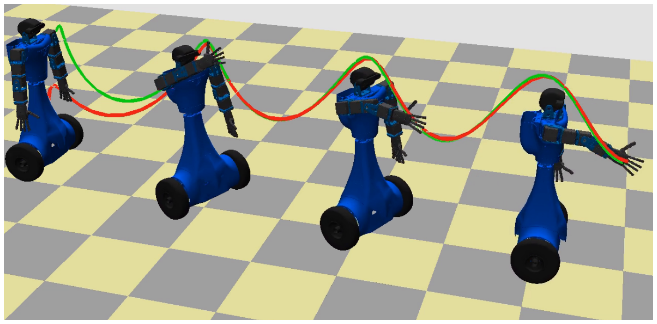

Fig. 6. Loco-manipulation task with ALTER-EGO: Stroboscopic view of the generated motion with actual (red) and desired (green) task trajectories.

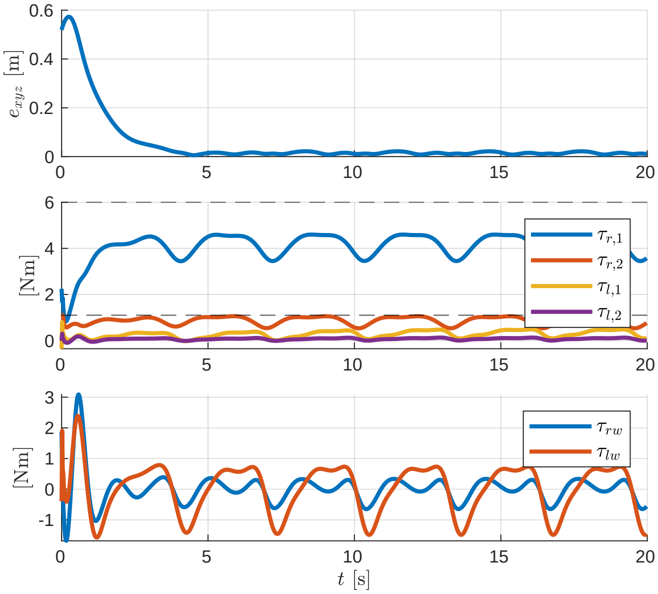

Fig. 7. Loco-manipulation task with ALTER-EGO: tracking error norm (top), arm torques (center), wheel torques (bottom). Arm torque limits are shown by dashed lines, while wheel torque limits are outside the plot range.

The weight matrices for the QP cost function are $\boldsymbol{W}_r = 20\,\boldsymbol{I}$, $\boldsymbol{W}_u = \mathrm{diag}\{0.1, 0.025, 0.05, 0.01, 1, 1\}$, $\boldsymbol{W}_p = \mathrm{diag}\{10^{-4}, 0.8, 1, 1\}$ and $\boldsymbol{W}_v = \mathrm{diag}\{0.01, 0.1, 1, 1\}$, while the gain in eq. (16) is again set to $\boldsymbol{K} = \boldsymbol{I}$.

A stroboscopic view of the generated motion is shown in Fig. 6, with the tracking error norm e_{xyz} and the input torques reported in Fig. 7. Again, the tracking error exhibits an initial transient after which is reduced essentially to zero, with some negligible fluctuations. In the right arm, the shoulder torque is doing most of work, by hovering around 4 N·m to compensate gravity, while the elbow torque saturates to the upper limit, fully confirming the validity of the input constraint transformation procedure discussed in Sect. IV-C. Left arm torques are much smaller, but still actively involved in motion generation.

Animation of this motion is also included in the accompanying video, along with two additional loco-manipulation scenarios concerning the right hand: tracking a mid-air circular trajectory and reaching a certain hand posture. See also https://youtu.be/PyyoZNOeklE.

VI. Conclusion

We presented a general task-oriented MPC algorithm for a WIP robot with arms, which uses an explicit stability constraint to handle the unstable pitch dynamics. We have discussed its application to navigation and loco-manipulation tasks, and validated its performance by simulations in such scenarios. Results show that the proposed approach generates stable motions that guarantee accurate task tracking. Our naive MATLAB implementation on a standard laptop computer is computationally very efficient, so one can confidently expect an optimized C++ implementation to provide full real-time performance on an experimental platform.

Future work will target experimental validation, a nonlinear formulation of the stability constraint, the analysis of recursive feasibility, and the use of NMPC methods such as Real Time Iteration.

References

[1] H. G. Nguyen, J. Morrell, K. D. Mullens, A. B. Burmeister, S. Miles, N. Farrington, K. M. Thomas, and D. W. Gage, "Segway robotic mobility platform," in *Mobile Robots XVII*, vol. 5609. SPIE, 2004, pp. 207–220.

[2] R. O. Ambrose, R. T. Savely, S. M. Goza, P. Strawser, M. A. Diftler, I. Spain, and N. Radford, "Mobile manipulation using NASA's Robonaut," in *2004 IEEE Int. Conf. on Robotics and Automation*, 2004, pp. 2104–2109.

