Model-predictive control in multi-contact based on stability polyhedrons

Hervé Audren, Abderrahmane Kheddar

Model-predictive control in multi-contact based on stability polyhedrons

Hervé Audren and Abderrahmane Kheddar

***Abstract*— We propose a new linear model-predictive control scheme in multi-contact based on the center of mass reduced model. In order to linearize the dynamics of the CoM, we exploit the notion of stability polyhedrons associated to given contact stances, inside which the existence of contact forces is guaranteed. To compute stability polyhedrons, we have first to specify a convex polytope inside which the center of mass's acceleration lies. We then generate a minimum jerk trajectory inside these successive stability polyhedrons that also yields contact transitions timings, and integrate it with our quadratic programming whole-body controller as part of the multi-objective tasks.**

I. Introduction

In this paper we introduce a model-predictive control (MPC) approach to generate multi-contact motion. Typical MPC adopt an hierarchical approach, see e.g. in [1], [2], [3], [4]: they compute trajectories to be best tracked by the underlying local whole-body controller –see e.g. in [5], [6]. Later controller decisions (in terms of desired instant posture position or torque) may impact balance in future motions. The most part of the MPC computed trajectories are discarded and only their newly computed head part (e.g. identified by a time interval) is fed to the whole-body controller. Indeed, we compensate for uncertainties and perturbations by recomputing very often the trajectory from the real state of the robot (and its interaction with the environment). Thus, the computation time must be small with respect to the trajectory length. A lot of approaches from earlier cited works use a simplified linear model that yields convex optimization problems.

In this work, we first recall the notion of robust stability [7] that is used in our MPC: it consists of defining a convex region of acceptable center of mass (CoM) acceleration $\mathcal{G}$, from which we can deduce an acceptable polyhedron of CoM positions $\mathcal{P}$. This allows us to formulate a linear minimum jerk problem in multi-contact and in 3 dimensions without further assumptions. Moreover, as the resulting problem can be solved fast, we extend the formulation to search for the best contact transition timings.

In the context of legged robotics, perhaps the most well-known application of MPC is [8] in which the authors optimized a Zero-Moment-Point (ZMP) trajectory. As it relied on the ZMP, it is however not applicable to multi-contact motion. This approach was extended in [9] where the authors optimized a jerk trajectory under constraints. Similarly, it was limited to walking on flat surfaces.

Since then, we have proposed a multi-contact MPC formulation [2]. However, in order to linearize the CoM dynamics, we had to specify the motion along one axis, leave one component of the momentum uncontrolled and specify stance transition timings in a heuristic (non satisfactory) way. In later work [10], the above restrictions were lifted in favor of generating momentum trajectories, but the algorithm no longer performed well enough for real-time operation.

In other approaches [11], [12] parallelism and general purpose GPU were exploited to solve very quickly a whole-body non-linear MPC problem giving a set of contact transitions. Unfortunately, it is hard to generate long enough whole-body trajectories in real-time, even on powerful computers. Also non-linear optimization solvers are often hard to tune and gradient-based ones suffer from local minima.

Finally, some MPC aim at generating spatial trajectories by solving non-linear problems, and then using Time Optimal Path Parametrization [13], [14], [15] to determine at which speed to execute the trajectory. Performing TOPP is fast, but solving the non-linear spatial problem can be costly unless specific assumptions or simplifications are made. Moreover, optimizing separately the kinematics from the dynamics can yield suboptimal solutions in terms of execution times. In our formulation, once the slightly expensive robust stability regions have been determined, each subsequent optimization is very fast, which makes it a great fit for MPC formulations.

We first recall in section II how we define and compute the stability regions in which CoM trajectories are generated. Then our contributions stands as follows:

- We introduce a linear, minimum jerk MPC formulation in section III, that aims at optimizing the spatial trajectory and finding stances transition timings;
- this MPC trajectory generator is then coupled with our whole-body, tasks-based controller in section IV;
- finally, simulation results are discussed in section V.

