Integration of Non-Inclusive Contacts in Posture Generation

Stanislas Brossette, Adrien Escande, Joris Vaillant, François Keith, Thomas Moulard, Abderrahmane Kheddar

***Abstract*— In this paper we propose a simple way to formulate geometric contact formation to have an arbitrary intersection shape in a robotic (humanoid) posture generation problem. The contact shape is the outcome of our posture generator that is formulated as a non-linear optimization programming to fulfill a large variety of robot intrinsic limitations (e.g. joint and torque limits) and tasks (e.g. desired contact). Starting by defining convex areas of contact on the robot's body and the environment, that we call contact patches, we can generate contacts with arbitrary intersection of a pair of any of these predefined patches. Our geometric contact modeling writes very simply as additional constraints and variables added to the optimization problem, translating the search for an ellipse inscribed in the intersection of the pair of patches we want in contact. The result of our posture generator is then a configuration where contact patches are not necessarily included in one another. This allows our posture generator to propose contacts of different shapes with a non-predefined number of contact points (used later to compute reaction/contact forces). We illustrate the efficiency of our method in multi-contact posture generation with the HRP-2 and ATLAS humanoid robots with results that can not be generated automatically by existing methods.**

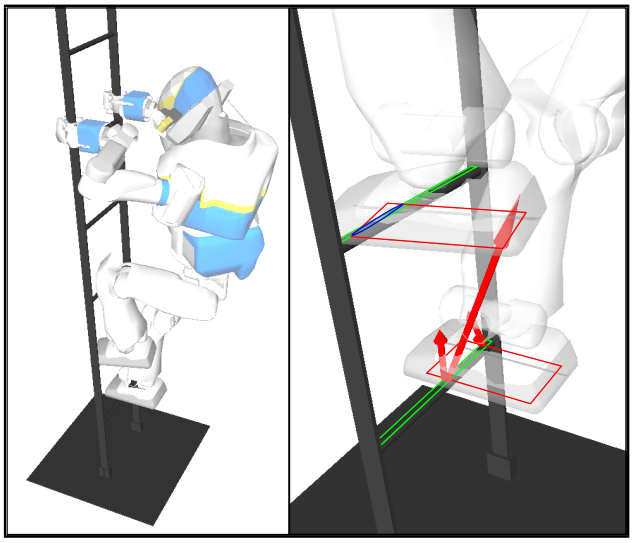

Fig. 1: Using non-inclusive contacts for ladder climbing (green/red: contact polygons; blue: contact ellipse; red arrows: contact forces resultants)

I. Introduction

Generating viable robotic postures is a common problem encountered in sampling-based planning techniques and simulation of virtual characters. Generating desired initial, intermediary or finale posture configurations requires defining static task goals (e.g. reach a target point in 6D) to be done under intrinsic constraints such as joint limits, torque limits, avoiding non-desired self-collisions... and perceptual or extrinsic ones such as keeping an object in the embedded camera field-of-view, avoiding non-desired collisions with surrounding objects, etc. A common task objective assigned to virtual characters, avatars, or humanoid robots is to contact one or more of its links with the environment (e.g. feet touching the ground). Since our main applications target humanoid robots, we consider here posture generation problems inherent to these robots. However, the formalism applies to any kind of robots achieving contacts with its surroundings or itself.

Our problem is to generate multi-contact viable postures. As far as humanoids are concerned, we may add to the previously cited constraints, equilibrium and non-sliding. This paper is dedicated to the focused issue of how contact constraints can be written geometrically. Generating postures often uses state-of-the-art enhanced inverse kinematics or general-purpose non-linear optimization programming (where inverse kinematics can be seen as a particular solver case).

In general, desired contacts write as hard constraints to fulfill in an optimization problem. Yet, we need to write the maths for, say, put the gripper on the wall and the left foot on the ground. In general, the maths of a gripper is a complex geometric description, so is often that of the environment. A contact is generally defined by a pair of points (one on each object in contact) and a pair of normal vectors. A hard contact constraint boils down to finding a posture in which the predefined authorized contact points and normal of each body match [1][2]. Likewise, in [3], the position of the feet of the NAO robot is manually tuned in order to obtain statically stable position during the climbing of a spiral staircase. In [4], the surface in contact is chosen according to two criteria: the position of the force sensors of the feet, and the type of contact desired. In [5], the problem of contact discovery in not considered. In [6], two surfaces are considered in contact as soon as the center point of one of them touches the other one and their normals match. Although used in many papers, it is not difficult to see that this definition of contact excludes a series of possibilities that would have been obtained if the predefined points were placed in different configurations within their respective patches. Once the contact is established, one determines the intersection of contacting surfaces in order to find points on which reaction forces are to be computed.