[3] P. Deegan, B. J. Thibodeau, and R. Grupen, "Designing a self-stabilizing robot for dynamic mobile manipulation," Massachusetts University Amherst, Dept. of Computer Science, Tech. Rep., 01 2006.

[4] S. Jeong and T. Takahashi, "Wheeled inverted pendulum type assistant robot: design concept and mobile control," *Intelligent Service Robotics*, vol. 1, no. 4, pp. 313–320, 2008.

[5] M. Stilman, J. Olson, and W. Gloss, "Golem Krang: Dynamically stable humanoid robot for mobile manipulation," in *2010 IEEE Int. Conf. on Robotics and Automation*, 2010, pp. 3304–3309.

[6] U. Nagarajan, G. Kantor, and R. Hollis, "The ballbot: An omnidirectional balancing mobile robot," *The Int. Journal of Robotics Research*, vol. 33, no. 6, pp. 917–930, 2014.

[7] G. Lentini, A. Settimi, D. Caporale, M. Garabini, G. Grioli, L. Pallottino, M. G. Catalano, and A. Bicchi, "ALTER-EGO: a mobile robot with a functionally anthropomorphic upper body designed for physical interaction," *IEEE Robotics & Automation Magazine*, vol. 26, no. 4, pp. 94–107, 2019.

[8] Y. Zhao, C. Woo, and J. Lee, "Balancing control of mobile manipulator with sliding mode controller," in *15th Int. Conf. on Control, Automation and Systems*, 2015, pp. 802–805.

[9] Y.-G. Bae and S. Jung, "Balancing control of a mobile manipulator with two wheels by an acceleration-based disturbance observer," *Int. Journal of Humanoid Robotics*, vol. 15, no. 03, p. 1850005, 2018.

[10] G. Zambella, G. Lentini, M. Garabini, G. Grioli, M. G. Catalano, A. Palleschi, L. Pallottino, A. Bicchi, A. Settimi, and D. Caporale, "Dynamic whole-body control of unstable wheeled humanoid robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3489–3496, 2019.

[11] C. Acar and T. Murakami, "Multi-task control for dynamically balanced two-wheeled mobile manipulator through task-priority," in *2011 IEEE Int. Symposium on Industrial Electronics*, 2011, pp. 2195–2200.

[12] M. Zafar and H. I. Christensen, "Whole body control of a wheeled inverted pendulum humanoid," in *16th IEEE-RAS Int. Conf. on Humanoid Robots*, 2016, pp. 89–94.

[13] M. Yue, C. An, and J.-Z. Sun, "An efficient model predictive control for trajectory tracking of wheeled inverted pendulum vehicles with various physical constraints," *Int. Journal of Control, Automation and Systems*, vol. 16, no. 1, pp. 265–274, 2018.

[14] M. Zafar, S. Hutchinson, and E. A. Theodorou, "Hierarchical optimization for whole-body control of wheeled inverted pendulum humanoids," in *2019 IEEE Int. Conf. on Robotics and Automation*, 2019, pp. 7535–7542.

[15] M. V. Minniti, F. Farshidian, R. Grandia, and M. Hutter, "Whole-body MPC for a dynamically stable mobile manipulator," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3687–3694, 2019.

[16] N. Scianca, D. De Simone, L. Lanari, and G. Oriolo, "MPC for humanoid gait generation: Stability and feasibility," *IEEE Transactions on Robotics*, vol. 36, no. 4, pp. 1171–1188, 2020.

[17] M. Beglini, L. Lanari, and G. Oriolo, "Anti-jackknifing control of tractor-trailer vehicles via intrinsically stable MPC," in *2020 IEEE Int. Conf. on Robotics and Automation*, 2020, pp. 8806–8812.

[18] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer, 2009.

[19] S. Kim and S. Kwon, "Dynamic modeling of a two-wheeled inverted pendulum balancing mobile robot," *Int. Journal of Control, Automation and Systems*, vol. 13, no. 4, pp. 926–933, 2015.

[20] U. Nagarajan and R. Hollis, "Shape space planner for shape-accelerated balancing mobile robots," *The Int. Journal of Robotics Research*, vol. 32, no. 11, pp. 1323–1341, 2013.

[21] M. Beglini, T. Belvedere, L. Lanari, and G. Oriolo, "An intrinsically stable MPC approach for anti-jackknifing control of tractor-trailer vehicles," *IEEE/ASME Transactions on Mechatronics*, 2022 (to appear).

[22] J. Pratt, J. Carff, S. Drakunov, and A. Goswami, "Capture point: A step toward humanoid push recovery," in *6th IEEE-RAS Int. Conf. on Humanoid Robots*, 2006, pp. 200–207.

[23] T. Takenaka, T. Matsumoto, and T. Yoshiike, "Real time motion generation and control for biped robot - 1st report: Walking gait pattern generation," in *2009 Int. Conf. on Intelligent Robots and Systems*, 2009, pp. 1084–1091.