Generation of Dynamic Multi-Contact Motions: 2D case studies

Sébastien Lengagne, Paul Mathieu, Abderrahmane Kheddar, Eiichi Yoshida

Generation of Dynamic Multi-Contact Motions: 2D case studies

Sébastien Lengagne, Paul Mathieu, Abderrahmane Kheddar and Eiichi Yoshida

***Abstract*— We present a multi-contact motion planning method that generates dynamic joint trajectories for multi-body robots that satisfy a set of continuous constraints. We highlight two variants when it comes to generate a single-contact or a multi-contact motion: the presence of the continuous equality geometrical constraints and of the contact forces. In this work, we compute the free-flyer trajectory and the contact forces from the joint trajectories provided by the optimization process. We assess our method on three dynamical multi-contact motions with 2D models. The comparison with intuitive adaptations of the single-contact motion planning methods shows the effectiveness of our method.**

***Index Terms*— Continuous constraint, multi-contact motion, dynamic motion, contact forces.**

INTRODUCTION

Humanoid robots transport themselves by sequentially alternating contacts with their surrounding environment. In our team, [1], [2] presented a contact-before-motion planning algorithm that generates a sequence of contact for a given motion problem. We successfully experimented it on a real HRP-2 robot [3], but the motion between two successive contacts and particularly the transitions were not fully dynamic.

This paper proposes a method to generate full dynamic motion between sequences of contacts, including dynamic transitions. It is based on our recent work [4] which applies semi-infinite optimization to dynamic motions with one contact such as a kicking motion. We extend it to generate smooth, dynamically stable and full-body multi-contact motions and transitions, from a given sequence of contact points as input.

In terms of computational speed, our method does certainly not compete with motion generation based on reduced models, e.g. the preview control used in [5], [6] where the trajectory of the COM is firstly generated. These methods deal with constraints (joint or torque limits, feasible inverse kinematics, equilibrium, etc.) *a posteriori*. For the time being, our priority is not closed-loop implementation of our trajectory generation algorithm. We are rather adopting a top-down approach: first, we aim at designing a method that solves the motion generation for the complete problem (whole-body) and considering all the constraints *a priori*. Thus, we consider a full-body model in order to take part of all the abilities of the robot and to produce any general feasible motion, that is not possible with reduce models since they fit only a small range of motions.

S. Lengagne, P. Mathieu, A. Kheddar and E. Yoshida are with CNRS-AIST Joint Robotics Laboratory (JRL), UMI3218/CRT, Tsukuba, Japan {sebastien.lengagne,e.yoshida}@aist.go.jp

A. Kheddar is also part of CNRS-UM2 LIRMM, Interactive Digital Human group, Montpellier, France kheddar@ieee.org

Nowadays, several methods emerge to generate full body single-contact motions [7], [8], [9]. But multi-contact motion was always considered as separate piecewise motions between fixed contacts that are connected afterwards. However, the generation of continuous full multi-contact motion is rather a complex problem, in one hand it is hybrid: alternating continuous and event phases (contact creations and contact breaks), on the other hand, during multi-contact motions, the closed chains affect the dynamical constraints: the internal contact forces constitute an infinite set of solutions. The motions must ensure all the limits of the robot (as for single-contact motions) and a set of kinematic and dynamical constraints inherent to multi-contact motions. In this paper, we get ride of the impact issue by considering perfectly inelastic contact (zero coefficient of restitution) and we ensure the joint position, velocity and acceleration continuity between two different contact phases.

We propose a method to solve whole-body multi-contact full motion under several constraints. We assess its effectiveness with three different 2D multi-contact scenarios.

I. PROBLEM STATEMENT

A. Optimal Control

We aim at computing the best whole-body joint trajectories that achieve a multi-contact dynamic motion for an entire sequence of successive contact stances (such as climbing or walking motion). Those joint trajectories $q(t)$ are refined by minimizing a cost function under a set of constraints:

$$\begin{array}{ll} \underset{q(t)}{\text{argmin}} & C(q(t)) \\ \forall i, \forall t \in [\Delta_i] & g_i(q(t)) \leq 0 \\ \forall j, \forall t \in [\Delta_j] & h_j(q(t)) = 0 \\ \forall t_k \in \{t_1, t_2, ...\} & z_k(q(t_k)) \leq 0 \end{array} \quad (1)$$

where:

- $q(t)$ is the joint trajectories vector (including $\dot{q}(t), \ddot{q}(t)$),
- g is the set of continuous inequality constraints, bounds on joint position, velocity, torques, contact forces...,
- h is the set of continuous equality constraints, (e.g. those translating geometric position of some robot's body),
- z is a set of discrete inequality constraints, to specify the position of a given robot's body at given time,
- C is the cost function, chosen by taking into account application contexts or intrinsic robotic performances,
- $[\Delta_i]$,$[\Delta_j]$ are time-intervals that define the validity duration of the continuous constraints.

g and h are continuous constraint sets, since they must be satisfied over time-intervals. Note that the single-contact

motion planning problem can be turned into the problem of Eq. (1) [4].