II. Robust stability

We recall briefly few results of our work in [7] in which we extended the notion of static equilibrium in multi-contact to the notion of *robust* equilibrium with respect to a predefined acceleration polytope. To do so, we used a flywheel model where the robot is reduced to a single rigid body mass (m):

Definition 1. *A CoM position c is said to be in robust equilibrium w.r.t. to a convex acceleration polytope $\mathcal{G}$ if and only if there exists contact forces f_i applied at points r_i that*

This work is supported by H2020 European project COMANOID RIA No: 645097 and by JSPS Grant-in-Aid for Scientific Research (B) Number 16H02886 ("Cutting-Edge multi-contact behaviors"

H. Audren and A. Kheddar are with CNRS-AIST JRL UMI3218/RL, Tsukuba, Japan and the CNRS-UM LIRMM, Interactive Digital Human group, Montpellier, France

satisfy:

$$\forall \ddot{c} \in \mathcal{G}, \exists f \text{ s.t. } \begin{cases} \sum f_i = m(\ddot{c} - g) \\ \sum f_i \times (c - r_i) = 0 \\ \|Bf\| \leq u^T f \end{cases} \quad (1)$$

where B, u define the non-linear friction cones relation, g is the gravity vector, u stacks the contacts' normals.

Considering that the contact points r_i are fixed, the above equations define a system of convex equalities and inequalities. It entails that there exists a convex shape, $\mathcal{P}$ such that:

Definition 2. *$\forall c \in \mathcal{P}$, c is in robust equilibrium w.r.t $\mathcal{G}$.*

We then show that by adapting the recursive projection algorithm [16] it is possible to obtain an explicit representation of $\mathcal{P}$, that is to say a set of inequalities $H_\mathcal{P}, b_\mathcal{P}$ such that:

$$\left\{c \in \mathbb{R}^3 | H_\mathcal{P} c \leq b_\mathcal{P}\right\} = \mathcal{P} \quad (2)$$

Note that the contact forces f do not appear in the above criteria: by restricting the possible accelerations to a known convex polytope $\mathcal{G}$, we have transformed the original non-linear problem eq. (1) into a linear one, eq. (2).

III. Polyhedron MPC

From the previous section, we know, given a convex acceleration polytope $\mathcal{G}$, how to compute a convex CoM polyhedron $\mathcal{P}$ such that: if $c \in \mathcal{P}$ and $\ddot{c} \in \mathcal{G}$, there exist contact forces that realize that acceleration. We now see how to exploit that fact to generate the CoM trajectories online.

We define that a motion is in robust equilibrium if and only if (iff) for every time instant t:

$$\forall t \quad c(t) \in \mathcal{P}(t) \text{ and } \ddot{c}(t) \in \mathcal{G}(t) \quad (3)$$

Given that $\mathcal{P}$ and $\mathcal{G}$ are entirely determined by the contact geometry and friction cones, we can denote $\mathcal{C}(t)$ the active contacts at the time t. The above condition thus write:

$$\forall t \quad c(t) \in \mathcal{P}\left(\mathcal{C}(t)\right) \text{ and } \ddot{c} \in \mathcal{G}\left(\mathcal{C}(t)\right) \quad (4)$$

Thus, to generate a trajectory we need to not only optimize the CoM trajectory itself ($c(t)$, $\ddot{c}(t)$) but the instants at which the contacts change ($\mathcal{C}$). As changing contacts is a discrete event, we parametrize those changes with integer variables.

A. Formulation

We consider a number of *stances*, i.e. consecutive sets of contacts, $\mathcal{C}_i$. We discretize the times into t_k. We can thus write that:

$$\forall k \quad c_k \in \mathcal{P}\left(\mathcal{C}_{i(k)}\right) \text{ and } \ddot{c} \in \mathcal{G}\left(\mathcal{C}_{i(k)}\right) \quad (5)$$

where $i(k)$ represents the active contact set index at instant k. It is a function over finite sets:

$$i : [\![0 \,..\, K]\!] \mapsto [\![0 \,..\, N]\!] \quad (6)$$
$$k \mapsto i(k) \quad (7)$$

Our problem is formulated as a minimum jerk cost. Thus let us first form the evolution equation:

$$\begin{bmatrix} c_{k+1} \\ \dot{c}_{k+1} \\ \ddot{c}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & dt & \frac{dt^2}{2} \\ 0 & 1 & dt \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_k \\ \dot{c}_k \\ \ddot{c}_k \end{bmatrix} + \begin{bmatrix} \frac{dt^3}{6} \\ \frac{dt^2}{2} \\ dt \end{bmatrix} \dddot{c}_k \quad (8)$$

$$x_{k+1} = Ax_k + B\dddot{c}_k \quad (9)$$

We can thus repeatedly integrate the above with initial condition x_0 to obtain the whole trajectory:

$$X = \begin{bmatrix} x_1^T & x_2^T & \dots & x_K^T \end{bmatrix}^T \quad (10)$$

$$= \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^K \end{bmatrix} x_0 + \begin{bmatrix} B & & & \\ AB & B & & \\ & & \ddots & \\ A^{K-1}B & A^{K-2}B & \dots & B \end{bmatrix} \begin{bmatrix} \dddot{c}_0 \\ \dddot{c}_1 \\ \vdots \\ \dddot{c}_n \end{bmatrix} \quad (11)$$

$$= \Phi x_0 + \Psi U \quad (12)$$

We can thus build a quadratic cost that is the distance to a reference and the weighted norm of the command:

$$\|X - \bar{X}\|^2 + W_u \|U\|^2 = U^T Q U + 2c^T U + \gamma \quad (13)$$

We introduce the positive, diagonal matrix W_x to weight the diffent parts of X independently:

$$Q = \Psi^T W_x \Psi + W_u \quad (14)$$
$$c^T = \left(x_0^T \Phi^T - \bar{X}^T\right) W_x \Psi \quad (15)$$

We can thus formulate the problem as:

$$\min_{U, i(k)} \ \frac{1}{2} U^T Q U + c^T U \quad (16)$$

$$\text{s.t. } \forall k \in [\![0 \,..\, K]\!] \quad \begin{matrix} c_k \in \mathcal{P}\left(\mathcal{C}_{i(k)}\right) \\ \ddot{c}_k \in \mathcal{G}\left(\mathcal{C}_{i(k)}\right) \end{matrix} \quad (17)$$

We obtained a quadratic objective, but the optimization is over a function, and the constraints are non-linear. The next sections present how to properly select the function $i(k)$.

B. Choosing a timing function

The family of functions $i(k)$ are defined over finite integer intervals. Thus, one could exhaustively search all $(N+1)^K$ of them. But this number is considerably large even for small values of N and K. Moreover, not any function can fit: we should go from one polyhedron to the next, without skipping one and without going backwards.

Hence, instead of looking at all possible $i(k)$, we consider a specific shape of $i(k)$: a strictly increasing by one function. That is to say, a stair-shaped function. It is defined by the τ_i that indicate at which time we switch from $\mathcal{P}_i$ to $\mathcal{P}_{i+1}$. Instead of looking for the value of K variables in $[\![0 \,..\, N]\!]$, we have to find the value of N variables in $[\![0 \,..\, K]\!]$, but those variables are known to be strictly increasing, which drastically reduces the size of the search space. See fig. 1 for a simple example of such a function.

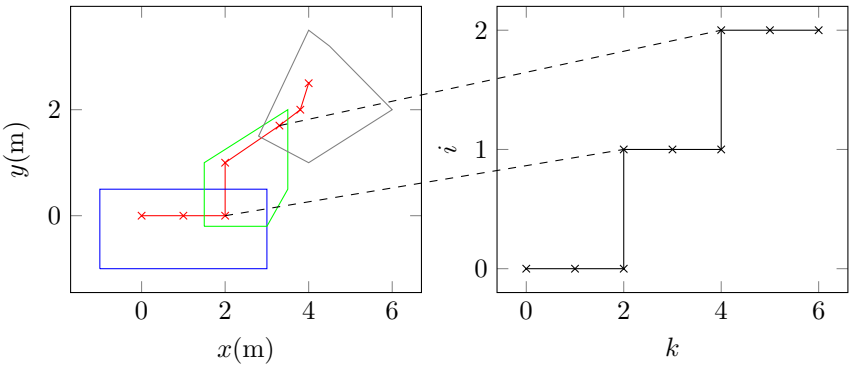

Fig. 1. Schema of the polyhedron CoM preview control. Left shows the spatial trajectory going through three different polygons. Right shows the evolution of the integer variable with time.

The actual number of possible functions is given by:

$$\sum_{i=1}^{N+1} (p_i(K)i!) \tag{18}$$

Where p stands for the partition function (see e.g. [17]).

Indeed the partition function tells us how many tuples of integers of a certain size sum to K. Those integers represent the duration of a step. As we consider N switching times, we are looking for the number of partitions of size $N+1$. We then account for the fact that we might not switch, and thus sum over all partitions of size less or equal than $N+1$. Finally, we consider that there are $i!$ possible orderings of a partition of size i.