CNRS-UM2 LIRMM Interactive Digital Humans, UMR0056, Montpellier, France

CNRS-AIST Joint Robotics Laboratory (JRL), UMI3218/CRT, Tsukuba, Japan

This work was partially supported by the FP7 KoroiBot and the FP7 RoboHow.Cog EU projects

Several approaches require fixing the number of contact points or to have inclusive contact (i.e. one patch is fully included in the other) [7].

We provide a simple solution that relaxes hard contact constraints and gets rid of predefining the contact points by allowing them to travel within the patches. We consider that a contact is valid if the intersection between two distinct patches has an area greater than a given threshold. To enforce this, we require this intersection to contain an ellipse whose surface can possibly be maximized. Convex patches allow writing the inscription constraints easily by means of half-spaces. We finally implement our solution and present some examples for complex multi-contact posture generations with the HRP-2 and ATLAS robots, as shown in Fig. 1.

II. Contact geometry formulation

For our formulation, we consider a situation where a set of contacts between the robot and its environment are already made, and therefore fixed, and we want to add a new contact to this set. This doesn't induce any loss of generality since it just comes down to adding the contacts one by one. To ensure that the new contact can be reached in a quasi-static way, we look for a configuration where the new contact is "barely" made: the position of contact is reached, but that contact does not support any contact forces (this is a necessary step to generate a sequence of quasi-static transitions). We call it a geometric contact. Let us consider that the contact to add is defined by two flat surfaces S_1 and S_2 which are respectively delimited by two convex polygons P_1 and P_2. For this contact to be valid, it is obviously necessary that the intersection $P_1 \cap P_2$ is not empty. We propose a method in which the size of the contact area is approximated by the size of an ellipse that is inscribed in it. If such an ellipse is found and is of a sufficient size, then the contact is valid. This allows to consider contacts between surfaces that do not necessarily include each other.

An important remark is that the number of sides of the intersection polygon is not known a priori and, as shown on Fig. 2, this number can change depending on the configuration. Each time this number changes, the gradient of the area of the intersection is discontinuous. This is an issue for integrating any constraint or objective based on the area because we use a solver for smooth optimization problems. This issue could be dealt with by using non-smooth optimization routines, but such algorithms are slower and less available, and our posture generator is not designed to use them. Moreover, supposing that we want to write constraints

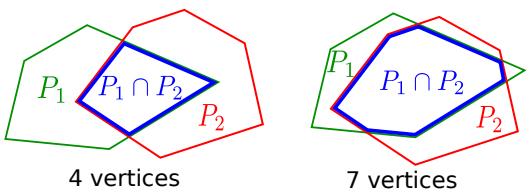

Fig. 2: Topological instability of $P_1 \cap P_2$

based on the sides of the contact area, then, the number of constraints would change with the number of vertex of $P_1 \cap P_2$. The large majority of the optimization softwares cannot deal with a non-constant number of constraints. The solution proposed in section IV overcomes these issues by defining a set of constraints that is independent from the topology of the intersection area.

III. Posture generation

The posture generation process aims at finding a posture that satisfies a set of tasks $\{\mathcal{T}_i\}$ and that minimizes a cost function *Cost* by solving the following problem:

$$\min_{\mathbf{q},\mathbf{f},\tau} \quad \mathit{Cost}(\mathbf{q},\mathbf{f},\tau)$$
$$\text{s.t.} \begin{cases} q_i^- \leq q_i \leq q_i^+ \quad \forall i = 1,...,n \\ \tau_i^- \leq \tau_i \leq \tau_i^+ \quad \forall i = 1,...,n \\ \epsilon_{ij} \leq d(r_i(\mathbf{q}), r_j(\mathbf{q})), \quad \forall (i,j) \in \mathcal{I}_{auto} \\ \epsilon_{ik} \leq d(r_i(\mathbf{q}), O_k)), \quad \forall (i,k) \in \mathcal{I}_{coll} \\ \tau + J(\mathbf{q})^T \mathbf{f} = \mathbf{g}(\mathbf{q}) \\ s(\mathbf{q},\mathbf{f}) \leq 0, \\ g_i(\mathbf{q},\mathbf{f},\tau) = 0 \quad \forall \mathcal{T}_i, \\ h_i(\mathbf{q},\mathbf{f},\tau) \leq 0 \quad \forall \mathcal{T}_i. \end{cases} \tag{1}$$

where the optimization variables $\mathbf{q}$, $\mathbf{f}$ and τ stand for the configuration, contact forces and joint torques of the robot. Those constraints are illustrated in Fig. 3 and are, in order of appearance:

- Joint limits
- Torque limits
- Auto-collisions, with $d(X,Y)$ the signed distance between objects X and Y and $r_i(\mathbf{q})$ is the i-th body of the robot at configuration $\mathbf{q}$. $\mathcal{I}_{auto}$ is the set of pairs of bodies to monitor.
- Collisions with the environment, O_k being the k-th object in the environment and $\mathcal{I}_{coll}$ the set of pairs to monitor
- Equation of static stability, with J the Jacobian matrix of all points where the contact forces are applied, and $\mathbf{g}$ the gravity term.
- Stability constraints describing the friction cones
- Equality constraints describing the task $\mathcal{T}_i$
- Inequality constraints describing the task $\mathcal{T}_i$

A usual task $\mathcal{T}_i$ in the posture generation is to ensure that two surfaces are in contact. We consider 2 polygons P_1 and P_2 described in 2 frames $\mathcal{F}_1$ and $\mathcal{F}_2$, respectively. Each frame is defined by an origin point $\mathbf{O_i} = (o_i^x, o_i^y, o_i^z)$ and 3 orthogonal vectors $[\mathbf{T_i}, \mathbf{B_i}, \mathbf{N_i}]$. In those frames, the polygons are described as a set of n_i bi-dimensional points $[p_i^0, p_i^1, \ldots, p_i^{n_i-1}]$ lying in the $[\mathbf{O_i}, \mathbf{T_i}, \mathbf{B_i}]$ plane. To ensure the contact between those two surfaces, it is necessary that the planes defined by $[\mathbf{O_1}, \mathbf{T_1}, \mathbf{B_1}]$ and $[\mathbf{O_2}, \mathbf{T_2}, \mathbf{B_2}]$ are coplanar. This is expressed by the set of equations (2). Those equations define a floating contact, where the co-planarity is ensured and the surfaces can translate along $\mathbf{T_1}$ and $\mathbf{B_1}$ and rotate around $\mathbf{N_1}$.

$$\begin{cases} (\mathbf{O_2} - \mathbf{O_1}).\mathbf{N_1} = 0 \\ \mathbf{T_2}.\mathbf{N_1} = 0 \\ \mathbf{B_2}.\mathbf{N_1} = 0 \\ -\mathbf{N_2}.\mathbf{N_1} \leq 0 \end{cases} \tag{2}$$

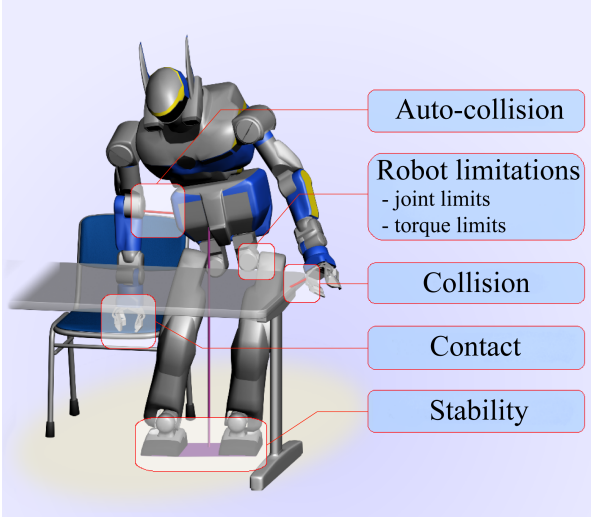

Fig. 3: Posture Generation's usual constraints

However, this is not sufficient since nothing ensures that the intersection of P_1 and P_2 is not empty. Up to now, to avoid the problems described in section II, we were requiring that one of the two polygons was completely included into the other (which restricted the contact configuration possibilities) or by defining a fixed contact position by hand, in which case the non-emptiness is ensured by the user. The next section presents a more general formulation

IV. Non inclusive contact constraints

A. Main Idea

We present our main contribution: a smooth formulation of the non empty intersection between two contact surfaces. We assume that co-planarity of S_1 and S_2 is obtained by using the constraints presented in the previous section. Here we focus on the intersection of the two polygons P_1 and P_2, respectively describing the contours of S_1 and S_2. As we pointed out earlier, a problem with computing the area of intersection of two polygons comes from the fact that depending on their positions in space, the number of edges of their intersection can change (cf. Fig. 2), which induces discontinuity of the gradient of the area and change of the number of constraints associated with this contact.