B. Semi-Infinite Programming

The motion planning problem (1) is an optimal control problem [10], [11] equivalent to an Infinite Programming (IP) one: it deals with finding a set of continuous functions that satisfy a set of continuous constraints (both can be seen as infinite sets of discrete values).

To make the problem tractable, most of the methods define a parameter set $\mathbf{P} \in \mathbb{R}^N$ used to compute the joint trajectories $q(t)$. The motion planning problem (1) turns into finding the best parameter set $\mathbf{P} \in \mathbb{R}^N$:

$$\begin{array}{ll} \underset{\mathbf{P}}{\text{argmin}} & C(\mathbf{P}) \\ \forall i, \forall t \in [\Delta_i] & g_i(\mathbf{P},t) \leq 0 \\ \forall j, \forall t \in [\Delta_j] & h_j(\mathbf{P},t) = 0 \\ \forall t_k \in \{t_1, t_2, ..., t_n\} & z_k(\mathbf{P},t_k) \leq 0 \end{array} \tag{2}$$

This problem is called Semi-Infinite Programming (SIP) [12] since it deals with a finite set of parameters $\mathbf{P} \in \mathbb{R}^N$, that must satisfy sets of continuous constraints (seen as an infinite set of discrete constraints).

C. Solving SIP: Time-discretization

Most of the time, SIP problems are solved by using a time-grid discretization of the constraints [12], [13], [14]. Thus the continuous constraints in Eq. (2) are replaced by:

$$\begin{array}{ll} \forall i, \forall t_k \in \mathbb{T}_i & g_i(\mathbf{P},t_k) \leq 0 \\ \forall j, \forall t_l \in \mathbb{T}_j & h_j(\mathbf{P},t_l) = 0 \end{array} \tag{3}$$

Where $\mathbb{T}_i$ and $\mathbb{T}_j$ are the time-grids (i.e. a set of discrete instant). As discussed in [15], it is not easy to set a time-grid discretization which guarantees in any circumstances that the constraints holds in-between a pair of sample time. Recently, we presented in [4], the time-interval discretization based on Taylor polynomial approximation of the constraints, that make possible to take into account:

$$\begin{array}{lll} \forall i, \forall \tau \in [\Delta_i] & \max\limits_{\tau} & g_i(\mathbf{P},\tau) \leq 0 \\ \forall j, \forall \tau \in [\Delta_j] & & h_j(\mathbf{P},\tau) = 0 \end{array} \tag{4}$$

The continuous functions are approximated by polynomial. By doing so, we can take into account all the constraints (discrete and continuous, inequalities and equalities) of a multi-contact motion planning problem. In this paper, we solve this problem thanks to IPOPT software [16] and consider an initial guess of $\mathbf{P}$ equal to zero (except the motion duration is initialized at one second) for all the optimization processes.

We point out the two main variants when it comes modeling the constraints of the multi-contact motion planning problem:

- *kinematics constraints*: how can we ensure that the set of contacts is sustained all along the motion?
- *dynamical constraints*: how can we compute the contact forces that ensure the balance without having the torques?

II. Kinematics Contact Constraints

A. Definition

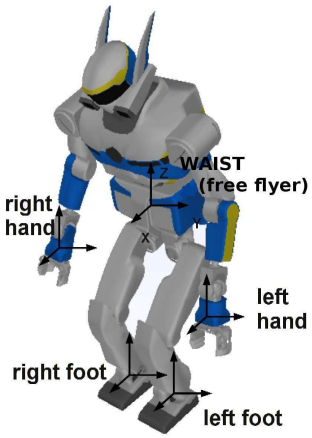

Fig. 1. Example of the contact frames for a Humanoid Robot. The robot can lean on its feet and on its hands.

To ensure the kinematics equality of one contact, we define two frames: one on the environment $\mathbf{X}_i^e$ and another one on the robot $\mathbf{X}_i^q$, see Fig. (1). To make a perfect rigid link, the optimization process tries to match the two frames:

$$\forall i, \forall t \in [\Delta t] \quad \mathbf{X}_i^q(t) = \mathbf{X}_i^e \tag{5}$$

We define one contact position $\mathbf{X}_i = [P_i, \theta_i]^T$ as a 6 dimensions vector in 3D which contains the position and the orientation of the contact frame i in the world frame.

Since the robot can move in its environment we specify its global position by tracking one of its bodies relative to the world coordinates: the free-flyer. We introduce the following notations:

- $\mathbf{X}_i^q$: the actual trajectory of the frame of the contact body i, which is computed thanks to the parameters for each step of the optimization process,
- $\mathbf{X}_i^e$: the expected position of the contact body i: frame on the environment,
- $\mathbf{X}^w$: the actual trajectory of the free-flyer,
- $\mathbf{X}_i^q$: the trajectory of the contact body i when $\mathbf{X}^w = 0$, this vector is only defined by the joint trajectories.