A numerical application for a typical scenario where $K = 15$ and $N = 2$ illustrates the huge reduction, 5 orders of magnitude, in the size of the search space:

- For the complete space search: $(N+1)^K = 14348907$.
- For the function search eq. (18): 129.

Whenever the transition times τ_i are fixed, the problem is reduced to a simple QP in few variables (typically a few dozens) and a lot of constraints (a few thousands).

Indeed, as c_k and $\ddot{c}_k$, are part of our state, eq. (5) can be written as linear constraints in U:

$$H_{\mathcal{P}_{i(k)}} c_k \leq b_{\mathcal{P}_{i(k)}} \tag{19}$$

$$\Leftrightarrow H_{\mathcal{P}_{i(k)}} S_{c_k} (\Phi x_0 + \Psi U) \leq b_{\mathcal{P}_{i(k)}} \tag{20}$$

$$\Leftrightarrow H_{\mathcal{P}_{i(k)}} S_{c_k} \Psi U \leq b_{\mathcal{P}_{i(k)}} - \Phi x_0 \tag{21}$$

And:

$$H_{\mathcal{G}_{i(k)}} \ddot{c}_k \leq b_{\mathcal{G}_{i(k)}} \tag{22}$$

$$\Leftrightarrow H_{\mathcal{G}_{i(k)}} S_{\ddot{c}_k} \Psi U \leq b_{\mathcal{G}_{i(k)}} - \Phi x_0 \tag{23}$$

With S_{c_k} and $S_{\ddot{c}_k}$ selection matrices that extract c_k and $\ddot{c}_k$ from X respectively.

This can be solved in milliseconds (using the quadratic program QL solver [18]), and thus exploring all the possible τ_i for a small number of transitions and instants, i.e. $N = 2$ and $K = 15$, can be done in about $500\,\mathrm{ms}$.

To further reduce the search space, we apply the following strategy: for each choice of τ_0, we test if $\tau_1 = N$ is feasible. If it is, we explore all possible τ_1 starting from N and working backwards towards τ_0 until the problem is no longer feasible. This amounts to discarding all solutions where it is not possible to stop in the intermediate polyhedron. As long as we consider a symmetric $\mathcal{G}$ that contains the gravity g, this is not highly restrictive. Indeed, by symmetry, at any point, decelerating as much as we previously accelerated is feasible while containing g means that stopping is a solution. To confirm this fact, we solved a hundred MPCs. In $97\,\%$ of the cases, there was no loss of optimality and in none of them did the reduction of the search space result in infeasibility.

We then select the solution (X, τ_0, τ_1) that yields the smallest cost. This easily extends to more τ_i but the combinatorial nature of the problem makes it quickly intractable. This method is sumarized in algorithm 1.

Algorithm 1 Algorithm `MINTIM` to determine the optimal timings

```
for τ0 ∈ ⟦0 .. K⟧ do
    for τ1 ∈ ⟦K .. τ0 + 1⟧ do
        res ← solveQP(τ0, τ1)
        if res then
            X, cost ← res
        else
            break
        end if
    end for
end for
return argmin_{τ0,τ1,X}(cost)
```

Although exhaustive search is typically not the most efficient solution, it is the best that we have implemented. Hill-climbing [19] is not adapted to our problem as the objective function (the optimal cost of our QP) is *not* convex. Simulated annealing [20] is not well adapted to small search spaces, and requires tuning of the temperature and acceptance function. Finally, we can recast our problem as a Mixed-Integer Quadratic Program (MIQP). Indeed, we would like to express the following relation between constraints:

$$\tau_i < j \leq \tau_{i+1} \implies c_j \in \mathcal{P}_i \wedge \ddot{c}_j \in \mathcal{G}_i \tag{24}$$

Which can be transformed into linear constraints by introducing binary slack variables. However, our tentative implementation did not show any improvement in terms of computation time and failed to converge in some instances.

IV. Integration for multi-contact control

In this section, we present how we integrated the trajectory generation of the previous section within our multi-contact planning and control framework.