To avoid dealing with these changes of topology, we consider using an ellipse $\mathcal{E}$ included in $P_1 \cap P_2$ to estimate the area of the intersection. Since P_1 and P_2 are convex polygons, then $P_1 \cap P_2$ is also a convex polygon. A convex polygon can be seen as an intersection of half-planes based on the lines supporting its edges. Thus, an ellipse is inside a convex polygon if it lies entirely in the corresponding half-planes. Having the ellipse be included in the intersection of two polygons is equivalent to having it included in both polygons:

$$\mathcal{E} \subset P_1 \cap P_2 \iff \mathcal{E} \subset P_1 \quad \wedge \quad \mathcal{E} \subset P_2 \tag{3}$$

Even if the number of edges of $P_1 \cap P_2$ can change, the numbers of edges of P_1 and P_2 respectively are fixed. To assert that an ellipse lies in a half-plane, we need a function that is positive when the ellipse is in it (with zero value when the ellipse is on the edge) and negative if not. The signed distance to the line defining the half-space is a good candidate (distance ellipse-line if the ellipse is in the half-plane, opposite of the penetration distance if not), but actually, any pseudo-distance does the job. And a sufficient condition for the ellipse to be inside the polygons intersection is that the pseudo-distance between the ellipse and each edge of both polygons is positive (see Fig. 4). By considering each edge separately as opposed to the (pseudo-)distance of the ellipse to a whole polygon, we can write smooth constraints with a simple pseudo-distance function. We develop such a pseudo-distance in the next subsection.

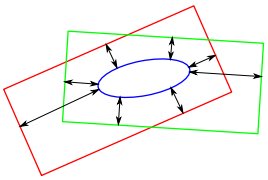

Fig. 4: Distance between $\mathcal{E}$ and $P_1 \cap P_2$

B. Pseudo-distance

To estimate the constraint of inclusion of the ellipse $\mathcal{E}$ in both polygons P_1 and P_2, we need to compute the signed distance between $\mathcal{E}$ and each segment of the polygons. Computing the distance between an ellipse and a line is not straightforward, whereas the distance between a line and a circle is very easy to compute. Also, we note that in the frame $F_\mathcal{E}$ defined by the ellipse's axes and pseudo-radius, the ellipse is a circle of radius $r_\mathcal{E} = 1$ (The x-unit along the first axis of the ellipse is r_x, the first radius of the ellipse, the y-unit along the second axis of the ellipse is r_y, the second radius). The transformation from the original frame F_0 in which the ellipse and the polygons are described to the ellipse's frame $F_\mathcal{E}$ is just the composition of a rotation and a scaling of the space along the axes of the ellipse with a scaling vector $[\frac{1}{r_x}, \frac{1}{r_y}]$. The effect of such a transformation applied to an ellipse and two polygons is shown in Fig. 5. We thus defined the following pseudo-distance from an ellipse to a half-plane as the signed Euclidean distance from the corresponding unit circle to the transformed half-plane in the frame $F_\mathcal{E}$.

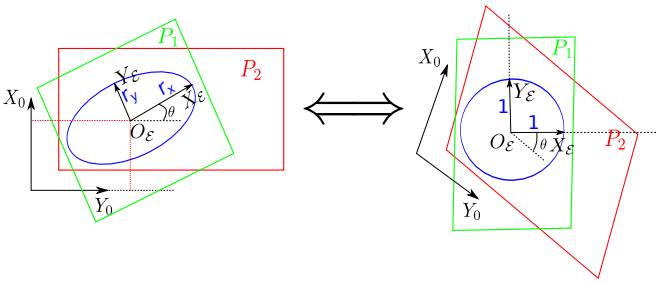

Fig. 5: Transformation from $F_0(O_0, X_0, Y_0)$ to $F_\mathcal{E}(O_\mathcal{E}, X_\mathcal{E}, Y_\mathcal{E})$