Each contact trajectory $\mathbf{X}_i^a$ is computed from the the free-flyer and joint angle trajectories:

$$\begin{bmatrix} P_i^a \\ \theta_i^a \end{bmatrix} = \begin{bmatrix} P^w \\ \theta^w \end{bmatrix} + \begin{bmatrix} A^w P_i^q \\ \theta_i^q \end{bmatrix} \tag{6}$$

Where A^w is the rotation matrix which depends on the orientation of the free-flyer θ^w.

B. Kinematics

When having a single-contact, we know the whole-body motion including the trajectory of the free-flyer (by selecting the contact body to be the reference free-flyer or by computing it from the joint trajectories and the contact position). For a motion with several contact phases, we do not assign

the free flyer reference body to one of the contacting bodies. We rather affix the free-flyer once for all and all along the motion. Yet, we need to compute its trajectory which must ensures that all the contact bodies are at the expected position all the time. The free-flyer trajectory can be computed by the optimization process or by directly from the joint trajectories.

C. With free-flyer parameterization

We can let the optimization solver find the trajectory of the free-flyer so that Eq. (5) is satisfied; this results in increasing the size of the optimization vector $\mathbf{P}$ with similar parameters to those used for joint trajectories. We apply this method without taking into account any cost function in the three scenarios described in the Annex I; we obtained the results presented in Table I. (We define 200 as the maximum number of iterations and use a $5-$order polynomial approximation).

TABLE I
CPU TIME WITH FREE-FLYER PARAMETERIZATION

motion	N^*	N_{eq}	N_{ctr}	*iter*	CPU time	status
1	108	0	246	11	1.7s	OK
		1	294	200	29.9s	MAX ITER
		2	342	44	6.9s	FAILED
		3	390	106	16.8s	FAILED
2	397	0	942	42	22.6s	OK
		1	1200	112	60.7s	FAILED
		2	1458	114	62.2s	FAILED
		3	1716	170	94.4s	FAILED
3	397	0	840	6	3.8s	OK
		1	996	11	6.5s	OK
		2	1152	18	9.8s	OK
		3	1308	18	10.0s	OK

(*) The parameter vector $\mathbf{P}$ contains the parameters to compute the joints and the free-flyer trajectories.

In Table I, N is the number of parameters, N_{ctr} the number of constraints and *iter* the number of iterations of the optimization process. We try to find a solution for several values N_{eq}: the order of the continuous equality constraint as defined in [4] (for a function $f(t) = \sum a_i t^i$, it ensures that $\forall i \in \{1, \cdots, N_{eq}\}\ a_i = 0$).

The walking motion (motion 3 in the Annex I), requires a simple trajectory of the free-flyer: the planning process can find a solution. For motions 1 and 2, a solution could not be found for all the values of N_{eq}. Those motions need a complex free-flyer trajectory that cannot be obtained by the parameterization we use (B-splines curves). Looking at Eq. (6), the contact position P_i is the result of the sum and multiplication between a parametrized polynomial function (P_w) and parameterized non polynomial functions (since A^w and P_i^q are obtained by addition of sines and cosines of parameterized polynomial functions). Implicitly, computing the free-flyer this way is equivalent to adding a set of continuous equality constraint that the solver is not able to handle correctly:

$$\forall t \in [\Delta t] \quad c_1 \sin\left(\sum_i b_i(t).P_i + c_2\right) = \sum_j b_j(t).P_j \qquad (7)$$

To sum-up, we cannot obtain good results from the optimization process for general cases, because parameterization of the free-flyer trajectory makes the problem over-constrained (cf. Eq. (7)). Therefore, we propose to compute the free-flyer trajectory from the joint trajectories.

D. Without free-flyer parameterization

Recall that for a N_c-contact motion, the joint and free-flyer trajectories writes:

$$\forall t \in [\Delta t],\ \forall i \in \{1, \ldots, N_c\} \quad \mathbf{X}_i^a(t) = \mathbf{X}_i^e \qquad (8)$$

Knowing the joint position and the expected contact positions $\mathbf{X}_i^e$, we have two cases:

- there is one solution for $\mathbf{X}^w$, thus we compute it.
- there is no solution $\mathbf{X}^w$ that satisfies the equality constraint. In this case, the optimization process will compute a new set $\mathbf{P}$ until a solution is found.