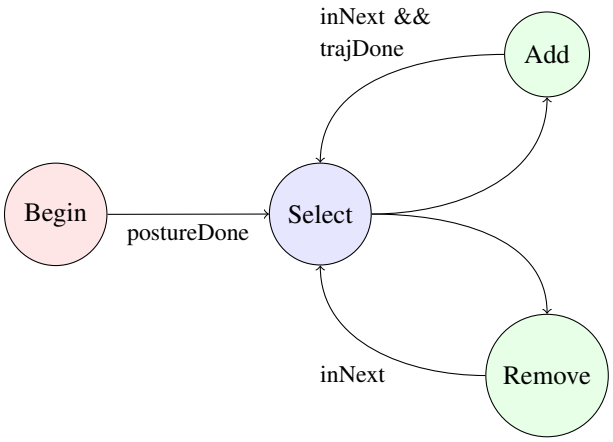

Fig. 2. Representation of the FSM that drives the whole-body controller

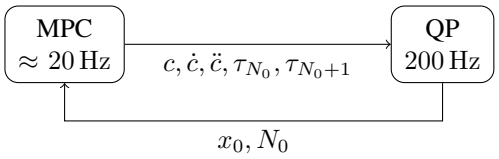

Fig. 3. MPC communication with the QP controller

First, we perform multi-contact planning [21], [22], [6] on our environment in order to obtain a sequence of stances. Each stance is composed of a set of contacts, a robot posture and an action. This action tells us if we are adding or removing a contact in this stance.

Then, from the contacts positions, we compute the polyhedrons $\mathcal{P}_i$ by appropriately setting the $\mathcal{G}_i$. We then set the number of stances to look ahead $N = 2$ and a length of the preview window $K = 10$. In this setting, we solve algorithm 1 in about $50\,\mathrm{ms}$.

We can now compute a CoM trajectory and timings by providing a starting state and a reference: $(X, \tau_0, \tau_1) = \texttt{MINTIM}(x_0, \bar{X})$. To do so, we connect the current state of the QP controller to the x_0 input. The $\bar{X}$ reference was chosen to be the simplest: it is a reference position (the endpoint) with zero velocity and acceleration. This means that we are only specifying where to go, but not giving any hints as to how to get there.

As for the gains tuning, we simply set W_u to $1e^{-4}$ and W_x to 1 from gathered experiments. Indeed, our primary objective is to reach the objective, while minimizing the jerk should have a lower priority.

Our tasks-base controller [22], [6] is built around solving a quadratic program in the generalized coordinates acceleration and contact forces, regrouped in the decision vector y:

$$\begin{aligned} \min_{y} \quad & \frac{1}{2} y^T Q y + c^T y \stackrel{\text{def}}{=} \sum_i \beta_i \mathcal{T}_i(y) \\ \text{s.t.} \quad & Ay \leq B \ ; \ Cy = D \end{aligned} \tag{25}$$

The task functions $\mathcal{T}$ allow us to express a variety of objectives in a quadratic form, and are weighted according to the β_i. In this paper, we only use the posture objective that minimizes the distance to a given posture and the CoM objective which tracks a CoM trajectory.

As for the constraints, they allow us to express a variety of physical limitations of the robot. Amongst them we are interested in:

- Joint limits constraints either in position, velocity or torque
- Discretized friction cones constraints
- Collision constraints

The controller is supervised by a three state finite state machine (FSM), illustrated in fig. 2. In each state, we always track the posture generated by the planner for the current stance, with a low weight. The states are:

1) Initial state: we simply track the posture of the initial stance.
2) Remove state: as we are removing a contact, the only additional task is to track the CoM trajectory while remaining in the current stance's polyhedron.
3) Add state: we are adding a contact, we insert two tasks to both track the CoM trajectory and track a spline for the end-effector, while remaining in the current polyhedron.

For the add state, the control points of this spline are determined by the geometry of the contacts while the time parametrization comes from the timings computed by the MPC. The spline passes through the initial contact position, with zero initial velocity and acceleration, a waypoint that is $10\,\%$ above the highest of the start and end points, and arrives at the final contact point with zero velocity and acceleration. From the MPC, we have the whole duration of the add phase, and thus specify that we pass through the waypoint at $30\,\%$ of the total duration. More complex settings would of course be necessary as the environment becomes more cluttered.

Now, we specify when to shift from one stance to the next:

1) Initial: whenever the posture error is low.
2) Remove: whenever we are inside the next stance's polyhedron.
3) Add: whenever we are inside the next stance's polyhedron and we are done executing the end-effector trajectory.

For each transition, the type and the contacts of the next stance are given by the plan.