Now let us consider a single segment $p_i p_j$ and an ellipse $\mathcal{E}$ defined in F_0. The expression of a vector $\mathbf{v}_{F_0}^T = [v_x, v_y]_{F_0}$ in $F_\mathcal{E}$ is obtained by applying the formula (4)

$$\mathbf{v}_{F_\mathcal{E}} = \begin{pmatrix} \frac{1}{r_x} & 0 \\ 0 & \frac{1}{r_y} \end{pmatrix} \cdot \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} . \mathbf{v}_{F_0} \tag{4}$$

In $F_{\mathcal{E}}$, the distance between the circumference of $\mathcal{E}$ and the segment p_ip_j is:

$$d_s(\mathcal{E}, p_ip_j) = \frac{(\overrightarrow{p_ip_j})_{F_{\mathcal{E}}} \times \left(\overrightarrow{p_iO_{\mathcal{E}}}\right)_{F_{\mathcal{E}}}}{\|\overrightarrow{p_ip_j}\|_{F_{\mathcal{E}}}} - r_{\mathcal{E}} \tag{5}$$

where $\times$ denotes the cross product and $O_{\mathcal{E}}$ is the center of the ellipse.

To avoid numerical problems when the segment p_ip_j is small, (see sec. IV-F), it is preferable to multiply this distance by $\|\overrightarrow{p_ip_j}\|_{F_{\mathcal{E}}}$ before using it as a constraint. Then we get the following constraint:

$$-(\overrightarrow{p_ip_j})_{F_{\mathcal{E}}} \times \left(\overrightarrow{p_iO_{\mathcal{E}}}\right)_{F_{\mathcal{E}}} + \|\overrightarrow{p_ip_j}\|_{F_{\mathcal{E}}} r_{\mathcal{E}} \leq 0 \tag{6}$$

The combination of these equations (4) and (6) applied for each edge of the polygons gives us all the necessary tools to develop a set of constraints that ensures that an ellipse is in the intersection of two polygons.

C. Modification of the optimization problem

To include the above idea in our posture generation, we need to modify the optimization problem (1) as follows. Each non inclusive geometrical contact adds five variables to the optimization vector, corresponding to the position, orientation and radiuses of the ellipse (x, y, θ, r_x and r_y). One constraint of ellipse inclusion (as described above) is added to the problem for each edge of the polygons. The parameters r_x and r_y are given lower positive bounds to ensure that the ellipse is not empty. The existence of a contact between S_1 and S_2 is thus transformed into the existence of r_x and r_y respecting their bounds. In summary, this kind of constraint adds 5 variables and $card(P_1) + card(P_2)$ constraints to the optimization problem, while the "usual" inclusion constraint adds 0 variable and $card(P_1)card(P_2)$ constraints. The existence of the contact can alternatively be enforced by imposing a minimum area for the ellipse.

D. Maximization of the contact area

The formulation in the above section only ensures the existence of a contact of minimal size. However, one could want to make sure to find a contact area as large as possible, so that it is more likely to be able to support strong forces and ensure strong friction forces, which is helpful to ensure the stability of the robot. Therefore, it seems appropriate to try and maximize the area of contact between two polygons. As explained before, computing the area of the intersection surface is not a good practice in our case. But we know that the ellipse computed as above gives a lower bound of the contact area.

$$\mathcal{E} \subset P_1 \cap P_2 \Longrightarrow \mathcal{A}(\mathcal{E}) \leq \mathcal{A}(P_1 \cap P_2) \tag{7}$$

with $\mathcal{A}(X)$ being the area of X.
Therefore we can maximize the size of the ellipse in order to maximize the contact area. This is readily obtained by minimizing the value $Cost = -\pi r_x r_y$ in the modified problem (1). In case there are other cost functions, the above cost can be added to them with a desired weight. This requires however to scale properly the cost so as to have a meaningful and easy-to-tune weight: the range of value of the ellipse's area goes from 0 to $\mathcal{A}(P_1 \cap P_2) \leq \min(\mathcal{A}(P_1), \mathcal{A}(P_2))$. This latter quantity can be small (a typical area of contact of a humanoid robot is about $0.01m^2$, some environment surfaces can be smaller). To get a basic cost (before weighting) of magnitude around 1, we use the following scaling:

$$\text{cost}(\mathcal{E}) = -\frac{\pi r_x r_y}{\min(\mathcal{A}(P_1), \mathcal{A}(P_2))} \tag{8}$$

This cost's absolute value will always be less than 1, but not much less around the optimum, in most cases.