In both cases, we compute the free-flyer trajectory that minimizes the distance between the expected and the real contact position:

$$\underset{\mathbf{X}^w}{\operatorname{argmin}} \quad \sum_{i=1}^{N_c} \|\mathbf{X}_i^a - \mathbf{X}_i^e\|^2 \qquad (9)$$

The solution of the Eq. (9) satisfies:

$$\sum_{i=1}^{N_c} \left(\begin{bmatrix} P^w \\ \theta^w \end{bmatrix} + \begin{bmatrix} A^w P_i^q \\ \theta_i^q \end{bmatrix} - \begin{bmatrix} P_i^e \\ \theta_i^e \end{bmatrix} \right) = 0 \qquad (10)$$

We can find the free-flyer orientation:

$$\theta^w = \frac{1}{N_c} \sum_{i=1} \left(-\theta_i^q + \theta_i^e\right) \qquad (11)$$

and, thanks to the orientation of the waist: $A^w = f(\theta^w)$, we compute the joint position:

$$P^w = \frac{1}{N_c} \sum_i \left(-A^w P_i^q + P_i^e\right) \qquad (12)$$

We apply this method for several planning processes without any cost function for the three scenarios described in the Annex I; obtained results are presented in Table II. (We define 200 as the maximum number of iterations and use a $5-$order polynomial approximation)

There are fewer parameters, since we do not have parameters for the free-flyer trajectory. Note that all the motion planning processes converge to a solution, even for the motions 1 and 2 that could not be solved before.

Unfortunately, the computation time of one iteration is bigger in Table II relatively to that in Table I since the computation of the constraint requires to evaluate the trajectory of the free-flyer from the joint trajectories. Nevertheless, we get less iterations than with a free-flyer parameterization.

TABLE II

CPU TIME WITHOUT FREE-FLYER PARAMETERIZATION

motion	N^*	N_{eq}	N_{ctr}	*iter*	CPU time	status
1	82	0	246	3	1.6s	OK
		1	294	4	2.1s	OK
		2	342	4	2.0s	OK
		3	390	4	2.1s	OK
2	298	0	942	39	62.6s	OK
		1	1200	68	126.7s	OK
		2	1458	50	77.4s	OK
		3	1716	45	63.9s	OK
3	298	0	840	4	6.5s	OK
		1	996	7	11.0s	OK
		2	1152	6	9.0s	OK
		3	1308	6	9.8s	OK

(*) The parameter vector **P** contains only the parameters to compute the joints trajectories.

E. Conclusion

We run all the planning processes presented in Tables I and II using a time-grid discretization method (10 points per intervals). None of them converged. Consequently, even if for single-contact motion this way of discretization produces results [8], it appears that it does not scale robustly to deal with multi-contact motion.

The computation of the free-flyer trajectory from the joint trajectories is used in the remaining and to all our problems. It is a much better option to solve the multi-contact motion generation problem; indeed the parameterization of the free-flyer over-constrains the problem and very often do not converge.

III. DYNAMICAL CONTACT CONSTRAINTS

A. Definition of a contact

In order to maintain a contact i over a time interval $[\Delta t]$ the kinematics constraints must hold (cf. Eq. (5)), yet we must take into account dynamical effects such as balance and sliding which relates to posture and contact forces.

We consider linear contact forces applied on several points of the contact body. Eventually, we get N_p forces F_i:

$$\begin{bmatrix} \Gamma \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbf{M}_1(q) \\ \mathbf{M}_2(q) \end{bmatrix} \ddot{q} + \begin{bmatrix} H_1(q,\dot{q}) \\ H_2(q,\dot{q}) \end{bmatrix} + \sum_{i=1}^{N_p} \begin{bmatrix} \mathbf{J}_{1,i}^T(q) \\ \mathbf{J}_{2,i}^T(q) \end{bmatrix} F_i \tag{13}$$

Where:

- $q \in \mathbb{R}^{n+6}$ is a vector containing the n joint position (q_i) and the position and orientation of the free-flyer ($\mathbf{X}_w$)
- $\Gamma \in \mathbb{R}^n$ is the vector of size n of the torques,
- $\mathbf{M} \in \mathbb{R}^{(n+6)\times(n+6)}$ is the inertial matrix,
- $H \in \mathbb{R}^{n+6}$ is the vector due to gravity, centrifugal and Coriolis effects,
- $\mathbf{J}_{1,i} \in \mathbb{R}^{n\times 3}$: the first part of the Jacobian matrix is the expression of the mapping of the contact forces i in the joint torque space,
- $\mathbf{J}_{2,i} \in \mathbb{R}^{6\times 3}$: the second part of the Jacobian matrix is the expression of the effects of the contact forces i on the forces applied to the free-flyer,
- $F_i \in \mathbb{R}^3 = [F_{i,x}, F_{i,y}, F_{i,z}]^T$ is the vector of the linear contact forces.