Finally, the current FSM stance, N_0, is sent back to the MPC component as shown in fig. 3. This value is used as the starting stance for the MPC: instead of looking for values of $i(k)$ in $[\![0\,..\,N]\!]$, we look for them in $[\![N_0\,..\,N_0 + N]\!]$. This

TABLE I

TASKS PART OF THE COST AND TASKS PART OF THE CONSTRAINTS FOR THE QP CONTROLLER

Cost Tasks	Constraint Tasks
CoM Trajectory	CoM in polyhedron
Posture Task	Contact constraints
	Joint limits constraints (position and velocity)
	Collision avoidance constraints

means we only look for a small number of transition timings, and only advance the starting polyhedron / polytope, when the whole-body controller has realized the current action. The MPC thus waits until the whole-body controller has performed the action before advancing to the next stance.

V. RESULTS

The following sections present two illustrations of the framework presented above: crossing a ridge and climbing stairs. In both cases, HRP2-Kai is able to perform the motion if we do not enforce real-torque limitations. This means that indeed, there exists contact forces that allow the controller to track the MPC CoM trajectory. However, those contact forces are not always realizable by the whole-body system. Indeed, during trajectory generation, we only relied on a reduced model, that does not take into account joint torques as those depend non-linearly on the joint angles.

The controller is also unable to perform the end-effector trajectory in time, but this is not problematic: the CoM remains in the previous polyhedron as long as the new contact has not been established. The MPC then computes new timings based on the current state.

In both scenarii, we used a single $\mathcal{G}$ for all stances, that is the convex hull of lateral accelerations with a magnitude of $a = 0.5\,\mathrm{m\,s^{-2}}$:

$$\mathcal{G} = \{\{a, 0, 0\}, \{-a, 0, 0\}, \{0, a, 0\}, \{0, -a, 0\}\} \qquad (26)$$

We also set the duration of each time-step to $dt = 0.15\,\mathrm{s}$, which yields a total duration of $1.5\,\mathrm{s}$ for the trajectories.

A. Ridge crossing

In this simulation, we want to cross a narrow ridge, while pushing on a flat, non-horizontal, support for added stability. This is the same environment as in [2], but the multi-contact plan we used is newly generated. A few keyframes are available in fig. 4. At most, we have three unilateral contacts with the environment.

Compared to our previous work [2], the CoM trajectory that we obtain is much more conservative. This is most evident in the first stance: HRP2-Kai leans quite heavily backwards with its CoM above the right foot. That stance is composed of a single unilateral contact, thus the region of stability is very small at that time. Indeed, as $\mathcal{G}$ contains the gravity, we are rather statically stable at all times. For this particular stance, it means that the CoM has to remain above the right foot, at a low altitude. However, we no longer need to use a carefully tuned timing heuristic and the timings automatically adjust based on the current state. Moreover, we no longer specify the z motion, nor do we have to carefully choose a $\bar{X}$.

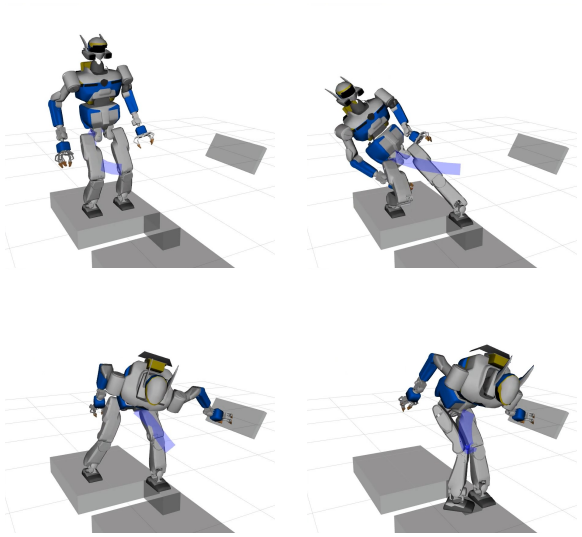

Fig. 4. Keyframes of HRP2-Kai crossing a ridge: the blue ribbon represents the current CoM trajectory being tracked.

B. Stairs climbing

In this scenario, HRP2-Kai is tasked with climbing a flight of stairs, using the handrail. Halfway up, it grasp again the handrail while using its other hand as unilateral support on the top platform. Thus, some stances have three unilateral contacts and a bilateral one. The resulting motion is illustrated in fig. 5.