E. Using a non inclusive contact to maintain stability

The method we presented so far allows finding a configuration in which a new non-inclusive contact is added, but this contact does not bear any force. It is found as a geometrical contact, but will eventually have to bear some forces, and thus, become a stability contact. Usually, for a stability contact, each vertex of the contact area is considered as the application point of a force that has to be in a friction cone. Since our method allows dealing with surfaces that are intersecting each other, the contact surface is not known beforehand. Therefore, as soon as a non inclusive contact is going to be used for the stability, we compute the intersection of the two polygons P_1 and P_2 that are involved, and that intersection $P_1 \cap P_2$ is the contact surface, and its vertices will bear the forces. We do not present here the algorithm to compute the intersection of two convex polygons, as it can be found in the literature easily.

F. Extension to singular cases

Our method can be extended to be used to approximate singular situations, such as finding an optimal contact with a linear or even punctual surface. This is done by giving a slight width to the point or the line. This approximation is physically grounded: in terms of real contacts, linear or punctual contacts do not exist. In fact, since all objects are deformable, even slightly, the contact area between two objects cannot be a perfect line, and must have a non-null area. Which justifies that linear and punctual contacts can be modeled as thin contact surfaces. By defining such a surface, we impose partly the orientation of the contact. Here again, one must be careful with numerical issues. Dealing with small numbers (here we would like to take width of a fraction of centimeter) may induce conditioning problems. Also, having two close parallel constraints of opposite direction (i.e. $g(x) \leq \alpha$ and $-g(x) \leq \alpha$ with α small) is not a good practice in optimization as it will lead the solver to take small steps. Therefore, it is best to apply a scaling to the constraints by applying a geometrical scaling to P_1 and P_2 in the appropriate direction.
Likewise, constraints on the area of the ellipse should be based on the same formulation as in equation (8).

V. Simulation results

We dedicated considerable efforts in proposing a general multi-contact motion planner to solve cases of non-gaited

acyclic planning. Given a humanoid robot, an environment, a start and a final desired postures, our planner generates a sequence of contact stances allowing any part of the humanoid to make contact with any part of the environment to achieve motion towards the goal. Our planner is thoroughly described in [8]. Extensions of this multi-contact planner to multi-agent robots and objects gathering locomotion and manipulation are presented in [7], and preliminary validations with some DARPA challenge scenarios, such as climbing a ladder, ingress/egress a utility car or crossing through a relatively constrained pathway are presented in [9]. In [8] and [7], we describe works in multi-contact that are achieved by other colleagues in robotics. In order to illustrate our method, we present some examples starring the HRP-2 and ATLAS humanoid robots, that are typical posture generations encountered in multi-contact planning. For the implementation of our posture generator, we use the RobOptim optimization framework [10] relying on the IPOPT solver [11].

A. Inclined ladder climbing

In this first example, we generate a posture that is part of an inclined ladder climbing planning. We consider that the robot HRP-2 reached a posture in which its right foot is on the first step and its right hand is grasping the right guardrail, both of those contacts are bearing forces. Those contacts are fixed, and we search a posture that adds to it a geometrical contact between the left foot and the second step. We require the contact to include an ellipse with both radiuses bigger than 40% of the ladder step's width. The resulting posture and a close-up view of the contact areas are shown in Fig. 6. The latter shows clearly how an ellipse of sufficient size is found, included in the contact area between the left foot and the second step. It also shows that the contact forces on the right foot are located on the vertex of the intersection of the contact surfaces between the right foot and the first step, which was also generated with our method, in a prior posture generation. One can note that we use a contact area slightly smaller than the actual surface under the foot of the robot. We use indeed safety margins to account for modeling errors so that the obtained posture is achievable by the real robot.

B. Vertical ladder climbing

In the second example we generate a posture in which the robot climbs a vertical ladder. In this particular step, the robot is using its right foot and both hands to maintain its stability on top of the first rung of the ladder. We search a posture that keeps those previous contacts and adds a geometrical contact between the left foot and the second rung of the ladder. The result of that optimization can be observed on Fig. 1, with the robot posture on the left and a close-up look at the contact areas on the right. The difficulty of this situation is that a contact has to be made with a very thin surface of the environment (the ladder rung). Usual contact generation method would reduce a lot the span of possible contact position (by patching the robot's foot with a very small surface or by imposing a set of authorized contact positions). Whereas with our method, the contact configuration is found during the optimization process without requiring any extra human work. The contact chosen by our software includes an ellipse which first axis is the width of the robot's foot and second axis is as thin as the ladder rung. One problem to expect is that numerical instability might happen if the surface of the rung is given too thin. But that would also happen with full inclusion constraints. This example also illustrates one limitation of our method: it only considers planar contacts and if one wants to model a purely linear contact an other contact model must be used, since our modeling of those singular cases is approximative.