To ensure that the contact is sustained at a given position, we avoid any sliding, rolling and taking off. Thus, we haven to ensure positive contact forces that remains within the friction cone. This constraint can be formulated as:

$$\forall i, \forall t \in [\Delta t] \quad \begin{cases} F_i^n(t) > 0 \\ |F_i^t(t)|^2 \leq \mu_i^2 F_i^n(t)^2 \end{cases} \tag{14}$$

where F_i^n and F_i^t are the normal and tangential components of the contact force F_i.

B. Balance

The balance of humanoid robots is usually described by the Zero Moment Point [17]. However, this method cannot be used for non-coplanar contacts. Hirukawa et al. [18] or Bretl [19] proposed a more general approach that, in a way or the other, relate the contact forces to their friction cone (Eq. (14)).

For multi-contact motions, we characterize the balance by searching the contact forces that ensure the inequality equation (14) and counterpart the dynamics effects:

$$D_2(q) + \sum_{i=1}^{N_p} \mathbf{J}_{2,i}^T(q) F_i = 0 \tag{15}$$

We simplify the notation of the Eq. (13) by bringing M and H into the single vector $D = [D_1, D_2]^T$ with the joint torques D_1 and the force applied on the free-flyer D_2 due to the free dynamic.

To have safe multi-contact motion, the contact forces must ensure the inequality constraints (14) and the dynamical equality constraint of Eq. (15) over the whole motion duration. Regarding from the optimization process we can set the contact forces as:

- *an input*: the optimization process computes the contact forces from the set of parameters **P** and has to ensure the continuous equality constraints Eq.(15),
- *an output*: the contact forces are a result of the joint trajectories.

In both cases, the constraints of Eq.(13) are checked by the optimization process.

C. Contact Forces as parameter

Here, the contact forces are set as variables of the optimization process. For each interval, we compute each component of the contact forces thanks to a 4^{th}-order B-splines parameterization.

$$\forall i, \circ \in \{x,y,z\} \quad F_{i,\circ} = \sum_{j=1}^{4} f_j(t) p_{i,\circ,j}^f \tag{16}$$

Where $f_j(t)$ is the set of basis functions and $p_{i,\circ,j}^f$ are optimization parameters. Moreover, we implement the continuous equality constraint of Eq. (15) of the balance. We try to obtain joint trajectories for motion 1 (the simplest one). We run the optimization process with $n = 338$ parameters and $N_{ctr} = 496$ constraints that cannot converge within 1000

iterations. As previously for the free-flyer trajectory, the parameterization of the contact forces make the problem over-constrained. Therefore, we propose a method to compute them from the joint trajectories.

IV. Computation of the contact forces

A. Problem

We need to compute the contact forces, from the joint trajectories, that fit with this equation:

$$D_2(q) + \mathbf{J}_2^T(q) F_c = 0 \tag{17}$$

Since the Matrix $\mathbf{J}_2 = [\mathbf{J}_{2,1}, \mathbf{J}_{2,2}, \cdots, \mathbf{J}_{2,N_p}]$ is not necessarily square, we can use the pseudo inverse : $(\mathbf{J}_2^T)^+ = (\mathbf{J}_2 \mathbf{J}_2^T)^{-1} \mathbf{J}_2$ to find $F_c = [F_1, F_2, ..., F_{N_p}]^T = (\mathbf{J}_2^T)^{-1} D_2$.

By definition, the pseudo inverse minimizes the Euclidean norm of the solution. However, we are not interested into minimizing the Euclidean norm and we propose to find the contact forces as close as possible to the normal direction of the contact, in order to increase the chances to fit within the friction cones Eq. (13).

B. Optimization problem formulation

We present a formal expression of the contact forces, that counterpart the free dynamics effects (cf. Eq. (17)) and are as close as possible to the normal direction. Doing so, we enhance the friction effect and unilateral constraints that are checked by the global optimization solver:

$$\min \frac{1}{2} \sum_{i=1}^{N_p} \beta_i (\alpha_i F_i^{t\,2} + F_i^{n2}) \tag{18}$$

$$\sum_{i=1}^{N_p} \left(\begin{bmatrix} \hat{P}_i A_i \\ A_i \end{bmatrix}^T [F_i] \right) + [D_2] = 0 \tag{19}$$

We decompose the Jacobian matrix $\mathbf{J}_{2,i}^T$, with $\hat{P}_i$ the screw operator of the contact position, $[D_2] = [M_x, M_y, M_z, F_x, F_y, F_z]^T$ the effort due to the free dynamics. β_i is a coefficient to equilibrate (or not) the repartition of the forces and α_i is used to weight the tangential regarding to the normal forces. If we set $\forall i \ \beta_i = \alpha_i = 1$, we end with the normal pseudo-inverse problem. To get forces as close as possible to the normal direction of the contact, we choose to set $\forall i \ \alpha_i = 10$.