We first remark that this motion appear much more plausible as compared to the previous scenario. The CoM trajectory is less constrained because the robot establishes more than one contact with the environment at all times: we do not produce the 'non-expected' poses of the previous scenario. However, as research on how humans decide to make or break contacts is scarce, it is difficult to quantify that feeling. It would seem coherent that humans use more contacts when confronted with a difficult and/or uncertain terrain. When walking under such conditions, humans indeed adopt stiffer, more static gaits [23].

Still, the motion is executed much faster than in the quasi-static case: we climb the whole flight in about $30\,\mathrm{s}$ whereas it took about $4\,\mathrm{min}$ in our previous work [24]. Indeed, we gain substantial time by not waiting for the CoM to reach a predetermined position. Yet, as this motion generates high-torques, we would probably have to slow it down to execute it on a real hardware, especially when approaching contact surfaces so as not to damage the robot in case of unpredicted contact.

VI. CONCLUSIONS AND FUTURE WORK

We have shown in this paper how the notion of stability regions allows us to generate a linear CoM MPC in multi-contact, without making any assumptions outside of restrictions on the possible CoM accelerations. Furthermore, we

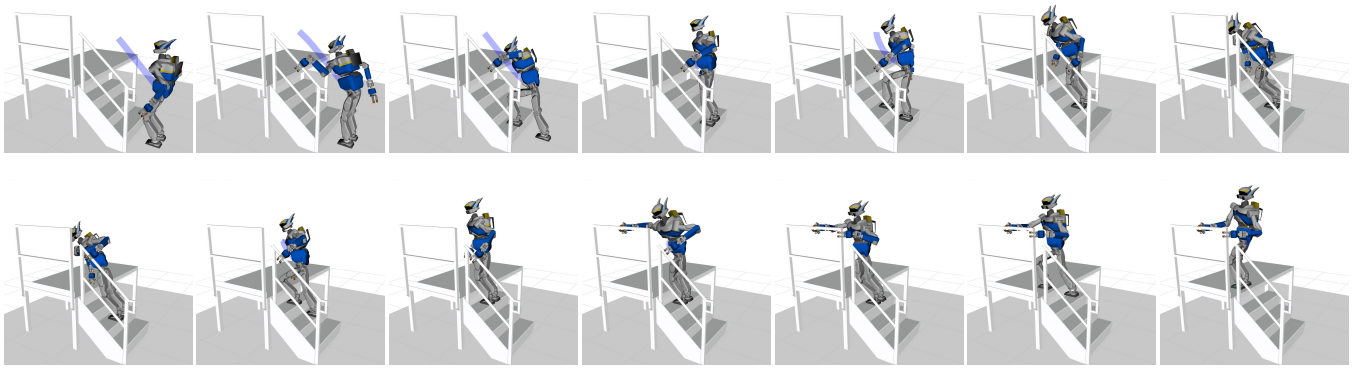

Fig. 5. Keyframes of HRP2-Kai climbing the stairs: the blue ribbon represents the current CoM trajectory.

exploit the linearity to solve many such problems at once, and find the optimal stance transitions timings. Finally, this approach was validated by simulation.

Yet, we have many avenues of improvement left open. Once all of the following points are solved, we implement our formulation on the real HRP2-Kai.

Firstly, we would like to improve even further the computation times. To do so, we could parallelize our algorithm in order to solve multiple QPs at the same time. We would probably gain the most by finding a better way to explore the possible timings than exhaustive search, for example by using MIQP. Our problem also has quite a specific form, and there is possibly a way to solve it faster.

Secondly, we need to make the trajectory generation trackable by the whole-body controller: this entails integrating part or whole of the whole-body constraints in either the trajectory generation or the polyhedron computation.

Finally, we have used only one $\mathcal{G}$ to constrain the accelerations. We would like to have it vary, for example through alternate optimizations, to obtain solutions that are no longer statically stable.

References

[1] P. M. Wensing and D. E. Orin, "Generation of dynamic humanoid behaviors through task-space control with conic optimization," in *IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany, 6-10 May 2013, pp. 3088–3094.

[2] H. Audren, J. Vaillant, A. Kheddar, A. Escande, K. Kaneko, and E. Yoshida, "Model preview control in multi-contact motion– application to a humanoid robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Chicago, Illinois, September 2014.

[3] C. Brasseur, A. Sherikov, C. Collette, D. Dimitrov, and P.-B. Wieber, "A robust linear MPC approach to online generation of 3d biped walking motion," in *IEEE-RAS International Conference onHumanoid Robots*, 2015, pp. 595–601.