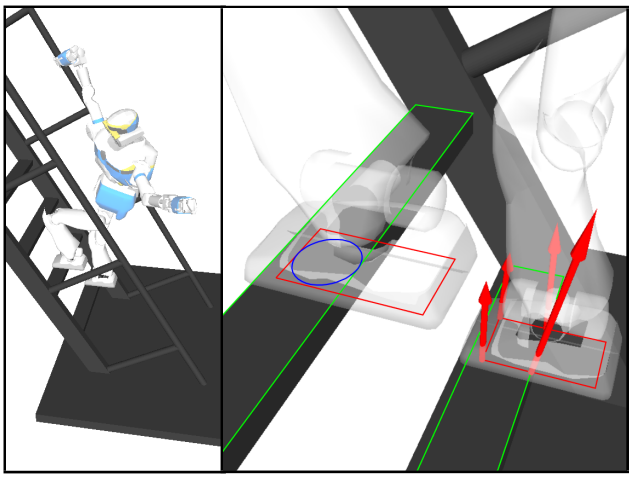

Fig. 6: HRP2-10 ladder climbing posture and up close view of the contact areas (green/red: contact polygons; blue: contact ellipse; red arrows: contact forces resultants)

C. Climbing Stairs

In a third simulation, the ATLAS robot climbs a flight of stairs. All the steps are too small for the robot to put its entire foot on. Therefore, it has to make a non inclusive contact and we propose to maximize the size of the contact area with the ellipse included in it, as explained in IV-D. The size of the contact area is limited by the fact that the foot cannot penetrate the wall behind each step. On Fig. 7, we present 3 postures generated on this environment. On each of those postures, we see that the ellipse's size is maximized until the foot enters in collision with the vertical wall behind each step. And when possible, like on the last step, the contact area is maximized without collision limitation and the foot is positioned as fully included in the support surface. We can see here that even when the size of the ellipse is maximized while competing with other non-linear constraints like collision avoidance, our method still works well and leads us to a satisfactory solution.

VI. Discussion and conclusion

Generating arbitrary shaped contact areas proved to be doable very simply in an optimization-based posture generation module. We focused on writing constraints that have continuous gradients, since the posture generate problem is dominantly smooth. Hence, our geometric contact model can be useful for other optimization-based purposes, for example