C. Analytical solution

To solve this problem, we write the Lagrange equation:

$$L = \sum_{i=1}^{N_p} \frac{\beta_i \alpha_i}{2} F_i^{t\,2} + \sum_i \frac{\beta_i}{2} F_i^{n2} + \left(\sum_{i=1}^{N_p} \begin{bmatrix} \hat{P}_i A_i \\ A_i \end{bmatrix}^T [F_i] + [D_2] \right) [\lambda] \tag{20}$$

Where $[\lambda] \in \mathbb{R}^6$ is the vector of the Lagrange multipliers. The solution of the Lagrange equation with respects to the condition of optimality:

$$\frac{\partial L}{\partial F_i} = 0 \ , \ \frac{\partial L}{\partial \lambda} = 0 \tag{21}$$

From the derivative $\frac{\partial L}{\partial F_i}$ we have:

$$F_i = -\gamma_i \begin{bmatrix} \hat{P}_i A_i & A_i \end{bmatrix} [\lambda] \tag{22}$$

With γ_i a three component diagonal matrix for which $\gamma_{i,\circ} = \frac{1}{\beta_i \alpha_i}$ if $F_\circ$ is one of the tangential component and $\gamma_{i,\circ} = \frac{1}{\beta_i}$ if $F_\circ$ is the normal component of the contact force. We replace the contact forces in the set of the equality constraints:

$$\sum_{i=1}^{N_p} \left(\begin{bmatrix} \hat{P}_i A_i \\ A_i \end{bmatrix}^T \gamma_i \begin{bmatrix} \hat{P}_i A_i & A_i \end{bmatrix} \right) [\lambda] = [D_2] \tag{23}$$

This equation can be turned into:

$$\Omega [\lambda] = [D_2] \tag{24}$$

We invert the (6×6) matrix Ω, using the Gauss-Jordan Algorithm, to find the value of the Lagrange multipliers:

$$[\lambda] = \Omega^{-1} [D_2] \tag{25}$$

We replace the Lagrange multipliers $[\lambda]$ in Eq. (22) to get the value of the contact forces and compute the torques, knowing the contact forces, thanks to Eq. (13).

D. Computational Results

We evaluate three different ways to set the coefficients β_i:

1) all the contact forces of all the bodies have the same weight, $\forall i \ \beta_i = 1$ over the whole motion duration,
2) β_i is the same for all the contact forces acting on the same body. The optimization solver tries to find a value of β_i for each contact body over each contact phase.
3) The optimization solver tries to find β_i for each contact forces over each contact phase.

In fact, the choice of the case 1, 2 or 3 impacts on the size of the feasible set of the contact forces and hence on the size of the feasible motion set. In case 1, we set all the values of β_i, thus it is not possible to modify the contact forces without changing the joint trajectories. In case 2 and 3, we are able to produce different (internal) contact forces from identical joint trajectories (by changing the value of β_i). With this method we do not make any assumption about the shape of the contact forces and find an analytical expression of them.

TABLE III
CPU time with the dynamical constraints

motion	case	N^*	N_{ctr}	*iter*	CPU time	status
1	1	82	406	40	54.2s	OK
	2	84	406	32	48.5s	OK
	3	89	406	45	68.7s	OK
2	1	298	1802	1000	5618.8s	MAX ITER
	2	312	1802	177	1055.8s	OK
	3	335	1802	82	533.3s	OK
3	1	298	1360	196	902.4s	OK
	2	303	1360	87	440.6s	OK
	3	317	1360	31	165.6s	OK

(*) The parameter vector **P** contains the parameters to compute the joints trajectories, and the values of β_i for cases 2 and 3..

Table III shows the results of the computation of the three motions with each of the three cases to specify β_i, with an $8-$order polynomial approximation of the continuous constraints and 1000 as the maximal number of iterations.

All the cases produce feasible motions, except for the motion 2; the case 1 does not lead to feasible motion. This was predictable since case1's motion cannot be realized without having internal forces.

The number of iteration relies on the choice of the case. For complex motions (2 and 3), case 3 is more efficient since the optimization converges within fewer iterations. Finally, we show that we can deal efficiently with the contact forces constraint during a multi-contact motion planning process.

V. Full motion planning results

Algorithm (1) summarizes the computation process of all the variables needed for multi-contact motion planning. First, we compute the trajectory of the free-flyer from the joint trajectories. Then, we compute the free dynamics effects D_2 to compute the contact forces that must counterpart it. Finally, the contact forces are used to get the evaluation of the torques.

Algorithm 1 Modeling of a multi-contact motion

- **Require:** $\mathbf{P}, \mathbf{X}_i^e$
- Joint trajectory: $\mathbf{P} \rightarrow q(t)$
- Kinematics computation
  - compute $\mathbf{X}_i^q = f(q(t))$
  - compute the trajectory of the waist $\mathbf{X}^w$
  - compute the trajectory of the contact body $\mathbf{X}_i^a$
- Dynamical computation
  - compute the free dynamics effects$[D_2]$
  - compute the contact forces F_i
  - compute the Torques $[\Gamma]$

We present the result of several full motion planning processes in Table IV, taking into account:

- the cost function as the sum of the square torques,
- a free motion duration for each support phase,
- bounds on joint values, velocities and on the contact forces (friction, unilateralism of the contact and balance),
- bounds on the joint torques.