[4] S. Caron and A. Kheddar, "Multi-contact walking pattern generation based on model preview control of 3d com accelerations," in *IEEE-RAS International Conference on Humanoid Robots*, November 2016.

[5] L. Righetti, J. Buchli, M. Mistry, M. Kalakrishnan, and S. Schaal, "Optimal Distribution of Contact Forces with Inverse-dynamics Control," *Int. J. Rob. Res.*, vol. 32, no. 3, pp. 280–298, 2013.

[6] J. Vaillant, A. Kheddar, H. Audren, F. Keith, S. Brossette, A. Escande, K. Bouyarmane, K. Kaneko, M. Morisawa, P. Gergondet, E. Yoshida, S. Kajita, and F. Kanehiro, "Multi-contact vertical ladder climbing with an hrp-2 humanoid," *Autonomous Robots*, vol. 40, no. 3, 2016.

[7] H. Audren and A. Kheddar, "3d robust stability polyhedron in multi-contact," *IEEE Transactions on Robotics*, vol. "Submitted", 2017. [Online]. Available: https://hal-lirmm.ccsd.cnrs.fr/lirmm-01477362

[8] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," in *IEEE International Conference on Robotics and Automation*, vol. 2, Sept 2003, pp. 1620–1626 vol.2.

[9] P.-B. Wieber, "Trajectory free linear model predictive control for stable walking in the presence of strong perturbations," in *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*. IEEE, 2006.

[10] A. Herzog, N. Rotella, S. Schaal, and L. Righetti, "Trajectory generation for multi-contact momentum-control," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, Nov. 2015.

[11] T. Erez, K. Lowrey, Y. Tassa, V. Kumar, S. Kolev, and E. Todorov, "An integrated system for real-time model predictive control of humanoid robots," in *Humanoid Robots (Humanoids), 2013 13th IEEE-RAS International Conference on*. IEEE, 2013, pp. 292–299.

[12] B. Chrétien, A. Escande, and A. Kheddar, "Gpu robot motion planning using semi-infinite nonlinear programming," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 10, Oct 2016.

[13] K. Hauser, "Fast interpolation and time-optimization on implicit contact submanifolds." in *Robotics: Science and systems*, 2013.

[14] M. Naveau, M. Kudruss, O. Stasse, C. Kirches, K. Mombaur, and P. Souères, "A reactive walking pattern generator based on nonlinear model predictive control," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 10–17, 2017.

[15] S. Caron and A. Kheddar, "Dynamic walking over rough terrains by nonlinear predictive control of the floating-base inverted pendulum," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, September 2017, to be presented.

[16] T. Bretl and S. Lall, "Testing static equilibrium for legged robots," *IEEE Transactions on Robotics*, vol. 24, no. 4, pp. 794–807, 2008.

[17] G. H. Hardy and E. M. Wright, *An introduction to the theory of numbers*. Oxford University Press, 1979.

[18] K. Schittkowski, "QL: A fortran code for convex quadratic programming - user's guide," University of Bayreuth, Tech. Rep., 2011.

[19] S. Russell and P. Norvig, *Artificial Intelligence: a Modern Approach*. Pearson, 2001.

[20] K. A. Dowsland, *Simulated annealing in Modern Heuristic Techniques for Combinatorial Problems*. McGraw-Hill, 1995, vol. 13, no. 2.

[21] A. Escande, A. Kheddar, and S. Miossec, "Planning contact points for humanoid robots," *Robotics and Autonomous Systems*, vol. 61, no. 5, pp. 428–442, 2013.

[22] K. Bouyarmane and A. Kheddar, "Humanoid robot locomotion and manipulation step planning," *Advanced Robotics*, vol. Special Issue on Cutting edge robotics in Japan, no. 26, pp. 1099–1126, 2012.

[23] G. Cappellini, Y. P. Ivanenko, N. Dominici, R. E. Poppele, and F. Lacquaniti, "Motor Patterns During Walking on a Slippery Walkway," *Journal of Neurophysiology*, vol. 103, no. 2, pp. 746–760, 2010.

[24] H. Audren, A. Kheddar, and P. Gergondet, "Stability polygons reshaping and morphing for smooth multi-contact transitions and force control of humanoid robots," in *IEEE-RAS International Conference on Humanoid Robots*, Cancun, Mexico, Nov. 2016, pp. 1037–1044.