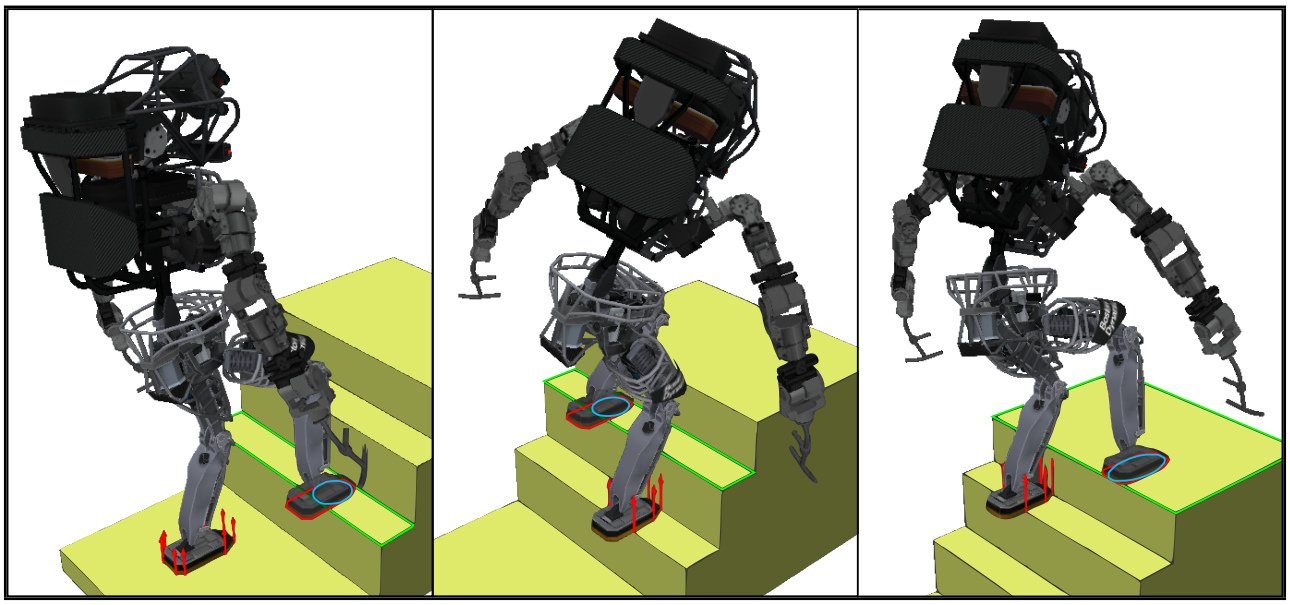

Fig. 7: Atlas climbing stairs with small steps by maximizing the size of the contact areas (green/red: contact polygons; blue: contact ellipse; red arrows: contact forces resultants)

control or trajectory optimization, and any gradient-based descent scheme which handles inequalities (e.g. [12]). While we were expecting an increase of computation time due to the addition of new variables in the problem, we noticed that the timings obtained with this method are sensibly the same that our previous version of the posture generator with full contact surface inclusion. Consequently, this method offering a richer contact search (exploration) during planning comes without degrading computation time. In fact, it truly allows us to substantially reduce the time spent by the user in ad-hoc tuning the shapes of the contact patches, or fixing the contact positions that were previously done by hand. Also, it is fairly easy to implement and extends a multi-contact planning algorithms like the one described in [8] to give it richer planning possibilities.

There are several opportunities for future work. For example, we could improve the generality of our contact constraint formulation even further to manage linear and punctual contacts without the approximations we currently use. Also we could extend our method to allow dealing with non-convex surfaces. And finally we would like to apply our method to other fields that use contact generation, like trajectory or control optimization. Our methods extends straightforwardly to point cloud data as far as polygonal patches can be extracted.

References

[1] Y. Zhang, J. Luo, K. Hauser, R. Ellenberg, P. Oh, H. Park, M. Paldhe, and C. Lee, "Motion planning of ladder climbing for humanoid robots," in *IEEE Conf. on Technologies for Practical Robot Applications*, 2013.

[2] K. Hauser, T. Bretl, J.-C. Latombe, K. Harada, and B. Wilcox, "Motion planning for legged robots on varied terrain," in *Intl. J. of Robotics Research 27(11-12):1325-1349*, 2008.

[3] S. Osswald, A. Gorog, A. Hornung, and M. Bennewitz, "Autonomous climbing of spiral staircases with humanoids," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, Sept 2011, pp. 4844–4849.

[4] J. Chestnutt, Y. Takaoka, K. Suga, K. Nishiwaki, J. Kuffner, and S. Kagami, "Biped navigation in rough environments using on-board sensing," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS'09)*. Piscataway, NJ, USA: IEEE Press, 2009, pp. 3543–3548.

[5] L. Sentis and O. Khatib, "Compliant control of multicontact and center-of-mass behaviors in humanoid robots," *IEEE Trans. Robot.*, vol. 26, no. 3, pp. 483–501, Jun. 2010.

[6] I. Mordatch, E. Todorov, and Z. Popović, "Discovery of complex behaviors through contact-invariant optimization," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 1–8, Jul. 2012.

[7] K. Bouyarmane and A. Kheddar, "Humanoid robot locomotion and manipulation step planning," *Advanced Robotics*, vol. 26, no. 10, pp. 1099–1126, 2012.

[8] 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.

[9] K. Bouyarmane, J. Vaillant, F. Keith, and A. Kheddar, "Exploring humanoid robot locomotion capabilities in virtual disaster response scenarios," in *IEEE/RSJ International Conference on Humanoid Robots*, Osaka, Japan, December 2012, pp. 337–342.

[10] T. Moulard, F. Lamiraux, K. Bouyarmane, and E. Yoshida, "Roboptim: an optimization framework for robotics," in *Japan Society for Mechanical Engineers: Robotics and Mechatronics Conference*, 2013.

[11] A. Wächter and L. Biegleri, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Math. Program.*, vol. 106, no. 1, pp. 25–57, 2006.

[12] A. Escande, N. Mansard, and P.-B. Wieber, "Fast resolution of hierarchized inverse kinematics with inequality constraints," in *IEEE International Conference on Robotics and Automation*, Anchorage, USA, 3-7 May 2010, pp. 3733 – 3738.