We do not run motion 2 with the case 1, since in Section IV-D we show it is not feasible without internal forces.

TABLE IV
CPU time for full motion planning

motion	case	*iter*	CPU time	criteria	status
1	1	84	218.4s	67.6	OK
	2	245	709.2s	53.4	OK
	3	422	1247.0s	49.9	OK
2	2	1572	19609.1s	736.5	OK
	3	878	11427.2s	527.3	OK
3	1	238	2299.8s	188.7	OK
	2	352	3546.8s	171.9	OK
	3	494	5194.6s	154.1	OK

Table IV shows that all the processes converge and produce an optimal motion. We pin out that the way to compute the contact forces impacts the objective function. In fact, the case 3 produces better results than case 2 which was better than case 1, since case 3 has the biggest feasible set of contact forces.

VI. Conclusion and discussion

In this paper, we presented efficient methods to plan whole-body multi-contact acyclic motion while dealing with continuous constraints. We consider a full-body model to take part of all the abilities of the robot in order to generate any possible motion. We found that it is better to compute the free-flyer trajectory from the joint ones to better satisfy the kinematics constraints. We also presented how to compute the contact forces from the free dynamics effects using a few optimization parameters (β_i). We ran several optimization processes on 2D cases to highlight that our method produces efficient multi-contact motions.

We are also able to generate motion with internal forces providing that during real experiments, we will be able to control the torques and hence the expected contact forces.

Now, we are working to extend this work to 3D-multi-contact motion generation, to integrate the self-collision avoidance and to perform experiments on a humanoid robot: HRP-2. We will also investigate the multi-contact impact motion, for which we need to remove the joint velocity continuity for an impact modeling.

Appendix I
2D validation

A. Decomposition of a motion

We applied our method to 2-D robots toy scenarios (it was important to be able to track every step reliably). We decompose a motion into several phases: one phase describes one contact stance and contains several time intervals (we choose 4 intervals per phase except for the first and the last phases, for which we consider only 3 intervals) and ensure the joint position, velocity and acceleration continuity, since we consider perfectly inelastic contact not to deal with impact forces (i.e. model discontinuity). On each contact phase, joint trajectories are computed from a set parameters and phase duration.

B. Benchmark motions

We defined three different motions, for a climbing robot for a walking robot and consider a friction parameter $\mu = 2.0$:

- **Motion 1** (Fig. 2): the climbing robot swings from a two-contact posture to another two-contact postures through a one-contact posture (as a cart wheel motion)
- **Motion 2** (Fig. 3): the climbing robot leans on a wall to climb on a slope,
- **Motion 3** (Fig. 4): the walking robot performs four steps of 0.3 meter on a flat ground,

Motion 1 is composed of 3 phases (10 intervals), motions 2 and 3 are composed of 9 phases (34 intervals).

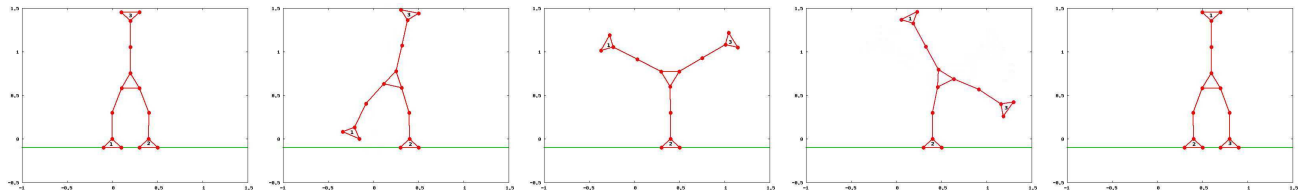

Fig. 2. Cart wheel motion: first motion used to evaluate our method.

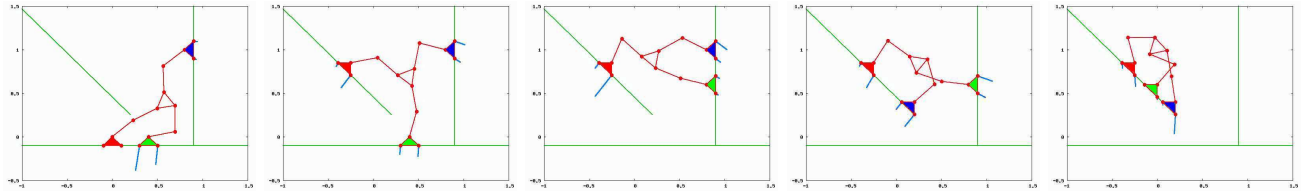

Fig. 3. Climbing motion: second motion used to evaluate our method.

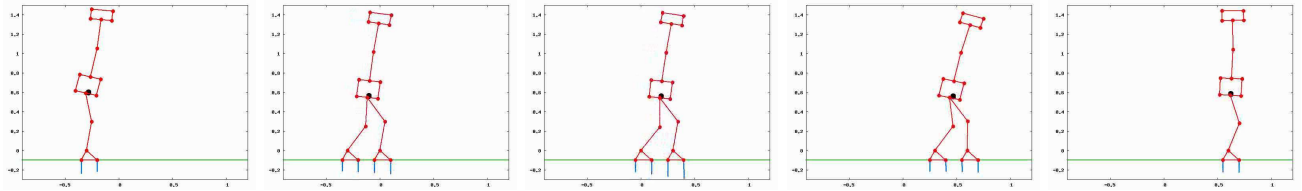

Fig. 4. Walking motion: third motion used to evaluate our method.

ACKNOWLEDGMENT

This research is partially supported by Japan Society for the Promotion of Science (JSPS) Grant-in-Aid for JSPS Fellows (P09809) and for Scientific Research (B), 22300071, 2010.

REFERENCES

[1] A. Escande and A. Kheddar, "Contact planning for acyclic motion with tasks constraints," in *IEEE/RSJ Int. Conf. on Intelligent RObots and Systems (IROS 2009)*, Oct. 11-15 2009.

[2] K. Bouyarmane, A. Escande, F. Lamiraux, and A. Kheddar, "Potential field guide for humanoid multicontacts acyclic motion planning," in *IEEE Int. Conf. on Robotics and Automation*, may 2009.

[3] A. Escande, A. Kheddar, S. Miossec, and S. Garsault, "Planning support contact-points for acyclic motions and experiments on hrp-2," in *ISER*, 2008, pp. 293–302.

[4] S. Lengagne, P. Mathieu, A. Kheddar, and E. Yoshida, "Generation of dynamic motions under continuous constraints: Efficient computation using b-splines and taylor polynomials," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010.

[5] 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, september 2003, pp. 1620 – 1626.

[6] P.-B. Wieber, "Trajectory Free Linear Model Predictive Control for Stable Walking in the Presence of Strong Perturbations," in *IEEE-RAS International Conference on Humanoid Robots*, Genova Italie, 2006.

[7] S.-H. Lee, J. Kim, F. Park, M. Kim, and J. E. Bobrow, "Newton-type algorithms for dynamics-based robot movement optimization," in *IEEE Transactions on robotics*, vol. 21, 2005, pp. 657– 667.

[8] S. Miossec, K. Yokoi, and A. Kheddar, "Development of a software for motion optimization of robots - application to the kick motion of the hrp-2 robot," in *IEEE Int. Conf. on Robotics and Biomimetics*, 2006, pp. 299–304.

[9] S. Lengagne, N. Ramdani, and P. Fraisse, "Safe motion planning and fast re-planning for humanoid robots. (submitted to ieee trans. on robotics)," 2010.

[10] O. von Stryk and R. Bulirsch, "Direct and indirect methods for trajectory optimization," *Ann. Oper. Res.*, vol. 37, no. 1-4, pp. 357–373, 1992.

[11] J. Denk and G. Schmidt, "Synthesis of a Walking Primitive Database for a Humanoid Robot using Optimal Control Techniques," in *IEEE-RAS Int. Conf. on Humanoid Robots*, Nov. 2001, pp. 319–326.

[12] R. Reemtsen and J.-J. Rückmann, *Nonconvex optimization optimization and its applications : Semi-infinite Programming*, R. Reemtsen and J.-J. Rückmann, Eds. Kluwer Academic Publishers, 1998.

[13] O. von Stryk, "Numerical solution of optimal control problems by direct collocation," 1993.

[14] R. Hettich and K. O. Kortanek, "Semi-infinite programming: theory, methods, and applications," *SIAM Rev.*, vol. 35, no. 3, pp. 380–429.

[15] S. Lengagne, N. Ramdani, and P. Fraisse, "Planning and fast re-planning of safe motions for humanoid robots : Application to a kicking motion," in *IEEE/RSJ Int. Conf. on Int. Rob. and Syst.*, 2009, pp. 441– 446.

[16] A. Wächter and L. T. Biegler, "On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, pp. 22–57, 2006.

[17] M. Vukobratović, "On the stability of anthropomorphic systems," *Mathematical Biosciences*, vol. 15, pp. 1–37, 1972.

[18] H. Hirukawa, S. Hattori, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, and M. Morisawa, "A universal stability criterion of the foot contact of legged robots - adios zmp," in *IEEE Int. Conf. on Robotics and Automation (ICRA).*, may 2006, pp. 1976– 1983.

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